

9.4

*IBM MQ* 애플리케이션 개발 참조서

**IBM**

#### 참고

이 정보와 이 정보가 지원하는 제품을 사용하기 전에, [2103 페이지의 『주의사항』](#)에 있는 정보를 확인하십시오.

이 개정판은 새 개정판에 별도로 명시하지 않는 한, IBM® MQ의 버전 9릴리스 4 및 모든 후속 릴리스와 수정에 적용됩니다.

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

© Copyright International Business Machines Corporation 2007년, 2024.

# 목차

<b>애플리케이션 참조 개발.....</b>	<b>7</b>
MQI 애플리케이션 참조.....	7
코드 예제.....	8
Constants.....	61
MQI에서 사용되는 데이터 유형.....	234
함수 호출.....	604
오브젝트 속성.....	768
리턴 코드.....	840
MQI 옵션의 유효성 검증을 위한 규칙.....	840
큐잉된 발행/구독 명령 메시지.....	843
시스템 인코딩.....	862
보고 옵션 및 메시지 플래그.....	865
데이터 변환 엑시트.....	869
MQRFH2 요소로 지정된 특성.....	890
코드 페이지 변환.....	898
64비트 플랫폼에서의 코드화 표준.....	951
IBM i 애플리케이션 프로그래밍 참조(ILE/RPG).....	955
IBM i의 데이터 유형 설명.....	956
IBM i의 함수 호출.....	1194
IBM i의 오브젝트 속성.....	1304
애플리케이션.....	1347
IBM i(ILE RPG)의 리턴 코드.....	1359
IBM i(ILE RPG)에 대해 MQI 옵션 유효성 검증 규칙.....	1360
IBM i의 시스템 인코딩.....	1362
IBM i의 보고 옵션 및 메시지 플래그.....	1365
IBM i의 데이터 변환.....	1368
IBM i의 변환 처리.....	1369
IBM i의 처리 규칙.....	1370
IBM i에서 보고 메시지의 변환.....	1373
IBM i의 MQDXP(데이터 변환 엑시트 매개변수).....	1374
IBM i의 MQXCNVC(문자 변환).....	1379
IBM i의 MQCONVX(데이터 변환 엑시트).....	1383
사용자 엑시트, API 엑시트 및 설치 가능 서비스 참조.....	1386
MQIEP 구조.....	1387
데이터 변환 엑시트 참조.....	1390
MQ_PUBLISH_EXIT - 발행 엑시트.....	1394
채널 엑시트 호출 및 데이터 구조.....	1402
클러스터 워크로드 엑시트 호출 및 데이터 구조.....	1463
API 엑시트 참조.....	1487
설치 가능 서비스 인터페이스 참조 정보.....	1546
IBM i에 설치 가능 서비스 인터페이스 참조 정보.....	1607
IBM MQ .NET 클래스 및 인터페이스.....	1646
MQAsyncStatus.NET 클래스.....	1646
MQAuthenticationInformationRecord.NET 클래스.....	1647
MQDestination.NET 클래스.....	1648
MQEnvironment.NET 클래스.....	1651
MQException.NET 클래스.....	1653
MQGetMessageOptions.NET 클래스.....	1654
MQManagedObject.NET 클래스.....	1657
MQMessage.NET 클래스.....	1659
MQProcess.NET 클래스.....	1671

MQPropertyDescriptor.NET 클래스.....	1673
MQPutMessageOptions.NET 클래스.....	1674
MQQueue.NET 클래스.....	1677
MQQueueManager.NET 클래스.....	1684
MQSubscription.NET 클래스.....	1696
MQTopic.NET 클래스.....	1698
IMQObjectTrigger.NET 인터페이스.....	1703
MQC.NET 인터페이스.....	1704
.NET 애플리케이션의 문자 세트 ID.....	1704
IBM MQ C++ 클래스.....	1706
C++ 및 MQI 상호 참조.....	1708
ImqAuthenticationRecord C++ 클래스.....	1722
ImqBinary C++ 클래스.....	1725
ImqCache C++ 클래스.....	1726
ImqChannel C++ 클래스.....	1729
ImqCICSBridgeHeader C++ 클래스.....	1734
ImqDeadLetterHeader C++ 클래스.....	1740
ImqDistributionList C++ 클래스.....	1743
ImqError C++ 클래스.....	1744
ImqGetMessageOptions C++ 클래스.....	1745
ImqHeader C++ 클래스.....	1748
ImqIMSBridgeHeader C++ 클래스.....	1750
ImqItem C++ 클래스.....	1752
ImqMessage C++ 클래스.....	1754
ImqMessageTracker C++ 클래스.....	1760
ImqNamelist C++ 클래스.....	1763
ImqObject C++ 클래스.....	1765
ImqProcess C++ 클래스.....	1770
ImqPutMessageOptions C++ 클래스.....	1771
ImqQueue C++ 클래스.....	1774
ImqQueueManager C++ 클래스.....	1784
ImqReferenceHeader C++ 클래스.....	1799
ImqString C++ 클래스.....	1801
ImqTrigger C++ 클래스.....	1806
ImqWorkHeader C++ 클래스.....	1809
IBM MQ classes for JMS 오브젝트의 특성.....	1811
IBM MQ classes for JMS 오브젝트 특성의 종속성.....	1814
APPLICATIONNAME.....	1816
ASYNCEXCEPTION.....	1817
기본 옵션.....	1818
BALTYPE.....	1818
BALTIMEOUT.....	1819
BROKERCCDURSUBQ.....	1819
BROKERCCSUBQ.....	1820
BROKERCONQ.....	1820
BROKERDURSUBQ.....	1821
BROKERPUBQ.....	1821
BROKERPUBQMGR.....	1821
BROKERQMGR.....	1822
BROKERSUBQ.....	1822
BROKERVER.....	1823
CCDTURL.....	1823
CCSID.....	1824
CERTVALPO.....	1824
CHANNEL.....	1825
CLEANUP.....	1825
CLEANUPINT.....	1826





CONNECTIONNAMELIST.....	1826
CLIENTRECONNECTOPTIONS.....	1827
CLIENTRECONNECTTIMEOUT.....	1827
CLIENTID.....	1828
CLONESUPP.....	1828
COMPHDR.....	1829
COMPMSG.....	1829
CONNOPT.....	1830
CONNTAG.....	1831
설명.....	1831
DIRECTAUTH.....	1831
ENCODING.....	1832
EXPIRY.....	1833
FAILIFQUIESCE.....	1833
호스트 이름.....	1834
LOCALADDRESS.....	1834
MAPNAMESTYLE.....	1835
MAXBUFFSIZE.....	1836
MDREAD.....	1836
MDWRITE.....	1837
MDMSGCTX.....	1837
MSGBATCHSZ.....	1838
MSGBODY.....	1838
MSGRETENTION.....	1839
MSGSELECTION.....	1839
MULTICAST.....	1840
OPTIMISTICPUBLICATION.....	1840
OUTCOMENOTIFICATION.....	1841
PERSISTENCE.....	1841
POLLINGINT.....	1842
포트.....	1842
PRIORITY.....	1843
PROCESSDURATION.....	1843
PROVIDERVERSION.....	1844
PROXYHOSTNAME.....	1846
PROXYPORT.....	1846
PUBACKINT.....	1847
PUTASYNCALLOWED.....	1847
QMANAGER.....	1848
큐.....	1848
READAHEADALLOWED.....	1848
READAHEADCLOSEPOLICY.....	1849
RECEIVECCSID.....	1850
RECEIVECONVERSION.....	1850
RECEIVEISOLATION.....	1851
RECEXIT.....	1851
RECEXITINIT.....	1851
REPLYTOSTYLE.....	1852
RESCANINT.....	1853
SECEXIT.....	1853
SECEXITINIT.....	1854
SENDCHECKCOUNT.....	1854
SENDEXIT.....	1854
SENDEXITINIT.....	1855
SHARECONVALLOWED.....	1855
SPARSESUBS.....	1856
SSLCIPHERSUITE.....	1856
SSLCRL.....	1857

SSLFIPSREQUIRED.....	1857
SSLPEERNAME.....	1858
SSLRESETCOUNT.....	1858
STATREFRESHINT.....	1859
SUBSTORE.....	1859
SYNCPOINTALLGETS.....	1860
TARGCLIENT.....	1860
TARGCLIENTMATCHING.....	1861
TEMPMODEL.....	1861
TEMPQPREFIX.....	1862
TEMPTOPICPREFIX.....	1862
TOPIC.....	1862
TRANSPORT.....	1863
WILDCARDFORMAT.....	1864
ENCODING 특성.....	1864
JMS 오브젝트의 TLS 특성.....	1865
IBM MQ Message Service Client (XMS) for .NET 참조.....	1866
.NET 인터페이스.....	1866
XMS 오브젝트의 특성.....	1945
Managed File Transfer 애플리케이션 참조 개발.....	2006
fteCreateTransfer를 사용한 프로그램 시작 예.....	2006
<b>fteAnt</b> : MFT 에서 Ant 태스크 실행.....	2008
사용자 정의 참조용 MFT 사용자 엑시트.....	2032
MFT 에이전트 명령 큐에 넣을 수 있는 메시지의 메시지 형식.....	2072
메시징 REST API 참조.....	2072
REST API 자원.....	2072
<b>주의사항.....</b>	<b>2103</b>
프로그래밍 인터페이스 정보.....	2104
상표.....	2104

## 애플리케이션 참조 개발

---

이 절에서 제공하는 링크를 사용하면 IBM MQ 애플리케이션 개발에 도움이 됩니다.

- 7 페이지의 『MQI 애플리케이션 참조』
-  955 페이지의 『IBM i 애플리케이션 프로그래밍 참조(ILE/RPG)』
-  1368 페이지의 『IBM i의 데이터 변환』
- 1386 페이지의 『사용자 엑시트, API 엑시트 및 설치 가능 서비스 참조』
- 1646 페이지의 『IBM MQ .NET 클래스 및 인터페이스』
- 1706 페이지의 『IBM MQ C++ 클래스』
- 1811 페이지의 『IBM MQ classes for JMS 오브젝트의 특성』
- 2072 페이지의 『메시징 REST API 참조』

### 관련 태스크

[애플리케이션 개발](#)

### 관련 참조

[Java 라이브러리에 대한 IBM MQ 클래스](#)

[JMS용 IBM MQ 클래스](#)

## MQI 애플리케이션 참조

---

이 절에서 제공하는 링크를 사용하면 MQI(Message Queue Interface) 애플리케이션 개발에 도움이 됩니다.

- 8 페이지의 『코드 예제』
- 61 페이지의 『Constants』
- 234 페이지의 『MQI에서 사용되는 데이터 유형』
- 604 페이지의 『함수 호출』
- 768 페이지의 『오브젝트 속성』
- 840 페이지의 『리턴 코드』
- 840 페이지의 『MQI 옵션의 유효성 검증을 위한 규칙』
- 862 페이지의 『시스템 인코딩』
- 865 페이지의 『보고 옵션 및 메시지 플래그』
- 869 페이지의 『데이터 변환 엑시트』
- 890 페이지의 『MQRFH2 요소로 지정된 특성』
- 898 페이지의 『코드 페이지 변환』

### 관련 개념

[1386 페이지의 『사용자 엑시트, API 엑시트 및 설치 가능 서비스 참조』](#)

사용자 엑시트, API 엑시트 및 설치 가능한 서비스 애플리케이션 개발에 유용한 이 섹션의 정보를 사용하십시오.

### 관련 태스크

[애플리케이션 개발](#)

### 관련 참조

[1646 페이지의 『IBM MQ .NET 클래스 및 인터페이스』](#)

IBM MQ .NET 클래스 및 인터페이스가 알파벳순으로 나열되어 있습니다. 특성, 메소드 및 구성자에 대해 설명합니다.

[1706 페이지의 『IBM MQ C++ 클래스』](#)

IBM MQ C++ 클래스는 IBM MQ MQI(Message Queue Interface)를 캡슐화합니다. 이러한 클래스를 모두 포함하는 단일 C++ 헤더 파일, **imqi.hpp**가 있습니다.

[Java 라이브러리의 IBM MQ 클래스](#)

[JMS 용 IBM MQ 클래스](#)

## 코드 예제

이 절의 참조 정보를 사용하여 비즈니스 요구를 해결하는 태스크를 달성합니다.

### C 언어 예제

이 토픽 컬렉션은 주로 IBM MQ for z/OS 샘플 애플리케이션에서 가져옵니다. 명시된 경우를 제외하고 모든 플랫폼에 적용할 수 있습니다.

#### 큐 관리자에 연결

이 예에서는 MQCONN 호출을 사용하여 프로그램을 z/OS 일괄처리의 큐 관리자에 연결하는 방법을 보여줍니다.

이 추출은 IBM MQ for z/OS에서 제공되는 찾아보기 샘플 애플리케이션(프로그램 CSQ4BCA1)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.

```
#include <cmqc.h>
...
static char Parm1[MQ_Q_MGR_NAME_LENGTH] ;

int main(int argc, char *argv[] )
{
    /*                                     */
    /*   Variables for MQ calls           */
    /*                                     */
    MQHCONN Hconn;      /* Connection handle   */
    MQLONG  CompCode;   /* Completion code    */
    MQLONG  Reason;    /* Qualifying reason  */

    /* Copy the queue manager name, passed in the */
    /* parm field, to Parm1                       */
    strncpy(Parm1,argv[1],MQ_Q_MGR_NAME_LENGTH);

    /*                                     */
    /* Connect to the specified queue manager.    */
    /* Test the output of the connect call.  If the */
    /* call fails, print an error message showing the */
    /* completion code and reason code, then leave the */
    /* program.                                     */
    /*                                     */
    MQCONN(Parm1,
           &Hconn,
           &CompCode,
           &Reason);
    if ((CompCode != MQCC_OK) | (Reason != MQRC_NONE))
    {
        sprintf(pBuff, MESSAGE_4_E,
                ERROR_IN_MQCONN, CompCode, Reason);
        PrintLine(pBuff);
        RetCode = CSQ4_ERROR;
        goto AbnormalExit2;
    }
    ...
}
```

#### 큐 관리자에서 연결 끊기

이 예에서는 MQDISC 호출을 사용하여 z/OS 일괄처리의 큐 관리자에서 프로그램의 연결을 끊는 방법을 보여줍니다.

이 코드 추출에서 사용된 변수는 8 페이지의 [『큐 관리자에 연결』](#)에서 설정된 변수입니다. 이 추출은 IBM MQ for z/OS에서 제공되는 찾아보기 샘플 애플리케이션(프로그램 CSQ4BCA1)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.

```

:
/*
/* Disconnect from the queue manager. Test the      */
/* output of the disconnect call. If the call        */
/* fails, print an error message showing the        */
/* completion code and reason code.                */
/*
MQDISC(&Hconn,
        &CompCode,
        &Reason);
if ((CompCode != MQCC_OK) || (Reason != MQRC_NONE))
{
    sprintf(pBuff, MESSAGE_4_E,
            ERROR_IN_MQDISC, CompCode, Reason);
    PrintLine(pBuff);
    RetCode = CSQ4_ERROR;
}
:

```

## 동적 큐 작성

이 예에서는 동적 큐를 작성하기 위해 MQOPEN 호출을 사용하는 방법을 설명합니다.

이 추출은 IBM MQ for z/OS에서 제공하는 메일 관리자 샘플 애플리케이션(프로그램 CSQ4TCD1)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)을 참조하십시오](#).

```

:
MQLONG  HCONN = 0; /* Connection handle */
MQHOBJS HOBJ; /* MailQ Object handle */
MQHOBJS HobjTempQ; /* TempQ Object Handle */
MQLONG  CompCode; /* Completion code */
MQLONG  Reason; /* Qualifying reason */
MQOD  ObjDesc = {MQOD_DEFAULT};
/* Object descriptor */
MQLONG  OpenOptions; /* Options control MQOPEN */

/*-----*/
/* Initialize the Object Descriptor (MQOD) */
/* control block. (The remaining fields */
/* are already initialized.) */
/*-----*/
strncpy( ObjDesc.ObjectName,
        SYSTEM_REPLY_MODEL,
        MQ_Q_NAME_LENGTH );
strncpy( ObjDesc.DynamicQName,
        SYSTEM_REPLY_INITIAL,
        MQ_Q_NAME_LENGTH );
OpenOptions = MQOO_INPUT_AS_Q_DEF;
/*-----*/
/* Open the model queue and, therefore, */
/* create and open a temporary dynamic */
/* queue */
/*-----*/
MQOPEN( HCONN,
        &ObjDesc,
        OpenOptions,
        &HobjTempQ,
        &CompCode,
        &Reason );
if ( CompCode == MQCC_OK ) {
}
else {
    /*-----*/
    /* Build an error message to report the */
    /* failure of the opening of the model */
    /* queue */
    /*-----*/
    MQMErrorHandling( "OPEN TEMPQ", CompCode,
                    Reason );
    ErrorFound = TRUE;
}
return ErrorFound;

```

```
}  
...
```

## 기존 큐 열기

이 예에서는 MQOPEN 호출을 사용하여 이미 정의된 큐를 여는 방법을 설명합니다.

이 추출은 IBM MQ for z/OS에서 제공되는 찾아보기 샘플 애플리케이션(프로그램 CSQ4BCA1)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.

```
#include <cmqc.h>  
...  
static char Parm1[MQ_Q_MGR_NAME_LENGTH];  
...  
int main(int argc, char *argv[] )  
{  
    /*  
    /*      Variables for MQ calls                               */  
    /*  
    MQHCONN Hconn ;           /* Connection handle             */  
    MQLONG  CompCode;        /* Completion code       */  
    MQLONG  Reason;         /* Qualifying reason     */  
    MQOD    ObjDesc = { MQOD_DEFAULT };  
    MQLONG  OpenOptions;     /* Options that control  */  
    /*      the MQOPEN call                                     */  
    MQHOBJ  Hobj;           /* Object handle         */  
    ...  
    /* Copy the queue name, passed in the parm field,        */  
    /* to Parm2 strncpy(Parm2,argv[2],                       */  
    /* MQ_Q_NAME_LENGTH);                                    */  
    ...  
    /* Initialize the object descriptor (MQOD) control      */  
    /* block. (The initialization default sets StrucId,     */  
    /* Version, ObjectType, ObjectQMgrName,                 */  
    /* DynamicQName, and AlternateUserid fields)           */  
    /*                                                      */  
    strncpy(ObjDesc.ObjectName,Parm2,MQ_Q_NAME_LENGTH);  
    ...  
    /* Initialize the other fields required for the open   */  
    /* call (Hobj is set by the MQCONN call).              */  
    /*                                                      */  
    OpenOptions = MQOO_BROWSE;  
    ...  
    /*                                                      */  
    /* Open the queue.                                       */  
    /* Test the output of the open call. If the call       */  
    /* fails, print an error message showing the           */  
    /* completion code and reason code, then bypass       */  
    /* processing, disconnect and leave the program.       */  
    /*                                                      */  
    MQOPEN(Hconn,  
           &ObjDesc,  
           OpenOptions,  
           &Hobj,  
           &CompCode,  
           &Reason);  
    ...  
    if ((CompCode != MQCC_OK) || (Reason != MQRC_NONE))  
    {  
        sprintf(pBuff, MESSAGE_4_E,  
                ERROR_IN_MQOPEN, CompCode, Reason);  
        PrintLine(pBuff);  
        RetCode = CSQ4_ERROR;  
        goto AbnormalExit1; /* disconnect processing */  
    }  
    ...  
} /* end of main */
```

## 큐 닫기

이 예에서는 MQCLOSE 호출을 사용하여 큐를 닫는 방법을 보여줍니다.

이 추출은 IBM MQ for z/OS에서 제공되는 찾아보기 샘플 애플리케이션(프로그램 CSQ4BCA1)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.


```

:
/*                                     */
/* Close the queue.                   */
/* Test the output of the close call  */
/* fails, print an error message showing the */
/* completion code and reason code.   */
/*                                     */
MQCLOSE(Hconn,
        &Hobj,
        MQCO_NONE,
        &CompCode,
        &Reason);
if ((CompCode != MQCC_OK) || (Reason != MQRC_NONE))
{
    sprintf(pBuff, MESSAGE_4_E,
            ERROR_IN_MQCLOSE, CompCode, Reason);
    PrintLine(pBuff);
    RetCode = CSQ4_ERROR;
}
:

```

## MQPUT을 사용하여 메시지 넣기

이 예에서는 MQPUT 호출을 사용하여 큐에 메시지를 넣는 방법을 설명합니다.

이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다. 샘플 애플리케이션의 이름 및 위치는 [샘플 프로시저 프로그램\(z/OS 이외의 플랫폼\)](#)  및 [IBM MQ for z/OS의 샘플 프로그램을 참조](#) 하십시오.

```

:
qput()
{
    MQMD    MsgDesc;
    MQPMO   PutMsgOpts;
    MQLONG  CompCode;
    MQLONG  Reason;
    MQHCONN Hconn;
    MQHOBJ  Hobj;
    char message_buffer[] = "MY MESSAGE";
    /*-----*/
    /* Set up PMO structure.          */
    /*-----*/
    memset(&PutMsgOpts, '\0', sizeof(PutMsgOpts));
    memcpy(PutMsgOpts.StrucId, MQPMO_STRUC_ID,
           sizeof(PutMsgOpts.StrucId));
    PutMsgOpts.Version = MQPMO_VERSION_1;
    PutMsgOpts.Options = MQPMO_SYNCPOINT;

    /*-----*/
    /* Set up MD structure.           */
    /*-----*/
    memset(&MsgDesc, '\0', sizeof(MsgDesc));
    memcpy(MsgDesc.StrucId, MQMD_STRUC_ID,
           sizeof(MsgDesc.StrucId));
    MsgDesc.Version      = MQMD_VERSION_1;
    MsgDesc.Expiry       = MQEI_UNLIMITED;
    MsgDesc.Report       = MQRO_NONE;
    MsgDesc.MsgType      = MQMT_DATAGRAM;
    MsgDesc.Priority     = 1;
    MsgDesc.Persistence  = MQPER_PERSISTENT;
    memset(MsgDesc.ReplyToQ,
           '\0',
           sizeof(MsgDesc.ReplyToQ));
    /*-----*/
    /* Put the message.               */
    /*-----*/
    MQPUT(Hconn, Hobj, &MsgDesc, &PutMsgOpts,
          sizeof(message_buffer), message_buffer,
          &CompCode, &Reason);
}
:

```

```

/*-----*/
/* Check completion and reason codes. */
/*-----*/
switch (CompCode)
{
    case MQCC_OK:
        break;
    case MQCC_FAILED:
        switch (Reason)
        {
            case MQRC_Q_FULL:
            case MQRC_MSG_TOO_BIG_FOR_Q:
                break;
            default:
                break; /* Perform error processing */
        }
        break;
    default:
        break; /* Perform error processing */
}
}

```

## MQPUT1을 사용하여 메시지 넣기

이 예에서는 MQPUT1 호출을 사용하여 큐를 열고 큐에 단일 메시지를 넣은 다음, 큐를 닫는 방법을 보여줍니다.

이 추출은 IBM MQ for z/OS에서 제공되는 신용 검사 샘플 애플리케이션(프로그램 CSQ4CCB5)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.

```

:
MQLONG  Hconn;           /* Connection handle      */
MQHOBJ  Hobj_CheckQ;   /* Object handle          */
MQLONG  CompCode;      /* Completion code        */
MQLONG  Reason;        /* Qualifying reason      */
MQOD    ObjDesc = {MQOD_DEFAULT}; /* Object descriptor      */
MQMD    MsgDesc = {MQMD_DEFAULT}; /* Message descriptor     */
MQLONG  OpenOptions;   /* Control the MQOPEN call */
MQGMO   GetMsgOpts = {MQGMO_DEFAULT}; /* Get Message Options   */
MQLONG  MsgBuffLen;    /* Length of message buffer */
CSQ4BCAQ MsgBuffer;    /* Message structure      */
MQLONG  DataLen;       /* Length of message      */

MQPMO   PutMsgOpts = {MQPMO_DEFAULT}; /* Put Message Options   */
CSQ4BQRM PutBuffer;    /* Message structure      */
MQLONG  PutBuffLen = sizeof(PutBuffer); /* Length of message buffer */
:

```

```

void Process_Query(void)
{
    /*
    /* Build the reply message
    /*
    /*
    /*
    /* Set the object descriptor, message descriptor and
    /* put message options to the values required to
    /* create the reply message.
    /*
    strncpy(ObjDesc.ObjectName, MsgDesc.ReplyToQ,
            MQ_Q_NAME_LENGTH);
    strncpy(ObjDesc.ObjectQMgrName, MsgDesc.ReplyToQMgr,
            MQ_Q_MGR_NAME_LENGTH);
    MsgDesc.MsgType = MQMT_REPLY;
    MsgDesc.Report = MQRO_NONE;
    memset(MsgDesc.ReplyToQ, ' ', MQ_Q_NAME_LENGTH);
    memset(MsgDesc.ReplyToQMgr, ' ', MQ_Q_MGR_NAME_LENGTH);
    memcpy(MsgDesc.MsgId, MQMI_NONE, sizeof(MsgDesc.MsgId));
    */
}

```



```

PutMsgOpts.Options = MQPMO_SYNCPOINT +
                    MQPMO_PASS_IDENTITY_CONTEXT;
PutMsgOpts.Context = Hobj_CheckQ;
PutBuffLen = sizeof(PutBuffer);
MQPUT1(Hconn,
        &ObjDesc,
        &MsgDesc,
        &PutMsgOpts,
        PutBuffLen,
        &PutBuffer,
        &CompCode,
        &Reason);

if (CompCode != MQCC_OK)
{
    strncpy(TS_Operation, "MQPUT1",
            sizeof(TS_Operation));
    strncpy(TS_ObjName, ObjDesc.ObjectName,
            MQ_Q_NAME_LENGTH);
    Record_Call_Error();
    Forward_Msg_To_DLQ();
}
return;
}
:

```

## 메시지 받기

이 예에서는 MQGET 호출을 사용하여 큐에서 메시지를 제거하는 방법을 설명합니다.

이 추출은 IBM MQ for z/OS에서 제공되는 찾아보기 샘플 애플리케이션(프로그램 CSQ4BCA1)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.

```

#include "cmqc.h"
:
#define BUFFERLENGTH 80
:
int main(int argc, char *argv[] )
{
    /*
    /* Variables for MQ calls
    /*
    /* Connection handle
    /* Completion code
    /* Qualifying reason
    /* Object handle
    /* Message descriptor
    /* Length of the message
    /* Area for message data
    /* Options which control
    /* the MQGET call
    /* Length of buffer
    :
    /* No need to change the message descriptor
    /* (MQMD) control block because initialization
    /* default sets all the fields.
    /*
    /* Initialize the get message options (MQGMO)
    /* control block (the copy file initializes all
    /* the other fields).
    /*
    /*
    MQGMO_NO_WAIT
    MQGMO_BROWSE_FIRST +
    MQGMO_ACCEPT_TRUNCATED_MSG;
    /*
    /* Get the first message.
    /* Test for the output of the call is carried out
    /* in the 'for' loop.
    /*
    /*
    MQGET(Hconn,
        Hobj,
        &MsgDesc,
        &GetMsgOpts,

```

```

    BufferLength,
    Buffer,
    &DataLength,
    &CompCode,
    &Reason);

```

```

/*                                     */
/* Process the message and get the next message, */
/* until no messages remaining. */
:
/*     If the call fails for any other reason, */
/*     print an error message showing the completion */
/*     code and reason code. */
/*                                     */
if ( (CompCode == MQCC_FAILED) &&
     (Reason == MQRC_NO_MSG_AVAILABLE) )
{
    :
}
else
{
    sprintf(pBuff, MESSAGE_4_E,
            ERROR_IN_MQGET, CompCode, Reason);
    PrintLine(pBuff);
    RetCode = CSQ4_ERROR;
}
:
} /* end of main */

```

## 대기 옵션을 사용하여 메시지 받기

이 예에서는 MQGET 호출의 대기 옵션을 사용하는 방법을 설명합니다.

이 코드는 잘린 메시지를 허용합니다. 이 추출은 IBM MQ for z/OS에서 제공되는 신용 검사 샘플 애플리케이션 (프로그램 CSQ4CCB5)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.

```

:
MQLONG  Hconn;                /* Connection handle */
MQHOBJ  Hobj_CheckQ;         /* Object handle */
MQLONG  CompCode;           /* Completion code */
MQLONG  Reason;             /* Qualifying reason */
MQOD    ObjDesc = {MQOD_DEFAULT};
MQMD    MsgDesc = {MQMD_DEFAULT};
MQLONG  OpenOptions;
MQGMO   GetMsgOpts = {MQGMO_DEFAULT};
MQLONG  MsgBufLen;          /* Length of message buffer */
CSQ4BCAQ MsgBuffer;         /* Message structure */
MQLONG  DataLen;           /* Length of message */

```

```

:
void main(void)
{
    :
    /*                                     */
    /* Initialize options and open the queue for input */
    /*                                     */
    :
    /*                                     */
    /* Get and process messages */
    /*                                     */
    GetMsgOpts.Options = MQGMO_WAIT +
                        MQGMO_ACCEPT_TRUNCATED_MSG +
                        MQGMO_SYNCPOINT;
    GetMsgOpts.WaitInterval = WAIT_INTERVAL;
    MsgBufLen = sizeof(MsgBuffer);
    memcpy(MsgDesc.MsgId, MQMI_NONE,
           sizeof(MsgDesc.MsgId));
    memcpy(MsgDesc.CorrelId, MQCI_NONE,
           sizeof(MsgDesc.CorrelId));
}

```

```

/*                                     */
/* Make the first MQGET call outside the loop */
/*                                     */
MQGET(Hconn,
      Hobj_CheckQ,
      &MsgDesc,
      &GetMsgOpts,
      MsgBuffLen,
      &MsgBuffer,
      &DataLen,
      &CompCode,
      &Reason);
:
/*                                     */
/* Test the output of the MQGET call. If the call */
/* failed, send an error message showing the */
/* completion code and reason code, unless the */
/* reason code is NO_MSG_AVAILABLE. */
/*                                     */
if (Reason != MQRC_NO_MSG_AVAILABLE)
{
    strncpy(TS_Operation, "MQGET", sizeof(TS_Operation));
    strncpy(TS_ObjName, ObjDesc.ObjectName,
            MQ_Q_NAME_LENGTH);
    Record_Call_Error();
}
:

```

## 신호보내기를 사용하여 메시지 받기

신호보내기는 *IBM MQ for z/OS*에서만 사용 가능합니다.

이 예제는 적당한 메시지가 큐에 도착할 때 알림을 받을 수 있도록 MQGET 호출을 사용하여 신호를 설정하는 방법을 예시합니다. 이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다.

```

:
get_set_signal()
{
    MQMD    MsgDesc;
    MQGMO   GetMsgOpts;
    MQLONG  CompCode;
    MQLONG  Reason;
    MQHCONN Hconn;
    MQHOBJ  Hobj;
    MQLONG  BufferLength;
    MQLONG  DataLength;
    char message_buffer[100];
    long int q_ecb, work_ecb;
    short int signal_sw, endloop;
    long int mask = 255;

    /*-----*/
    /* Set up GMO structure. */
    /*-----*/
    memset(&GetMsgOpts, '\0', sizeof(GetMsgOpts));
    memcpy(GetMsgOpts.StrucId, MQGMO_STRUC_ID,
           sizeof(GetMsgOpts.StrucId));
    GetMsgOpts.Version = MQGMO_VERSION_1;
    GetMsgOpts.WaitInterval = 1000;
    GetMsgOpts.Options = MQGMO_SET_SIGNAL +
                        MQGMO_BROWSE_FIRST;
    q_ecb = 0;
    GetMsgOpts.Signal1 = &q_ecb;
    /*-----*/
    /* Set up MD structure. */
    /*-----*/
    memset(&MsgDesc, '\0', sizeof(MsgDesc));
    memcpy(MsgDesc.StrucId, MQMD_STRUC_ID,
           sizeof(MsgDesc.StrucId));
    MsgDesc.Version = MQMD_VERSION_1;
    MsgDesc.Report = MQRO_NONE;
    memcpy(MsgDesc.MsgId, MQMI_NONE,
           sizeof(MsgDesc.MsgId));
    memcpy(MsgDesc.CorrelId, MQCI_NONE,
           sizeof(MsgDesc.CorrelId));
}

```

```

/*-----*/
/* Issue the MQGET call. */
/*-----*/
BufferLength = sizeof(message_buffer);
signal_sw = 0;

MQGET(Hconn, Hobj, &MsgDesc, &GetMsgOpts,
      BufferLength, message_buffer, &DataLength,
      &CompCode, &Reason);
/*-----*/
/* Check completion and reason codes. */
/*-----*/
switch (CompCode)
{
  case (MQCC_OK):          /* Message retrieved */
    break;
  case (MQCC_WARNING):
    switch (Reason)
    {
      case (MQRC_SIGNAL_REQUEST_ACCEPTED):
        signal_sw = 1;
        break;
      default:
        break; /* Perform error processing */
    }
    break;
  case (MQCC_FAILED):
    switch (Reason)
    {
      case (MQRC_Q_MGR_NOT_AVAILABLE):
      case (MQRC_CONNECTION_BROKEN):
      case (MQRC_Q_MGR_STOPPING):
        break;
      default:
        break; /* Perform error processing. */
    }
    break;
  default:
    break; /* Perform error processing. */
}
/*-----*/
/* If the SET_SIGNAL was accepted, set up a loop to */
/* check whether a message has arrived at one second */
/* intervals. The loop ends if a message arrives or */
/* the wait interval specified in the MQGMO */
/* structure has expired. */
/* */
/* If a message arrives on the queue, another MQGET */
/* must be issued to retrieve the message. If other */
/* MQM calls have been made in the intervening */
/* period, this may necessitate reinitializing the */
/* MQMD and MQGMO structures. */
/* In this code, no intervening calls */
/* have been made, so the only change required to */
/* the structures is to specify MQGMO_NO_WAIT, */
/* since we now know the message is there. */
/* */
/* This code uses the EXEC CICS DELAY command to */
/* suspend the program for a second. A batch program */
/* may achieve the same effect by calling an */
/* assembler language subroutine which issues a */
/* z/OS STIMER macro. */
/*-----*/

```

```

if (signal_sw == 1)
{
  endloop = 0;
  do
  {
    EXEC CICS DELAY FOR HOURS(0) MINUTES(0) SECONDS(1);
    work_ecb = q_ecb & mask;
    switch (work_ecb)
    {
      case (MQEC_MSG_ARRIVED):
        endloop = 1;
        mqgmo_options = MQGMO_NO_WAIT;
        MQGET(Hconn, Hobj, &MsgDesc, &GetMsgOpts,
              BufferLength, message_buffer,

```

```

        &DataLength, &CompCode, &Reason);
        if (CompCode != MQCC_OK)
            ; /* Perform error processing. */
        break;
        case (MQEC_WAIT_INTERVAL_EXPIRED):
        case (MQEC_WAIT_CANCELED):
            endloop = 1;
            break;
        default:
            break;
    }
} while (endloop == 0);
}
return;
}
}

```

## 오브젝트의 속성 조회

이 예에서는 MQINQ 호출을 사용하여 큐의 속성을 조회하는 방법을 설명합니다.

이 추출은 IBM MQ for z/OS에서 제공되는 큐 속성 샘플 애플리케이션(프로그램 CSQ4CCC1)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.

```

#include <mqc.h> /* MQ API header file */
:
#define NUMBEROFSELECTORS 2

const MQHCONN Hconn = MQHC_DEF_HCONN;
:
static void InquireGetAndPut(char *Message,
                             PMQHOBJ pHobj,
                             char *Object)
{
/* Declare local variables */
/* */
MQLONG SelectorCount = NUMBEROFSELECTORS;
/* Number of selectors */
MQLONG IntAttrCount = NUMBEROFSELECTORS;
/* Number of int attrs */
MQLONG CharAttrLength = 0;
/* Length of char attribute buffer */
MQCHAR *CharAttrs ;
/* Character attribute buffer */
MQLONG SelectorsTable[NUMBEROFSELECTORS];
/* attribute selectors */
MQLONG IntAttrsTable[NUMBEROFSELECTORS];
/* integer attributes */
MQLONG CompCode;
/* Completion code */
MQLONG Reason;
/* Qualifying reason */
/* */
/* Open the queue. If successful, do the inquire */
/* call. */
/* */
/* */
/* Initialize the variables for the inquire */
/* call: */
/* - Set SelectorsTable to the attributes whose */
/* status is */
/* required */
/* - All other variables are already set */
/* */
SelectorsTable[0] = MQIA_INHIBIT_GET;
SelectorsTable[1] = MQIA_INHIBIT_PUT;
/* */
/* Issue the inquire call */
/* Test the output of the inquire call. If the */
/* call failed, display an error message */
/* showing the completion code and reason code, */
/* otherwise display the status of the */
/* INHIBIT-GET and INHIBIT-PUT attributes */
/* */
MQINQ(Hconn,
      *pHobj,
      SelectorCount,

```

```

        SelectorTable,
        IntAttrCount,
        IntAttrsTable,
        CharAttrLength,
        CharAttrs,
        &CompCode,
        &Reason);
    if (CompCode != MQCC_OK)
    {
        sprintf(Message, MESSAGE_4_E,
                ERROR_IN_MQINQ, CompCode, Reason);
        SetMsg(Message);
    }
    else
    {
        /* Process the changes */
    } /* end if CompCode */

```

## 큐의 속성 설정

이 예에서는 MQSET 호출을 사용하여 큐의 속성을 변경하는 방법을 설명합니다.

이 추출은 IBM MQ for z/OS에서 제공되는 큐 속성 샘플 애플리케이션(프로그램 CSQ4CCC1)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.

```

#include <mqc.h>      /* MQ API header file      */
:
#define NUMBEROFSELECTORS 2

const MQHCONN Hconn = MQHC_DEF_HCONN;

static void InhibitGetAndPut(char *Message,
                             PMQHOBJ pHobj,
                             char *Object)
{
    /*
    /* Declare local variables
    /*
    /*
    MQLONG SelectorCount = NUMBEROFSELECTORS;
    /* Number of selectors
    MQLONG IntAttrCount = NUMBEROFSELECTORS;
    /* Number of int attrs
    MQLONG CharAttrLength = 0;
    /* Length of char attribute buffer
    MQCHAR *CharAttrs ;
    /* Character attribute buffer
    MQLONG SelectorTable[NUMBEROFSELECTORS];
    /* attribute selectors
    MQLONG IntAttrsTable[NUMBEROFSELECTORS];
    /* integer attributes
    MQLONG CompCode;
    /* Completion code
    MQLONG Reason;
    /* Qualifying reason
    :
    /*
    /* Open the queue. If successful, do the
    /* inquire call.
    /*
    :
    /*
    /* Initialize the variables for the set call:
    /* - Set SelectorTable to the attributes to be
    /* set
    /* - Set IntAttrsTable to the required status
    /* - All other variables are already set
    /*
    SelectorTable[0] = MQIA_INHIBIT_GET;
    SelectorTable[1] = MQIA_INHIBIT_PUT;
    IntAttrsTable[0] = MQQA_GET_INHIBITED;
    IntAttrsTable[1] = MQQA_PUT_INHIBITED;
    :

    /*
    /* Issue the set call.
    /* Test the output of the set call. If the
    /* call fails, display an error message
    */

```

```

/*      showing the completion code and reason      */
/*      code; otherwise move INHIBITED to the      */
/*      relevant screen map fields                  */
/*      */
MQSET(Hconn,
      *pHobj,
      SelectorCount,
      SelectorsTable,
      IntAttrCount,
      IntAttrsTable,
      CharAttrLength,
      CharAttrs,
      &CompCode,
      &Reason);
if (CompCode != MQCC_OK)
{
    sprintf(Message, MESSAGE_4_E,
            ERROR_IN_MQSET, CompCode, Reason);
    SetMsg(Message);
}
else
{
    /* Process the changes */
} /* end if CompCode */

```

## MQSTAT으로 상태 정보 검색

이 예에서는 비동기 MQPUT을 실행하고 MQSTAT으로 상태 정보를 검색하는 방법을 설명합니다.

이 추출은 IBM MQ for Windows 시스템에서 제공되는 MQSTAT 호출 샘플 애플리케이션(프로그램 amqsapt0)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.

```

/*****
/*
/* Program name: AMQSAPT0
/*
/* Description: Sample C program that asynchronously puts messages
/*              to a message queue (example using MQPUT & MQSTAT).
/*
/* Licensed Materials - Property of IBM
/*
/* 63H9336
/* (c) Copyright IBM Corp. 2006, 2024. All Rights Reserved.
/*
/* US Government Users Restricted Rights - Use, duplication or
/* disclosure restricted by GSA ADP Schedule Contract with
/* IBM Corp.
/*
/*****
/* Function:
/*
/* AMQSAPT0 is a sample C program to put messages on a message
/* queue with asynchronous response option, querying the success
/* of the put operations with MQSTAT.
/*
/* -- messages are sent to the queue named by the parameter
/*
/* -- gets lines from StdIn, and adds each to target
/* queue, taking each line of text as the content
/* of a datagram message; the sample stops when a null
/* line (or EOF) is read.
/* New-line characters are removed.
/* If a line is longer than 99 characters it is broken up
/* into 99-character pieces. Each piece becomes the
/* content of a datagram message.
/* If the length of a line is a multiple of 99 plus 1, for
/* example, 199, the last piece will only contain a
/* new-line character so will terminate the input.
/*
/* -- writes a message for each MQI reason other than
/* MQRC_NONE; stops if there is a MQI completion code
/* of MQCC_FAILED
/*
/* -- summarizes the overall success of the put operations
/* through a call to MQSTAT to query MQSTAT_TYPE_ASYNC_ERROR*/
/*

```

```

/* Program logic: */
/* MQOPEN target queue for OUTPUT */
/* while end of input file not reached, */
/* . read next line of text */
/* . MQPUT datagram message with text line as data */
/* MQCLOSE target queue */
/* MQSTAT connection */
/* */
/* */
/*****
/* AMQSAPT0 has the following parameters */
/* required: */
/* (1) The name of the target queue */
/* optional: */
/* (2) Queue manager name */
/* (3) The open options */
/* (4) The close options */
/* (5) The name of the target queue manager */
/* (6) The name of the dynamic queue */
/* */
/*****
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
/* includes for MQI */
#include <cmqc.h>

int main(int argc, char **argv)
{
/* Declare file and character for sample input */
FILE *fp;

/* Declare MQI structures needed */
MQOD od = {MQOD_DEFAULT}; /* Object Descriptor */
MQMD md = {MQMD_DEFAULT}; /* Message Descriptor */
MQPMO pmo = {MQPMO_DEFAULT}; /* put message options */
MQSTS sts = {MQSTS_DEFAULT}; /* status information */
/** note, sample uses defaults where it can */
MQHCONN Hcon; /* connection handle */
MQHOBJ Hobj; /* object handle */
MQLONG Q_options; /* MQOPEN options */
MQLONG C_options; /* MQCLOSE options */
MQLONG CompCode; /* completion code */
MQLONG OpenCode; /* MQOPEN completion code */
MQLONG Reason; /* reason code */
MQLONG CReason; /* reason code for MQCONN */
MQLONG messlen; /* message length */
char buffer[100]; /* message buffer */
char QMName[50]; /* queue manager name */

printf("Sample AMQSAPT0 start\n");
if (argc < 2)
{
printf("Required parameter missing - queue name\n");
exit(99);
}

/*****
/* Connect to queue manager */
/*****
QMName[0] = 0; /* default */
if (argc > 2)
strcpy(QMName, argv[2]);
MQCONN(QMName, /* queue manager */
&Hcon, /* connection handle */
&Compcode, /* completion code */
&Reason); /* reason code */
/* report reason and stop if it failed */
if (CompCode == MQCC_FAILED)
{
printf("MQCONN ended with reason code %d\n", CReason);
exit( (int)CReason );
}

/*****
/* Use parameter as the name of the target queue */
/*****

```



```

strncpy(od.ObjectName, argv[1], (size_t)MQ_Q_NAME_LENGTH);
printf("target queue is %s\n", od.ObjectName);

if (argc > 5)
{
    strncpy(od.ObjectQMgrName, argv[5], (size_t) MQ_Q_MGR_NAME_LENGTH);
    printf("target queue manager is %s\n", od.ObjectQMgrName);
}

if (argc > 6)
{
    strncpy(od.DynamicQName, argv[6], (size_t) MQ_Q_NAME_LENGTH);
    printf("dynamic queue name is %s\n", od.DynamicQName);
}

/*****
/*
/*   Open the target message queue for output
/*
/*
/*****
if (argc > 3)
{
    O_options = atoi( argv[3] );
    printf("open options are %d\n", O_options);
}
else
{
    O_options = MQOO_OUTPUT          /* open queue for output
    | MQOO_FAIL_IF QUIESCING        /* but not if MQM stopping
    ;                               /* = 0x2010 = 8208 decimal
}

MQOPEN(Hcon,                      /* connection handle
        &od,                      /* object descriptor for queue
        O_options,                /* open options
        &Hobj,                   /* object handle
        &OpenCode,               /* MQOPEN completion code
        &Reason);                /* reason code

/* report reason, if any; stop if failed
if (Reason != MQRC_NONE)
{
    printf("MQOPEN ended with reason code %d\n", Reason);
}

if (OpenCode == MQCC_FAILED)
{
    printf("unable to open queue for output\n");
}

/*****
/*
/*   Read lines from the file and put them to the message queue
/*
/*   Loop until null line or end of file, or there is a failure
/*
/*
/*****
CompCode = OpenCode;          /* use MQOPEN result for initial test
fp = stdin;

memcpy(md.Format,             /* character string format
        MQFMT_STRING, (size_t)MQ_FORMAT_LENGTH);

/*****
/* These options specify that put operation should occur
/*
/* asynchronously and the application will check the success
/*
/* using MQSTAT at a later time.
/*
/*****
md.Persistence = MQPER_NOT_PERSISTENT;
pmo.Options |= MQPMO_ASYNC_RESPONSE;

/*****
/* These options cause the MsgId and CorrelId to be replaced, so
/*
/* that there is no need to reset them before each MQPUT
/*
/*****
pmo.Options |= MQPMO_NEW_MSG_ID;
pmo.Options |= MQPMO_NEW_CORREL_ID;

while (CompCode != MQCC_FAILED)
{
    if (fgets(buffer, sizeof(buffer), fp) != NULL)
    {
        messlen = (MQLONG)strlen(buffer); /* length without null

```

```

    if (buffer[messlen-1] == '\n') /* last char is a new-line */
    {
        buffer[messlen-1] = '\0'; /* replace new-line with null */
        --messlen; /* reduce buffer length */
    }
else messlen = 0; /* treat EOF same as null line */

/*****
/*
/* Put each buffer to the message queue */
/*
/*****
if (messlen > 0)
{
    MQPUT(Hcon, /* connection handle */
        Hobj, /* object handle */
        &md, /* message descriptor */
        &pmo, /* default options (datagram) */
        messlen, /* message length */
        buffer, /* message buffer */
        &CompCode, /* completion code */
        &Reason); /* reason code */

    /* report reason, if any */
    if (Reason != MQRC_NONE)
    {
        printf("MQPUT ended with reason code %d\n", Reason);
    }
}
else /* satisfy end condition when empty line is read */
    CompCode = MQCC_FAILED;
}

/*****
/*
/* Close the target queue (if it was opened) */
/*
/*****
if (OpenCode != MQCC_FAILED)
{
    if (argc > 4)
    {
        C_options = atoi( argv[4] );
        printf("close options are %d\n", C_options);
    }
    else
    {
        C_options = MQCO_NONE; /* no close options */
    }

    MQCLOSE(Hcon, /* connection handle */
        &Hobj, /* object handle */
        C_options, /* completion code */
        &CompCode, /* completion code */
        &Reason); /* reason code */

    /* report reason, if any */
    if (Reason != MQRC_NONE)
    {
        printf("MQCLOSE ended with reason code %d\n", Reason);
    }
}

/*****
/*
/* Query how many asynchronous puts succeeded */
/*
/*****
MQSTAT(&Hcon, /* connection handle */
    MQSTAT_TYPE_ASYNC_ERROR, /* status type */
    &Sts, /* MQSTS structure */
    &CompCode, /* completion code */
    &Reason); /* reason code */

/* report reason, if any */
if (Reason != MQRC_NONE)
{
    printf("MQSTAT ended with reason code %d\n", Reason);
}
else
{

```

```

/* Display results */
printf("Succeeded putting %d messages\n",
      sts.PutSuccessCount);
printf("%d messages were put with a warning\n",
      sts.PutWarningCount);
printf("Failed to put %d messages\n",
      sts.PutFailureCount);

if(sts.CompCode == MQCC_WARNING)
{
  printf("The first warning that occurred had reason code %d\n",
        sts.Reason);
}
else if(sts.CompCode == MQCC_FAILED)
{
  printf("The first error that occurred had reason code %d\n",
        sts.Reason);
}
}

/*****
/*
/* Disconnect from MQM if not already connected
/*
/*
/*****
if (CReason != MQRC_ALREADY_CONNECTED)
{
  MQDISC(&Hcon,
        &CompCode,
        &Reason);
  /* connection handle
  /* completion code
  /* reason code

  /* report reason, if any
  if (Reason != MQRC_NONE)
  {
    printf("MQDISC ended with reason code %d\n", Reason);
  }
}

/*****
/*
/* END OF AMQSAPTO
/*
/*
/*****
printf("Sample AMQSAPTO end\n");
return(0);
}

```

## COBOL 예제

토픽의 이 컬렉션은 IBM MQ for z/OS 샘플 애플리케이션에서 가져옵니다. 이는 별도의 기술이 없으면 모든 플랫폼에 적용 가능합니다.

### 큐 관리자에 연결

이 예에서는 MQCONN 호출을 사용하여 프로그램을 z/OS 일괄처리의 큐 관리자에 연결하는 방법을 보여줍니다.

이 추출은 IBM MQ for z/OS에서 제공되는 찾아보기 샘플 애플리케이션(프로그램 CSQ4BVA1)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.

```

* -----*
WORKING-STORAGE SECTION.
* -----*
* W02 - Data fields derived from the PARM field
01 W02-MQM PIC X(48) VALUE SPACES.
* W03 - MQM API fields
01 W03-HCONN PIC S9(9) BINARY.
01 W03-COMPCODE PIC S9(9) BINARY.
01 W03-REASON PIC S9(9) BINARY.
*
* MQV contains constants (for filling in the control
* blocks)
* and return codes (for testing the result of a call)
*
01 W05-MQM-CONSTANTS.
COPY CMQV SUPPRESS.

```

```

:
*
* Separate into the relevant fields any data passed
* in the PARM statement
*
UNSTRING PARM-STRING DELIMITED BY ALL ','
          INTO W02-MQM
          W02-OBJECT.
:
*
* Connect to the specified queue manager.
*
CALL 'MQCONN' USING W02-MQM
                  W03-HCONN
                  W03-COMPCODE
                  W03-REASON.
*
* Test the output of the connect call. If the call
* fails, print an error message showing the
* completion code and reason code.
*
IF (W03-COMPCODE NOT = MQCC-OK) THEN
:
END-IF.
:

```

### 큐 관리자에서 연결 끊기

이 예에서는 MQDISC 호출을 사용하여 z/OS 일괄처리의 큐 관리자에서 프로그램의 연결을 끊는 방법을 보여줍니다.

이 코드 추출에서 사용된 변수는 23 페이지의 『큐 관리자에 연결』에서 설정된 변수입니다. 이 추출은 IBM MQ for z/OS에서 제공되는 찾아보기 샘플 애플리케이션(프로그램 CSQ4BVA1)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.

```

:
*
* Disconnect from the queue manager
*
CALL 'MQDISC' USING W03-HCONN
                  W03-COMPCODE
                  W03-REASON.
*
* Test the output of the disconnect call. If the
* call fails, print an error message showing the
* completion code and reason code.
*
IF (W03-COMPCODE NOT = MQCC-OK) THEN
:
END-IF.
:

```

### 동적 큐 작성

이 예에서는 동적 큐를 작성하기 위해 MQOPEN 호출을 사용하는 방법을 설명합니다.

이 추출은 IBM MQ for z/OS에서 제공되는 신용 검사 샘플 애플리케이션(프로그램 CSQ4CVB1)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.

```

:
* -----*
WORKING-STORAGE SECTION.
* -----*
*
* W02 - Queues processed in this program
*
01 W02-MODEL-QNAME PIC X(48) VALUE
   'CSQ4SAMP.B1.MODEL'
01 W02-NAME-PREFIX PIC X(48) VALUE
   'CSQ4SAMP.B1.*'
01 W02-TEMPORARY-Q PIC X(48).
*
* W03 - MQM API fields
*
01 W03-HCONN PIC S9(9) BINARY VALUE ZERO.

```

```

01 W03-OPTIONS      PIC S9(9) BINARY.
01 W03-HOBJ        PIC S9(9) BINARY.
01 W03-COMPCODE    PIC S9(9) BINARY.
01 W03-REASON      PIC S9(9) BINARY.
*
*   API control blocks
*
01 MQM-OBJECT-DESCRIPTOR.
   COPY CMQODV.
*
*   CMQV contains constants (for setting or testing
*   field values) and return codes (for testing the
*   result of a call)
*
01 MQM-CONSTANTS.
   COPY CMQV SUPPRESS.
* -----*
PROCEDURE DIVISION.
* -----*
:
* -----*
OPEN-TEMP-RESPONSE-QUEUE SECTION.
* -----*

```

```

*
*   This section creates a temporary dynamic queue
*   using a model queue
*
* -----*
*
*   Change three fields in the Object Descriptor (MQOD)
*   control block. (MQODV initializes the other fields)
*
   MOVE MQOT-Q          TO MQOD-OBJECTTYPE.
   MOVE W02-MODEL-QNAME TO MQOD-OBJECTNAME.
   MOVE W02-NAME-PREFIX TO MQOD-DYNAMICQNAME.
*
   COMPUTE W03-OPTIONS = MQOD-INPUT-EXCLUSIVE.
*
   CALL 'MQOPEN' USING W03-HCONN
                      MQOD
                      W03-OPTIONS
                      W03-HOBJ-MODEL
                      W03-COMPCODE
                      W03-REASON.
*
   IF W03-COMPCODE NOT = MQCC-OK
       MOVE 'MQOPEN'      TO M01-MSG4-OPERATION
       MOVE W03-COMPCODE TO M01-MSG4-COMPCODE
       MOVE W03-REASON   TO M01-MSG4-REASON
       MOVE M01-MESSAGE-4 TO M00-MESSAGE
   ELSE
       MOVE MQOD-OBJECTNAME TO W02-TEMPORARY-Q
   END-IF.
*
OPEN-TEMP-RESPONSE-QUEUE-EXIT.
*
*   Return to performing section.
*
   EXIT.
   EJECT
*

```

## 기존 큐 열기

이 예에서는 MQOPEN 호출을 사용하여 기존 큐를 여는 방법을 설명합니다.

이 추출은 IBM MQ for z/OS에서 제공되는 찾아보기 샘플 애플리케이션(프로그램 CSQ4BVA1)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.

```

:
* -----*
WORKING-STORAGE SECTION.
* -----*
*

```

```

* W01 - Fields derived from the command area input
*
01 W01-OBJECT          PIC X(48).
*
* W02 - MQM API fields
*
01 W02-HCONN          PIC S9(9) BINARY VALUE ZERO.
01 W02-OPTIONS        PIC S9(9) BINARY.
01 W02-HOBJ           PIC S9(9) BINARY.
01 W02-COMPCODE        PIC S9(9) BINARY.
01 W02-REASON          PIC S9(9) BINARY.
*
* CMQODV defines the object descriptor (MQOD)
*
01 MQM-OBJECT-DESCRIPTOR.
   COPY CMQODV.
*
* CMQV contains constants (for setting or testing
* field values) and return codes (for testing the
* result of a call)
*
01 MQM-CONSTANTS.
   COPY CMQV SUPPRESS.
* -----*
E-OPEN-QUEUE SECTION.
* -----*
*
* This section opens the queue
*
* Initialize the Object Descriptor (MQOD) control
* block
* (The copy file initializes the remaining fields.)
*
MOVE MQOT-Q           TO MQOD-OBJECTTYPE.
MOVE W01-OBJECT        TO MQOD-OBJECTNAME.
*
* Initialize W02-OPTIONS to open the queue for both
* inquiring about and setting attributes
*
COMPUTE W02-OPTIONS = MQ00-INQUIRE + MQ00-SET.

```

```

*
* Open the queue
*
CALL 'MQOPEN' USING W02-HCONN
                   MQOD
                   W02-OPTIONS
                   W02-HOBJ
                   W02-COMPCODE
                   W02-REASON.
*
* Test the output from the open
*
* If the completion code is not OK, display a
* separate error message for each of the following
* errors:
*
* Q-MGR-NOT-AVAILABLE - MQM is not available
* CONNECTION-BROKEN   - MQM is no longer connected to CICS
* UNKNOWN-OBJECT-NAME - The queue does not exist
* NOT-AUTHORIZED      - The user is not authorized to open
*                       the queue
*
* For any other error, display an error message
* showing the completion and reason codes
*
IF W02-COMPCODE NOT = MQCC-OK
   EVALUATE TRUE
*
   WHEN W02-REASON = MQRC-Q-MGR-NOT-AVAILABLE
      MOVE M01-MESSAGE-6 TO M00-MESSAGE
*
   WHEN W02-REASON = MQRC-CONNECTION-BROKEN
      MOVE M01-MESSAGE-6 TO M00-MESSAGE
*
   WHEN W02-REASON = MQRC-UNKNOWN-OBJECT-NAME
      MOVE M01-MESSAGE-2 TO M00-MESSAGE
*
   WHEN W02-REASON = MQRC-NOT-AUTHORIZED
      MOVE M01-MESSAGE-3 TO M00-MESSAGE

```

```

*
      WHEN OTHER
          MOVE 'MOPEN'          TO M01-MSG4-OPERATION
          MOVE W02-COMPCODE    TO M01-MSG4-COMPCODE
          MOVE W02-REASON      TO M01-MSG4-REASON
          MOVE M01-MESSAGE-4   TO M00-MESSAGE
      END-EVALUATE
      END-IF.
E-EXIT.
*
*   Return to performing section
*
      EXIT.
      EJECT

```

## 큐 닫기

이 예에서는 MQCLOSE 호출을 사용하는 방법을 설명합니다.

이 코드 추출에서 사용된 변수는 23 페이지의 『[큐 관리자에 연결](#)』에서 설정된 변수입니다. 이 추출은 IBM MQ for z/OS에서 제공되는 찾아보기 샘플 애플리케이션(프로그램 CSQ4BVA1)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.

```

:
*
*   Close the queue
*
      MOVE MQCO-NONE TO W03-OPTIONS.
*
      CALL 'MQCLOSE' USING W03-HCONN
                          W03-HOBJ
                          W03-OPTIONS
                          W03-COMPCODE
                          W03-REASON.
*
*   Test the output of the MQCLOSE call.  If the call
*   fails, print an error message showing the
*   completion code and reason code.
*
      IF (W03-COMPCODE NOT = MQCC-OK) THEN
          MOVE 'CLOSE'          TO W04-MSG4-TYPE
          MOVE W03-COMPCODE    TO W04-MSG4-COMPCODE
          MOVE W03-REASON      TO W04-MSG4-REASON
          MOVE W04-MESSAGE-4   TO W00-PRINT-DATA
          PERFORM PRINT-LINE
          MOVE W06-CSQ4-ERROR TO W00-RETURN-CODE
      END-IF.
*

```

## MQPUT을 사용하여 메시지 넣기

이 예에서는 컨텍스트를 사용하여 MQPUT 호출을 사용하는 방법을 설명합니다.

이 추출은 IBM MQ for z/OS에서 제공되는 신용 검사 샘플 애플리케이션(프로그램 CSQ4CVB1)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.

```

:
* -----*
*   WORKING-STORAGE SECTION.
* -----*
*
*   W02 - Queues processed in this program
*
      01 W02-TEMPORARY-Q          PIC X(48).
*
*   W03 - MQM API fields
*
      01 W03-HCONN                PIC S9(9) BINARY VALUE ZERO.
      01 W03-HOBJ-INQUIRY        PIC S9(9) BINARY.
      01 W03-OPTIONS             PIC S9(9) BINARY.
      01 W03-BUFFLEN             PIC S9(9) BINARY.
      01 W03-COMPCODE            PIC S9(9) BINARY.
      01 W03-REASON              PIC S9(9) BINARY.
*

```

```

01 W03-PUT-BUFFER.
*
05 W03-CSQ4BIIM.
COPY CSQ4VB1.
*
* API control blocks
*
01 MQM-MESSAGE-DESCRIPTOR.
COPY CMQMDV.
01 MQM-PUT-MESSAGE-OPTIONS.
COPY CMQPMOV.
*
* MQV contains constants (for filling in the
* control blocks) and return codes (for testing
* the result of a call).
*
01 MQM-CONSTANTS.
COPY CMQV SUPPRESS.
* -----*
PROCEDURE DIVISION.
* -----*
:
* Open queue and build message.
:

```

```

*
* Set the message descriptor and put-message options to
* the values required to create the message.
* Set the length of the message.
*
MOVE MQMT-REQUEST TO MQMD-MSGTYPE.
MOVE MQCI-NONE TO MQMD-CORRELID.
MOVE MQMI-NONE TO MQMD-MSGID.
MOVE W02-TEMPORARY-Q TO MQMD-REPLYTOQ.
MOVE SPACES TO MQMD-REPLYTOQMGR.
MOVE 5 TO MQMD-PRIORITY.
MOVE MQPER-NOT-PERSISTENT TO MQMD-PERSISTENCE.
COMPUTE MQPMO-OPTIONS = MQPMO-NO-SYNCPOINT +
MQPMO-DEFAULT-CONTEXT.
MOVE LENGTH OF CSQ4BIIM-MSG TO W03-BUFFLEN.
*
CALL 'MQPUT' USING W03-HCONN
W03-HOBJ-INQUIRY
MQMD
MQPMO
W03-BUFFLEN
W03-PUT-BUFFER
W03-COMPCODE
W03-REASON.
IF W03-COMPCODE NOT = MQCC-OK
:
END-IF.

```

## MQPUT1을 사용하여 메시지 넣기

이 예에서는 MQPUT1 호출을 사용하는 방법을 설명합니다.

이 추출은 IBM MQ for z/OS에서 제공되는 신용 검사 샘플 애플리케이션(프로그램 CSQ4CVB5)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.

```

:
* -----*
WORKING-STORAGE SECTION.
* -----*
*
* W03 - MQM API fields
*
01 W03-HCONN PIC S9(9) BINARY VALUE ZERO.
01 W03-OPTIONS PIC S9(9) BINARY.
01 W03-COMPCODE PIC S9(9) BINARY.
01 W03-REASON PIC S9(9) BINARY.
01 W03-BUFFLEN PIC S9(9) BINARY.
*
01 W03-PUT-BUFFER.

```



```
05 W03-CSQ4BQRM.
COPY CSQ4VB4.
```

```
*
*   API control blocks
*
* 01 MQM-OBJECT-DESCRIPTOR.
*   COPY CMQODV.
* 01 MQM-MESSAGE-DESCRIPTOR.
*   COPY CMQMDV.
* 01 MQM-PUT-MESSAGE-OPTIONS.
*   COPY CMQPMOV.
*
* CMQV contains constants (for filling in the
* control blocks) and return codes (for testing
* the result of a call).
*
* 01 MQM-MQV.
*   COPY CMQV SUPPRESS.
* -----*
* PROCEDURE DIVISION.
* -----*
*
*   Get the request message.
*
* -----*
* PROCESS-QUERY SECTION.
* -----*
*
*   Build the reply message.
*
*
* Set the object descriptor, message descriptor and
* put-message options to the values required to create
* the message.
* Set the length of the message.
*
* MOVE MQMD-REPLYTOQ      TO MQOD-OBJECTNAME.
* MOVE MQMD-REPLYTOQMGR   TO MQOD-OBJECTQMGRNAME.
* MOVE MQMT-REPLY         TO MQMD-MSGTYPE.
* MOVE SPACES             TO MQMD-REPLYTOQ.
* MOVE SPACES             TO MQMD-REPLYTOQMGR.
* MOVE LOW-VALUES         TO MQMD-MSGID.
* COMPUTE MQPMO-OPTIONS = MQPMO-SYNCPOINT +
*                       MQPMO-PASS-IDENTITY-CONTEXT.
* MOVE W03-HOBJ-CHECKQ    TO MQPMO-CONTEXT.
* MOVE LENGTH OF CSQ4BQRM-MSG TO W03-BUFFLEN.
*
*   CALL 'MQPUT1' USING W03-HCONN
*                       MQOD
*                       MQMD
*                       MQPMO
*                       W03-BUFFLEN
*                       W03-PUT-BUFFER
*                       W03-COMPCODE
*                       W03-REASON.
*   IF W03-COMPCODE NOT = MQCC-OK
*     MOVE 'MQPUT1'       TO M02-OPERATION
*     MOVE MQOD-OBJECTNAME TO M02-OBJECTNAME
*     PERFORM RECORD-CALL-ERROR
*     PERFORM FORWARD-MSG-TO-DLQ
*   END-IF.
*
```

## 메시지 받기

이 예에서는 MQGET 호출을 사용하여 큐에서 메시지를 제거하는 방법을 설명합니다.

이 추출은 IBM MQ for z/OS에서 제공되는 신용 검사 샘플 애플리케이션(프로그램 CSQ4CVB1)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.

```

* -----*
* WORKING-STORAGE SECTION.
* -----*
```

```

*
*   W03 - MQM API fields
*
01  W03-HCONN          PIC S9(9) BINARY VALUE ZERO.
01  W03-HOBJ-RESPONSE PIC S9(9) BINARY.
01  W03-OPTIONS       PIC S9(9) BINARY.
01  W03-BUFFLEN       PIC S9(9) BINARY.
01  W03-DATALEN       PIC S9(9) BINARY.
01  W03-COMPCODE      PIC S9(9) BINARY.
01  W03-REASON        PIC S9(9) BINARY.
*
01  W03-GET-BUFFER.
    05 W03-CSQ4BAM.
    COPY CSQ4VB2.
*
*   API control blocks
*
01  MQM-MESSAGE-DESCRIPTOR.
    COPY CMQMDV.
01  MQM-GET-MESSAGE-OPTIONS.
    COPY CMQGMV.
*
*   MQV contains constants (for filling in the
*   control blocks) and return codes (for testing
*   the result of a call).
*
01  MQM-CONSTANTS.
    COPY CMQV SUPPRESS.
* -----*
A-MAIN SECTION.
* -----*
:
*   Open response queue.
:
* -----*
PROCESS-RESPONSE-SCREEN SECTION.
* -----*
*
*   This section gets a message from the response queue. *
*
*   When a correct response is received, it is *
*   transferred to the map for display; otherwise *
*   an error message is built. *
*
* -----*

```

```

*
*   Set get-message options
*
    COMPUTE MQGMO-OPTIONS = MQGMO-SYNCPOINT +
                          MQGMO-ACCEPT-TRUNCATED-MSG +
                          MQGMO-NO-WAIT.
*
*   Set msgid and correlid in MQMD to nulls so that any
*   message will qualify.
*   Set length to available buffer length.
*
    MOVE MQMI-NONE TO MQMD-MSGID.
    MOVE MQCI-NONE TO MQMD-CORRELID.
    MOVE LENGTH OF W03-GET-BUFFER TO W03-BUFFLEN.
*
    CALL 'MQGET' USING W03-HCONN
                      W03-HOBJ-RESPONSE
                      MQMD
                      MQGMO
                      W03-BUFFLEN
                      W03-GET-BUFFER
                      W03-DATALEN
                      W03-COMPCODE
                      W03-REASON.
    EVALUATE TRUE
      WHEN W03-COMPCODE NOT = MQCC-FAILED
      :
*     Process the message
      :
      WHEN (W03-COMPCODE = MQCC-FAILED AND
            W03-REASON = MQRC-NO-MSG-AVAILABLE)
        MOVE M01-MESSAGE-9 TO M00-MESSAGE
        PERFORM CLEAR-RESPONSE-SCREEN

```

```

*
      WHEN OTHER
        MOVE 'MQGET '      TO M01-MSG4-OPERATION
        MOVE W03-COMPCODE TO M01-MSG4-COMPCODE
        MOVE W03-REASON   TO M01-MSG4-REASON
        MOVE M01-MESSAGE-4 TO M00-MESSAGE
        PERFORM CLEAR-RESPONSE-SCREEN
      END-EVALUATE.

```

## 대기 옵션을 사용하여 메시지 받기

이 예에서는 대기 옵션과 함께 MQGET 호출을 사용하고 잘린 메시지를 수락하는 방법을 설명합니다.

이 추출은 IBM MQ for z/OS에서 제공되는 신용 검사 샘플 애플리케이션(프로그램 CSQ4CVB5)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.

```

:
* -----*
* WORKING-STORAGE SECTION.
* -----*
*
*   W00 - General work fields
*
*   01 W00-WAIT-INTERVAL   PIC S9(09) BINARY VALUE 30000.
*
*   W03 - MQM API fields
*
*   01 W03-HCONN          PIC S9(9) BINARY VALUE ZERO.
*   01 W03-OPTIONS       PIC S9(9) BINARY.
*   01 W03-HOBJ-CHECKQ   PIC S9(9) BINARY.
*   01 W03-COMPCODE      PIC S9(9) BINARY.
*   01 W03-REASON        PIC S9(9) BINARY.
*   01 W03-DATALEN       PIC S9(9) BINARY.
*   01 W03-BUFFLEN       PIC S9(9) BINARY.
*
*   01 W03-MSG-BUFFER.
*     05 W03-CSQ4BCAQ.
*     COPY CSQ4VB3.
*
*   API control blocks
*
*   01 MQM-MESSAGE-DESCRIPTOR.
*     COPY CMQMDV.
*   01 MQM-GET-MESSAGE-OPTIONS.
*     COPY CMQGMOV.
*
*   CMQV contains constants (for filling in the
*   control blocks) and return codes (for testing
*   the result of a call).
*
*   01 MQM-MQV.
*     COPY CMQV SUPPRESS.
* -----*
* PROCEDURE DIVISION.
* -----*
:
*   Open input queue.
:

```

```

*
*   Get and process messages.
*
*   COMPUTE MQGMO-OPTIONS = MQGMO-WAIT +
*                           MQGMO-ACCEPT-TRUNCATED-MSG +
*                           MQGMO-SYNCPPOINT.
*   MOVE LENGTH OF W03-MSG-BUFFER TO W03-BUFFLEN.
*   MOVE W00-WAIT-INTERVAL TO MQGMO-WAITINTERVAL.
*   MOVE MQMI-NONE TO MQMD-MSGID.
*   MOVE MQCI-NONE TO MQMD-CORRELID.
*
*   Make the first MQGET call outside the loop.
*
*   CALL 'MQGET' USING W03-HCONN
*                     W03-HOBJ-CHECKQ
*                     MQMD

```

```

MQGMO
W03-BUFFLEN
W03-MSG-BUFFER
W03-DATALEN
W03-COMPCODE
W03-REASON.

*
* Test the output of the MQGET call using the
* PERFORM loop that follows.
*
* Perform whilst no failure occurs
* - process this message
* - reset the call parameters
* - get another message
* End-perform
*
*
* Test the output of the MQGET call. If the call
* fails, send an error message showing the
* completion code and reason code, unless the
* completion code is NO-MSG-AVAILABLE.
*
IF (W03-COMPCODE NOT = MQCC-FAILED) OR
(W03-REASON NOT = MQRC-NO-MSG-AVAILABLE)
MOVE 'MQGET ' TO M02-OPERATION
MOVE MQ0D-OBJECTNAME TO M02-OBJECTNAME
PERFORM RECORD-CALL-ERROR
END-IF.
:

```

## 신호보내기를 사용하여 메시지 받기

이 예제는 신호보내기에서 MQGET 호출을 사용하는 방법을 보여줍니다. 이 추출은 IBM MQ for z/OS에서 제공되는 신용 검사 샘플 애플리케이션(프로그램 CSQ4CVB2)에서 가져옵니다.

신호보내기는 *IBM MQ for z/OS*에서만 사용 가능합니다.

```

:
* -----*
WORKING-STORAGE SECTION.
* -----*
*
* W00 - General work fields
:
01 W00-WAIT-INTERVAL PIC S9(09) BINARY VALUE 30000.
*
* W03 - MQM API fields
*
01 W03-HCONN PIC S9(9) BINARY VALUE ZERO.
01 W03-HOBJ-REPLYQ PIC S9(9) BINARY.
01 W03-COMPCODE PIC S9(9) BINARY.
01 W03-REASON PIC S9(9) BINARY.
01 W03-DATALEN PIC S9(9) BINARY.
01 W03-BUFFLEN PIC S9(9) BINARY.
:
01 W03-GET-BUFFER.
05 W03-CSQ4BQRM.
COPY CSQ4VB4.
*
05 W03-CSQ4BIIM REDEFINES W03-CSQ4BQRM.
COPY CSQ4VB1.
*
05 W03-CSQ4BPGM REDEFINES W03-CSQ4BIIM.
COPY CSQ4VB5.
:
* API control blocks
*
01 MQM-MESSAGE-DESCRIPTOR.
COPY CMQMDV.
01 MQM-GET-MESSAGE-OPTIONS.
COPY CMQGMV.
:
* MQV contains constants (for filling in the
* control blocks) and return codes (for testing
* the result of a call).

```

```

*
01 MQM-MQV.
COPY CMQV SUPPRESS.
* -----*
LINKAGE SECTION.
* -----*
01 L01-ECB-ADDR-LIST.
05 L01-ECB-ADDR1          POINTER.
05 L01-ECB-ADDR2          POINTER.

*
01 L02-ECBS.
05 L02-INQUIRY-ECB1      PIC S9(09) BINARY.
05 L02-REPLY-ECB2       PIC S9(09) BINARY.
01 REDEFINES L02-ECBS.
05                       PIC X(02).
05 L02-INQUIRY-ECB1-CC  PIC S9(04) BINARY.
05                       PIC X(02).
05 L02-REPLY-ECB2-CC   PIC S9(04) BINARY.

*
* -----*
PROCEDURE DIVISION.
* -----*
:
* Initialize variables, open queues, set signal on
* inquiry queue.
:
* -----*
PROCESS-SIGNAL-ACCEPTED SECTION.
* -----*
* This section gets a message with signal. If a      *
* message is received, process it. If the signal    *
* is set or is already set, the program goes into   *
* an operating system wait.                          *
* Otherwise an error is reported and call error set. *
* -----*
PERFORM REPLYQ-GETSIGNAL.
*
EVALUATE TRUE
  WHEN (W03-COMPCODE = MQCC-OK AND
        W03-REASON = MQRC-NONE)
    PERFORM PROCESS-REPLYQ-MESSAGE
*
  WHEN (W03-COMPCODE = MQCC-WARNING AND
        W03-REASON = MQRC-SIGNAL-REQUEST-ACCEPTED)
    OR
    (W03-COMPCODE = MQCC-FAILED AND
     W03-REASON = MQRC-SIGNAL-OUTSTANDING)
    PERFORM EXTERNAL-WAIT
*
  WHEN OTHER
    MOVE 'MQGET SIGNAL' TO M02-OPERATION
    MOVE MQ0D-OBJECTNAME TO M02-OBJECTNAME
    PERFORM RECORD-CALL-ERROR
    MOVE W06-CALL-ERROR TO W06-CALL-STATUS
END-EVALUATE.
*
PROCESS-SIGNAL-ACCEPTED-EXIT.
* Return to performing section
EXIT.
EJECT
*

* -----*
EXTERNAL-WAIT SECTION.
* -----*
* This section performs an external CICS wait on two *
* ECBs until at least one is posted. It then calls  *
* the sections to handle the posted ECB.            *
* -----*
EXEC CICS WAIT EXTERNAL
      ECBLIST(W04-ECB-ADDR-LIST-PTR)
      NUMEVENTS(2)
END-EXEC.
*
* At least one ECB must have been posted to get to this

```

```

* point. Test which ECB has been posted and perform
* the appropriate section.
*
  IF L02-INQUIRY-ECB1 NOT = 0
    PERFORM TEST-INQUIRYQ-ECB
  ELSE
    PERFORM TEST-REPLYQ-ECB
  END-IF.
*
EXTERNAL-WAIT-EXIT.
*
  Return to performing section.
*
  EXIT.
  EJECT
  :
* -----*
REPLYQ-GETSIGNAL SECTION.
* -----*
*
* This section performs an MQGET call (in syncpoint with *
* signal) on the reply queue. The signal field in the *
* MQGMO is set to the address of the ECB. *
* Response handling is done by the performing section. *
* -----*
*
  COMPUTE MQGMO-OPTIONS          = MQGMO-SYNCPOINT +
                                MQGMO-SET-SIGNAL.
  MOVE W00-WAIT-INTERVAL        TO MQGMO-WAITINTERVAL.
  MOVE LENGTH OF W03-GET-BUFFER TO W03-BUFFLEN.
*
  MOVE ZEROS                    TO L02-REPLY-ECB2.
  SET MQGMO-SIGNAL1 TO ADDRESS OF L02-REPLY-ECB2.

```

```

*
* Set msgid and correlid to nulls so that any message
* will qualify.
*
  MOVE MQMI-NONE TO MQMD-MSGID.
  MOVE MQCI-NONE TO MQMD-CORRELID.
*
  CALL 'MQGET' USING W03-HCONN
                    W03-HOBJ-REPLYQ
                    MQMD
                    MQGMO
                    W03-BUFFLEN
                    W03-GET-BUFFER
                    W03-DATALEN
                    W03-COMPCODE
                    W03-REASON.
*
REPLYQ-GETSIGNAL-EXIT.
*
  Return to performing section.
*
  EXIT.
  EJECT
*
  :

```

## 오브젝트의 속성 조회

이 예에서는 MQINQ 호출을 사용하여 큐의 속성을 조회하는 방법을 설명합니다.

이 추출은 IBM MQ for z/OS에서 제공되는 큐 속성 샘플 애플리케이션(프로그램 CSQ4CVC1)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 [샘플 프로시저 프로그램\(z/OS를 제외한 플랫폼\)](#)을 참조하십시오.

```

:
* -----*
WORKING-STORAGE SECTION.
* -----*
*
* W02 - MQM API fields
*

```



이 추출은 IBM MQ for z/OS에서 제공되는 큐 속성 샘플 애플리케이션(프로그램 CSQ4CVC1)에서 가져옵니다. 다른 플랫폼에 있는 샘플 애플리케이션의 이름과 위치는 샘플 프로시저 프로그램(z/OS 이외의 플랫폼)을 참조하십시오.

```

:
* -----*
WORKING-STORAGE SECTION.
* -----*
*
*   W02 - MQM API fields
*
01 W02-SELECTORCOUNT    PIC S9(9) BINARY VALUE 2.
01 W02-INTATTRCOUNT    PIC S9(9) BINARY VALUE 2.
01 W02-CHARATTRLENGTH   PIC S9(9) BINARY VALUE ZERO.
01 W02-CHARATTRS        PIC X      VALUE LOW-VALUES.
01 W02-HCONN             PIC S9(9) BINARY VALUE ZERO.
01 W02-HOBJ              PIC S9(9) BINARY.
01 W02-COMPCODE          PIC S9(9) BINARY.
01 W02-REASON            PIC S9(9) BINARY.
01 W02-SELECTORS-TABLE.
   05 W02-SELECTORS      PIC S9(9) BINARY OCCURS 2 TIMES.
01 W02-INTATTRS-TABLE.
   05 W02-INTATTRS      PIC S9(9) BINARY OCCURS 2 TIMES.
*
*   CMQODV defines the object descriptor (MQOD).
*
01 MQM-OBJECT-DESCRIPTOR.
   COPY CMQODV.
*
*   CMQV contains constants (for setting or testing
*   field values) and return codes (for testing the
*   result of a call).
*
01 MQM-CONSTANTS.
   COPY CMQV SUPPRESS.
* -----*
PROCEDURE DIVISION.
* -----*

```

```

*
*   Get the queue name and open the queue.
*
:
*
*
* Initialize the variables required for the set call:
* - Set W02-SELECTORS-TABLE to the attributes to be set
* - Set W02-INTATTRS-TABLE to the required status
* - All other variables are already set
*
   MOVE MQIA-INHIBIT-GET    TO W02-SELECTORS(1).
   MOVE MQIA-INHIBIT-PUT    TO W02-SELECTORS(2).
   MOVE MQQA-GET-INHIBITED TO W02-INTATTRS(1).
   MOVE MQQA-PUT-INHIBITED TO W02-INTATTRS(2).
*
*   Set the attributes.
*
   CALL 'MQSET' USING W02-HCONN,
                     W02-HOBJ,
                     W02-SELECTORCOUNT,
                     W02-SELECTORS-TABLE,
                     W02-INTATTRCOUNT,
                     W02-INTATTRS-TABLE,
                     W02-CHARATTRLENGTH,
                     W02-CHARATTRS,
                     W02-COMPCODE,
                     W02-REASON.
*
* Test the output from the call:
*
* - If the completion code is not OK, display an error
*   message showing the completion and reason codes
*
* - Otherwise, move 'INHIBITED' into the relevant
*   screen map fields
*
   IF W02-COMPCODE NOT = MQCC-OK

```



```

MOVE 'MQSET'      TO M01-MSG4-OPERATION
MOVE W02-COMPCODE TO M01-MSG4-COMPCODE
MOVE W02-REASON   TO M01-MSG4-REASON
MOVE M01-MESSAGE-4 TO M00-MESSAGE
ELSE
*
*   Process the changes.
*
*
*
END-IF.

```

## ▶ z/OS System/390 어셈블러 언어 예제

이 토픽 컬렉션은 주로 IBM MQ for z/OS 샘플 애플리케이션에서 가져옵니다.

### ▶ z/OS 큐 관리자에 연결

이 예에서는 MQCONN 호출을 사용하여 프로그램을 z/OS 일괄처리의 큐 관리자에 연결하는 방법을 보여줍니다.

이 추출은 IBM MQ for z/OS에서 제공되는 찾아보기 샘플 프로그램(CSQ4BAA1)에서 가져옵니다.

```

:
WORKAREA DSECT
*
PARMLIST CALL ,(0,0,0,0,0,0,0,0,0,0,0),VL,MF=L
*
COMPCODE DS F      Completion code
REASON   DS F      Reason code
HCONN   DS F      Connection handle
          ORG
PARMADDR DS F      Address of parm field
PARMLEN DS H      Length of parm field
*
MQMNAME  DS CL48   Queue manager name
*
*
*****
* SECTION NAME : MAINPARM *
*****
MAINPARM DS 0H
          MVI  MQMNAME,X'40'
          MVC  MQMNAME+1(L'MQMNAME-1),MQMNAME
*
* Space out first byte and initialize
*
* Code to address and verify parameters passed omitted
*
*
PARM1MVE DS 0H
          SR   R1,R3      Length of data
          LA  R4,MQMNAME  Address for target
          BCTR R1,R0      Reduce for execute
          EX  R1,MOVEPARM Move the data
*
*****
* EXECUTES *
*****
MOVEPARM MVC 0(*-*,R4),0(R3)
*
          EJECT

```

```

*****
* SECTION NAME : MAINCONN *
*****
*
*
MAINCONN DS 0H
          XC   HCONN,HCONN Null connection handle
*
          CALL MQCONN, X
              (MQMNAME, X
              HCONN, X
              COMPCODE, X
              REASON), X
              MF=(E,PARMLIST),VL

```



```

MVC WOD_OBJECTNAME,MOD_Q COPY IN THE MODEL Q NAME
MVC WOD_DYNAMICQNAME,DYN_Q COPY IN THE DYNAMIC Q NAME
L R5,=AL4(MQOO_OUTPUT) OPEN FOR OUTPUT AND
A R5,=AL4(MQOO_INQUIRE) INQUIRE
ST R5,OPTIONS

```

```

*
* ISSUE MQI OPEN REQUEST USING REENTRANT
* FORM OF CALL MACRO
*
CALL MQOPEN, X
(HCONN, X
WOD, X
OPTIONS, X
HOBJ, X
COMPCODE, X
REASON),VL,MF=(E,CALLST)
*
LA R5,MQCC_OK CHECK THE COMPLETION CODE
C R5,COMPCODE FROM THE REQUEST AND BRANCH
BNE BADCALL TO ERROR ROUTINE IF NOT MQCC_OK
*
MVC TEMP_Q,WOD_OBJECTNAME SAVE NAME OF TEMPORARY Q
CREATED BY OPEN OF MODEL Q
*
*
*
BADCALL DS 0H
*
*
*
CONSTANTS:
*
MOD_Q DC CL48'QUERY.REPLY.MODEL' MODEL QUEUE NAME
DYN_Q DC CL48'QUERY.TEMPQ.*' DYNAMIC QUEUE NAME
*
CMQODA DSECT=NO,LIST=YES CONSTANT VERSION OF MQOD
MQA
*
*
WORKING STORAGE
*
DFHEISTG
HCONN DS F CONNECTION HANDLE
OPTIONS DS F OPEN OPTIONS
HOBJ DS F OBJECT HANDLE
COMPCODE DS F MQI COMPLETION CODE
REASON DS F MQI REASON CODE
TEMP_Q DS CL(MQ_Q_NAME_LENGTH) SAVED QNAME AFTER OPEN
*
WOD CMQODA DSECT=NO,LIST=YES WORKING VERSION OF MQOD
*
CALLST CALL ,(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),VL,MF=L LIST FORM
OF CALL
MACRO
*
*
*
END

```

## z/OS 기존 큐 열기

이 예에서는 MQOPEN 호출을 사용하여 이미 정의된 큐를 여는 방법을 설명합니다.

이는 두 옵션을 지정하는 방법을 표시합니다. 이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다.

```

*
*
* R5 = WORK REGISTER.
*
OPEN DS 0H
*
MVC WOD_AREA,MQOD_AREA INITIALIZE WORKING VERSION OF
MQOD WITH DEFAULTS
MVC WOD_OBJECTNAME,Q_NAME SPECIFY Q NAME TO OPEN
LA R5,MQOO_INPUT_EXCLUSIVE OPEN FOR MQGET CALLS
*
ST R5,OPTIONS
*

```



```

*          CONSTANTS
*
*          CMQA
*
*          WORKING STORAGE (REENTRANT)
*
WEG4      DSECT
*
CALLLST   CALL  ,(0,0,0,0,0,0,0,0,0,0,0,0),VL,MF=L
*
HCONN     DS    F
HOBJ      DS    F
OPTIONS   DS    F
COMPCODE  DS    F
REASON    DS    F
*
*
LEG4      EQU   *-WKEG4
END

```

## MQPUT을 사용하여 메시지 넣기

이 예에서는 MQPUT 호출을 사용하여 큐에 메시지를 넣는 방법을 설명합니다.

이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다.

```

:
*          CONNECT TO QUEUE MANAGER
*
CONN      DS    0H
:
*
*          OPEN A QUEUE
*
OPEN      DS    0H
:
*
*          R4,R5,R6,R7 = WORK REGISTER.
*
PUT       DS    0H
          LA    R4,MQMD           SET UP ADDRESSES AND
          LA    R5,MQMD_LENGTH    LENGTH FOR USE BY MVCL
          LA    R6,WMD           INSTRUCTION, AS MQMD IS
          LA    R7,WMD_LENGTH    OVER 256 BYES LONG.
          MVCL  R6,R4           INITIALIZE WORKING VERSION
*                               OF MESSAGE DESCRIPTOR
*
*          MVC  WPMO_AREA,MQPMO_AREA  INITIALIZE WORKING MQPMO
*
          LA    R5,BUFFER_LEN    RETRIEVE THE BUFFER LENGTH
          ST    R5,BUFFLEN      AND SAVE IT FOR MQM USE
*
          MVC  BUFFER,TEST_MSG   SET THE MESSAGE TO BE PUT
*
*          ISSUE MQI PUT REQUEST USING REENTRANT FORM
*          OF CALL MACRO
*
*          HCONN WAS SET BY PREVIOUS MQCONN REQUEST
*          HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST
*
          CALL  MQPUT,           X
                (HCONN,        X
                HOBJ,          X
                WMD,           X
                WPMO,          X
                BUFFLEN,       X
                BUFFER,        X
                COMPCODE,      X
                REASON),VL,MF=(E,CALLLST)
*
          LA    R5,MQCC_OK
          C     R5,COMPCODE
          BNE  BADCALL
*
          :
BADCALL   DS    0H
:

```

```

*
*   CONSTANTS
*
CMQMDA DSECT=NO,LIST=YES,PERSISTENCE=MQPER_PERSISTENT
CMQPMOA DSECT=NO,LIST=YES
CMQA
TEST_MSG DC CL80'THIS IS A TEST MESSAGE'
*
*   WORKING STORAGE DSECT
*
WORKSTG DSECT
*
COMPCODE DS F
REASON  DS F
BUFFLEN DS F
OPTIONS DS F
HCONN   DS F
HOBJ    DS F
*
BUFFER  DS CL80
BUFFER_LEN EQU *-BUFFER
*
WMD     CMQMDA DSECT=NO,LIST=NO
WPMO    CMQPMOA DSECT=NO,LIST=NO
*
CALLLST CALL , (0,0,0,0,0,0,0,0,0,0),VL,MF=L
*
:
:
END

```

## ▶ z/OS MQPUT1을 사용하여 메시지 넣기

이 예에서는 MQPUT1 호출을 사용하여 큐를 열고 큐에 단일 메시지를 넣은 다음, 큐를 닫는 방법을 보여줍니다.

이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다.

```

:
*
*   CONNECT TO QUEUE MANAGER
*
CONN    DS 0H
:
*
*   R4,R5,R6,R7 = WORK REGISTER.
*
PUT     DS 0H
*
MVC    WOD_AREA,MQOD_AREA      INITIALIZE WORKING VERSION OF
*                                MQOD WITH DEFAULTS
MVC    WOD_OBJECTNAME,Q_NAME  SPECIFY Q NAME FOR PUT1
*
LA     R4,MQMD                 SET UP ADDRESSES AND
LA     R5,MQMD_LENGTH          LENGTH FOR USE BY MVCL
LA     R6,WMD                  INSTRUCTION, AS MQMD IS
LA     R7,WMD_LENGTH           OVER 256 BYES LONG.
MVCL   R6,R4                   INITIALIZE WORKING VERSION
*                                OF MESSAGE DESCRIPTOR

*
MVC    WPMO_AREA,MQPMO_AREA    INITIALIZE WORKING MQPMO
*
LA     R5,BUFFER_LEN           RETRIEVE THE BUFFER LENGTH
ST     R5,BUFFLEN              AND SAVE IT FOR MQM USE
*
MVC    BUFFER,TEST_MSG        SET THE MESSAGE TO BE PUT
*
*   ISSUE MQI PUT REQUEST USING REENTRANT FORM OF CALL MACRO
*
*   HCONN WAS SET BY PREVIOUS MQCONN REQUEST
*   HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST
*
CALL   MQPUT1,                 X
      (HCONN,                  X
      LMQOD,                   X

```

```

                LMQMD,                X
                LMQPMO,               X
                BUFFERLENGTH,        X
                BUFFER,               X
                COMPCODE,            X
                REASON),VL,MF=(E,CALLST)
*
        LA R5,MQCC_OK
        C R5,COMPCODE
        BNE BADCALL
*
        :
BADCALL DS 0H
        :
*

```

```

*      CONSTANTS
*
CMQMDA DSECT=NO,LIST=YES,PERSISTENCE=MQPER_PERSISTENT
CMQPMOA DSECT=NO,LIST=YES
CMQODA DSECT=NO,LIST=YES
CMQA
*
TEST_MSG DC CL80'THIS IS ANOTHER TEST MESSAGE'
Q_NAME DC CL48'TEST.QUEUE.NAME'
*
*      WORKING STORAGE DSECT
*
WORKSTG DSECT
*
COMPCODE DS F
REASON DS F
BUFFLEN DS F
OPTIONS DS F
HCONN DS F
HOBJ DS F
*
BUFFER DS CL80
BUFFER_LEN EQU *-BUFFER
*
WOD CMQODA DSECT=NO,LIST=YES WORKING VERSION OF MQOD
WMD CMQMDA DSECT=NO,LIST=NO
WPMO CMQPMOA DSECT=NO,LIST=NO
*
CALLLST CALL ,(0,0,0,0,0,0,0,0,0,0,0),VL,MF=L
*
        :
        END

```

## 메시지 받기

이 예에서는 MQGET 호출을 사용하여 큐에서 메시지를 제거하는 방법을 설명합니다.

이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다.

```

:
*
*      CONNECT TO QUEUE MANAGER
*
CONN DS 0H
:
*
*      OPEN A QUEUE FOR GET
*
OPEN DS 0H
:
*
*      R4,R5,R6,R7 = WORK REGISTER.
*
GET DS 0H
LA R4,MQMD SET UP ADDRESSES AND
LA R5,MQMD_LENGTH LENGTH FOR USE BY MVCL
LA R6,WMD INSTRUCTION, AS MQMD IS
LA R7,WMD_LENGTH OVER 256 BYES LONG.
MVCL R6,R4 INITIALIZE WORKING VERSION
* OF MESSAGE DESCRIPTOR

```

```

*
*   MVC   WGMO_AREA,MQGMO_AREA   INITIALIZE WORKING MQGMO
*
*   LA    R5,BUFFER_LEN          RETRIEVE THE BUFFER LENGTH
*   ST    R5,BUFFLEN            AND SAVE IT FOR MQM USE
*
*
*   * ISSUE MQI GET REQUEST USING REENTRANT FORM OF CALL MACRO
*
*   HCONN WAS SET BY PREVIOUS MQCONN REQUEST
*   HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST
*
*   CALL  MQGET,                  X
*         (HCONN,                X
*          HOBJ,                  X
*          WMD,                    X
*          WGMO,                  X
*          BUFFLEN,              X
*          BUFFER,                X
*          DATALEN,             X
*          COMPCODE,             X
*          REASON),              X
*         VL,MF=(E,CALLLST)
*
*   LA    R5,MQCC_OK
*   C     R5,COMPCODE
*   BNE  BADCALL
*
*   :
*   BADCALL DS 0H
*   :

```

```

*
*   CONSTANTS
*
*   CMQMDA DSECT=NO,LIST=YES
*   CMQGMOA DSECT=NO,LIST=YES
*   CMQA
*
*   WORKING STORAGE DSECT
*
*   WORKSTG DSECT
*
*   COMPCODE DS F
*   REASON   DS F
*   BUFFLEN  DS F
*   DATALEN DS F
*   OPTIONS  DS F
*   HCONN    DS F
*   HOBJ     DS F
*
*   BUFFER   DS CL80
*   BUFFER_LEN EQU *-BUFFER
*
*   WMD      CMQMDA DSECT=NO,LIST=NO
*   WGMO     CMQGMOA DSECT=NO,LIST=NO
*
*   CALLLST  CALL ,(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),VL,MF=L
*
*   :
*   END

```

## 대기 옵션을 사용하여 메시지 받기

이 예에서는 MQGET 호출의 대기 옵션을 사용하는 방법을 설명합니다.

이 코드는 잘린 메시지를 허용합니다. 이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다.

```

:
*   CONNECT TO QUEUE MANAGER
*   CONN   DS 0H
*   :
*   OPEN A QUEUE FOR GET
*   OPEN  DS 0H

```



```

:
*      R4,R5,R6,R7 = WORK REGISTER.
GET  DS  0H
    LA  R4,MQMD                SET UP ADDRESSES AND
    LA  R5,MQMD_LENGTH        LENGTH FOR USE BY MVCL
    LA  R6,WMD                INSTRUCTION, AS MQMD IS
    LA  R7,WMD_LENGTH        OVER 256 BYES LONG.
    MVCL R6,R4                INITIALIZE WORKING VERSION
*                               OF MESSAGE DESCRIPTOR

*
MVC  WGMO_AREA,MQGMO_AREA    INITIALIZE WORKING MQGMO
L    R5,=AL4(MQGMO_WAIT)
A    R5,=AL4(MQGMO_ACCEPT_TRUNCATED_MSG)
ST   R5,WGMO_OPTIONS
MVC  WGMO_WAITINTERVAL,TWO_MINUTES  WAIT UP TO TWO
                                         MINUTES BEFORE
                                         FAILING THE
                                         CALL

*
LA   R5,BUFFER_LEN          RETRIEVE THE BUFFER LENGTH
ST   R5,BUFFLEN             AND SAVE IT FOR MQM USE

*
*  ISSUE MQI GET REQUEST USING REENTRANT FORM OF CALL MACRO
*
*      HCONN WAS SET BY PREVIOUS MQCONN REQUEST
*      HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST
*
CALL  MQGET,                X
      (HCONN,                X
      HOBJ,                  X
      WMD,                   X
      WGMO,                  X
      BUFFLEN,               X
      BUFFER,                X
      DATALEN,              X
      COMPCODE,              X
      REASON),               X
      VL,MF=(E,CALLLST)

*
LA   R5,MQCC_OK              DID THE MQGET REQUEST
C    R5,COMPCODE             WORK OK?
BE  GETOK                   YES, SO GO AND PROCESS.
LA   R5,MQCC_WARNING        NO, SO CHECK FOR A WARNING.
C    R5,COMPCODE             IS THIS A WARNING?
BE  CHECK_W                 YES, SO CHECK THE REASON.

*
LA   R5,MQRC_NO_MSG_AVAILABLE  IT MUST BE AN ERROR.
                                         IS IT DUE TO AN EMPTY
C    R5,REASON              QUEUE?
BE  NOMSG                  YES, SO HANDLE THE ERROR
B   BADCALL                NO, SO GO TO ERROR ROUTINE

*
CHECK_W DS  0H
      LA  R5,MQRC_TRUNCATED_MSG_ACCEPTED  IS THIS A
                                         TRUNCATED
C      R5,REASON                          MESSAGE?
BE  GETOK                                  YES, SO GO AND PROCESS.
B   BADCALL                                NO, SOME OTHER WARNING

*
NOMSG DS  0H
:
GETOK DS  0H
:

BADCALL DS  0H
:
*
*  CONSTANTS
*
      CMQMDA DSECT=NO,LIST=YES
      CMQGMOA DSECT=NO,LIST=YES
      CMQA

*
TWO_MINUTES DC F'120000'  GET WAIT INTERVAL
*
*  WORKING STORAGE DSECT

```

```

*
WORKSTG DSECT
*
COMPCODE DS F
REASON DS F
BUFFLEN DS F
DATALEN DS F
OPTIONS DS F
HCONN DS F
HOBJ DS F
*
BUFFER DS CL80
BUFFER_LEN EQU *-BUFFER
*
WMD CMQMDA DSECT=NO,LIST=NO
WGMO CMQGMOA DSECT=NO,LIST=NO
*
CALLLST CALL ,(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),VL,MF=L
*
:
END

```

## ▶ z/OS 신호보내기를 사용하여 메시지 받기

이 예는 MQGET 호출을 사용하여 적절한 메시지가 큐에 도착할 때 알림을 받도록 신호를 설정하는 방법을 보여줍니다.

이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다.

```

:
*
* CONNECT TO QUEUE MANAGER
*
CONN DS 0H
:
*
* OPEN A QUEUE FOR GET
*
OPEN DS 0H
:
*
* R4,R5,R6,R7 = WORK REGISTER.
*
GET DS 0H
LA R4,MQMD SET UP ADDRESSES AND
LA R5,MQMD_LENGTH LENGTH FOR USE BY MVCL
LA R6,WMD INSTRUCTION, AS MQMD IS
LA R7,WMD_LENGTH OVER 256 BYES LONG.
MVCL R6,R4 INITIALIZE WORKING VERSION
OF MESSAGE DESCRIPTOR
*

```

```

*
MVC WGMO_AREA,MQGMO_AREA INITIALIZE WORKING MQGMO
LA R5,MQGMO_SET_SIGNAL
ST R5,WGMO_OPTIONS
MVC WGMO_WAITINTERVAL,FIVE_MINUTES WAIT UP TO FIVE
MINUTES BEFORE
FAILING THE CALL
*
*
XC SIG_ECB,SIG_ECB CLEAR THE ECB
LA R5,SIG_ECB GET THE ADDRESS OF THE ECB
ST R5,WGMO_SIGNAL1 AND PUT IT IN THE WORKING
MQGMO
*
*
LA R5,BUFFER_LEN RETRIEVE THE BUFFER LENGTH
ST R5,BUFFLEN AND SAVE IT FOR MQM USE
*
*
* ISSUE MQI GET REQUEST USING REENTRANT FORM OF CALL MACRO
*
* HCONN WAS SET BY PREVIOUS MQCONN REQUEST
* HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST
*

```

```

CALL  MQGET,           X
      (HCONN,         X
       HOBJ,          X
       WMD,           X
       WGM0,          X
       BUFFLEN,       X
       BUFFER,        X
       DATALEN,      X
       COMPCODE,      X
       REASON),       X
      VL,MF=(E,CALLST)

```

```

*
LA  R5,MQCC_OK        DID THE MQGET REQUEST
C  R5,COMPCODE        WORK OK?
BE  GETOK             YES, SO GO AND PROCESS.
LA  R5,MQCC_WARNING   NO, SO CHECK FOR A WARNING.
C  R5,COMPCODE        IS THIS A WARNING?
BE  CHECK_W          YES, SO CHECK THE REASON.
B  BADCALL            NO, SO GO TO ERROR ROUTINE
*

```

```

CHECK_W DS 0H
LA  R5,MQRC_SIGNAL_REQUEST_ACCEPTED
C  R5,REASON          SIGNAL REQUEST SIGNAL SET?
BNE BADCALL          NO, SOME ERROR OCCURRED
B  DOWORK             YES, SO DO SOMETHING
                     ELSE
*

```

```

CHECKSIG DS 0H
CLC SIG_ECB+1(3),=AL3(MQEC_MSG_ARRIVED)
                     IS A MESSAGE AVAILABLE?
BE  GET              YES, SO GO AND GET IT
*

```

```

CLC SIG_ECB+1(3),=AL3(MQEC_WAIT_INTERVAL_EXPIRED)
                     HAVE WE WAITED LONG ENOUGH?
BE  NOMSG            YES, SO SAY NO MSG AVAILABLE
B  BADCALL            IF IT'S ANYTHING ELSE
                     GO TO ERROR ROUTINE.
*

```

```

DOWORK DS 0H
:
TM  SIG_ECB,X'40'    HAS THE SIGNAL ECB BEEN POSTED?
BO  CHECKSIG        YES, SO GO AND CHECK WHY
B  DOWORK            NO, SO GO AND DO MORE WORK
*

```

```

NOMSG DS 0H
:

```

```

GETOK DS 0H
:

```

```

BADCALL DS 0H
:

```

```

*
*   CONSTANTS
*

```

```

CMQMDA DSECT=NO,LIST=YES
CMQGMOA DSECT=NO,LIST=YES
CMQA

```

```

*   FIVE_MINUTES DC F'300000'          GET SIGNAL INTERVAL
*

```

```

*   WORKING STORAGE DSECT
*

```

```

WORKSTG DSECT
*

```

```

COMPCODE DS F
REASON   DS F
BUFFLEN  DS F
DATALEN  DS F
OPTIONS  DS F
HCONN    DS F
HOBJ     DS F
SIG_ECB  DS F

```

```

*
BUFFER   DS CL80
BUFFER_LEN EQU *-BUFFER
*

```



```

ST R0,INTATTRS+0      Place in field
LA R0,MQQA_PUT_ALLOWED Load attribute value
ST R0,INTATTRS+4      Place in field
B UPDCALL              Go and do call
*
UPDCALL DS 0H
CALL MQSET,           C
      (HCONN,         C
      HOBJ,           C
      SELECTORCOUNT, C
      SELECTOR,       C
      INTATTRCOUNT, C
      INTATTRS,      C
      CHARATTRLENGTH, C
      CHARATTRS,     C
      COMPCODE,       C
      REASON),        C
      VL,MF=(E,CALLLIST)
*
      LA R0,MQCC_OK   Load expected compcode
      C R0,COMPCODE   Was set successful?
      :
* SECTION NAME : INQUIRE *
* FUNCTION : Inquires on the objects attributes *
* CALLED BY : PROCESS *
* CALLS : OPEN, CLOSE, CODES *
* RETURN : To Register 6 *
INQUIRE DS 0H
      :

```

```

* Initialize the variables for the inquire call
*
SR R0,R0              Clear register zero
ST R0,CHARATTRLENGTH Set char length to zero
LA R0,2               Load to set
ST R0,SELECTORCOUNT selectors add
ST R0,INTATTRCOUNT  integer attributes
*
LA R0,MQIA_INHIBIT_GET Load attribute value
ST R0,SELECTOR+0      Place in field
LA R0,MQIA_INHIBIT_PUT Load attribute value
ST R0,SELECTOR+4      Place in field
CALL MQINQ,           C
      (HCONN,         C
      HOBJ,           C
      SELECTORCOUNT, C
      SELECTOR,       C
      INTATTRCOUNT, C
      INTATTRS,      C
      CHARATTRLENGTH, C
      CHARATTRS,     C
      COMPCODE,       C
      REASON),        C
      VL,MF=(E,CALLLIST)
LA R0,MQCC_OK         Load expected compcode
C R0,COMPCODE         Was inquire successful?
:

```

**z/OS PL/I 예제**

PL/I 사용은 z/OS에서만 지원합니다. 토픽의 이 컬렉션은 PL/I 예제를 사용하는 기법을 보여줍니다.

**z/OS 큐 관리자에 연결**

이 예에서는 MQCONN 호출을 사용하여 프로그램을 z/OS 일괄처리의 큐 관리자에 연결하는 방법을 보여줍니다. 이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다.

```

%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****
/* STRUCTURE BASED ON PARAMETER INPUT AREA (PARAM) */
*****/

```

```

DCL 1 INPUT_PARAM      BASED(ADDR(PARAM)),
  2 PARAM_LENGTH      FIXED BIN(15),
  2 PARAM_MQMNAME     CHAR(48);
:
/*****
/* WORKING STORAGE DECLARATIONS
*****/
DCL MQMNAME            CHAR(48);
DCL COMPCODE          BINARY FIXED (31);
DCL REASON            BINARY FIXED (31);
DCL HCONN            BINARY FIXED (31);
:
/*****
/* COPY QUEUE MANAGER NAME PARAMETER
/* TO LOCAL STORAGE
*****/
MQMNAME = ' ';
MQMNAME = SUBSTR(PARAM_MQMNAME,1,PARAM_LENGTH);
:
/*****
/* CONNECT FROM THE QUEUE MANAGER
*****/
CALL MQCONN (MQMNAME, /* MQM SYSTEM NAME
                    HCONN, /* CONNECTION HANDLE
                    COMPCODE, /* COMPLETION CODE
                    REASON); /* REASON CODE
:
/*****
/* TEST THE COMPLETION CODE OF THE CONNECT CALL.
/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE
/* SHOWING THE COMPLETION CODE AND THE REASON CODE.
*****/
IF COMPCODE /= MQCC_OK
  THEN DO;
  :
  CALL ERROR_ROUTINE;
END;

```

## z/OS 큐 관리자에서 연결 끊기

이 예에서는 MQDISC 호출을 사용하여 z/OS 일괄처리의 큐 관리자에서 프로그램의 연결을 끊는 방법을 보여줍니다.

이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다.

```

%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****
/* WORKING STORAGE DECLARATIONS
*****/
DCL COMPCODE          BINARY FIXED (31);
DCL REASON            BINARY FIXED (31);
DCL HCONN            BINARY FIXED (31);
:
/*****
/* DISCONNECT FROM THE QUEUE MANAGER
*****/
CALL MQDISC (HCONN, /* CONNECTION HANDLE
             COMPCODE, /* COMPLETION CODE
             REASON); /* REASON CODE
:
/*****
/* TEST THE COMPLETION CODE OF THE DISCONNECT CALL.
/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE
/* SHOWING THE COMPLETION CODE AND THE REASON CODE.
*****/
IF COMPCODE /= MQCC_OK
  THEN DO;
  :
  CALL ERROR_ROUTINE;
END;

```

## z/OS 동적 큐 작성

이 예에서는 동적 큐를 작성하기 위해 MQOPEN 호출을 사용하는 방법을 설명합니다.

이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다.

```
%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****
/* WORKING STORAGE DECLARATIONS */
*****/
DCL COMPCODE          BINARY FIXED (31);
DCL REASON            BINARY FIXED (31);
DCL HCONN             BINARY FIXED (31);
DCL HOBJ              BINARY FIXED (31);
DCL OPTIONS           BINARY FIXED (31);
:
DCL MODEL_QUEUE_NAME CHAR(48) INIT('PL1.REPLY.MODEL');
DCL DYNAMIC_NAME_PREFIX CHAR(48) INIT('PL1.TEMPQ.*');
DCL DYNAMIC_QUEUE_NAME CHAR(48) INIT(' ');
:
/*****
/* LOCAL COPY OF OBJECT DESCRIPTOR */
*****/
DCL 1 LMQOD LIKE MQOD;
:
/*****
/* SET UP OBJECT DESCRIPTOR FOR OPEN OF REPLY QUEUE */
*****/
LMQOD.OBJECTTYPE =MQOT_Q;
LMQOD.OBJECTNAME = MODEL_QUEUE_NAME;
LMQOD.DYNAMICQNAME = DYNAMIC_NAME_PREFIX;
OPTIONS = MQOO_INPUT_EXCLUSIVE;

CALL MQOPEN (HCONN,
             LMQOD,
             OPTIONS,
             HOBJ,
             COMPCODE,
             REASON);

/*****
/* TEST THE COMPLETION CODE OF THE OPEN CALL. */
/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE */
/* SHOWING THE COMPLETION CODE AND THE REASON CODE. */
/* IF THE CALL HAS SUCCEEDED THEN EXTRACT THE NAME OF */
/* THE NEWLY CREATED DYNAMIC QUEUE FROM THE OBJECT */
/* DESCRIPTOR. */
*****/
IF COMPCODE = MQCC_OK
  THEN DO;
  :
  CALL ERROR_ROUTINE;
END;
ELSE
  DYNAMIC_QUEUE_NAME = LMQOD_OBJECTNAME;
```

## 기존 큐 열기

이 예에서는 MQOPEN 호출을 사용하여 기존 큐를 여는 방법을 설명합니다.

이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다.

```
%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****
/* WORKING STORAGE DECLARATIONS */
*****/
DCL COMPCODE          BINARY FIXED (31);
DCL REASON            BINARY FIXED (31);
DCL HCONN             BINARY FIXED (31);
DCL HOBJ              BINARY FIXED (31);
DCL OPTIONS           BINARY FIXED (31);
:
DCL QUEUE_NAME        CHAR(48) INIT('PL1.LOCAL.QUEUE');
:
/*****
/* LOCAL COPY OF OBJECT DESCRIPTOR */
*****/
```

```

DCL 1 LMQOD LIKE MQOD;
:
/*****
/* SET UP OBJECT DESCRIPTOR FOR OPEN OF REPLY QUEUE */
/*****
LMQOD.OBJECTTYPE = MQOT_Q;
LMQOD.OBJECTNAME = QUEUE_NAME;
OPTIONS = MQOO_INPUT_EXCLUSIVE;

CALL MQOPEN (HCONN,
             LMQOD,
             OPTIONS,
             HOBJ,
             COMPCODE,
             REASON);

/*****
/* TEST THE COMPLETION CODE OF THE OPEN CALL. */
/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE */
/* SHOWING THE COMPLETION CODE AND THE REASON CODE. */
/*****
IF COMPCODE = MQCC_OK
THEN DO;
:
CALL ERROR_ROUTINE;
END;

```

## 큐 닫기

이 예에서는 MQCLOSE 호출을 사용하는 방법을 설명합니다.

이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다.

```

%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****
/* WORKING STORAGE DECLARATIONS */
/*****
DCL COMPCODE          BINARY FIXED (31);
DCL REASON            BINARY FIXED (31);
DCL HCONN             BINARY FIXED (31);
DCL HOBJ              BINARY FIXED (31);
DCL OPTIONS           BINARY FIXED (31);
:
/*****
/* SET CLOSE OPTIONS */
/*****
OPTIONS=MQCO_NONE;

/*****
/* CLOSE QUEUE */
/*****
CALL MQCLOSE (HCONN, /* CONNECTION HANDLE */
             HOBJ,   /* OBJECT HANDLE */
             OPTIONS, /* CLOSE OPTIONS */
             COMPCODE, /* COMPLETION CODE */
             REASON); /* REASON CODE */

/*****
/* TEST THE COMPLETION CODE OF THE CLOSE CALL. */
/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE */
/* SHOWING THE COMPLETION CODE AND THE REASON CODE. */
/*****
IF COMPCODE = MQCC_OK
THEN DO;
:
CALL ERROR_ROUTINE;
END;

```

## MQPUT을 사용하여 메시지 넣기

이 예에서는 컨텍스트를 사용하여 MQPUT 호출을 사용하는 방법을 설명합니다.



이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다.

```
%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****/
/* WORKING STORAGE DECLARATIONS */
/*****/
DCL COMPCODE          BINARY FIXED (31);
DCL REASON            BINARY FIXED (31);
DCL HCONN            BINARY FIXED (31);
DCL HOBJ            BINARY FIXED (31);
DCL OPTIONS          BINARY FIXED (31);
DCL BUFFLEN          BINARY FIXED (31);
DCL BUFFER            CHAR(80);
:
DCL PL1_TEST_MESSAGE  CHAR(80)
INIT('***** THIS IS A TEST MESSAGE *****');
:
/*****/
/* LOCAL COPY OF MESSAGE DESCRIPTOR */
/* AND PUT MESSAGE OPTIONS */
/*****/
DCL 1 LMQMD LIKE MQMD;
DCL 1 LMQPMO LIKE MQPMO;
:
/*****/
/* SET UP MESSAGE DESCRIPTOR */
/*****/
LMQMD.MSGTYPE = MQMT_DATAGRAM;
LMQMD.PRIORITY = 1;
LMQMD.PERSISTENCE = MQPER_PERSISTENT;
LMQMD.REPLYTOQ = ' ';
LMQMD.REPLYTOQMGR = ' ';
LMQMD.MSGID = MQMI_NONE;
LMQMD.CORRELID = MQCI_NONE;

/*****/
/* SET UP PUT MESSAGE OPTIONS */
/*****/
LMQPMO.OPTIONS = MQPMO_NO_SYNCPOINT;

/*****/
/* SET UP LENGTH OF MESSAGE BUFFER AND THE MESSAGE */
/*****/
BUFFLEN = LENGTH(BUFFER);
BUFFER = PL1_TEST_MESSAGE;
/*****/
/*
*/
/* HCONN WAS SET BY PREVIOUS MQCONN REQUEST. */
/* HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST. */
/*
*/
/*****/
CALL MQPUT (HCONN,
            HOBJ,
            LMQMD,
            LMQPMO,
            BUFFLEN,
            BUFFER,
            COMPCODE,
            REASON);

/*****/
/* TEST THE COMPLETION CODE OF THE PUT CALL. */
/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE */
/* SHOWING THE COMPLETION CODE AND THE REASON CODE. */
/*****/
IF COMPCODE = MQCC_OK
THEN DO;
:
CALL ERROR_ROUTINE;
END;
```

## ▶ z/OS MQPUT1을 사용하여 메시지 넣기

이 예에서는 MQPUT1 호출을 사용하는 방법을 설명합니다.

이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다.

```
%INCLUDE SYSLIB(CMQEPP);
%INCLUDE SYSLIB(CMQP);
:
/*****
/* WORKING STORAGE DECLARATIONS */
*****/
DCL COMPCODE      BINARY FIXED (31);
DCL REASON        BINARY FIXED (31);
DCL HCONN         BINARY FIXED (31);
DCL OPTIONS       BINARY FIXED (31);
DCL BUFFLEN       BINARY FIXED (31);
DCL BUFFER        CHAR(80);
:
DCL REPLY_TO_QUEUE CHAR(48) INIT('PL1.REPLY.QUEUE');
DCL QUEUE_NAME     CHAR(48) INIT('PL1.LOCAL.QUEUE');
DCL PL1_TEST_MESSAGE CHAR(80)
INIT('***** THIS IS ANOTHER TEST MESSAGE *****');
:
/*****
/* LOCAL COPY OF OBJECT DESCRIPTOR, MESSAGE DESCRIPTOR */
/* AND PUT MESSAGE OPTIONS */
*****/
DCL 1 LMQOD LIKE MQOD;
DCL 1 LMQMD LIKE MQMD;
DCL 1 LMQPMO LIKE MQPMO;
:
/*****
/* SET UP OBJECT DESCRIPTOR AS REQUIRED. */
*****/
LMQOD.OBJECTTYPE = MQOT_Q;
LMQOD.OBJECTNAME = QUEUE_NAME;

/*****
/* SET UP MESSAGE DESCRIPTOR AS REQUIRED. */
*****/
LMQMD.MSGTYPE = MQMT_REQUEST;
LMQMD.PRIORITY = 5;
LMQMD.PERSISTENCE = MQPER_PERSISTENT;
LMQMD.REPLYTOQ = REPLY_TO_QUEUE;
LMQMD.REPLYTOQMGR = 'T';
LMQMD.MSGID = MQMI_NONE;
LMQMD.CORRELID = MQCI_NONE;

/*****
/* SET UP PUT MESSAGE OPTIONS AS REQUIRED */
*****/
LMQPMO.OPTIONS = MQPMO_NO_SYNCPOINT;

/*****
/* SET UP LENGTH OF MESSAGE BUFFER AND THE MESSAGE */
*****/
BUFFLEN = LENGTH(BUFFER);
BUFFER = PL1_TEST_MESSAGE;

CALL MQPUT1 (HCONN,
LMQOD,
LMQMD,
LMQPMO,
BUFFLEN,
BUFFER,
COMPCODE,
REASON);

/*****
/* TEST THE COMPLETION CODE OF THE PUT1 CALL. */
/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE SHOWING */
/* THE COMPLETION CODE AND THE REASON CODE. */
*****/
IF COMPCODE /= MQCC_OK
THEN DO;
:
CALL ERROR_ROUTINE;
END;
```

## z/OS 메시지 받기

이 예에서는 MQGET 호출을 사용하여 큐에서 메시지를 제거하는 방법을 설명합니다.

이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다.

```
%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****
/* WORKING STORAGE DECLARATIONS */
*****/
DCL COMPCODE          BINARY FIXED (31);
DCL REASON            BINARY FIXED (31);
DCL HCONN            BINARY FIXED (31);
DCL HOBJ            BINARY FIXED (31);
DCL BUFFLEN          BINARY FIXED (31);
DCL DATALEN         BINARY FIXED (31);
DCL BUFFER           CHAR(80);
:

/*****
/* LOCAL COPY OF MESSAGE DESCRIPTOR AND */
/* GET MESSAGE OPTIONS */
*****/
DCL 1 LMQMD LIKE MQMD;
DCL 1 LMQGMO LIKE MQGMO;
:
/*****
/* SET UP MESSAGE DESCRIPTOR AS REQUIRED. */
/* MSGID AND CORRELID IN MQMD SET TO NULLS SO FIRST */
/* AVAILABLE MESSAGE WILL BE RETRIEVED. */
*****/
LMQMD.MSGID = MQMI_NONE;
LMQMD.CORRELID = MQCI_NONE;

/*****
/* SET UP GET MESSAGE OPTIONS AS REQUIRED. */
*****/
LMQGMO.OPTIONS = MQGMO_NO_SYNCPOINT;

/*****
/* SET UP LENGTH OF MESSAGE BUFFER. */
*****/
BUFFLEN = LENGTH(BUFFER);
/*
/* HCONN WAS SET BY PREVIOUS MQCONN REQUEST.
/* HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST.
/*
*****/

CALL MQGET (HCONN,
           HOBJ,
           LMQMD,
           LMQGMO,
           BUFFERLEN,
           BUFFER,
           DATALEN,
           COMPCODE,
           REASON);

/*****
/* TEST THE COMPLETION CODE OF THE GET CALL.
/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE
/* SHOWING THE COMPLETION CODE AND THE REASON CODE.
*****/
IF COMPCODE = MQCC_OK
  THEN DO;
  :
  CALL ERROR_ROUTINE;
END;
```

## z/OS 대기 옵션을 사용하여 메시지 받기

이 예에서는 대기 옵션과 함께 MQGET 호출을 사용하고 잘린 메시지를 수락하는 방법을 설명합니다.

이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다.

```
%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****
/* WORKING STORAGE DECLARATIONS */
*****/
DCL COMPCODE          BINARY FIXED (31);
DCL REASON            BINARY FIXED (31);
DCL HCONN            BINARY FIXED (31);
DCL HOBJ            BINARY FIXED (31);
DCL BUFFLEN          BINARY FIXED (31);
DCL DATALEN        BINARY FIXED (31);
DCL BUFFER           CHAR(80);
:
/*****
/* LOCAL COPY OF MESSAGE DESCRIPTOR AND GET MESSAGE */
/* OPTIONS */
*****/
DCL 1 LMQMD LIKE MQMD;
DCL 1 LMQGMO LIKE MQGMO;
:
/*****
/* SET UP MESSAGE DESCRIPTOR AS REQUIRED. */
/* MSGID AND CORRELID IN MQMD SET TO NULLS SO FIRST */
/* AVAILABLE MESSAGE WILL BE RETRIEVED. */
*****/
LMQMD.MSGID = MQMI_NONE;
LMQMD.CORRELID = MQCI_NONE;

/*****
/* SET UP GET MESSAGE OPTIONS AS REQUIRED. */
/* WAIT INTERVAL SET TO ONE MINUTE. */
*****/
LMQGMO.OPTIONS = MQGMO_WAIT +
                MQGMO_ACCEPT_TRUNCATED_MSG +
                MQGMO_NO_SYNCPOINT;
LMQGMO.WAITINTERVAL=60000;

/*****
/* SET UP LENGTH OF MESSAGE BUFFER. */
*****/
BUFFLEN = LENGTH(BUFFER);

/*****
/*
/* HCONN WAS SET BY PREVIOUS MQCONN REQUEST.
/* HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST.
/*
*****/

CALL MQGET (HCONN,
           HOBJ,
           LMQMD,
           LMQGMO,
           BUFFERLEN,
           BUFFER,
           DATALEN,
           COMPCODE,
           REASON);

/*****
/* TEST THE COMPLETION CODE OF THE GET CALL.
/* TAKE APPROPRIATE ACTION BASED ON COMPLETION CODE AND
/* REASON CODE.
*****/

SELECT(COMPCODE);
  WHEN (MQCC_OK) DO; /* GET WAS SUCCESSFUL */
  :
  END;
  WHEN (MQCC_WARNING) DO;
    IF REASON = MQRC_TRUNCATED_MSG_ACCEPTED
    THEN DO; /* GET WAS SUCCESSFUL */
    :
  END;
```

```

ELSE DO;
  :
  CALL ERROR_ROUTINE;
END;
END;
WHEN (MQCC_FAILED) DO;
  :
  CALL ERROR_ROUTINE;
END;
END;
OTHERWISE;
END;

```

## z/OS 신호보내기를 사용하여 메시지 받기

신호보내기에서 MQGET 호출을 사용하는 방법을 보여줍니다.

신호보내기는 IBM MQ for z/OS에서만 사용 가능합니다.

이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다.

```

%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****
/* WORKING STORAGE DECLARATIONS */
*****/
DCL COMPCODE          BINARY FIXED (31);
DCL REASON            BINARY FIXED (31);
DCL HCONN             BINARY FIXED (31);
DCL HOBJ              BINARY FIXED (31);
DCL DATALEN          BINARY FIXED (31);
DCL BUFLLEN           BINARY FIXED (31);
DCL BUFFER            CHAR(80);
:
DCL ECB_FIXED          FIXED BIN(31);
DCL 1 ECB_OVERLAY BASED(ADDR(ECB_FIXED)),
    3 ECB_WAIT BIT,
    3 ECB_POSTED BIT,
    3 ECB_FLAG3_8 BIT(6),
    3 ECB_CODE PIC'999';
:
/*****
/* LOCAL COPY OF MESSAGE DESCRIPTOR AND GET MESSAGE */
/* OPTIONS */
*****/
DCL 1 LMQMD LIKE MQMD;
DCL 1 LMQGMO LIKE MQGMO;
:
/*****
/* CLEAR ECB FIELD. */
*****/
ECB_FIXED = 0;
:
/*****
/* SET UP MESSAGE DESCRIPTOR AS REQUIRED. */
/* MSGID AND CORRELID IN MQMD SET TO NULLS SO FIRST */
/* AVAILABLE MESSAGE WILL BE RETRIEVED. */
*****/
LMQMD.MSGID = MQMI_NONE;
LMQMD.CORRELID = MQCI_NONE;
/*****
/* SET UP GET MESSAGE OPTIONS AS REQUIRED. */
/* WAIT INTERVAL SET TO ONE MINUTE. */
*****/
LMQGMO.OPTIONS = MQGMO_SET_SIGNAL +
                MQGMO_NO_SYNCPOINT;
LMQGMO.WAITINTERVAL=60000;
LMQGMO.SIGNAL1 = ADDR(ECB_FIXED);

/*****
/* SET UP LENGTH OF MESSAGE BUFFER. */
/* CALL MESSAGE RETRIEVAL ROUTINE. */
*****/
BUFLLEN = LENGTH(BUFFER);

```

```

CALL GET_MSG;

/*****
/* TEST THE COMPLETION CODE OF THE GET CALL.          */
/* TAKE APPROPRIATE ACTION BASED ON COMPLETION CODE AND */
/* REASON CODE.                                       */
*****/

SELECT;
  WHEN ((COMPCODE = MQCC_OK) &
        (REASON = MQCC_NONE)) DO
    :
    CALL MSG_ROUTINE;
    :
  END;
  WHEN ((COMPCODE = MQCC_WARNING) &
        (REASON = MQRC_SIGNAL_REQUEST_ACCEPTED)) DO;
    :
    CALL DO_WORK;
    :
  END;
  WHEN ((COMPCODE = MQCC_FAILED) &
        (REASON = MQRC_SIGNAL_OUTSTANDING)) DO;
    :
    CALL DO_WORK;
    :
  END;
  OTHERWISE DO;          /* FAILURE CASE */
/*****
/* ISSUE AN ERROR MESSAGE SHOWING THE COMPLETION CODE */
/* AND THE REASON CODE.                               */
*****/
    :
    CALL ERROR_ROUTINE;
    :
  END;
END;
:
:

```

```

DO_WORK: PROC;
:
IF ECB_POSTED
THEN DO;
  SELECT(ECB_CODE);
  WHEN(MQEC_MSG_ARRIVED) DO;
    :
    CALL GET_MSG;
    :
  END;
  WHEN(MQEC_WAIT_INTERVAL_EXPIRED) DO;
    :
    CALL NO_MSG;
    :
  END;
  OTHERWISE DO;          /* FAILURE CASE */
/*****
/* ISSUE AN ERROR MESSAGE SHOWING THE COMPLETION CODE */
/* AND THE REASON CODE.                               */
*****/
    :
    CALL ERROR_ROUTINE;
    :
  END;
END;
:
END;
END DO_WORK;

GET_MSG: PROC;

```

```

/*****
/*
/* HCONN WAS SET BY PREVIOUS MQCONN REQUEST.          */
/* HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST.          */
*****/

```

```

/* MD AND GMO SET UP AS REQUIRED.          */
/*                                         */
/*****
CALL MQGET (HCONN,
            HOBJ,
            LMQMD,
            LMQGMO,
            BUFFLEN,
            BUFFER,
            DATALEN,
            COMPCODE,
            REASON);

END GET_MSG;

NO_MSG: PROC;
:
END NO_MSG;

```

## **z/OS** 오브젝트의 속성 조회

이 예에서는 MQINQ 호출을 사용하여 큐의 속성을 조회하는 방법을 설명합니다.

이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다.

```

%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****
/* WORKING STORAGE DECLARATIONS          */
/*****
DCL COMPCODE          BINARY FIXED (31);
DCL REASON            BINARY FIXED (31);
DCL HCONN             BINARY FIXED (31);
DCL HOBJ              BINARY FIXED (31);
DCL OPTIONS           BINARY FIXED (31);
DCL SELECTORCOUNT   BINARY FIXED (31);
DCL INTATTRCOUNT   BINARY FIXED (31);
DCL 1 SELECTOR_TABLE,
   3 SELECTORS(5)      BINARY FIXED (31);
DCL 1 INTATTR_TABLE,
   3 INTATTRS(5)      BINARY FIXED (31);
DCL CHARATTRLENGTH   BINARY FIXED (31);
DCL CHARATTRS        CHAR(100);
:

/*****
/* SET VARIABLES FOR INQUIRE CALL        */
/* INQUIRE ON THE CURRENT QUEUE DEPTH    */
/*****

SELECTORS(01) = MQIA_CURRENT_Q_DEPTH;

SELECTORCOUNT = 1;
INTATTRCOUNT  = 1;

CHARATTRLENGTH = 0;
/*****
/*                                         */
/* HCONN WAS SET BY PREVIOUS MQCONN REQUEST. */
/* HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST.  */
/*                                         */
/*****
CALL MQINQ (HCONN,
            HOBJ,
            SELECTORCOUNT,
            SELECTORS,
            INTATTRCOUNT,
            INTATTRS,
            CHARATTRLENGTH,
            CHARATTRS,
            COMPCODE,
            REASON);

/*****
/* TEST THE COMPLETION CODE OF THE INQUIRE CALL. */

```

```

/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE SHOWING */
/* THE COMPLETION CODE AND THE REASON CODE. */
/*****
  IF COMPCODE = MQCC_OK
  THEN DO;
  :
  CALL ERROR_ROUTINE;
END;

```

## z/OS 큐의 속성 설정

이 예에서는 MQSET 호출을 사용하여 큐의 속성을 변경하는 방법을 설명합니다.

이 내용은 IBM MQ와 함께 제공된 샘플 애플리케이션에서 추출되지 않습니다.

```

%INCLUDE SYSLIB(CMQP);
%INCLUDE SYSLIB(CMQEPP);
:
/*****
/* WORKING STORAGE DECLARATIONS */
/*****
  DCL COMPCODE          BINARY FIXED (31);
  DCL REASON            BINARY FIXED (31);
  DCL HCONN             BINARY FIXED (31);
  DCL HOBJ              BINARY FIXED (31);
  DCL OPTIONS           BINARY FIXED (31);
  DCL SELECTORCOUNT   BINARY FIXED (31);
  DCL INTATTRCOUNT    BINARY FIXED (31);
  DCL 1 SELECTOR_TABLE,
      3 SELECTORS(5)      BINARY FIXED (31);
  DCL 1 INTATTR_TABLE,
      3 INTATTRS(5)      BINARY FIXED (31);
  DCL CHARATTRLENGTH   BINARY FIXED (31);
  DCL CHARATTRS        CHAR(100);
  :

/*****
/* SET VARIABLES FOR SET CALL */
/* SET GET AND PUT INHIBITED */
/*****

  SELECTORS(01) = MQIA_INHIBIT_GET;
  SELECTORS(02) = MQIA_INHIBIT_PUT;

  INTATTRS(01) = MQQA_GET_INHIBITED;
  INTATTRS(02) = MQQA_PUT_INHIBITED;

  SELECTORCOUNT = 2;
  INTATTRCOUNT = 2;

  CHARATTRLENGTH = 0;

/*****
/*
/* HCONN WAS SET BY PREVIOUS MQCONN REQUEST. */
/* HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST. */
/*
/*****
  CALL MQSET (HCONN,
             HOBJ,
             SELECTORCOUNT,
             SELECTORS,
             INTATTRCOUNT,
             INTATTRS,
             CHARATTRLENGTH,
             CHARATTRS,
             COMPCODE,
             REASON);

/*****
/* TEST THE COMPLETION CODE OF THE SET CALL. */
/* IF THE CALL HAS FAILED ISSUE AN ERROR MESSAGE SHOWING */
/* THE COMPLETION CODE AND THE REASON CODE. */
/*****
  IF COMPCODE = MQCC_OK
  THEN DO;
  :

```



```
CALL ERROR_ROUTINE;
END;
```

## Constants

이 절의 참조 정보를 사용하여 비즈니스 요구를 해결하는 태스크를 달성합니다.

### IBM MQ COPY, 헤더, 포함 및 모듈 파일

이 정보는 범용 프로그래밍 인터페이스 정보입니다.

이 절에는 다음과 같이 다양한 프로그래밍 언어의 MQI를 사용하는 데 도움이 되는 정보가 있습니다.

#### C 헤더 파일

헤더 파일은 MQI를 사용하는 C 애플리케이션 프로그램의 작성에 도움이 되도록 제공됩니다.

C 헤더 파일은 다음 테이블에 요약되어 있습니다.

표 1. C 헤더 파일 - 호출 프로토타입, 데이터 유형, 리턴 코드, 상수 및 구조					
파일 이름	설명	IBM i	AIX® and Linux® 시스템	Windows	z/OS
호출 프로토타입, 데이터 유형, 리턴 코드, 상수 및 구조					
CMQC	MQI 정의	C	C	C	C
CMQBC	MQAI 정의	C	C	C	
CMQEC	인터페이스 시작점 정의(CMQC, CMQXC 및 CMQZC 포함)		C	C	
CMQCFC	PCF 정의	C	C	C	C
CMQPSC	발행/구독 정의	C	C	C	C
CMQXC	채널 및 엑시트 정의	C	C	C	C
CMQZC	설치 가능 서비스 정의	C	C	C	
키: C= 파일이 제공됨					

#### COBOL COPY 파일

다양한 COPY 파일이 MQI를 사용하는 COBOL 애플리케이션 프로그램의 작성에 도움이 되도록 제공됩니다.

표 2. COBOL 복사 파일 - 리턴 코드, 상수 및 구조					
파일 이름	설명	IBM i	AIX and Linux	Windows	z/OS
리턴 코드 및 상수					
CMQx	MQI 정의	V	V	V	V
CMQCFx	PCF 정의	V	V	V	V
CMQPSx	발행/구독 정의	V	V	V	V
CMQXx	채널 및 엑시트 정의	V	V	V	V
구조					
CMQAIRx	MQAIR - 인증 정보 레코드		V L	V L	
CMQBOX	MQBO - 시작 옵션	V L	V L	V L	

표 2. COBOL 복사 파일 - 리턴 코드, 상수 및 구조 (계속)					
파일 이름	설명	IBM i	AIX and Linux	Windows	z/OS
CMQCDx	MQCD - 채널 정의	V L	V L	V L	V L
CMQCFBFx	MQCFBF - PCF 바이트 문자열 필터 매개변수	V L	V L	V L	V L
CMQCFBSx	MQCFBS - PCF 바이트 문자열 매개변수	V L	V L	V L	V L
CMQCFGRx	MQCFGR - PCF 그룹 매개변수	V L	V L	V L	V L
CMQCFHx	MQCFH - PCF 헤더	V L	V L	V L	V L
CMQCFIFx	MQCFIF - PCF 정수 필터 매개변수	V L	V L	V L	V L
CMQCFILx	MQCFIL - PCF 정수 목록 매개변수	V L	V L	V L	V L
CMQCFINx	MQCFIN - PCF 정수 매개변수	V L	V L	V L	V L
CMQCFSFx	MQCFSF - PCF 문자열 필터 매개변수	V L	V L	V L	V L
CMQCFSLx	MQCFSL - PCF 문자열 목록 매개변수	V L	V L	V L	V L
CMQCFSTx	MQCFST - PCF 문자열 매개변수	V L	V L	V L	V L
CMQCFXLx	MQCFIL64 - PCF 64비트 정수 목록 매개변수	V L	V L	V L	V L
CMQCFXNx	MQCFIN64 - PCF 64비트 정수 매개변수	V L	V L	V L	V L
CMQCHARVx	MQCHARV - 가변 길이 문자열	V L	V L	V L	V L
CMQCIHx	MQCIH - CICS® bridge 헤더	V L	V L	V L	V L
CMQCNOx	MQCNO - 연결 옵션	V L	V L	V L	V L
CMQCSPx	MQCSP - 보안 매개변수	V L	V L	V L	V L
CMQCXPx	MQCXP - 채널 엑시트 매개변수	V L			V L
CMQDHx	MQDH - 분배 헤더	V L	V L	V L	V L
CMQDLHx	MQDLH - 데드-레터 헤더	V L	V L	V L	V L
CMQDXPx	MQDXP - 데이터 변환 엑시트 매개변수	V L		V L	
CMQEPHx	MQEPH - 임베드된 PCF 헤더	V L	V L	V L	V L
CMQGMox	MQGMo - 메시지 가져오기 옵션	V L	V L	V L	V L
CMQIIHx	MQIIH - IMS 정보 헤더	V L	V L	V L	V L
CMQMDx	MQMD - 메시지 디스크립터	V L	V L	V L	V L
CMQMD1x	MQMD1 - 메시지 디스크립터 버전 1	V L	V L	V L	V L
CMQMD2x	MQMD2 - 메시지 디스크립터 버전 2	V L	V L	V L	V L
CMQMDEx	MQMDE - 메시지 디스크립터 확장	V L	V L	V L	V L
CMQODx	MQOD - 오브젝트 디스크립터	V L	V L	V L	V L
CMQORx	MQOR - 오브젝트 레코드	V L	V L	V L	V L

표 2. COBOL 복사 파일 - 리턴 코드, 상수 및 구조 (계속)					
파일 이름	설명	IBM i	AIX and Linux	Windows	z/OS
CMQPMOx	MQPMO - 메시지 넣기 옵션	V L	V L	V L	V L
CMQRFHx	MQRFH - 규칙 및 형식화 헤더	V L	V L	V L	V L
CMQRFH2x	MQRFH2 - 규칙 및 형식화 헤더 2	V L	V L	V L	V L
CMQRMHx	MQRMH - 참조 메시지 헤더	V L	V L	V L	V L
CMQRRx	MQRR - 응답 레코드	V L	V L	V L	
CMQSCOx	MQSCO - TLS 구성 옵션		V L	V L	
CMQTMx	MQTM - 트리거 메시지	V L		V L	V L
CMQTMCx	MQTM - 트리거 메시지 문자	V L	V L		
CMQTM2x	MQTM2 - 트리거 메시지 2 문자	V L	V L	V L	V L
CMQWIHx	MQWIH - 작업 정보 헤더	V L	V L	V L	V L
CMQXQHx	MQXQH - 전송 큐 헤더	V L	V L	V L	V L

**키:**

- 초기값이 제공된 파일, x=V
- 초기값이 제공되지 않은 파일, x=L

### z/OS PL/I include files

A number of INCLUDE files are provided for the PL/I programming language. These files are available on z/OS only.

Table 3. PL/I include files - data types, return codes, constants, and structures					
File name	Description	IBM i	AIX and Linux	Windows	z/OS
<b>Data types, return codes, constants, and structures</b>					
CMQP	MQI definitions				P
CMQCFP	PCF definitions				P
CMQEPP	Entry point definitions				P
CMQSP	Publish/Subscribe definitions				P
CMQXP	Channel and exit definitions				P

**Key:** P= File provided

### IBM i RPG 복사 파일

RPG COPY 파일이 RPG 프로그래밍 언어에 대해 제공됩니다. 이러한 파일은 IBM i에서만 사용 가능합니다.

표 4. RPG 복사 파일 - 리턴 코드, 상수 및 구조					
파일 이름	설명	IBM i	AIX and Linux	Windows	z/OS
<b>리턴 코드 및 상수</b>					
CMQx	MQI 정의	G R			

표 4. RPG 복사 파일 - 리턴 코드, 상수 및 구조 (계속)					
파일 이름	설명	IBM i	AIX and Linux	Windows	z/OS
CMQCFx	PCF 정의	G			
CMQPSx	발행/구독 정의	G			
CMQXx	채널 및 엑시트 정의	G R			
<b>구조</b>					
CMQBOx	MQBO - 시작 옵션	G H			
CMQCDx	MQCD - 채널 정의	G H R			
CMQCFBFx	MQCFBF - PCF 바이트 문자열 필터 매개변수	G H			
CMQCFBSx	MQCFBS - PCF 바이트 문자열 매개변수	G H			
CMQCFGRx	MQCFGR - PCF 그룹 매개변수	G H			
CMQCFHx	MQCFH - PCF 헤더	G H			
CMQCFIFx	MQCFIF - PCF 정수 필터 매개변수	G H			
CMQCFILx	MQCFIL - PCF 정수 목록 매개변수	G H			
CMQCFINx	MQCFIN - PCF 정수 매개변수	G H			
CMQCFSFx	MQCFSF - PCF 문자열 필터 매개변수	G H			
CMQCFSLx	MQCFSL - PCF 문자열 목록 매개변수	G H			
CMQCFSTx	MQCFST - PCF 문자열 매개변수	G H			
CMQCFXLx	MQCFIL64 - PCF 64비트 정수 목록 매개변수	G H			
CMQCFXNx	MQCFIN64 - PCF 64비트 정수 매개변수	G H			
CMQCHARVx	MQCHARV - 가변 길이 문자열	G H			
CMQCIHx	MQCIH - CICS bridge 헤더	G H			
CMQCNOx	MQCNO - 연결 옵션	G H			
CMQCSPx	MQCSP - 보안 매개변수	G H			
CMQCXPx	MQCXP - 채널 엑시트 매개변수	G H R			
CMQDHx	MQDH - 분배 헤더	G H R			
CMQDLHx	MQDLH - 데드-레터 헤더	G H R			
CMQDXPx	MQDXP - 데이터 변환 엑시트 매개변수	G H R			
CMQEPHx	MQEPH - 임베드된 PCF 헤더	G H			
CMQGMOx	MQGMO - 메시지 가져오기 옵션	G H R			
CMQIIHx	MQIIH - IMS 정보 헤더	G H R			
CMQMDx	MQMD - 메시지 디스크립터	G H R			

표 4. RPG 복사 파일 - 리턴 코드, 상수 및 구조 (계속)					
파일 이름	설명	IBM i	AIX and Linux	Windows	z/OS
CMQMD1x	MQMD1 - 메시지 디스크립터 버전 1	G H R			
CMQMD2x	MQMD2 - 메시지 디스크립터 버전 2	G H			
CMQMDEx	MQMDE - 메시지 디스크립터 확장	G H R			
CMQODx	MQOD - 오브젝트 디스크립터	G H R			
CMQORx	MQOR - 오브젝트 레코드	G H R			
CMQPMOx	MQPMO - 메시지 넣기 옵션	G H R			
CMQXPx	MQXPx - 발행/구독 라우팅 엑시트 매개변수	G H			
CMQRFHx	MQRFH - 규칙 및 형식화 헤더	G H			
CMQRFH2x	MQRFH2 - 규칙 및 형식화 헤더 2	G H			
CMQRMHx	MQRMH - 참조 메시지 헤더	G H R			
CMQRRx	MQRR - 응답 레코드	G H R			
CMQTMx	MQTM - 트리거 메시지	G H R			
CMQTMcx	MQTMC - 트리거 메시지 문자	G H R			
CMQTM2x	MQTMC2 - 트리거 메시지 2 문자	G H R			
CMQWIHx	MQWIH - 작업 정보 헤더	G H			
CMQXQHx	MQXQH - 전송 큐 헤더	G H R			

**키:**

- 정적 링크용의 초기화되고 제공된 파일, x=G
- 정적 링크용의 초기화되지 않고 제공된 파일, x=H
- 동적 링크용의 초기화되고 제공된 파일, x=R

**Windows Visual Basic 모듈 파일**

헤더(또는 양식) 파일은 MQI를 사용하는 Visual Basic 애플리케이션 프로그램의 작성에 도움이 되도록 제공됩니다. 이러한 헤더 파일은 32비트 버전으로만 제공됩니다.

표 5. Visual Basic 모듈 파일 - 호출 선언, 데이터 유형, 리턴 코드, 상수 및 구조					
파일 이름	설명	IBM i	AIX and Linux 시스템	Windows	z/OS
<b>호출 선언, 데이터 유형, 리턴 코드, 상수 및 구조</b>					
CMQB	MQI 정의			B	
CMQBB	MQAI 정의			B	
CMQCFB	PCF 정의			B	
CMQXB	채널 및 엑시트 정의			B	

**키:** B= 파일이 제공됨

**z/OS** **z/OS Assembler COPY files**

Various COPY files are provided to help you write z/OS Assembler application programs that use the MQI.

<i>Table 6. z/OS Assembler copy files - data types, return codes, constants, and structures</i>					
<b>File name</b>	<b>Description</b>	<b>IBM i</b>	<b>AIX and Linux</b>	<b>Windows</b>	<b>z/OS</b>
<b>Data types, return codes, and constants</b>					
CMQA	MQI definitions				A
CMQCFA	PCF definitions				A
CMQPSA	Publish/Subscribe definitions				A
CMQVERA	Structure version control				A
CMQXA	Channel and exit definitions				A
<b>Structures</b>					
CMQCD	MQCD - Channel definition				
CMQCFBFA	MQCFBF - PCF byte string filter parameter				
CMQCFBSA	MQCFBS - PCF byte string parameter				A
CMQCFGRA	MQCFGR - PCF group parameter				A
CMQCFHA	MQCFH - PCF header				A
CMQCFIFA	MQCFIF - PCF integer filter parameter				A
CMQCFILA	MQCFIL - PCF integer list parameter				A
CMQCFINA	MQCFIN - PCF integer parameter				A
CMQCFSTA	MQCFST - PCF string parameter				A
CMQCFSLA	MQCFSL - PCF string list parameter				A
CMQCFXLA	MQCFIL64 - PCF 64-bit integer list parameter				A
CMQCFXNA	MQCFIN64 - PCF 64-bit integer parameter				A
CMQCHARVA	MQCHARV - Variable length string				A
CMQCIHA	MQCIH - CICS bridge header				A
CMQCNOA	MQCNO - Connect options				A
CMQCSPA	MQCSP - Security parameters				A
CMQCXPA	MQCXP - Channel exit parameters				A
CMQDHA	MQDH - Distribution header				A
CMQDLHA	MQDLH - Dead-letter header				A

Table 6. z/OS Assembler copy files - data types, return codes, constants, and structures (continued)

File name	Description	IBM i	AIX and Linux	Windows	z/OS
CMQDXPA	MQDXP - Data conversion exit parameters				A
CMQEPHA	MQEPH - Embedded PCF header				A
CMQGMOA	MQGMO - Get message options				A
CMQIIHA	MQIIH - IMS information header				A
CMQMDA	MQMD - Message descriptor				A
CMQMD1A	MQMD1 - Message descriptor version 1				A
CMQMD2A	MQMD2 - Message descriptor version 2				A
CMQMDEA	MQMDE - Message descriptor extended				A
CMQODA	MQOD - Object descriptor				A
CMQORA	MQOR - Object record				A
CMQPMOA	MQPMO - Put message options				A
CMQRFHA	MQRFH - Rules and formatting header				A
CMQRFH2A	MQRFH2 - Rules and formatting header 2				A
CMQRMHA	MQRMH - Reference message header				A
CMQTMA	MQTM - Trigger message				A
CMQTMC2A	MQTMC2 - Trigger message 2 character				A
CMQWCRA	MQWCR - Cluster workload cluster record				A
CMQWDRA	MQWDR - Cluster workload destination record				A
CMQWDR1A	MQWDR1 - Cluster workload destination record version 1				A
CMQWDR2A	MQWDR2 - Cluster workload destination record version 2				A
CMQWIHA	MQWIH - Work information header				A
CMQWQRA	MQWQR - Cluster workload queue record				A
CMQWQR1A	MQWQR1 - Cluster workload queue record version 1				A
CMQWQR2A	MQWQR2 - Cluster workload queue record version 2				A

Table 6. z/OS Assembler copy files - data types, return codes, constants, and structures (continued)

File name	Description	IBM i	AIX and Linux	Windows	z/OS
CMQWXP	MQWXP - Cluster workload exit parameters				A
CMQWXP1A	MQWXP1 - Cluster workload exit parameters version 1				A
CMQWXP2A	MQWXP2 - Cluster workload exit parameters version 2				A
CMQWXP3A	MQWXP3 - Cluster workload exit parameters version 3				A
CMQXPA	MQXP - CICS API-crossing exit parameters				A
CMQXQHA	MQXQH - Transmission queue header				A
CMQXWDA	MQXWD - Exit wait descriptor				A

**Key:** A= File provided

### MQ\_\*(문자열 길이)

표 7. 상수 값		
이름	10진수 값	16진수 값
MQ_ABEND_CODE_LENGTH	4	X'00000004'
MQ_ACCOUNTING_TOKEN_LENGTH	32	X'00000020'
MQ_APPL_FUNCTION_NAME_LENGTH	10	X'0000000A'
MQ_APPL_IDENTITY_DATA_LENGTH	32	X'00000020'
MQ_APPL_NAME_LENGTH	28	X'0000001C'
MQ_APPL_ORIGIN_DATA_LENGTH	4	X'00000004'
MQ_APPL_TAG_LENGTH	28	X'0000001C'
MQ_ARM_SUFFIX_LENGTH	2	X'00000002'
MQ_ATTENTION_ID_LENGTH	4	X'00000004'
MQ_AUTH_INFO_CONN_NAME_LENGTH	264	X'00000108'
MQ_AUTH_INFO_DESC_LENGTH	64	X'00000040'
MQ_AUTH_INFO_NAME_LENGTH	48	X'00000030'
MQ_AUTH_INFO_OCSP_URL_LENGTH	256	X'00000100'
MQ_AUTHENTICATOR_LENGTH	8	X'00000008'
MQ_AUTO_REORG_CATALOG_LENGTH	44	X'0000002C'
MQ_AUTO_REORG_TIME_LENGTH	4	X'00000004'
MQ_BATCH_INTERFACE_ID_LENGTH	8	X'00000008'
MQ_BRIDGE_NAME_LENGTH	24	X'00000018'
MQ_CANCEL_CODE_LENGTH	4	X'00000004'
MQ_CF_STRUC_DESC_LENGTH	64	X'00000040'
MQ_CF_STRUC_NAME_LENGTH	12	X'0000000C'



표 7. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQ_CHANNEL_DATE_LENGTH	12	X'0000000C'
MQ_CHANNEL_DESC_LENGTH	64	X'00000040'
MQ_CHANNEL_NAME_LENGTH	20	X'00000014'
MQ_CHANNEL_TIME_LENGTH	8	X'00000008'
MQ_CHINIT_SERVICE_PARM_LENGTH	32	X'00000020'
MQ_CICS_FILE_NAME_LENGTH	8	X'00000008'
MQ_CLIENT_ID_LENGTH	23	X'00000017'
MQ_CLUSTER_NAME_LENGTH	48	X'00000030'
MQ_CONN_NAME_LENGTH	264	X'00000108'
MQ_CONN_TAG_LENGTH	128	X'00000080'
MQ_CONNECTION_ID_LENGTH	24	X'00000018'
MQ_CORREL_ID_LENGTH	24	X'00000018'
MQ_CREATION_DATE_LENGTH	12	X'0000000C'
MQ_CREATION_TIME_LENGTH	8	X'00000008'
MQ_DATE_LENGTH	12	X'0000000C'
MQ_DISTINGUISHED_NAME_LENGTH	1024	X'00000400'
MQ_DNS_GROUP_NAME_LENGTH	18	X'00000012'
MQ_EXIT_DATA_LENGTH	32	X'00000020'
MQ_EXIT_INFO_NAME_LENGTH	48	X'00000030'
MQ_EXIT_NAME_LENGTH	(value differs by platform or version)	
MQ_EXIT_PD_AREA_LENGTH	48	X'00000030'
MQ_EXIT_USER_AREA_LENGTH	16	X'00000010'
MQ_FACILITY_LENGTH	8	X'00000008'
MQ_FACILITY_LIKE_LENGTH	4	X'00000004'
MQ_FORMAT_LENGTH	8	X'00000008'
MQ_FUNCTION_LENGTH	4	X'00000004'
MQ_GROUP_ID_LENGTH	24	X'00000018'
MQ_LDAP_PASSWORD_LENGTH	32	X'00000020'
MQ_LISTENER_NAME_LENGTH	48	X'00000030'
MQ_LISTENER_DESC_LENGTH	64	X'00000040'
MQ_LOCAL_ADDRESS_LENGTH	48	X'00000030'
MQ_LTERM_OVERRIDE_LENGTH	8	X'00000008'
MQ_LU_NAME_LENGTH	8	X'00000008'
MQ_LUWID_LENGTH	16	X'00000010'
MQ_MAX_EXIT_NAME_LENGTH	128	X'00000080'
MQ_MAX_MCA_USER_ID_LENGTH	64	X'00000040'
MQ_MAX_PROPERTY_NAME_LENGTH	4095	X'00000FFF'
MQ_MAX_USER_ID_LENGTH	64	X'00000040'
MQ_MCA_JOB_NAME_LENGTH	28	X'0000001C'

표 7. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQ_MCA_NAME_LENGTH	20	X'00000014'
MQ_MCA_USER_DATA_LENGTH	32	X'00000020'
MQ_MCA_USER_ID_LENGTH	(value differs by platform or version)	(value differs by platform or version)
MQ_MFS_MAP_NAME_LENGTH	8	X'00000008'
MQ_MODE_NAME_LENGTH	8	X'00000008'
MQ_MSG_HEADER_LENGTH	4000	X'00000FA0'
MQ_MSG_ID_LENGTH	24	X'00000018'
MQ_MSG_TOKEN_LENGTH	16	X'00000010'
MQ_NAMELIST_DESC_LENGTH	64	X'00000040'
MQ_NAMELIST_NAME_LENGTH	48	X'00000030'
MQ_NHA_INSTANCE_NAME_LENGTH	48	X'00000030'
MQ_OBJECT_INSTANCE_ID_LENGTH	24	X'00000018'
MQ_OBJECT_NAME_LENGTH	48	X'00000030'
MQ_PASS_TICKET_APPL_LENGTH	8	X'00000008'
MQ_PASSWORD_LENGTH	12	X'0000000C'
MQ_PROCESS_APPL_ID_LENGTH	256	X'00000100'
MQ_PROCESS_DESC_LENGTH	64	X'00000040'
MQ_PROCESS_ENV_DATA_LENGTH	128	X'00000080'
MQ_PROCESS_NAME_LENGTH	48	X'00000030'
MQ_PROCESS_USER_DATA_LENGTH	128	X'00000080'
MQ_PROGRAM_NAME_LENGTH	20	X'00000014'
MQ_PUT_APPL_NAME_LENGTH	28	X'0000001C'
MQ_PUT_DATE_LENGTH	8	X'00000008'
MQ_PUT_TIME_LENGTH	8	X'00000008'
MQ_Q_DESC_LENGTH	64	X'00000040'
MQ_Q_MGR_DESC_LENGTH	64	X'00000040'
MQ_Q_MGR_IDENTIFIER_LENGTH	48	X'00000030'
MQ_Q_MGR_NAME_LENGTH	48	X'00000030'
MQ_Q_NAME_LENGTH	48	X'00000030'
MQ_QSG_NAME_LENGTH	4	X'00000004'
MQ_REMOTE_SYS_ID_LENGTH	4	X'00000004'
MQ_SECURITY_ID_LENGTH	40	X'00000028'
MQ_SELECTOR_LENGTH	10240	X'00002800'
MQ_SERVICE_ARGS_LENGTH	255	X'000000FF'
MQ_SERVICE_COMMAND_LENGTH	255	X'000000FF'
MQ_SERVICE_DESC_LENGTH	64	X'00000040'
MQ_SERVICE_NAME_LENGTH	32	X'00000020'
MQ_SERVICE_PATH_LENGTH	255	X'000000FF'
MQ_SERVICE_STEP_LENGTH	8	X'00000008'

표 7. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQ_SHORT_CONN_NAME_LENGTH	20	X'00000014'
MQ_SHORT_DNAME_LENGTH	256	X'00000100'
MQ_SSL_CIPHER_SPEC_LENGTH	32	X'00000020'
MQ_SSL_CRYPTOHARDWARE_LENGTH	256	X'00000100'
MQ_SSL_HANDSHAKE_STAGE_LENGTH	32	X'00000020'
MQ_SSL_KEY_LIBRARY_LENGTH	44	X'0000002C'
MQ_SSL_KEY_MEMBER_LENGTH	8	X'00000008'
MQ_SSL_KEY_REPOSITORY_LENGTH	256	X'00000100'
MQ_SSL_PEER_NAME_LENGTH	1024	X'00000400'
MQ_SSL_SHORT_PEER_NAME_LENGTH	256	X'00000100'
MQ_START_CODE_LENGTH	4	X'00000004'
MQ_STORAGE_CLASS_DESC_LENGTH	64	X'00000040'
MQ_STORAGE_CLASS_LENGTH	8	X'00000008'
MQ_SUB_IDENTITY_LENGTH	128	X'00000080'
MQ_SUB_POINT_LENGTH	128	X'00000080'
MQ_SUITE_B_128_BIT	2	X'00000002'
MQ_SUITE_B_192_BIT	4	X'00000004'
MQ_SUITE_B_NONE	1	X'00000001'
MQ_SUITE_B_NOT_AVAILABLE	0	X'00000000'
MQ_TCP_NAME_LENGTH	8	X'00000008'
MQ_TIME_LENGTH	8	X'00000008'
MQ_TOPIC_DESC_LENGTH	64	X'00000040'
MQ_TOPIC_NAME_LENGTH	48	X'00000030'
MQ_TOPIC_STR_LENGTH	10240	X'00002800'
MQ_TOTAL_EXIT_DATA_LENGTH	999	X'000003E7'
MQ_TOTAL_EXIT_NAME_LENGTH	999	X'000003E7'
MQ_TP_NAME_LENGTH	64	X'00000040'
MQ_TPIPE_NAME_LENGTH	8	X'00000008'
MQ_TRAN_INSTANCE_ID_LENGTH	16	X'00000010'
MQ_TRANSACTION_ID_LENGTH	4	X'00000004'
MQ_TRIGGER_DATA_LENGTH	64	X'00000040'
MQ_TRIGGER_PROGRAM_NAME_LENGTH	8	X'00000008'
MQ_TRIGGER_TERM_ID_LENGTH	4	X'00000004'
MQ_TRIGGER_TRANS_ID_LENGTH	4	X'00000004'
MQ_USER_ID_LENGTH	12	X'0000000C'
MQ_VERSION_LENGTH	8	X'00000008'
MQ_XCF_GROUP_NAME_LENGTH	8	X'00000008'
MQ_XCF_MEMBER_NAME_LENGTH	16	X'00000010'

## MQ \*(명령 형식 문자열 길이)

표 8. 상수 값		
이름	10진수 값	16진수 값
MQ_ARCHIVE_PFX_LENGTH	36	X'00000024'
MQ_ARCHIVE_UNIT_LENGTH	8	X'00000008'
MQ_ASID_LENGTH	4	X'00000004'
MQ_AUTH_PROFILE_NAME_LENGTH	48	X'00000030'
MQ_CF_LEID_LENGTH	12	X'0000000C'
MQ_COMMAND_MQSC_LENGTH	32768	X'00008000'
MQ_DATA_SET_NAME_LENGTH	44	X'0000002C'
MQ_DB2_NAME_LENGTH	4	X'00000004'
MQ_DSG_NAME_LENGTH	8	X'00000008'
MQ_ENTITY_NAME_LENGTH	1024	X'00000400'
MQ_ENV_INFO_LENGTH	96	X'00000060'
MQ_IP_ADDRESS_LENGTH	48	X'00000030'
MQ_LOG_CORREL_ID_LENGTH	8	X'00000008'
MQ_LOG_EXTENT_NAME_LENGTH	24	X'00000018'
MQ_LOG_PATH_LENGTH	1024	X'00000400'
MQ_LRSN_LENGTH	12	X'0000000C'
MQ_ORIGIN_NAME_LENGTH	8	X'00000008'
MQ_PSB_NAME_LENGTH	8	X'00000008'
MQ_PST_ID_LENGTH	8	X'00000008'
MQ_Q_MGR_CPF_LENGTH	4	X'00000004'
MQ_RESPONSE_ID_LENGTH	24	X'00000018'
MQ_RBA_LENGTH	16	X'00000010'
MQ_SECURITY_PROFILE_LENGTH	40	X'00000028'
MQ_SERVICE_COMPONENT_LENGTH	48	X'00000030'
MQ_SUB_NAME_LENGTH	10240	X'00002800'
MQ_SYSP_SERVICE_LENGTH	32	X'00000020'
MQ_SYSTEM_NAME_LENGTH	8	X'00000008'
MQ_TASK_NUMBER_LENGTH	8	X'00000008'
MQ_TPIPE_PFX_LENGTH	4	X'00000004'
MQ_UOW_ID_LENGTH	256	X'00000100'
MQ_USER_DATA_LENGTH	10240	X'00002800'
MQ_VOLSER_LENGTH	6	X'00000006'

## MQACH \*(API 엑시트 체인 영역 헤더 구조)

표 9. 상수 구조	
이름	구조
MQACH_STRUC_ID	"ACH-"
MQACH_STRUC_ID_ARRAY	'A', 'C', 'H', '-'

참고: ↵ 기호는 단일 공백 문자를 나타냅니다.

표 10. 상수 값		
이름	10진수 값	16진수 값
MQACH_VERSION_1	1	X'00000001'
MQACH_CURRENT_VERSION	1	X'00000001'
MQACH_LENGTH_1	(value differs by platform or version)	(value differs by platform or version)
MQACH_CURRENT_LENGTH	(value differs by platform or version)	(value differs by platform or version)

### MQACT\_\*(계정 토큰)

표 11. 상수 이름 및 값	
이름	값
MQACT_NONE	X'00...00' (32개널)
MQACT_NONE_ARRAY	'\0', '\0', ... (32개널)

### MQACT\_\*(명령 형식 조치 옵션)

표 12. 상수 값		
이름	10진수 값	16진수 값
MQACT_FORCE_REMOVE	1	X'00000001'
MQACT_ADVANCE_LOG	2	X'00000002'
MQACT_COLLECT_STATISTICS	3	X'00000003'
MQACT_PUBSUB	4	X'00000004'

### MQACTP\_\*(조치)

표 13. 상수 값		
이름	10진수 값	16진수 값
MQACTP_NEW	0	X'00000000'
MQACTP_FORWARD	1	X'00000001'
MQACTP_REPLY	2	X'00000002'
MQACTP_REPORT	3	X'00000003'

### MQACTT\_\*(계정 토큰 유형)

표 14. 상수 값	
이름	16진수 값
MQACTT_UNKNOWN	X'00'
MQACTT_CICS_LUOW_ID	X'01'
MQACTT_OS2_DEFAULT	X'04'
MQACTT_DOS_DEFAULT	X'05'
MQACTT_UNIX_NUMERIC_ID	X'06'
MQACTT_OS400_ACCOUNT_TOKEN	X'08'
MQACTT_WINDOWS_DEFAULT	X'09'

표 14. 상수 값 (계속)	
이름	16진수 값
MQACTT_NT_SECURITY_ID	X'0B'
MQACTT_USER	X'19'

## MQADOPT\_\*(새 MCA 확인 채택 및 새 MCA 유형 채택)

### 새 MCA 확인 채택

표 15. 상수 값		
이름	10진수 값	16진수 값
MQADOPT_CHECK_NONE	0	X'00000000'
MQADOPT_CHECK_ALL	1	X'00000001'
MQADOPT_CHECK_Q_MGR_NAME	2	X'00000002'
MQADOPT_CHECK_NET_ADDR	4	X'00000004'

### 새 MCA 유형 채택

표 16. 상수 값		
이름	10진수 값	16진수 값
MQADOPT_TYPE_NO	0	X'00000000'
MQADOPT_TYPE_ALL	1	X'00000001'
MQADOPT_TYPE_SVR	2	X'00000002'
MQADOPT_TYPE_SDR	4	X'00000004'
MQADOPT_TYPE_RCVR	8	X'00000008'
MQADOPT_TYPE_CLUSRCVR	16	X'00000010'

## MQAIR\_\*(인증 정보 레코드 구조)

표 17. 상수 구조	
이름	구조
MQAIR_STRUC_ID	"AIR~"
MQAIR_STRUC_ID_ARRAY	'A', 'I', 'R', '~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 18. 상수 값		
이름	10진수 값	16진수 값
MQAIR_VERSION_1	1	X'00000001'
MQAIR_VERSION_2	2	X'00000002'
MQAIR_CURRENT_VERSION	2	X'00000002'

## MQAIT\_\*(인증 정보 유형)

표 19. 상수 값		
이름	10진수 값	16진수 값
MQAIT_ALL	0	X'00000000'

표 19. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQAIT_CRL_LDAP	1	X'00000001'
MQAIT_OCSP	2	X'00000002'
MQAIT_IDPW_OS	3	X'00000003'
MQAIT_IDPW_LDAP	4	X'00000004'

### MQAS\_\*(명령 형식 비동기 상태 값)

표 20. 상수 값		
이름	10진수 값	16진수 값
MQAS_NONE	0	X'00000000'
MQAS_STARTED	1	X'00000001'
MQAS_START_WAIT	2	X'00000002'
MQAS_STOPPED	3	X'00000003'
MQAS_SUSPENDED	4	X'00000004'
MQAS_SUSPENDED_TEMPORARY	5	X'00000005'
MQAS_ACTIVE	6	X'00000006'
MQAS_INACTIVE	7	X'00000007'

### MQAT\_\*(Put 애플리케이션 유형)

표 21. 상수 값		
이름	10진수 값	16진수 값
MQAT_UNKNOWN	-1	X'FFFFFFFF'
MQAT_NO_CONTEXT	0	X'00000000'
MQAT_CICS	1	X'00000001'
MQAT_MVS	2	X'00000002'
MQAT_OS390	2	X'00000002'
MQAT_ZOS	2	X'00000002'
MQAT_IMS	3	X'00000003'
MQAT_OS2	4	X'00000004'
MQAT_DOS	5	X'00000005'
MQAT_AIX	6	X'00000006'
MQAT_UNIX	6	X'00000006'
MQAT_QMGR	7	X'00000007'
MQAT_OS400	8	X'00000008'
MQAT_WINDOWS	9	X'00000009'
MQAT_CICS_VSE	10	X'0000000A'
MQAT_WINDOWS_NT	11	X'0000000B'
MQAT_VMS	12	X'0000000C'
MQAT_GUARDIAN	13	X'0000000D'
MQAT_NSK	13	X'0000000D'
MQAT_VOS	14	X'0000000E'

표 21. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQAT_OPEN_TP1	15	X'0000000F'
MQAT_VM	18	X'00000012'
MQAT_IMS_BRIDGE	19	X'00000013'
MQAT_XCF	20	X'00000014'
MQAT_CICS_BRIDGE	21	X'00000015'
MQAT_NOTES_AGENT	22	X'00000016'
MQAT_TPF	23	X'00000017'
MQAT_USER	25	X'00000019'
MQAT_BROKER	26	X'0000001A'
MQAT_QMGR_PUBLISH	26	X'0000001A'
MQAT_JAVA	28	X'0000001C'
MQAT_DQM	29	X'0000001D'
MQAT_CHANNEL_INITIATOR	30	X'0000001E'
MQAT_WLM	31	X'0000001F'
MQAT_BATCH	32	X'00000020'
MQAT_RRS_BATCH	33	X'00000021'
MQAT_SIB	34	X'00000022'
MQAT_DEFAULT	(value differs by platform or version)	(value differs by platform or version)
MQAT_USER_FIRST	65536	X'00010000'
MQAT_USER_LAST	999999999	X'3B9AC9FF'

### MQAUTH\_\*(명령 형식 권한 값)

표 22. 상수 값		
이름	10진수 값	16진수 값
MQAUTH_NONE	0	X'00000000'
MQAUTH_ALT_USER_AUTHORITY	1	X'00000001'
MQAUTH_BROWSE	2	X'00000002'
MQAUTH_CHANGE	3	X'00000003'
MQAUTH_CLEAR	4	X'00000004'
MQAUTH_CONNECT	5	X'00000005'
MQAUTH_CREATE	6	X'00000006'
MQAUTH_DELETE	7	X'00000007'
MQAUTH_DISPLAY	8	X'00000008'
MQAUTH_INPUT	9	X'00000009'
MQAUTH_INQUIRE	10	X'0000000A'
MQAUTH_OUTPUT	11	X'0000000B'
MQAUTH_PASS_ALL_CONTEXT	12	X'0000000C'
MQAUTH_PASS_IDENTITY_CONTEXT	13	X'0000000D'
MQAUTH_SET	14	X'0000000E'



표 22. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQAUTH_SET_ALL_CONTEXT	15	X'0000000F'
MQAUTH_SET_IDENTITY_CONTEXT	16	X'00000010'
MQAUTH_CONTROL	17	X'00000011'
MQAUTH_CONTROL_EXTENDED	18	X'00000012'
MQAUTH_PUBLISH	19	X'00000013'
MQAUTH_SUBSCRIBE	20	X'00000014'
MQAUTH_RESUME	21	X'00000015'
MQAUTH_SYSTEM	22	X'00000016'

### MQAUTHOPT\_\*(명령 형식 권한 옵션)

표 23. 상수 값		
이름	10진수 값	16진수 값
MQAUTHOPT_CUMULATIVE	256	X'00000100'
MQAUTHOPT_ENTITY_EXPLICIT	1	X'00000001'
MQAUTHOPT_ENTITY_SET	2	X'00000002'
MQAUTHOPT_NAME_ALL_MATCHING	32	X'00000020'
MQAUTHOPT_NAME_AS_WILDCARD	64	X'00000040'
MQAUTHOPT_NAME_EXPLICIT	16	X'00000010'

### MQAXC\_\*(API 엑시트 컨텍스트 구조)

표 24. 상수 구조	
이름	구조
MQAXC_STRUC_ID	"AXC~"
MQAXC_STRUC_ID_ARRAY	'A', 'X', 'C', '~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 25. 상수 값		
이름	10진수 값	16진수 값
MQAXC_VERSION_1	1	X'00000001'
MQAXC_CURRENT_VERSION	1	X'00000001'

### MQAXP\_\*(API 엑시트 매개변수 구조)

표 26. 상수 구조	
이름	구조
MQAXP_STRUC_ID	"AXP~"
MQAXP_STRUC_ID_ARRAY	'A', 'X', 'P', '~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 27. 상수 값		
이름	10진수 값	16진수 값
MQAXP_VERSION_1	1	X'00000001'
MQAXP_VERSION_2	2	X'00000002'
MQAXP_CURRENT_VERSION	2	X'00000002'

### MQBA\_\*(바이트 속성 선택자)

표 28. 상수 값		
이름	10진수 값	16진수 값
MQBA_FIRST	6001	X'00001771'
MQBA_LAST	8000	X'00001F40'

### MQBACF\_\*(명령 형식 바이트 매개변수 유형)

표 29. 상수 값		
이름	10진수 값	16진수 값
MQBACF_FIRST	7001	X'00001B59'
MQBACF_EVENT_ACCOUNTING_TOKEN	7001	X'00001B59'
MQBACF_EVENT_SECURITY_ID	7002	X'00001B5A'
MQBACF_RESPONSE_SET	7003	X'00001B5B'
MQBACF_RESPONSE_ID	7004	X'00001B5C'
MQBACF_EXTERNAL_UOW_ID	7005	X'00001B5D'
MQBACF_CONNECTION_ID	7006	X'00001B5E'
MQBACF_GENERIC_CONNECTION_ID	7007	X'00001B5F'
MQBACF_ORIGIN_UOW_ID	7008	X'00001B60'
MQBACF_Q_MGR_UOW_ID	7009	X'00001B61'
MQBACF_ACCOUNTING_TOKEN	7010	X'00001B62'
MQBACF_CORREL_ID	7011	X'00001B63'
MQBACF_GROUP_ID	7012	X'00001B64'
MQBACF_MSG_ID	7013	X'00001B65'
MQBACF_CF_LEID	7014	X'00001B66'
MQBACF_DESTINATION_CORREL_ID	7015	X'00001B67'
MQBACF_SUB_ID	7016	X'00001B68'
MQBACF_LAST_USED	7016	X'00001B68'

### MQBL\_\*(mqAddString 및 mqSetString의 버퍼 길이)

표 30. 상수 값		
이름	10진수 값	16진수 값
MQBL_NULL_TERMINATED	-1	X'FFFFFFFF'

## MQBMHO\_\*(버퍼 대 메시지 핸들 옵션 및 구조)

### 버퍼 대 메시지 핸들 옵션 구조

표 31. 상수 구조	
이름	구조
MQBMHO_STRUC_ID	"BMHO"
MQBMHO_STRUC_ID_ARRAY	'B', 'M', 'H', 'O'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 32. 상수 값		
이름	10진수 값	16진수 값
MQBMHO_VERSION_1	1	X'00000001'
MQBMHO_CURRENT_VERSION	1	X'00000001'

### 버퍼 대 메시지 핸들 옵션

표 33. 상수 값		
이름	10진수 값	16진수 값
MQBMHO_NONE	0	X'00000000'
MQBMHO_DELETE_PROPERTIES	1	X'00000001'

## MQBND\_\*(기본 바인딩)

표 34. 상수 값		
이름	10진수 값	16진수 값
MQBND_BIND_ON_OPEN	0	X'00000000'
MQBND_BIND_NOT_FIXED	1	X'00000001'
MQBND_BIND_ON_GROUP	2	X'00000002'

## MQBO\_\*(시작 옵션 및 구조)

### 시작 옵션 구조

표 35. 상수 구조	
이름	구조
MQBO_STRUC_ID	"BO~"
MQBO_STRUC_ID_ARRAY	'B', 'O', '~', '~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 36. 상수 값		
이름	10진수 값	16진수 값
MQBO_VERSION_1	1	X'00000001'
MQBO_CURRENT_VERSION	1	X'00000001'

## 시작 옵션

표 37. 상수 값		
이름	10진수 값	16진수 값
MQBO_NONE	0	X'00000000'

## MQBT\_\*(명령 형식 브릿지 유형)

표 38. 상수 값		
이름	10진수 값	16진수 값
MQBT_OTMA	1	X'00000001'

## MQCA\_\*(문자 속성 선택자)

표 39. 상수 값		
이름	10진수 값	16진수 값
MQCA_ADMIN_TOPIC_NAME	2105	X'00000839'
MQCA_ALTERATION_DATE	2027	X'000007EB'
MQCA_ALTERATION_TIME	2028	X'000007EC'
MQCA_APPL_ID	2001	X'000007D1'
MQCA_AUTH_INFO_CONN_NAME	2053	X'00000805'
MQCA_AUTH_INFO_DESC	2046	X'000007FE'
MQCA_AUTH_INFO_NAME	2045	X'000007FD'
MQCA_AUTH_INFO_OCSP_URL	2109	X'0000083D'
MQCA_AUTO_REORG_CATALOG	2091	X'0000082B'
MQCA_AUTO_REORG_START_TIME	2090	X'0000082A'
MQCA_BACKOUT_REQ_Q_NAME	2019	X'000007E3'
MQCA_BASE_OBJECT_NAME	2002	X'000007D2'
MQCA_BASE_Q_NAME	2002	X'000007D2'
MQCA_BATCH_INTERFACE_ID	2068	X'00000814'
MQCA_CF_STRUC_DESC	2052	X'00000804'
MQCA_CF_STRUC_NAME	2039	X'000007F7'
MQCA_CHANNEL_AUTO_DEF_EXIT	2026	X'000007EA'
MQCA_CHILD	2101	X'00000835'
MQCA_CHINIT_SERVICE_PARM	2076	X'0000081C'
MQCA_CICS_FILE_NAME	2060	X'0000080C'
MQCA_CLUS_CHL_NAME	2124	X'0000084C'
MQCA_CLUSTER_DATE	2037	X'000007F5'
MQCA_CLUSTER_NAME	2029	X'000007ED'
MQCA_CLUSTER_NAMELIST	2030	X'000007EE'
MQCA_CLUSTER_Q_MGR_NAME	2031	X'000007EF'
MQCA_CLUSTER_TIME	2038	X'000007F6'
MQCA_CLUSTER_WORKLOAD_DATA	2034	X'000007F2'
MQCA_CLUSTER_WORKLOAD_EXIT	2033	X'000007F1'

표 39. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCA_COMMAND_INPUT_Q_NAME	2003	X'000007D3'
MQCA_COMMAND_REPLY_Q_NAME	2067	X'00000813'
MQCA_CREATION_DATE	2004	X'000007D4'
MQCA_CREATION_TIME	2005	X'000007D5'
MQCA_DEAD_LETTER_Q_NAME	2006	X'000007D6'
MQCA_DEF_XMIT_Q_NAME	2025	X'000007E9'
MQCA_DNS_GROUP	2071	X'00000817'
MQCA_ENV_DATA	2007	X'000007D7'
MQCA_FIRST	2001	X'000007D1'
MQCA_IGQ_USER_ID	2041	X'000007F9'
MQCA_INITIATION_Q_NAME	2008	X'000007D8'
MQCA_LAST	4000	X'00000FA0'
MQCA_LAST_USED	2109	X'0000083D'
MQCA_LDAP_PASSWORD	2048	X'00000800'
MQCA_LDAP_USER_NAME	2047	X'000007FF'
MQCA_LU_GROUP_NAME	2072	X'00000818'
MQCA_LU_NAME	2073	X'00000819'
MQCA_LU62_ARM_SUFFIX	2074	X'0000081A'
MQCA_MODEL_DURABLE_Q	2096	X'00000830'
MQCA_MODEL_NON_DURABLE_Q	2097	X'00000831'
MQCA_MONITOR_Q_NAME	2066	X'00000812'
MQCA_NAMELIST_DESC	2009	X'000007D9'
MQCA_NAMELIST_NAME	2010	X'000007DA'
MQCA_NAMES	2020	X'000007E4'
MQCA_PARENT	2102	X'00000836'
MQCA_PASS_TICKET_APPL	2086	X'00000826'
MQCA_PROCESS_DESC	2011	X'000007DB'
MQCA_PROCESS_NAME	2012	X'000007DC'
MQCA_Q_DESC	2013	X'000007DD'
MQCA_Q_MGR_DESC	2014	X'000007DE'
MQCA_Q_MGR_IDENTIFIER	2032	X'000007F0'
MQCA_Q_MGR_NAME	2015	X'000007DF'
MQCA_Q_NAME	2016	X'000007E0'
MQCA_QSG_NAME	2040	X'000007F8'
MQCA_REMOTE_Q_MGR_NAME	2017	X'000007E1'
MQCA_REMOTE_Q_NAME	2018	X'000007E2'
MQCA_REPOSITORY_NAME	2035	X'000007F3'
MQCA_REPOSITORY_NAMELIST	2036	X'000007F4'
MQCA_RESUME_DATE	2098	X'00000832'

표 39. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCA_RESUME_TIME	2099	X'00000833'
MQCA_SERVICE_DESC	2078	X'0000081E'
MQCA_SERVICE_NAME	2077	X'0000081D'
MQCA_SERVICE_START_ARGS	2080	X'00000820'
MQCA_SERVICE_START_COMMAND	2079	X'0000081F'
MQCA_SERVICE_STOP_ARGS	2082	X'00000822'
MQCA_SERVICE_STOP_COMMAND	2081	X'00000821'
MQCA_STDERR_DESTINATION	2084	X'00000824'
MQCA_STDOUT_DESTINATION	2083	X'00000823'
MQCA_SSL_CRL_NAMELIST	2050	X'00000802'
MQCA_SSL_CRYPTO_HARDWARE	2051	X'00000803'
MQCA_SSL_KEY_LIBRARY	2069	X'00000815'
MQCA_SSL_KEY_MEMBER	2070	X'00000816'
MQCA_SSL_KEY_REPOSITORY	2049	X'00000801'
MQCA_STORAGE_CLASS	2022	X'000007E6'
MQCA_STORAGE_CLASS_DESC	2042	X'000007FA'
MQCA_SYSTEM_LOG_Q_NAME	2065	X'00000811'
MQCA_TCP_NAME	2075	X'0000081B'
MQCA_TOPIC_DESC	2093	X'0000082D'
MQCA_TOPIC_NAME	2092	X'0000082C'
MQCA_TOPIC_STRING_FILTER	2108	X'0000083C'
MQCA_TOPIC_STRING	2094	X'0000082E'
MQCA_TPIPE_NAME	2085	X'00000825'
MQCA_TRIGGER_CHANNEL_NAME	2064	X'00000810'
MQCA_TRIGGER_DATA	2023	X'000007E7'
MQCA_TRIGGER_PROGRAM_NAME	2062	X'0000080E'
MQCA_TRIGGER_TERM_ID	2063	X'0000080F'
MQCA_TRIGGER_TRANS_ID	2061	X'0000080D'
MQCA_USER_DATA	2021	X'000007E5'
MQCA_USER_LIST	4000	X'00000FA0'
MQCA_VERSION	2120	X'00000848'
MQCA_XCF_GROUP_NAME	2043	X'000007FB'
MQCA_XCF_MEMBER_NAME	2044	X'000007FC'
MQCA_XMIT_Q_NAME	2024	X'000007E8'

### MQCACF\_\*(명령 형식 문자 매개변수 유형)

표 40. 상수 값		
이름	10진수 값	16진수 값
MQCACF_FIRST	3001	X'00000BB9'

표 40. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCACF_FROM_Q_NAME	3001	X'00000BB9'
MQCACF_TO_Q_NAME	3002	X'00000BBA'
MQCACF_FROM_PROCESS_NAME	3003	X'00000BBB'
MQCACF_TO_PROCESS_NAME	3004	X'00000BBC'
MQCACF_FROM_NAMELIST_NAME	3005	X'00000BBD'
MQCACF_TO_NAMELIST_NAME	3006	X'00000BBE'
MQCACF_FROM_CHANNEL_NAME	3007	X'00000BBF'
MQCACF_TO_CHANNEL_NAME	3008	X'00000BC0'
MQCACF_FROM_AUTH_INFO_NAME	3009	X'00000BC1'
MQCACF_TO_AUTH_INFO_NAME	3010	X'00000BC2'
MQCACF_Q_NAMES	3011	X'00000BC3'
MQCACF_PROCESS_NAMES	3012	X'00000BC4'
MQCACF_NAMELIST_NAMES	3013	X'00000BC5'
MQCACF_ESCAPE_TEXT	3014	X'00000BC6'
MQCACF_LOCAL_Q_NAMES	3015	X'00000BC7'
MQCACF_MODEL_Q_NAMES	3016	X'00000BC8'
MQCACF_ALIAS_Q_NAMES	3017	X'00000BC9'
MQCACF_REMOTE_Q_NAMES	3018	X'00000BCA'
MQCACF_SENDER_CHANNEL_NAMES	3019	X'00000BCB'
MQCACF_SERVER_CHANNEL_NAMES	3020	X'00000BCC'
MQCACF_REQUESTER_CHANNEL_NAMES	3021	X'00000BCD'
MQCACF_RECEIVER_CHANNEL_NAMES	3022	X'00000BCE'
MQCACF_OBJECT_Q_MGR_NAME	3023	X'00000BCF'
MQCACF_APPL_NAME	3024	X'00000BD0'
MQCACF_USER_IDENTIFIER	3025	X'00000BD1'
MQCACF_AUX_ERROR_DATA_STR_1	3026	X'00000BD2'
MQCACF_AUX_ERROR_DATA_STR_2	3027	X'00000BD3'
MQCACF_AUX_ERROR_DATA_STR_3	3028	X'00000BD4'
MQCACF_BRIDGE_NAME	3029	X'00000BD5'
MQCACF_STREAM_NAME	3030	X'00000BD6'
MQCACF_TOPIC	3031	X'00000BD7'
MQCACF_PARENT_Q_MGR_NAME	3032	X'00000BD8'
MQCACF_CORREL_ID	3033	X'00000BD9'
MQCACF_PUBLISH_TIMESTAMP	3034	X'00000BDA'
MQCACF_STRING_DATA	3035	X'00000BDB'
MQCACF_SUPPORTED_STREAM_NAME	3036	X'00000BDC'
MQCACF_REG_TOPIC	3037	X'00000BDD'
MQCACF_REG_TIME	3038	X'00000BDE'
MQCACF_REG_USER_ID	3039	X'00000BDF'

표 40. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCACF_CHILD_Q_MGR_NAME	3040	X'00000BE0'
MQCACF_REG_STREAM_NAME	3041	X'00000BE1'
MQCACF_REG_Q_MGR_NAME	3042	X'00000BE2'
MQCACF_REG_Q_NAME	3043	X'00000BE3'
MQCACF_REG_CORREL_ID	3044	X'00000BE4'
MQCACF_EVENT_USER_ID	3045	X'00000BE5'
MQCACF_OBJECT_NAME	3046	X'00000BE6'
MQCACF_EVENT_Q_MGR	3047	X'00000BE7'
MQCACF_AUTH_INFO_NAMES	3048	X'00000BE8'
MQCACF_EVENT_APPL_IDENTITY	3049	X'00000BE9'
MQCACF_EVENT_APPL_NAME	3050	X'00000BEA'
MQCACF_EVENT_APPL_ORIGIN	3051	X'00000BEB'
MQCACF_SUBSCRIPTION_NAME	3052	X'00000BEC'
MQCACF_REG_SUB_NAME	3053	X'00000BED'
MQCACF_SUBSCRIPTION_IDENTITY	3054	X'00000BEE'
MQCACF_REG_SUB_IDENTITY	3055	X'00000BEF'
MQCACF_SUBSCRIPTION_USER_DATA	3056	X'00000BF0'
MQCACF_REG_SUB_USER_DATA	3057	X'00000BF1'
MQCACF_APPL_TAG	3058	X'00000BF2'
MQCACF_DATA_SET_NAME	3059	X'00000BF3'
MQCACF_UOW_START_DATE	3060	X'00000BF4'
MQCACF_UOW_START_TIME	3061	X'00000BF5'
MQCACF_UOW_LOG_START_DATE	3062	X'00000BF6'
MQCACF_UOW_LOG_START_TIME	3063	X'00000BF7'
MQCACF_UOW_LOG_EXTENT_NAME	3064	X'00000BF8'
MQCACF_PRINCIPAL_ENTITY_NAMES	3065	X'00000BF9'
MQCACF_GROUP_ENTITY_NAMES	3066	X'00000BFA'
MQCACF_AUTH_PROFILE_NAME	3067	X'00000BFB'
MQCACF_ENTITY_NAME	3068	X'00000BFC'
MQCACF_SERVICE_COMPONENT	3069	X'00000BFD'
MQCACF_RESPONSE_Q_MGR_NAME	3070	X'00000BFE'
MQCACF_CURRENT_LOG_EXTENT_NAME	3071	X'00000BFF'
MQCACF_RESTART_LOG_EXTENT_NAME	3072	X'00000C00'
MQCACF_MEDIA_LOG_EXTENT_NAME	3073	X'00000C01'
MQCACF_LOG_PATH	3074	X'00000C02'
MQCACF_COMMAND_MQSC	3075	X'00000C03'
MQCACF_Q_MGR_CPF	3076	X'00000C04'
MQCACF_USAGE_LOG_RBA	3078	X'00000C06'
MQCACF_USAGE_LOG_LRSN	3079	X'00000C07'



표 40. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCACF_COMMAND_SCOPE	3080	X'00000C08'
MQCACF_ASID	3081	X'00000C09'
MQCACF_PSB_NAME	3082	X'00000C0A'
MQCACF_PST_ID	3083	X'00000C0B'
MQCACF_TASK_NUMBER	3084	X'00000C0C'
MQCACF_TRANSACTION_ID	3085	X'00000C0D'
MQCACF_Q_MGR_UOW_ID	3086	X'00000C0E'
MQCACF_ORIGIN_NAME	3088	X'00000C10'
MQCACF_ENV_INFO	3089	X'00000C11'
MQCACF_SECURITY_PROFILE	3090	X'00000C12'
MQCACF_CONFIGURATION_DATE	3091	X'00000C13'
MQCACF_CONFIGURATION_TIME	3092	X'00000C14'
MQCACF_FROM_CF_STRUC_NAME	3093	X'00000C15'
MQCACF_TO_CF_STRUC_NAME	3094	X'00000C16'
MQCACF_CF_STRUC_NAMES	3095	X'00000C17'
MQCACF_FAIL_DATE	3096	X'00000C18'
MQCACF_FAIL_TIME	3097	X'00000C19'
MQCACF_BACKUP_DATE	3098	X'00000C1A'
MQCACF_BACKUP_TIME	3099	X'00000C1B'
MQCACF_SYSTEM_NAME	3100	X'00000C1C'
MQCACF_CF_STRUC_BACKUP_START	3101	X'00000C1D'
MQCACF_CF_STRUC_BACKUP_END	3102	X'00000C1E'
MQCACF_CF_STRUC_LOG_Q_MGRS	3103	X'00000C1F'
MQCACF_FROM_STORAGE_CLASS	3104	X'00000C20'
MQCACF_TO_STORAGE_CLASS	3105	X'00000C21'
MQCACF_STORAGE_CLASS_NAMES	3106	X'00000C22'
MQCACF_DSG_NAME	3108	X'00000C24'
MQCACF_DB2_NAME	3109	X'00000C25'
MQCACF_SYSP_CMD_USER_ID	3110	X'00000C26'
MQCACF_SYSP_OTMA_GROUP	3111	X'00000C27'
MQCACF_SYSP_OTMA_MEMBER	3112	X'00000C28'
MQCACF_SYSP_OTMA_DRU_EXIT	3113	X'00000C29'
MQCACF_SYSP_OTMA_TPIPE_PFX	3114	X'00000C2A'
MQCACF_SYSP_ARCHIVE_PFX1	3115	X'00000C2B'
MQCACF_SYSP_ARCHIVE_UNIT1	3116	X'00000C2C'
MQCACF_SYSP_LOG_CORREL_ID	3117	X'00000C2D'
MQCACF_SYSP_UNIT_VOLSER	3118	X'00000C2E'
MQCACF_SYSP_Q_MGR_TIME	3119	X'00000C2F'
MQCACF_SYSP_Q_MGR_DATE	3120	X'00000C30'

표 40. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCACF_SYSP_Q_MGR_RBA	3121	X'00000C31'
MQCACF_SYSP_LOG_RBA	3122	X'00000C32'
MQCACF_SYSP_SERVICE	3123	X'00000C33'
MQCACF_FROM_LISTENER_NAME	3124	X'00000C34'
MQCACF_TO_LISTENER_NAME	3125	X'00000C35'
MQCACF_FROM_SERVICE_NAME	3126	X'00000C36'
MQCACF_TO_SERVICE_NAME	3127	X'00000C37'
MQCACF_LAST_PUT_DATE	3128	X'00000C38'
MQCACF_LAST_PUT_TIME	3129	X'00000C39'
MQCACF_LAST_GET_DATE	3130	X'00000C3A'
MQCACF_LAST_GET_TIME	3131	X'00000C3B'
MQCACF_OPERATION_DATE	3132	X'00000C3C'
MQCACF_OPERATION_TIME	3133	X'00000C3D'
MQCACF_ACTIVITY_DESC	3134	X'00000C3E'
MQCACF_APPL_IDENTITY_DATA	3135	X'00000C3F'
MQCACF_APPL_ORIGIN_DATA	3136	X'00000C40'
MQCACF_PUT_DATE	3137	X'00000C41'
MQCACF_PUT_TIME	3138	X'00000C42'
MQCACF_REPLY_TO_Q	3139	X'00000C43'
MQCACF_REPLY_TO_Q_MGR	3140	X'00000C44'
MQCACF_RESOLVED_Q_NAME	3141	X'00000C45'
MQCACF_STRUC_ID	3142	X'00000C46'
MQCACF_VALUE_NAME	3143	X'00000C47'
MQCACF_SERVICE_START_DATE	3144	X'00000C48'
MQCACF_SERVICE_START_TIME	3145	X'00000C49'
MQCACF_SYSP_OFFLINE_RBA	3146	X'00000C4A'
MQCACF_SYSP_ARCHIVE_PFX2	3147	X'00000C4B'
MQCACF_SYSP_ARCHIVE_UNIT2	3148	X'00000C4C'
MQCACF_TO_TOPIC_NAME	3149	X'00000C4D'
MQCACF_FROM_TOPIC_NAME	3150	X'00000C4E'
MQCACF_TOPIC_NAMES	3151	X'00000C4F'
MQCACF_SUB_NAME	3152	X'00000C50'
MQCACF_DESTINATION_Q_MGR	3153	X'00000C51'
MQCACF_DESTINATION	3154	X'00000C52'
MQCACF_SUB_USER_ID	3156	X'00000C54'
MQCACF_SUB_USER_DATA	3159	X'00000C57'
MQCACF_SUB_SELECTOR	3160	X'00000C58'
MQCACF_LAST_PUB_DATE	3161	X'00000C59'
MQCACF_LAST_PUB_TIME	3162	X'00000C5A'

표 40. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCACF_FROM_SUB_NAME	3163	X'00000C5B'
MQCACF_TO_SUB_NAME	3164	X'00000C5C'
MQCACF_LAST_MSG_TIME	3167	X'00000C5F'
MQCACF_LAST_MSG_DATE	3168	X'00000C60'
MQCACF_SUBSCRIPTION_POINT	3169	X'00000C61'
MQCACF_FILTER	3170	X'00000C62'
MQCACF_NONE	3171	X'00000C63'
MQCACF_ADMIN_TOPIC_NAMES	3172	X'00000C64'
MQCACF_ROUTING_FINGER_PRINT	3173	X'00000C65'
MQCACF_APPL_DESC	3174	X'00000C66'
MQCACF_Q_MGR_START_DATE	3175	X'00000C67'
MQCACF_Q_MGR_START_TIME	3176	X'00000C68'
MQCACF_FROM_COMM_INFO_NAME	3177	X'00000C69'
MQCACF_TO_COMM_INFO_NAME	3178	X'00000C6A'
MQCACF_CF_OFFLOAD_SIZE1	3179	X'00000C6B'
MQCACF_CF_OFFLOAD_SIZE2	3180	X'00000C6C'
MQCACF_CF_OFFLOAD_SIZE3	3181	X'00000C6D'
MQCACF_CF_SMDG_GENERIC_NAME	3182	X'00000C6E'
MQCACF_CF_SMDG	3183	X'00000C6F'
MQCACF_RECOVERY_DATE	3184	X'00000C70'
MQCACF_RECOVERY_TIME	3185	X'00000C71'
MQCACF_CF_SMDGCONN	3186	X'00000C72'
MQCACF_CF_STRUC_NAME	3187	X'00000C73'
MQCACF_ALTERNATE_USERID	3188	X'00000C74'
MQCACF_CHAR_ATTRS	3189	X'00000C75'
MQCACF_DYNAMIC_Q_NAME	3190	X'00000C76'
MQCACF_HOST_NAME	3191	X'00000C77'
MQCACF_MQCB_NAME	3192	X'00000C78'
MQCACF_OBJECT_STRING	3193	X'00000C79'
MQCACF_RESOLVED_LOCAL_Q_MGR	3194	X'00000C7A'
MQCACF_RESOLVED_LOCAL_Q_NAME	3195	X'00000C7B'
MQCACF_RESOLVED_OBJECT_STRING	3196	X'00000C7C'
MQCACF_RESOLVED_Q_MGR	3197	X'00000C7D'
MQCACF_SELECTION_STRING	3198	X'00000C7E'
MQCACF_XA_INFO	3199	X'00000C7F'
MQCACF_APPL_FUNCTION	3200	X'00000C80'
MQCACF_XQH_REMOTE_Q_NAME	3201	X'00000C81'
MQCACF_XQH_REMOTE_Q_MGR	3202	X'00000C82'
MQCACF_XQH_PUT_TIME	3203	X'00000C83'

표 40. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCACF_XQH_PUT_DATE	3204	X'00000C84'
MQCACF_EXCL_OPERATOR_MESSAGES	3205	X'00000C85'
MQCACF_CSP_USER_IDENTIFIER	3206	X'00000C86'
MQCACF_AMQP_CLIENT_ID	3207	X'00000C87'
MQCACF_ARCHIVE_LOG_EXTENT_NAME	3208	X'00000C88'
MQCACF_APPL_IMMOVABLE_DATE	3209	X'00000C89'
MQCACF_APPL_IMMOVABLE_TIME	3210	X'00000C8A'
MQCACF_NHA_INSTANCE_NAME	3211	X'00000C8B'
MQCACF_LAST_USED	3211	X'00000C8B'

### MQCACH\_\*(명령 형식 문자 채널 매개변수 유형)

표 41. 상수 값		
이름	10진수 값	16진수 값
MQCACH_FIRST	3501	X'00000DAD'
MQCACH_CHANNEL_NAME	3501	X'00000DAD'
MQCACH_DESC	3502	X'00000DAE'
MQCACH_MODE_NAME	3503	X'00000DAF'
MQCACH_TP_NAME	3504	X'00000DB0'
MQCACH_XMIT_Q_NAME	3505	X'00000DB1'
MQCACH_CONNECTION_NAME	3506	X'00000DB2'
MQCACH_MCA_NAME	3507	X'00000DB3'
MQCACH_SEC_EXIT_NAME	3508	X'00000DB4'
MQCACH_MSG_EXIT_NAME	3509	X'00000DB5'
MQCACH_SEND_EXIT_NAME	3510	X'00000DB6'
MQCACH_RCV_EXIT_NAME	3511	X'00000DB7'
MQCACH_CHANNEL_NAMES	3512	X'00000DB8'
MQCACH_SEC_EXIT_USER_DATA	3513	X'00000DB9'
MQCACH_MSG_EXIT_USER_DATA	3514	X'00000DBA'
MQCACH_SEND_EXIT_USER_DATA	3515	X'00000DBB'
MQCACH_RCV_EXIT_USER_DATA	3516	X'00000DBC'
MQCACH_USER_ID	3517	X'00000DBD'
MQCACH_PASSWORD	3518	X'00000DBE'
MQCACH_LOCAL_ADDRESS	3520	X'00000DC0'
MQCACH_LOCAL_NAME	3521	X'00000DC1'
MQCACH_LAST_MSG_TIME	3524	X'00000DC4'
MQCACH_LAST_MSG_DATE	3525	X'00000DC5'
MQCACH_MCA_USER_ID	3527	X'00000DC7'
MQCACH_CHANNEL_START_TIME	3528	X'00000DC8'
MQCACH_CHANNEL_START_DATE	3529	X'00000DC9'

표 41. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCACH_MCA_JOB_NAME	3530	X'00000DCA'
MQCACH_LAST_LUWID	3531	X'00000DCB'
MQCACH_CURRENT_LUWID	3532	X'00000DCC'
MQCACH_FORMAT_NAME	3533	X'00000DCD'
MQCACH_MR_EXIT_NAME	3534	X'00000DCE'
MQCACH_MR_EXIT_USER_DATA	3535	X'00000DCF'
MQCACH_SSL_CIPHER_SPEC	3544	X'00000DD8'
MQCACH_SSL_PEER_NAME	3545	X'00000DD9'
MQCACH_SSL_HANDSHAKE_STAGE	3546	X'00000DDA'
MQCACH_SSL_SHORT_PEER_NAME	3547	X'00000ddb'
MQCACH_REMOTE_APPL_TAG	3548	X'00000DDC'
MQCACH_SSL_CERT_USER_ID	3549	X'00000DDD'
MQCACH_SSL_CERT_ISSUER_NAME	3550	X'00000DDE'
MQCACH_LU_NAME	3551	X'00000DDF'
MQCACH_IP_ADDRESS	3552	X'00000DE0'
MQCACH_TCP_NAME	3553	X'00000DE1'
MQCACH_LISTENER_NAME	3554	X'00000DE2'
MQCACH_LISTENER_DESC	3555	X'00000DE3'
MQCACH_LISTENER_START_DATE	3556	X'00000DE4'
MQCACH_LISTENER_START_TIME	3557	X'00000DE5'
MQCACH_SSL_KEY_RESET_DATE	3558	X'00000DE6'
MQCACH_SSL_KEY_RESET_TIME	3559	X'00000DE7'
MQCACH_LAST_USED	3559	X'00000DE7'

### MQCADSD\_\*(CICS 정보 헤더 ADS 디스크립터)

표 42. 상수 값		
이름	10진수 값	16진수 값
MQCADSD_NONE	0	X'00000000'
MQCADSD_SEND	1	X'00000001'
MQCADSD_RECV	16	X'00000010'
MQCADSD_MSGFORMAT	256	X'00000100'

### MQCAFTY\_\*(연결 연관관계 값)

표 43. 상수 값		
이름	10진수 값	16진수 값
MQCAFTY_NONE	0	X'00000000'
MQCAFTY_PREFERRED	1	X'00000001'

## MQCAMO\_\*(명령 형식 문자 모니터링 매개변수 유형)

표 44. 상수 값		
이름	10진수 값	16진수 값
MQCAMO_FIRST	2701	X'00000A8D'
MQCAMO_CLOSE_DATE	2701	X'00000A8D'
MQCAMO_CLOSE_TIME	2702	X'00000A8E'
MQCAMO_CONN_DATE	2703	X'00000A8F'
MQCAMO_CONN_TIME	2704	X'00000A90'
MQCAMO_DISC_DATE	2705	X'00000A91'
MQCAMO_DISC_TIME	2706	X'00000A92'
MQCAMO_END_DATE	2707	X'00000A93'
MQCAMO_END_TIME	2708	X'00000A94'
MQCAMO_OPEN_DATE	2709	X'00000A95'
MQCAMO_OPEN_TIME	2710	X'00000A96'
MQCAMO_START_DATE	2711	X'00000A97'
MQCAMO_START_TIME	2712	X'00000A98'
MQCAMO_LAST_USED	2712	X'00000A98'

## MQCBC\_\*(MQCBC 상수 구조)

표 45. 상수 구조	
이름	구조
MQCBC_STRUC_ID	"CBC~"
MQCBC_STRUC_ID_ARRAY	'C','B','C','~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 46. 상수 값		
이름	10진수 값	16진수 값
MQCBC_VERSION_1	1	X'00000001'
MQCBC_CURRENT_VERSION	1	X'00000001'

## MQCBCF\_\*(MQCBC 상수 플래그)

표 47. 상수 값		
이름	10진수 값	16진수 값
MQCBCF_NONE	0	X'00000000'
MQCBCF_READA_BUFFER_EMPTY	1	X'00000001'

## MQCBCT\_\*(MQCBC 상수 콜백 유형)

표 48. 상수 값		
이름	10진수 값	16진수 값
MQCBCT_START_CALL	1	X'00000001'
MQCBCT_STOP_CALL	2	X'00000002'

표 48. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCBCT_REGISTER_CALL	3	X'00000003'
MQCBCT_DEREGISTER_CALL	4	X'00000004'
MQCBCT_EVENT_CALL	5	X'00000005'
MQCBCT_MSG_REMOVED	6	X'00000006'
MQCBCT_MSG_NOT_REMOVED	7	X'00000007'

### MQCBD\_\*(MQCBD 상수 구조)

표 49. 상수 구조	
이름	구조
MQCBD_STRUC_ID	"CBD~"
MQCBD_STRUC_ID_ARRAY	'C', 'B', 'D', '~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 50. 상수 값		
이름	10진수 값	16진수 값
MQCBD_VERSION_1	1	X'00000001'
MQCBD_CURRENT_VERSION	1	X'00000001'

### MQCBDO\_\*(MQCBD 상수 콜백 옵션)

표 51. 상수 값		
이름	10진수 값	16진수 값
MQCBDO_NONE	0	X'00000000'
MQCBDO_START_CALL	1	X'00000001'
MQCBDO_STOP_CALL	4	X'00000004'
MQCBDO_REGISTER_CALL	256	X'00000100'
MQCBDO_DEREGISTER_CALL	512	X'00000200'
MQCBDO_FAIL_IF QUIESCING	8192	X'00002000'

### MQCBO\_\*(mqCreateBag의 Create-Bag 옵션)

표 52. 상수 값		
이름	10진수 값	16진수 값
MQCBO_NONE	0	X'00000000'
MQCBO_USER_BAG	0	X'00000000'
MQCBO_ADMIN_BAG	1	X'00000001'
MQCBO_COMMAND_BAG	16	X'00000010'
MQCBO_SYSTEM_BAG	32	X'00000020'
MQCBO_GROUP_BAG	64	X'00000040'
MQCBO_LIST_FORM_ALLOWED	2	X'00000002'
MQCBO_LIST_FORM_INHIBITED	0	X'00000000'
MQCBO_REORDER_AS_REQUIRED	4	X'00000004'

표 52. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCBO_DO_NOT_REORDER	0	X'00000000'
MQCBO_CHECK_SELECTORS	8	X'00000008'
MQCBO_DO_NOT_CHECK_SELECTORS	0	X'00000000'

### MQCBT\_\*(MQCBD 상수, 이는 콜백 함수의 유형임)

표 53. 상수 값		
이름	10진수 값	16진수 값
MQCBT_MESSAGE_CONSUMER	1	X'00000001'
MQCBT_EVENT_HANDLER	2	X'00000002'

### MQCC\_\*(완료 코드)

표 54. 상수 값		
이름	10진수 값	16진수 값
MQCC_OK	0	X'00000000'
MQCC_WARNING	1	X'00000001'
MQCC_FAILED	2	X'00000002'
MQCC_UNKNOWN	-1	X'FFFFFFFF'

### MQCCSI\_\*(코드화 문자 세트 ID)

표 55. 상수 값		
이름	10진수 값	16진수 값
MQCCSI_UNDEFINED	0	X'00000000'
MQCCSI_DEFAULT	0	X'00000000'
MQCCSI_Q_MGR	0	X'00000000'
MQCCSI_INHERIT	-2	X'FFFFFFFE'
MQCCSI_EMBEDDED	-1	X'FFFFFFF'
MQCCSI_APPL	-3	X'FFFFFFFD'






### MQCCT\_\*(CICS 정보 헤더 대화식 태스크 옵션)

표 56. 상수 값		
이름	10진수 값	16진수 값
MQCCT_YES	1	X'00000001'
MQCCT_NO	0	X'00000000'

### MQCD\_\*(채널 정의 구조)

표 57. 상수 값		
이름	10진수 값	16진수 값
MQCD_VERSION_1	1	X'00000001'
MQCD_VERSION_2	2	X'00000002'
MQCD_VERSION_3	3	X'00000003'



표 57. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCD_VERSION_4	4	X'00000004'
MQCD_VERSION_5	5	X'00000005'
MQCD_VERSION_6	6	X'00000006'
MQCD_VERSION_7	7	X'00000007'
MQCD_VERSION_8	8	X'00000008'
MQCD_VERSION_9	9	X'00000009'
MQCD_VERSION_10	10	X'0000000A'
 MQCD_VERSION_11	11	X'0000000B'
 MQCD_CURRENT_VERSION	11	X'0000000B'
 MQCD_VERSION_12	12	X'0000000C'
 MQCD_CURRENT_VERSION	12	X'0000000C'
MQCD_LENGTH_4	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_5	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_6	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_7	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_8	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_9	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_10	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_11	(value differs by platform or version)	(value differs by platform or version)
 MQCD_LENGTH_12	(value differs by platform or version)	(value differs by platform or version)
MQCD_CURRENT_LENGTH	(value differs by platform or version)	(value differs by platform or version)

### MQCDC\_\*(채널 데이터 변환)

표 58. 상수 값		
이름	10진수 값	16진수 값
MQCDC_SENDER_CONVERSION	1	X'00000001'
MQCDC_NO_SENDER_CONVERSION	0	X'00000000'

### MQCERT\_\*(인증서 유효성 검증 정책 유형)

MQ_CERT_VAL_POLICY_DEFAULT	0	X'00000000'
MQ_CERT_VAL_POLICY_ANY	0	X'00000000'
MQ_CERT_VAL_POLICY_RFC5280	1	X'00000001'

V9.4.0	V9.4.0	MQ_CERT_VAL_POLICY_NONE	2	X'00000002'
--------	--------	-------------------------	---	-------------

### MQCF\_\*(기능 플래그)

표 59. 상수 값		
이름	10진수 값	16진수 값
MQCF_NONE	0	X'00000000'
MQCF_DIST_LISTS	1	X'00000001'

### MQCFAC\_\*(CICS 정보 헤더 기능)

표 60. 상수 이름 및 값	
이름	16진수 값
MQCFAC_NONE	X'00...00' (8닐)
MQCFAC_NONE_ARRAY	'\0', '\0', ... (8닐)

### MQCFBF\_\*(명령 형식 바이트 문자열 필터 매개변수 구조)

표 61. 상수 값		
이름	10진수 값	16진수 값
MQCFBF_STRUC_LENGTH_FIXED	20	X'00000014'

### MQCFBS\_\*(명령 형식 바이트 문자열 매개변수 구조)

표 62. 상수 값		
이름	10진수 값	16진수 값
MQCFBS_STRUC_LENGTH_FIXED	16	X'00000010'

### MQCF\_C\*(명령 형식 헤더 제어 옵션)

표 63. 상수 값		
이름	10진수 값	16진수 값
MQCF_C_LAST	1	X'00000001'
MQCF_C_NOT_LAST	0	X'00000000'

### MQCFGR\_\*(명령 형식 그룹 매개변수 구조)

표 64. 상수 값		
이름	10진수 값	16진수 값
MQCFGR_STRUC_LENGTH	16	X'00000010'

### MQCFH\_\*(명령 형식 헤더 구조)

표 65. 상수 값		
이름	10진수 값	16진수 값
MQCFH_STRUC_LENGTH	36	X'00000024'
MQCFH_VERSION_1	1	X'00000001'
MQCFH_VERSION_2	2	X'00000002'

표 65. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCFH_VERSION_3	3	X'00000003'
MQCFH_CURRENT_VERSION	3	X'00000003'

### MQCFIF\_\*(명령 형식 정수 필터 매개변수 구조)

표 66. 상수 값		
이름	10진수 값	16진수 값
MQCFIF_STRUC_LENGTH	20	X'00000014'

### MQCFIL\_\*(명령 형식 정수 목록 매개변수 구조)

표 67. 상수 값		
이름	10진수 값	16진수 값
MQCFIL_STRUC_LENGTH_FIXED	16	X'00000010'

### MQCFIL64\_\*(명령 형식 64비트 정수 목록 매개변수 구조)

표 68. 상수 값		
이름	10진수 값	16진수 값
MQCFIL64_STRUC_LENGTH_FIXED	16	X'00000010'

### MQCFIN\_\*(명령 형식 정수 매개변수 구조)

표 69. 상수 값		
이름	10진수 값	16진수 값
MQCFIN_STRUC_LENGTH	16	X'00000010'

### MQCFIN64\_\*(명령 형식 64비트 정수 매개변수 구조)

표 70. 상수 값		
이름	10진수 값	16진수 값
MQCFIN64_STRUC_LENGTH	24	X'00000018'

### MQCFO\_\*(명령 형식 저장소 새로 고치기 옵션 및 명령 형식 큐 제거 옵션)

#### 명령 형식 저장소 새로 고치기 옵션

표 71. 상수 값		
이름	10진수 값	16진수 값
MQCFO_REFRESH_REPOSITORY_YES	1	X'00000001'
MQCFO_REFRESH_REPOSITORY_NO	0	X'00000000'

#### 명령 형식 큐 제거 옵션

표 72. 상수 값		
이름	10진수 값	16진수 값
MQCFO_REMOVE_QUEUES_YES	1	X'00000001'

표 72. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCFO_REMOVE_QUEUES_NO	0	X'00000000'

### MQCFOP\_\*(명령 형식 필터 연산자)

표 73. 상수 값		
이름	10진수 값	16진수 값
MQCFOP_LESS	1	X'00000001'
MQCFOP_EQUAL	2	X'00000002'
MQCFOP_GREATER	4	X'00000004'
MQCFOP_NOT_LESS	6	X'00000006'
MQCFOP_NOT_EQUAL	5	X'00000005'
MQCFOP_NOT_GREATER	3	X'00000003'
MQCFOP_LIKE	18	X'00000012'
MQCFOP_NOT_LIKE	21	X'00000015'
MQCFOP_CONTAINS	10	X'0000000A'
MQCFOP_EXCLUDES	13	X'0000000D'
MQCFOP_CONTAINS_GEN	26	X'0000001A'
MQCFOP_EXCLUDES_GEN	29	X'0000001D'

### MQCFR\_\*(CF 복구 가능성)

표 74. 상수 값		
이름	10진수 값	16진수 값
MQCFR_YES	1	X'00000001'
MQCFR_NO	0	X'00000000'

### MQCFSF\_\*(명령 형식 문자열 필터 매개변수 구조)

표 75. 상수 값		
이름	10진수 값	16진수 값
MQCFSF_STRUC_LENGTH_FIXED	24	X'00000018'

### MQCFSL\_\*(명령 형식 문자열 목록 매개변수 구조)

표 76. 상수 값		
이름	10진수 값	16진수 값
MQCFSL_STRUC_LENGTH_FIXED	24	X'00000018'

### MQCFST\_\*(명령 형식 문자열 매개변수 구조)

표 77. 상수 값		
이름	10진수 값	16진수 값
MQCFST_STRUC_LENGTH_FIXED	20	X'00000014'

## MQCFSTATUS\_\*(명령 형식 CF 상태)

표 78. 상수 값		
이름	10진수 값	16진수 값
MQCFSTATUS_NOT_FOUND	0	X'00000000'
MQCFSTATUS_ACTIVE	1	X'00000001'
MQCFSTATUS_IN_RECOVER	2	X'00000002'
MQCFSTATUS_IN_BACKUP	3	X'00000003'
MQCFSTATUS_FAILED	4	X'00000004'
MQCFSTATUS_NONE	5	X'00000005'
MQCFSTATUS_UNKNOWN	6	X'00000006'
MQCFSTATUS_ADMIN_INCOMPLETE	20	X'00000014'
MQCFSTATUS_NEVER_USED	21	X'00000015'
MQCFSTATUS_NO_BACKUP	22	X'00000016'
MQCFSTATUS_NOT_FAILED	23	X'00000017'
MQCFSTATUS_NOT_RECOVERABLE	24	X'00000018'
MQCFSTATUS_XES_ERROR	25	X'00000019'

## MQCFT\_\*(명령 형식 구조 유형)

표 79. 상수 값		
이름	10진수 값	16진수 값
MQCFT_NONE	0	X'00000000'
MQCFT_COMMAND	1	X'00000001'
MQCFT_RESPONSE	2	X'00000002'
MQCFT_INTEGER	3	X'00000003'
MQCFT_STRING	4	X'00000004'
MQCFT_INTEGER_LIST	5	X'00000005'
MQCFT_STRING_LIST	6	X'00000006'
MQCFT_EVENT	7	X'00000007'
MQCFT_USER	8	X'00000008'
MQCFT_BYTE_STRING	9	X'00000009'
MQCFT_TRACE_ROUTE	10	X'0000000A'
MQCFT_REPORT	12	X'0000000C'
MQCFT_INTEGER_FILTER	13	X'0000000D'
MQCFT_STRING_FILTER	14	X'0000000E'
MQCFT_BYTE_STRING_FILTER	15	X'0000000F'
MQCFT_COMMAND_XR	16	X'00000010'
MQCFT_XR_MSG	17	X'00000011'
MQCFT_XR_ITEM	18	X'00000012'
MQCFT_XR_SUMMARY	19	X'00000013'
MQCFT_GROUP	20	X'00000014'
MQCFT_STATISTICS	21	X'00000015'

표 79. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCFT_ACCOUNTING	22	X'00000016'
MQCFT_INTEGER64	23	X'00000017'
MQCFT_INTEGER64_LIST	25	X'00000019'

### MQCFTYPE\_\*(명령 형식 CF 유형)

표 80. 상수 값		
이름	10진수 값	16진수 값
MQCFTYPE_APPL	0	X'00000000'
MQCFTYPE_ADMIN	1	X'00000001'

### MQCFUNC\_\*(CICS 정보 헤더 함수)

표 81. 상수 구조	
이름	구조
MQCFUNC_MQCONN	"CONN"
MQCFUNC_MQGET	"GET~"
MQCFUNC_MQINQ	"INQ~"
MQCFUNC_MQOPEN	"OPEN"
MQCFUNC_MQPUT	"PUT~"
MQCFUNC_MQPUT1	"PUT1"
MQCFUNC_NONE	"~~~~"
MQCFUNC_MQCONN_ARRAY	'C','O','N','N'
MQCFUNC_MQGET_ARRAY	'G','E','T','~'
MQCFUNC_MQINQ_ARRAY	'I','N','Q','~'
MQCFUNC_MQOPEN_ARRAY	'O','P','E','N'
MQCFUNC_MQPUT_ARRAY	'P','U','T','~'
MQCFUNC_MQPUT1_ARRAY	'P','U','T','1'
MQCFUNC_NONE_ARRAY	'~','~','~','~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

### MQCGWI\_\*(CICS 정보 헤더 가져오기 대기 간격)

표 82. 상수 값		
이름	10진수 값	16진수 값
MQCGWI_DEFAULT	-2	X'FFFFFFFE'

### MQCHAD\_\*(채널 자동 정의)

표 83. 상수 값		
이름	10진수 값	16진수 값
MQCHAD_DISABLED	0	X'00000000'
MQCHAD_ENABLED	1	X'00000001'

### MQCHIDS\_\*(명령 형식 인다우트(in-doubt) 상태)

표 84. 상수 값		
이름	10진수 값	16진수 값
MQCHIDS_NOT_INDOUBT	0	X'00000000'
MQCHIDS_INDOUBT	1	X'00000001'

### MQCHLD\_\*(명령 형식 채널 배치)

표 85. 상수 값		
이름	10진수 값	16진수 값
MQCHLD_ALL	-1	X'FFFFFFFF'
MQCHLD_DEFAULT	1	X'00000001'
MQCHLD_SHARED	2	X'00000002'
MQCHLD_PRIVATE	4	X'00000004'
MQCHLD_FIXSHARED	5	X'00000005'

### MQCHS\_\*(명령 형식 채널 상태)

표 86. 상수 값		
이름	10진수 값	16진수 값
MQCHS_INACTIVE	0	X'00000000'
MQCHS_BINDING	1	X'00000001'
MQCHS_STARTING	2	X'00000002'
MQCHS_RUNNING	3	X'00000003'
MQCHS_STOPPING	4	X'00000004'
MQCHS_RETRYING	5	X'00000005'
MQCHS_STOPPED	6	X'00000006'
MQCHS_REQUESTING	7	X'00000007'
MQCHS_PAUSED	8	X'00000008'
MQCHS_INITIALIZING	13	X'0000000D'
MQCHS_SWITCHING	14	X'0000000E'

### MQCHSH\_\*(명령 형식 채널 공유 재시작 옵션)

표 87. 상수 값		
이름	10진수 값	16진수 값
MQCHSH_RESTART_NO	0	X'00000000'
MQCHSH_RESTART_YES	1	X'00000001'

### MQCHSR\_\*(명령 형식 채널 중지 옵션)

표 88. 상수 값		
이름	10진수 값	16진수 값
MQCHSR_STOP_NOT_REQUESTED	0	X'00000000'
MQCHSR_STOP_REQUESTED	1	X'00000001'

## MQCHSSTATE\_\*(명령 형식 채널 하위 상태)

표 89. 상수 값		
이름	10진수 값	16진수 값
MQCHSSTATE_OTHER	0	X'00000000'
MQCHSSTATE_END_OF_BATCH	100	X'00000064'
MQCHSSTATE_SENDING	200	X'000000C8'
MQCHSSTATE_RECEIVING	300	X'0000012C'
MQCHSSTATE_SERIALIZING	400	X'00000190'
MQCHSSTATE_RESYNCHING	500	X'000001F4'
MQCHSSTATE_HEARTBEATING	600	X'00000258'
MQCHSSTATE_IN_SCYEXIT	700	X'000002BC'
MQCHSSTATE_IN_RCSEXIT	800	X'00000320'
MQCHSSTATE_IN_SENDEXIT	900	X'00000384'
MQCHSSTATE_IN_MSGEXIT	1000	X'000003E8'
MQCHSSTATE_IN_MREXIT	1100	X'0000044C'
MQCHSSTATE_IN_CHADEXIT	1200	X'000004B0'
MQCHSSTATE_NET_CONNECTING	1250	X'000004E2'
MQCHSSTATE_SSL_HANDSHAKING	1300	X'00000514'
MQCHSSTATE_NAME_SERVER	1400	X'00000578'
MQCHSSTATE_IN_MQPUT	1500	X'000005DC'
MQCHSSTATE_IN_MQGET	1600	X'00000640'
MQCHSSTATE_IN_MQI_CALL	1700	X'000006A4'
MQCHSSTATE_COMPRESSING	1800	X'00000708'

## MQCHT\_\*(채널 유형)

표 90. 상수 값		
이름	10진수 값	16진수 값
MQCHT_SENDER	1	X'00000001'
MQCHT_SERVER	2	X'00000002'
MQCHT_RECEIVER	3	X'00000003'
MQCHT_REQUESTER	4	X'00000004'
MQCHT_ALL	5	X'00000005'
MQCHT_CLNTCONN	6	X'00000006'
MQCHT_SVRCONN	7	X'00000007'
MQCHT_CLUSRCVR	8	X'00000008'
MQCHT_CLUSSDR	9	X'00000009'

## MQCHTAB\_\*(명령 형식 채널 테이블 유형)

표 91. 상수 값		
이름	10진수 값	16진수 값
MQCHTAB_Q_MGR	1	X'00000001'



표 91. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCHTAB_CLNTCONN	2	X'00000002'

### MQCI\_\*(상관 ID)

표 92. 상수 이름 및 값	
이름	값
MQCI_NONE	X'00...00' (24년)
MQCI_NONE_ARRAY	'\0','\0',... (24년)
MQCI_NEW_SESSION	X'414D5121...'
MQCI_NEW_SESSION_ARRAY	'\x41','\x4D','\51','\x21',...

### MQCIH\_\*(CICS 정보 헤더 구조 및 플래그)

#### CICS 정보 헤더 구조

표 93. 상수 구조	
이름	구조
MQCIH_STRUC_ID	"CIH~"
MQCIH_STRUC_ID_ARRAY	'C','I','H','~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 94. 상수 값		
이름	10진수 값	16진수 값
MQCIH_VERSION_1	1	X'00000001'
MQCIH_VERSION_2	2	X'00000002'
MQCIH_CURRENT_VERSION	2	X'00000002'
MQCIH_LENGTH_1	164	X'000000A4'
MQCIH_LENGTH_2	180	X'000000B4'
MQCIH_CURRENT_LENGTH	180	X'000000B4'

#### CICS 정보 헤더 플래그

표 95. 상수 값		
이름	10진수 값	16진수 값
MQCIH_NONE	0	X'00000000'
MQCIH_PASS_EXPIRATION	1	X'00000001'
MQCIH_UNLIMITED_EXPIRATION	0	X'00000000'
MQCIH_REPLY_WITHOUT_NULLS	2	X'00000002'
MQCIH_REPLY_WITH_NULLS	0	X'00000000'
MQCIH_SYNC_ON_RETURN	4	X'00000004'
MQCIH_NO_SYNC_ON_RETURN	0	X'00000000'

## MQCLCT\_\*(클러스터 캐시 유형)

표 96. 상수 값		
이름	10진수 값	16진수 값
MQCLCT_STATIC	0	X'00000000'
MQCLCT_DYNAMIC	1	X'00000001'

## MQCLRS\_\*(명령 형식 토픽 문자열 지우기 범위)

표 97. 상수 값		
이름	10진수 값	16진수 값
MQCLRS_LOCAL	1	X'00000001'
MQCLRS_GLOBAL	2	X'00000002'

## MQCLRT\_\*(명령 형식 토픽 문자열 지우기 유형)

표 98. 상수 값		
이름	10진수 값	16진수 값
MQCLRT_RETAINED	1	X'00000001'

## MQCLT\_\*(CICS 정보 헤더 링크 유형)

표 99. 상수 값		
이름	10진수 값	16진수 값
MQCLT_PROGRAM	1	X'00000001'
MQCLT_TRANSACTION	2	X'00000002'

## MQCLWL\_\*(클러스터 워크로드)

표 100. 상수 값		
이름	10진수 값	16진수 값
MQCLWL_USEQ_LOCAL	0	X'00000000'
MQCLWL_USEQ_ANY	1	X'00000001'
MQCLWL_USEQ_AS_Q_MGR	-3	X'FFFFFFFD'

## MQCLXQ\_\*(클러스터 전송 큐 유형)

MQCLXQ\_\*는 DEFCLXQ 큐 관리자 속성에 설정할 수 있는 값입니다. DEFCLXQ 속성은 클러스터 송신자 채널이 기본적으로 메시지를 가져오고 클러스터 수신자 채널로 메시지를 보내기 위해 선택하는 전송 큐를 제어합니다.

표 101. 상수 값		
이름	10진수 값	16진수 값
MQCLXQ_SCTQ	0	X'00000000'
MQCLXQ_CHANNEL	1	X'00000001'

### 관련 참조

786 페이지의 『DefClusterXmitQueueType(MQLONG)』

DefClusterXmitQueueType 속성은 클러스터-송신자 채널이 기본적으로 메시지를 가져오고 클러스터-수신자 채널로 메시지를 보내기 위해 선택하는 전송 큐를 제어합니다.

[큐 관리자 변경](#)

큐 관리자 조회

큐 관리자 조회(응답)

680 페이지의 『MQINQ - 오브젝트 속성 조회』

MQINQ 호출은 오브젝트의 속성이 포함된 문자열의 세트 및 정수의 배열을 리턴합니다.

## MQCMD\_\*(명령 코드)

표 102. 상수 값		
이름	10진수 값	16진수 값
MQCMD_NONE	0	X'00000000'
MQCMD_CHANGE_Q_MGR	1	X'00000001'
MQCMD_INQUIRE_Q_MGR	2	X'00000002'
MQCMD_CHANGE_PROCESS	3	X'00000003'
MQCMD_COPY_PROCESS	4	X'00000004'
MQCMD_CREATE_PROCESS	5	X'00000005'
MQCMD_DELETE_PROCESS	6	X'00000006'
MQCMD_INQUIRE_PROCESS	7	X'00000007'
MQCMD_CHANGE_Q	8	X'00000008'
MQCMD_CLEAR_Q	9	X'00000009'
MQCMD_COPY_Q	10	X'0000000A'
MQCMD_CREATE_Q	11	X'0000000B'
MQCMD_DELETE_Q	12	X'0000000C'
MQCMD_INQUIRE_Q	13	X'0000000D'
MQCMD_REFRESH_Q_MGR	16	X'00000010'
MQCMD_RESET_Q_STATS	17	X'00000011'
MQCMD_INQUIRE_Q_NAMES	18	X'00000012'
MQCMD_INQUIRE_PROCESS_NAMES	19	X'00000013'
MQCMD_INQUIRE_CHANNEL_NAMES	20	X'00000014'
MQCMD_CHANGE_CHANNEL	21	X'00000015'
MQCMD_COPY_CHANNEL	22	X'00000016'
MQCMD_CREATE_CHANNEL	23	X'00000017'
MQCMD_DELETE_CHANNEL	24	X'00000018'
MQCMD_INQUIRE_CHANNEL	25	X'00000019'
MQCMD_PING_CHANNEL	26	X'0000001A'
MQCMD_RESET_CHANNEL	27	X'0000001B'
MQCMD_START_CHANNEL	28	X'0000001C'
MQCMD_STOP_CHANNEL	29	X'0000001D'
MQCMD_START_CHANNEL_INIT	30	X'0000001E'
MQCMD_START_CHANNEL_LISTENER	31	X'0000001F'
MQCMD_CHANGE_NAMELIST	32	X'00000020'
MQCMD_COPY_NAMELIST	33	X'00000021'
MQCMD_CREATE_NAMELIST	34	X'00000022'
MQCMD_DELETE_NAMELIST	35	X'00000023'

표 102. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCMD_INQUIRE_NAMELIST	36	X'00000024'
MQCMD_INQUIRE_NAMELIST_NAMES	37	X'00000025'
MQCMD_ESCAPE	38	X'00000026'
MQCMD_RESOLVE_CHANNEL	39	X'00000027'
MQCMD_PING_Q_MGR	40	X'00000028'
MQCMD_INQUIRE_Q_STATUS	41	X'00000029'
MQCMD_INQUIRE_CHANNEL_STATUS	42	X'0000002A'
MQCMD_CONFIG_EVENT	43	X'0000002B'
MQCMD_Q_MGR_EVENT	44	X'0000002C'
MQCMD_PERFM_EVENT	45	X'0000002D'
MQCMD_CHANNEL_EVENT	46	X'0000002E'
MQCMD_DELETE_PUBLICATION	60	X'0000003C'
MQCMD_DEREGISTER_PUBLISHER	61	X'0000003D'
MQCMD_DEREGISTER_SUBSCRIBER	62	X'0000003E'
MQCMD_PUBLISH	63	X'0000003F'
MQCMD_REGISTER_PUBLISHER	64	X'00000040'
MQCMD_REGISTER_SUBSCRIBER	65	X'00000041'
MQCMD_REQUEST_UPDATE	66	X'00000042'
MQCMD_BROKER_INTERNAL	67	X'00000043'
MQCMD_ACTIVITY_MSG	69	X'00000045'
MQCMD_INQUIRE_CLUSTER_Q_MGR	70	X'00000046'
MQCMD_RESUME_Q_MGR_CLUSTER	71	X'00000047'
MQCMD_SUSPEND_Q_MGR_CLUSTER	72	X'00000048'
MQCMD_REFRESH_CLUSTER	73	X'00000049'
MQCMD_RESET_CLUSTER	74	X'0000004A'
MQCMD_TRACE_ROUTE	75	X'0000004B'
MQCMD_REFRESH_SECURITY	78	X'0000004E'
MQCMD_CHANGE_AUTH_INFO	79	X'0000004F'
MQCMD_COPY_AUTH_INFO	80	X'00000050'
MQCMD_CREATE_AUTH_INFO	81	X'00000051'
MQCMD_DELETE_AUTH_INFO	82	X'00000052'
MQCMD_INQUIRE_AUTH_INFO	83	X'00000053'
MQCMD_INQUIRE_AUTH_INFO_NAMES	84	X'00000054'
MQCMD_INQUIRE_CONNECTION	85	X'00000055'
MQCMD_STOP_CONNECTION	86	X'00000056'
MQCMD_INQUIRE_AUTH_RECS	87	X'00000057'
MQCMD_INQUIRE_ENTITY_AUTH	88	X'00000058'
MQCMD_DELETE_AUTH_REC	89	X'00000059'
MQCMD_SET_AUTH_REC	90	X'0000005A'

표 102. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCMD_LOGGER_EVENT	91	X'0000005B'
MQCMD_RESET_Q_MGR	92	X'0000005C'
MQCMD_CHANGE_LISTENER	93	X'0000005D'
MQCMD_COPY_LISTENER	94	X'0000005E'
MQCMD_CREATE_LISTENER	95	X'0000005F'
MQCMD_DELETE_LISTENER	96	X'00000060'
MQCMD_INQUIRE_LISTENER	97	X'00000061'
MQCMD_INQUIRE_LISTENER_STATUS	98	X'00000062'
MQCMD_COMMAND_EVENT	99	X'00000063'
MQCMD_CHANGE_SECURITY	100	X'00000064'
MQCMD_CHANGE_CF_STRUC	101	X'00000065'
MQCMD_CHANGE_STG_CLASS	102	X'00000066'
MQCMD_CHANGE_TRACE	103	X'00000067'
MQCMD_ARCHIVE_LOG	104	X'00000068'
MQCMD_BACKUP_CF_STRUC	105	X'00000069'
MQCMD_CREATE_BUFFER_POOL	106	X'0000006A'
MQCMD_CREATE_PAGE_SET	107	X'0000006B'
MQCMD_CREATE_CF_STRUC	108	X'0000006C'
MQCMD_CREATE_STG_CLASS	109	X'0000006D'
MQCMD_COPY_CF_STRUC	110	X'0000006E'
MQCMD_COPY_STG_CLASS	111	X'0000006F'
MQCMD_DELETE_CF_STRUC	112	X'00000070'
MQCMD_DELETE_STG_CLASS	113	X'00000071'
MQCMD_INQUIRE_ARCHIVE	114	X'00000072'
MQCMD_INQUIRE_CF_STRUC	115	X'00000073'
MQCMD_INQUIRE_CF_STRUC_STATUS	116	X'00000074'
MQCMD_INQUIRE_CMD_SERVER	117	X'00000075'
MQCMD_INQUIRE_CHANNEL_INIT	118	X'00000076'
MQCMD_INQUIRE_QSG	119	X'00000077'
MQCMD_INQUIRE_LOG	120	X'00000078'
MQCMD_INQUIRE_SECURITY	121	X'00000079'
MQCMD_INQUIRE_STG_CLASS	122	X'0000007A'
MQCMD_INQUIRE_SYSTEM	123	X'0000007B'
MQCMD_INQUIRE_THREAD	124	X'0000007C'
MQCMD_INQUIRE_TRACE	125	X'0000007D'
MQCMD_INQUIRE_USAGE	126	X'0000007E'
MQCMD_MOVE_Q	127	X'0000007F'
MQCMD_RECOVER_BSDES	128	X'00000080'
MQCMD_RECOVER_CF_STRUC	129	X'00000081'

표 102. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCMD_RESET_TPIPE	130	X'00000082'
MQCMD_RESOLVE_INDOUBT	131	X'00000083'
MQCMD_RESUME_Q_MGR	132	X'00000084'
MQCMD_REVERIFY_SECURITY	133	X'00000085'
MQCMD_SET_ARCHIVE	134	X'00000086'
MQCMD_SET_LOG	136	X'00000088'
MQCMD_SET_SYSTEM	137	X'00000089'
MQCMD_START_CMD_SERVER	138	X'0000008A'
MQCMD_START_Q_MGR	139	X'0000008B'
MQCMD_START_TRACE	140	X'0000008C'
MQCMD_STOP_CHANNEL_INIT	141	X'0000008D'
MQCMD_STOP_CHANNEL_LISTENER	142	X'0000008E'
MQCMD_STOP_CMD_SERVER	143	X'0000008F'
MQCMD_STOP_Q_MGR	144	X'00000090'
MQCMD_STOP_TRACE	145	X'00000091'
MQCMD_SUSPEND_Q_MGR	146	X'00000092'
MQCMD_INQUIRE_CF_STRUC_NAMES	147	X'00000093'
MQCMD_INQUIRE_STG_CLASS_NAMES	148	X'00000094'
MQCMD_CHANGE_SERVICE	149	X'00000095'
MQCMD_COPY_SERVICE	150	X'00000096'
MQCMD_CREATE_SERVICE	151	X'00000097'
MQCMD_DELETE_SERVICE	152	X'00000098'
MQCMD_INQUIRE_SERVICE	153	X'00000099'
MQCMD_INQUIRE_SERVICE_STATUS	154	X'0000009A'
MQCMD_START_SERVICE	155	X'0000009B'
MQCMD_STOP_SERVICE	156	X'0000009C'
MQCMD_DELETE_BUFFER_POOL	157	X'0000009D'
MQCMD_DELETE_PAGE_SET	158	X'0000009E'
MQCMD_CHANGE_BUFFER_POOL	159	X'0000009F'
MQCMD_CHANGE_PAGE_SET	160	X'000000A0'
MQCMD_INQUIRE_Q_MGR_STATUS	161	X'000000A1'
MQCMD_CREATE_LOG	162	X'000000A2'
MQCMD_STATISTICS_MQI	164	X'000000A4'
MQCMD_STATISTICS_Q	165	X'000000A5'
MQCMD_STATISTICS_CHANNEL	166	X'000000A6'
MQCMD_ACCOUNTING_MQI	167	X'000000A7'
MQCMD_ACCOUNTING_Q	168	X'000000A8'
MQCMD_INQUIRE_AUTH_SERVICE	169	X'000000A9'
MQCMD_CHANGE_TOPIC	170	X'000000AA'

표 102. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCMD_COPY_TOPIC	171	X'000000AB'
MQCMD_CREATE_TOPIC	172	X'000000AC'
MQCMD_DELETE_TOPIC	173	X'000000AD'
MQCMD_INQUIRE_TOPIC	174	X'000000AE'
MQCMD_INQUIRE_TOPIC_NAMES	175	X'000000AF'
MQCMD_INQUIRE_SUBSCRIPTION	176	X'000000B0'
MQCMD_CREATE_SUBSCRIPTION	177	X'000000B1'
MQCMD_CHANGE_SUBSCRIPTION	178	X'000000B2'
MQCMD_DELETE_SUBSCRIPTION	179	X'000000B3'
MQCMD_COPY_SUBSCRIPTION	181	X'000000B5'
MQCMD_INQUIRE_SUB_STATUS	182	X'000000B6'
MQCMD_INQUIRE_TOPIC_STATUS	183	X'000000B7'
MQCMD_CLEAR_TOPIC_STRING	184	X'000000B8'
MQCMD_INQUIRE_PUBSUB_STATUS	185	X'000000B9'
MQCMD_PURGE_CHANNEL	195	X'000000C3'

### MQCMDI\_\*(명령 형식 명령 정보 값)

표 103. 상수 값		
이름	10진수 값	16진수 값
MQCMDI_CMDScope_ACCEPTED	1	X'00000001'
MQCMDI_CMDScope_GENERATED	2	X'00000002'
MQCMDI_CMDScope_COMPLETED	3	X'00000003'
MQCMDI_QSG_DISP_COMPLETED	4	X'00000004'
MQCMDI_COMMAND_ACCEPTED	5	X'00000005'
MQCMDI_CLUSTER_REQUEST_QUEUED	6	X'00000006'
MQCMDI_CHANNEL_INIT_STARTED	7	X'00000007'
MQCMDI_RECOVER_STARTED	11	X'0000000B'
MQCMDI_BACKUP_STARTED	12	X'0000000C'
MQCMDI_RECOVER_COMPLETED	13	X'0000000D'
MQCMDI_SEC_TIMER_ZERO	14	X'0000000E'
MQCMDI_REFRESH_CONFIGURATION	16	X'00000010'
MQCMDI_SEC_SIGNOFF_ERROR	17	X'00000011'
MQCMDI_IMS_BRIDGE_SUSPENDED	18	X'00000012'
MQCMDI_DB2_SUSPENDED	19	X'00000013'
MQCMDI_DB2_OBSOLETE_MSGS	20	X'00000014'
MQCMDI_SEC_UPPERCASE	21	X'00000015'
MQCMDI_SEC_MIXEDCASE	22	X'00000016'

## MQCMDL\_\*(명령 레벨)

표 104. 상수 이름 및 값	
이름	값
MQCMDL_LEVEL_800	800
MQCMDL_LEVEL_801	801
MQCMDL_LEVEL_802	802
MQCMDL_LEVEL_900	900
MQCMDL_LEVEL_901	901
MQCMDL_LEVEL_902	902
MQCMDL_LEVEL_903	903
MQCMDL_LEVEL_904	904
MQCMDL_LEVEL_905	905
MQCMDL_LEVEL_910	910
MQCMDL_LEVEL_912	912
MQCMDL_LEVEL_913	913
MQCMDL_LEVEL_914	914
MQCMDL_LEVEL_915	915
MQCMDL_LEVEL_920	920
MQCMDL_LEVEL_921	921
MQCMDL_LEVEL_922	922
MQCMDL_LEVEL_923	923
MQCMDL_LEVEL_924	924
MQCMDL_LEVEL_925	925
MQCMDL_LEVEL_930	930
MQCMDL_LEVEL_931	931
MQCMDL_LEVEL_932	932

## MQCMHO\_\*(메시지 핸들 작성 옵션 및 구조)

### 메시지 핸들 작성 옵션 구조

표 105. 상수 구조	
이름	구조
MQCMHO_STRUC_ID	"CMHO"
MQCMHO_STRUC_ID_ARRAY	'C', 'M', 'H', 'O'

**참고:** ~ 기호는 단일 공백 문자를 나타냅니다.

표 106. 상수 값		
이름	10진수 값	16진수 값
MQCMHO_VERSION_1	1	X'00000001'
MQCMHO_CURRENT_VERSION	1	X'00000001'



## 메시지 핸들 작성 옵션

표 107. 상수 값		
이름	10진수 값	16진수 값
MQCMHO_DEFAULT_VALIDATION	0	X'00000000'
MQCMHO_NO_VALIDATION	1	X'00000001'
MQCMHO_VALIDATE	2	X'00000002'
MQCMHO_NONE	0	X'00000000'

## MQCNO\_\*(연결 옵션 및 구조)

### 연결 옵션 구조

표 108. 상수 구조	
이름	구조
MQCNO_STRUC_ID	"CNO~"
MQCNO_STRUC_ID_ARRAY	'C', 'N', 'O', '~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 109. 상수 값		
이름	10진수 값	16진수 값
MQCNO_VERSION_1	1	X'00000001'
MQCNO_VERSION_2	2	X'00000002'
MQCNO_VERSION_3	3	X'00000003'
MQCNO_VERSION_4	4	X'00000004'
MQCNO_VERSION_5	5	X'00000005'
MQCNO_CURRENT_VERSION	5	X'00000005'

### 연결 옵션

표 110. 상수 값		
이름	10진수 값	16진수 값
MQCNO_STANDARD_BINDING	0	X'00000000'
MQCNO_FASTPATH_BINDING	1	X'00000001'
MQCNO_SERIALIZE_CONN_TAG_Q_MGR	2	X'00000002'
MQCNO_SERIALIZE_CONN_TAG_QSG	4	X'00000004'
MQCNO_RESTRICT_CONN_TAG_Q_MGR	8	X'00000008'
MQCNO_RESTRICT_CONN_TAG_QSG	16	X'00000010'
MQCNO_HANDLE_SHARE_NONE	32	X'00000020'
MQCNO_HANDLE_SHARE_BLOCK	64	X'00000040'
MQCNO_HANDLE_SHARE_NO_BLOCK	128	X'00000080'
MQCNO_SHARED_BINDING	256	X'00000100'
MQCNO_ISOLATED_BINDING	512	X'00000200'
MQCNO_LOCAL_BINDING	1024	X'00000400'
MQCNO_CLIENT_BINDING	2048	X'00000800'

표 110. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCNO_ACCOUNTING_MQI_ENABLED	4096	X'00001000'
MQCNO_ACCOUNTING_MQI_DISABLED	8192	X'00002000'
MQCNO_ACCOUNTING_Q_ENABLED	16384	X'00004000'
MQCNO_ACCOUNTING_Q_DISABLED	32768	X'00008000'
MQCNO_NO_CONV_SHARING	65536	X'00010000'
MQCNO_ALL_CONVS_SHARE	262144	X'00040000'
MQCNO_CD_FOR_OUTPUT_ONLY	524288	X'00080000'
MQCNO_USE_CD_SELECTION	1048576	X'00100000'
MQCNO_RECONNECT	16777216	X'01000000'
MQCNO_RECONNECT_AS_DEF	0	X'00000000'
MQCNO_RECONNECT_DISABLED	33554432	X'02000000'
MQCNO_RECONNECT_Q_MGR	67108864	X'04000000'
MQCNO_ACTIVITY_TRACE_ENABLED	134217728	X'08000000'
MQCNO_ACTIVITY_TRACE_DISABLED	268435456	X'10000000'
MQCNO_NONE	0	X'00000000'

### MQCO\_\*(닫기 옵션)

표 111. 상수 값		
이름	10진수 값	16진수 값
MQCO_IMMEDIATE	0	X'00000000'
MQCO_NONE	0	X'00000000'
MQCO_DELETE	1	X'00000001'
MQCO_DELETE_PURGE	2	X'00000002'
MQCO_KEEP_SUB	4	X'00000004'
MQCO_REMOVE_SUB	8	X'00000008'
MQCO_QUIESCE	32	X'00000020'

### MQCODL\_\*(CICS 정보 헤더 출력 데이터 길이)

표 112. 상수 값		
이름	10진수 값	16진수 값
MQCODL_AS_INPUT	-1	X'FFFFFFFF'

### MQCOMPRESS\_\*(채널 압축)

표 113. 상수 값		
이름	10진수 값	16진수 값
MQCOMPRESS_NOT_AVAILABLE	-1	X'FFFFFFFF'
MQCOMPRESS_NONE	0	X'00000000'
MQCOMPRESS_RLE	1	X'00000001'
MQCOMPRESS_ZLIBFAST	2	X'00000002'
MQCOMPRESS_ZLIBHIGH	4	X'00000004'

표 113. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCOMPRESS_SYSTEM	8	X'00000008'
MQCOMPRESS_ANY	268435455	X'0FFFFFFF'

### MQCONNID\_\*(연결 ID)

표 114. 상수 이름 및 값	
이름	값
MQCONNID_NONE	X'00...00' (24널)
MQCONNID_NONE_ARRAY	'\0', '\0', ... (24널)

### MQCOPY\_\*(특성 복사 옵션)

표 115. 상수 값		
이름	10진수 값	16진수 값
MQCOPY_NONE	0	X'00000000'
MQCOPY_ALL	1	X'00000001'
MQCOPY_FORWARD	2	X'00000002'
MQCOPY_PUBLISH	4	X'00000004'
MQCOPY_REPLY	8	X'00000008'
MQCOPY_REPORT	16	X'00000010'
MQCOPY_DEFAULT	22	X'00000016'

### MQCQT\_\*(클러스터 큐 유형)

표 116. 상수 값		
이름	10진수 값	16진수 값
MQCQT_LOCAL_Q	1	X'00000001'
MQCQT_ALIAS_Q	2	X'00000002'
MQCQT_REMOTE_Q	3	X'00000003'
MQCQT_Q_MGR_ALIAS	4	X'00000004'

### MQCRC\_\*(CICS 정보 헤더 리턴 코드)

표 117. 상수 값		
이름	10진수 값	16진수 값
MQCRC_OK	0	X'00000000'
MQCRC_CICS_EXEC_ERROR	1	X'00000001'
MQCRC_MQ_API_ERROR	2	X'00000002'
MQCRC_BRIDGE_ERROR	3	X'00000003'
MQCRC_BRIDGE_ABEND	4	X'00000004'
MQCRC_APPLICATION_ABEND	5	X'00000005'
MQCRC_SECURITY_ERROR	6	X'00000006'
MQCRC_PROGRAM_NOT_AVAILABLE	7	X'00000007'

표 117. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQCRC_BRIDGE_TIMEOUT	8	X'00000008'
MQCRC_TRANSID_NOT_AVAILABLE	9	X'00000009'

### MQCS\_\*(MQCBC 상수 이용자 상태)

표 118. 상수 값		
이름	10진수 값	16진수 값
MQCS_NONE	0	X'00000000'
MQCS_SUSPENDED_TEMPORARY	1	X'00000001'
MQCS_SUSPENDED_USER_ACTION	2	X'00000002'
MQCS_SUSPENDED	3	X'00000003'
MQCS_STOPPED	4	X'00000004'

### MQCSC\_\*(CICS 정보 헤더 시작 코드)

표 119. 상수 구조	
이름	구조
MQCSC_START	"S-"
MQCSC_STARTDATA	"SD-"
MQCSC_TERMINPUT	"TD-"
MQCSC_NONE	"-"
MQCSC_START_ARRAY	'S','-',',','-',',','-',',','-'
MQCSC_STARTDATA_ARRAY	'S','D','-',',','-',',','-',',','-'
MQCSC_TERMINPUT_ARRAY	'T','D','-',',','-',',','-',',','-'
MQCSC_NONE_ARRAY	'-',',','-',',','-',',','-',',','-'

참고: - 기호는 단일 공백 문자를 나타냅니다.

### MQCSP\_\*(연결 보안 매개변수 구조 및 인증 유형)

#### 연결 보안 매개변수 구조

표 120. 상수 구조	
이름	구조
MQCSP_STRUC_ID	"CSP-"
MQCSP_STRUC_ID_ARRAY	'C','S','P','-',',','-',',','-',',','-'

참고: - 기호는 단일 공백 문자를 나타냅니다.

표 121. 상수 값		
이름	10진수 값	16진수 값
MQCSP_VERSION_1	1	X'00000001'
MQCSP_VERSION_2	2	X'00000002'
MQCSP_VERSION_3	3	X'00000003'
MQCSP_CURRENT_VERSION	3	X'00000003'

## 연결 보안 매개변수 인증 유형

표 122. 상수 값		
이름	10진수 값	16진수 값
MQCSP_AUTH_NONE	0	X'00000000'
MQCSP_AUTH_USER_ID_AND_PWD	1	X'00000001'
MQCSP_AUTH_ID_TOKEN	2	X'00000002'

## MQCSRV\_\*(명령 서버 옵션)

표 123. 상수 값		
이름	10진수 값	16진수 값
MQCSRV_CONVERT_NO	0	X'00000000'
MQCSRV_CONVERT_YES	1	X'00000001'
MQCSRV_DLQ_NO	0	X'00000000'
MQCSRV_DLQ_YES	1	X'00000001'

## MQCT\_\*(큐 관리자 연결 태그)

표 124. 상수 이름 및 값	
이름	값
MQCT_NONE	X'00...00' (128닐)
MQCT_NONE_ARRAY	'\0','\0',... (128닐)

## MQCTES\_\*(CICS 정보 헤더 태스크 종료 상태)

표 125. 상수 값		
이름	10진수 값	16진수 값
MQCTES_NOSYNC	0	X'00000000'
MQCTES_COMMIT	256	X'00000100'
MQCTES_BACKOUT	4352	X'00001100'
MQCTES_ENDTASK	65536	X'00010000'

## MQCTLO\_\*(MQCTL 옵션 구조 및 사용자 제어 옵션)

### MQCTL 옵션 구조

표 126. 상수 구조	
이름	구조
MQCTLO_STRUC_ID	"CTLO"
MQCTLO_STRUC_ID_ARRAY	'C','T','L','O'

참고: ↵ 기호는 단일 공백 문자를 나타냅니다.

표 127. 상수 값		
이름	10진수 값	16진수 값
MQCTLO_VERSION_1	1	X'00000001'
MQCTLO_CURRENT_VERSION	1	X'00000001'

## MQCTL 옵션 이용자 제어 옵션

표 128. 상수 값		
이름	10진수 값	16진수 값
MQCTLO_NONE	0	X'00000000'
MQCTLO_THREAD_AFFINITY	1	X'00000001'
MQCTLO_FAIL_IF_QUIESCING	8192	X'00002000'

## MQCUOWC\_\*(CICS 정보 헤더 작업 단위 제어)

표 129. 상수 값		
이름	10진수 값	16진수 값
MQCUOWC_ONLY	273	X'00000111'
MQCUOWC_CONTINUE	65536	X'00010000'
MQCUOWC_FIRST	17	X'00000011'
MQCUOWC_MIDDLE	16	X'00000010'
MQCUOWC_LAST	272	X'00000110'
MQCUOWC_COMMIT	256	X'00000100'
MQCUOWC_BACKOUT	4352	X'00001100'

## MQCXP\_\*(채널 엑시트 매개변수 구조)

표 130. 상수 구조	
이름	구조
MQCXP_STRUC_ID	"CXP~"
MQCXP_STRUC_ID_ARRAY	'C', 'X', 'P', '~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 131. 상수 값		
이름	10진수 값	16진수 값
MQCXP_VERSION_1	1	X'00000001'
MQCXP_VERSION_2	2	X'00000002'
MQCXP_VERSION_3	3	X'00000003'
MQCXP_VERSION_4	4	X'00000004'
MQCXP_VERSION_5	5	X'00000005'
MQCXP_VERSION_6	6	X'00000006'
MQCXP_VERSION_7	7	X'00000007'
MQCXP_VERSION_8	8	X'00000008'
MQCXP_VERSION_9	9	X'00000009'
MQCXP_CURRENT_VERSION	9	X'00000009'

## MQDC\_\*(목적지 클래스)

표 132. 상수 값		
이름	10진수 값	16진수 값
MQDC_MANAGED	1	X'00000001'

표 132. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQDC_PROVIDED	2	X'00000002'

### MQDCC\_\*(변환 옵션 및 마스크 및 요인)

#### 변환 옵션

표 133. 상수 값		
이름	10진수 값	16진수 값
MQDCC_DEFAULT_CONVERSION	1	X'00000001'
MQDCC_FILL_TARGET_BUFFER	2	X'00000002'
MQDCC_INT_DEFAULT_CONVERSION	4	X'00000004'
MQDCC_SOURCE_ENC_NATIVE	(value differs by platform or version)	(value differs by platform or version)
MQDCC_SOURCE_ENC_NORMAL	16	X'00000010'
MQDCC_SOURCE_ENC_REVERSED	32	X'00000020'
MQDCC_SOURCE_ENC_UNDEFINED	0	X'00000000'
MQDCC_TARGET_ENC_NATIVE	(value differs by platform or version)	(value differs by platform or version)
MQDCC_TARGET_ENC_NORMAL	256	X'00000100'
MQDCC_TARGET_ENC_REVERSED	512	X'00000200'
MQDCC_TARGET_ENC_UNDEFINED	0	X'00000000'
MQDCC_NONE	0	X'00000000'

#### 변환 옵션 마스크 및 요인

표 134. 상수 값		
이름	10진수 값	16진수 값
MQDCC_SOURCE_ENC_MASK	240	X'000000F0'
MQDCC_TARGET_ENC_MASK	3840	X'00000F00'
MQDCC_SOURCE_ENC_FACTOR	16	X'00000010'
MQDCC_TARGET_ENC_FACTOR	256	X'00000100'

### MQDELO\_\*(발행/구독 삭제 옵션)

표 135. 상수 값		
이름	10진수 값	16진수 값
MQDELO_NONE	0	X'00000000'
MQDELO_LOCAL	4	X'00000004'

### MQDH\_\*(분배 헤더 구조)

표 136. 상수 구조	
이름	구조
MQDH_STRUC_ID	"DH--"

표 136. 상수 구조 (계속)	
이름	구조
MQDH_STRUC_ID_ARRAY	'D','H',' ',' '

참고: 궡 기호는 단일 공백 문자를 나타냅니다.

표 137. 상수 값		
이름	10진수 값	16진수 값
MQDH_VERSION_1	1	X'00000001'
MQDH_CURRENT_VERSION	1	X'00000001'

### MQDHF\_\*(분배 헤더 플래그)

표 138. 상수 값		
이름	10진수 값	16진수 값
MQDHF_NEW_MSG_IDS	1	X'00000001'
MQDHF_NONE	0	X'00000000'

### MQDISCONNECT\_\*(명령 형식 연결 끊기 유형)

표 139. 상수 값		
이름	10진수 값	16진수 값
MQDISCONNECT_NORMAL	0	X'00000000'
MQDISCONNECT_IMPLICIT	1	X'00000001'
MQDISCONNECT_Q_MGR	2	X'00000002'

### MQDL\_\*(분배 목록)

표 140. 상수 값		
이름	10진수 값	16진수 값
MQDL_SUPPORTED	1	X'00000001'
MQDL_NOT_SUPPORTED	0	X'00000000'

### MQDLH\_\*(데드 레터 헤더 구조)

표 141. 상수 구조	
이름	구조
MQDLH_STRUC_ID	"DLH궡"
MQDLH_STRUC_ID_ARRAY	'D','L','H',' '

참고: 궡 기호는 단일 공백 문자를 나타냅니다.

MQDLH_VERSION_1	1	X'00000001'
MQDLH_CURRENT_VERSION	1	X'00000001'



## MQDLV\_\*(지속/비지속 메시지 전달)

표 142. 상수 값		
이름	10진수 값	16진수 값
MQDLV_AS_PARENT	0	X'00000000'
MQDLV_ALL	1	X'00000001'
MQDLV_ALL_DUR	2	X'00000002'
MQDLV_ALL_AVAIL	3	X'00000003'

## MQDMHO\_\*(메시지 핸들 삭제 옵션 및 구조)

### 메시지 핸들 삭제 옵션 구조

표 143. 상수 구조	
이름	구조
MQDMHO_STRUC_ID	"DMHO"
MQDMHO_STRUC_ID_ARRAY	'D', 'M', 'H', 'O'

참고: - 기호는 단일 공백 문자를 나타냅니다.

표 144. 상수 값		
이름	10진수 값	16진수 값
MQDMHO_VERSION_1	1	X'00000001'
MQDMHO_CURRENT_VERSION	1	X'00000001'

### 메시지 핸들 삭제 옵션

표 145. 상수 값		
이름	10진수 값	16진수 값
MQDMHO_NONE	0	X'00000000'

## MQDMPO\_\*(메시지 특성 삭제 옵션 및 구조)

### 메시지 특성 삭제 옵션 구조

표 146. 상수 구조	
이름	구조
MQDMPO_STRUC_ID	"DMPO"
MQDMPO_STRUC_ID_ARRAY	'D', 'M', 'P', 'O'

참고: - 기호는 단일 공백 문자를 나타냅니다.

표 147. 상수 값		
이름	10진수 값	16진수 값
MQDMPO_VERSION_1	1	X'00000001'
MQDMPO_CURRENT_VERSION	1	X'00000001'

## 메시지 특성 삭제 옵션

표 148. 상수 값		
이름	10진수 값	16진수 값
MQDMPO_DEL_FIRST	0	X'00000000'
MQDMPO_DEL_PROP_UNDER_CURSOR	1	X'00000001'
MQDMPO_NONE	0	X'00000000'

## MQDNSWLM\_\*(DNS WLM)

표 149. 상수 값		
이름	10진수 값	16진수 값
MQDNSWLM_NO	0	X'00000000'
MQDNSWLM_YES	1	X'00000001'

## MQDT\_\*(목적지 유형)

표 150. 상수 값		
이름	10진수 값	16진수 값
MQDT_APPL	1	X'00000001'
MQDT_BROKER	2	X'00000002'

## MQDXP\_\*(변환 엑시트 매개변수 구조)

표 151. 상수 구조	
이름	구조
MQDXP_STRUC_ID	"DXP~"
MQDXP_STRUC_ID_ARRAY	'D', 'X', 'P', '~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 152. 상수 값		
이름	10진수 값	16진수 값
MQDXP_VERSION_1	1	X'00000001'
MQDXP_VERSION_2	2	X'00000002'
MQDXP_CURRENT_VERSION	2	X'00000002'

## MQEC\_\*(신호 값)

표 153. 상수 값		
이름	10진수 값	16진수 값
MQEC_MSG_ARRIVED	2	X'00000002'
MQEC_WAIT_INTERVAL_EXPIRED	3	X'00000003'
MQEC_WAIT_CANCELED	4	X'00000004'
MQEC_Q_MGR QUIESCING	5	X'00000005'
MQEC_CONNECTION QUIESCING	6	X'00000006'

## MQEI\_\*(만료)

표 154. 상수 값		
이름	10진수 값	16진수 값
MQEI_UNLIMITED	-1	X'FFFFFFFF'

## MQENC\_\*(인코딩)

### MQENC\_\*(인코딩)

표 155. 플랫폼별 상수 값			
이름	플랫폼	10진수 값	16진수 값
MQENC_NATIVE	IBM i	273	X'00000111'
	Linux	546	X'00000222'
	SPARC의 Linux	273	X'00000111'
	x86의 Linux	546	X'00000222'
	AIX and Linux	273	X'00000111'
	Windows	546	X'00000222'
	Windows의 Micro Focus COBOL	17	X'00000011'
	z/OS	785	X'00000311'

## MQENC\_\*(인코딩 마스크)

표 156. 상수 값		
이름	10진수 값	16진수 값
MQENC_INTEGER_MASK	15	X'0000000F'
MQENC_DECIMAL_MASK	240	X'000000F0'
MQENC_FLOAT_MASK	3840	X'00000F00'
MQENC_RESERVED_MASK	-4096	X'FFFFFF000'

## MQENC\_\*(2진 정수 인코딩)

표 157. 상수 값		
이름	10진수 값	16진수 값
MQENC_INTEGER_UNDEFINED	0	X'00000000'
MQENC_INTEGER_NORMAL	1	X'00000001'
MQENC_INTEGER_REVERSED	2	X'00000002'

## MQENC\_\*(팩형 10진수 정수 인코딩)

표 158. 상수 값		
이름	10진수 값	16진수 값
MQENC_DECIMAL_UNDEFINED	0	X'00000000'
MQENC_DECIMAL_NORMAL	16	X'00000010'
MQENC_DECIMAL_REVERSED	32	X'00000020'

## MQENC\_\*(부동 소수점 숫자 인코딩)

표 159. 상수 값		
이름	10진수 값	16진수 값
MQENC_FLOAT_UNDEFINED	0	X'00000000'
MQENC_FLOAT_IEEE_NORMAL	256	X'00000100'
MQENC_FLOAT_IEEE_REVERSED	512	X'00000200'
MQENC_FLOAT_S390	768	X'00000300'
MQENC_FLOAT_TNS	1024	X'00000400'

## MQEPH\_\*(임베드된 명령 형식 헤더 구조 및 플래그)

### 임베드된 명령 형식 헤더 구조

표 160. 상수 구조	
이름	구조
MQEPH_STRUC_ID	"EPH↵"
MQEPH_STRUC_ID_ARRAY	'E', 'P', 'H', '↵'

참고: ↵ 기호는 단일 공백 문자를 나타냅니다.

표 161. 상수 값		
이름	10진수 값	16진수 값
MQEPH_STRUC_LENGTH_FIXED	68	X'00000044'
MQEPH_VERSION_1	1	X'00000001'
MQEPH_CURRENT_VERSION	1	X'00000001'

### 임베드된 명령 형식 헤더 플래그

표 162. 상수 값		
이름	10진수 값	16진수 값
MQEPH_NONE	0	X'00000000'
MQEPH_CCSID_EMBEDDED	1	X'00000001'

## MQET\_\*(명령 형식 이스케이프 유형)

표 163. 상수 값		
이름	10진수 값	16진수 값
MQET_MQSC	1	X'00000001'

## MQEVO\_\*(명령 형식 이벤트 원본)

표 164. 상수 값		
이름	10진수 값	16진수 값
MQEVO_OTHER	0	X'00000000'
MQEVO_CONSOLE	1	X'00000001'
MQEVO_INIT	2	X'00000002'
MQEVO_MSG	3	X'00000003'

표 164. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQEVO_MQSET	4	X'00000004'
MQEVO_INTERNAL	5	X'00000005'
MQEVO_MQSUB	6	X'00000006'
MQEVO_CTLMSG	7	X'00000007'
MQEVO_REST	8	X'00000008'

### MQEVR\_\*(명령 형식 이벤트 레코딩)

표 165. 상수 값		
이름	10진수 값	16진수 값
MQEVR_DISABLED	0	X'00000000'
MQEVR_ENABLED	1	X'00000001'
MQEVR_EXCEPTION	2	X'00000002'
MQEVR_NO_DISPLAY	3	X'00000003'

### MQEXPI\_\*(만기 스캔 간격)

표 166. 상수 값		
이름	10진수 값	16진수 값
MQEXPI_OFF	0	X'00000000'

### MQFB\_\*(피드백 값)

표 167. 상수 값		
이름	10진수 값	16진수 값
MQFB_NONE	0	X'00000000'
MQFB_SYSTEM_FIRST	1	X'00000001'
MQFB_QUIT	256	X'00000100'
MQFB_EXPIRATION	258	X'00000102'
MQFB_COA	259	X'00000103'
MQFB_COD	260	X'00000104'
MQFB_CHANNEL_COMPLETED	262	X'00000106'
MQFB_CHANNEL_FAIL_RETRY	263	X'00000107'
MQFB_CHANNEL_FAIL	264	X'00000108'
MQFB_APPL_CANNOT_BE_STARTED	265	X'00000109'
MQFB_TM_ERROR	266	X'0000010A'
MQFB_APPL_TYPE_ERROR	267	X'0000010B'
MQFB_STOPPED_BY_MSG_EXIT	268	X'0000010C'
MQFB_ACTIVITY	269	X'0000010D'
MQFB_XMIT_Q_MSG_ERROR	271	X'0000010F'
MQFB_PAN	275	X'00000113'
MQFB_NAN	276	X'00000114'
MQFB_STOPPED_BY_CHAD_EXIT	277	X'00000115'

표 167. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQFB_STOPPED_BY_PUBSUB_EXIT	279	X'00000117'
MQFB_NOT_A_REPOSITORY_MSG	280	X'00000118'
MQFB_BIND_OPEN_CLUSRCVR_DEL	281	X'00000119'
MQFB_MAX_ACTIVITIES	282	X'0000011A'
MQFB_NOT_FORWARDED	283	X'0000011B'
MQFB_NOT_DELIVERED	284	X'0000011C'
MQFB_UNSUPPORTED_FORWARDING	285	X'0000011D'
MQFB_UNSUPPORTED_DELIVERY	286	X'0000011E'
MQFB_DATA_LENGTH_ZERO	291	X'00000123'
MQFB_DATA_LENGTH_NEGATIVE	292	X'00000124'
MQFB_DATA_LENGTH_TOO_BIG	293	X'00000125'
MQFB_BUFFER_OVERFLOW	294	X'00000126'
MQFB_LENGTH_OFF_BY_ONE	295	X'00000127'
MQFB_IIH_ERROR	296	X'00000128'
MQFB_NOT_AUTHORIZED_FOR_IMS	298	X'0000012A'
MQFB_IMS_ERROR	300	X'0000012C'
MQFB_IMS_FIRST	301	X'0000012D'
MQFB_IMS_LAST	399	X'0000018F'
MQFB_CICS_INTERNAL_ERROR	401	X'00000191'
MQFB_CICS_NOT_AUTHORIZED	402	X'00000192'
MQFB_CICS_BRIDGE_FAILURE	403	X'00000193'
MQFB_CICS_CORREL_ID_ERROR	404	X'00000194'
MQFB_CICS_CCSDID_ERROR	405	X'00000195'
MQFB_CICS_ENCODING_ERROR	406	X'00000196'
MQFB_CICS_CIH_ERROR	407	X'00000197'
MQFB_CICS_UOW_ERROR	408	X'00000198'
MQFB_CICS_COMMAREA_ERROR	409	X'00000199'
MQFB_CICS_APPL_NOT_STARTED	410	X'0000019A'
MQFB_CICS_APPL_ABENDED	411	X'0000019B'
MQFB_CICS_DLQ_ERROR	412	X'0000019C'
MQFB_CICS_UOW_BACKED_OUT	413	X'0000019D'
MQFB_PUBLICATIONS_ON_REQUEST	501	X'000001F5'
MQFB_SUBSCRIBER_IS_PUBLISHER	502	X'000001F6'
MQFB_MSG_SCOPE_MISMATCH	503	X'000001F7'
MQFB_SELECTOR_MISMATCH	504	X'000001F8'
MQFB_IMS_NACK_1A_REASON_FIRST	600	X'00000258'
MQFB_IMS_NACK_1A_REASON_LAST	855	X'00000357'
MQFB_SYSTEM_LAST	65535	X'0000FFFF'
MQFB_APPL_FIRST	65536	X'00010000'

표 167. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQFB_APPL_LAST	999999999	X'3B9AC9FF'

### MQFC\_\*(명령 형식 강제 실행 옵션)

표 168. 상수 값		
이름	10진수 값	16진수 값
MQFC_YES	1	X'00000001'
MQFC_NO	0	X'00000000'

### MQFMT\_\*(형식)

표 169. 상수 이름 및 값	
이름	값
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_EMBEDDED_PCF	"MQHEPCF~"
MQFMT_EVENT	"MQEVENT~"
MQFMT_IMS	"MQIMS~~~"
MQFMT_IMS_VAR_STRING	"MQIMSVS~"
MQFMT_MD_EXTENSION	"MQHMDE~"
MQFMT_PCF	"MQPCF~~~"
MQFMT_REF_MSG_HEADER	"MQHREF~"
MQFMT_RF_HEADER	"MQHRF~~~"
MQFMT_RF_HEADER_1	"MQHRF~~~"
MQFMT_RF_HEADER_2	"MQHRF2~~"
MQFMT_STRING	"MQSTR~~~"
MQFMT_TRIGGER	"MQTRIG~"
MQFMT_WORK_INFO_HEADER	"MQHWIH~"
MQFMT_XMIT_Q_HEADER	"MQXMIT~"
MQFMT_NONE_ARRAY	'~','~','~','~','~','~','~','~','~'
MQFMT_ADMIN_ARRAY	'M','Q','A','D','M','I','N','~'
MQFMT_CHANNEL_COMPLETED_ARRAY	'M','Q','C','H','C','O','M','~'
MQFMT_CICS_ARRAY	'M','Q','C','I','C','S','~','~'
MQFMT_COMMAND_1_ARRAY	'M','Q','C','M','D','1','~','~'
MQFMT_COMMAND_2_ARRAY	'M','Q','C','M','D','2','~','~'

표 169. 상수 이름 및 값 (계속)	
이름	값
MQFMT_DEAD_LETTER_HEADER_ARRAY	'M','Q','D','E','A','D','-', '-'
MQFMT_DIST_HEADER_ARRAY	'M','Q','H','D','I','S','T','-', '-'
MQFMT_EMBEDDED_PCF_ARRAY	'M','Q','H','E','P','C','F','-', '-'
MQFMT_EVENT_ARRAY	'M','Q','E','V','E','N','T','-', '-'
MQFMT_IMS_ARRAY	'M','Q','I','M','S','-', '-', '-'
MQFMT_IMS_VAR_STRING_ARRAY	'M','Q','I','M','S','V','S','-', '-'
MQFMT_MD_EXTENSION_ARRAY	'M','Q','H','M','D','E','-', '-'
MQFMT_PCF_ARRAY	'M','Q','P','C','F','-', '-', '-'
MQFMT_REF_MSG_HEADER_ARRAY	'M','Q','H','R','E','F','-', '-'
MQFMT_RF_HEADER_ARRAY	'M','Q','H','R','F','-', '-', '-'
MQFMT_RF_HEADER_1_ARRAY	'M','Q','H','R','F','-', '-', '-'
MQFMT_RF_HEADER_2_ARRAY	'M','Q','H','R','F','2','-', '-'
MQFMT_STRING_ARRAY	'M','Q','S','T','R','-', '-', '-'
MQFMT_TRIGGER_ARRAY	'M','Q','T','R','I','G','-', '-'
MQFMT_WORK_INFO_HEADER_ARRAY	'M','Q','H','W','I','H','-', '-'
MQFMT_XMIT_Q_HEADER_ARRAY	'M','Q','X','M','I','T','-', '-'

참고: - 기호는 단일 공백 문자를 나타냅니다.

### MQFUN\_\*(애플리케이션 함수 유형)

표 170. 상수 값		
이름	10진수 값	16진수 값
MQFUN_TYPE_UNKNOWN	0	X'00000000'
MQFUN_TYPE_JVM	1	X'00000001'
MQFUN_TYPE_PROGRAM	2	X'00000002'
MQFUN_TYPE_PROCEDURE	3	X'00000003'
MQFUN_TYPE_USERDEF	4	X'00000004'
MQFUN_TYPE_COMMAND	5	X'00000005'

### MQGA\_\*(그룹 속성 선택자)

표 171. 상수 값		
이름	10진수 값	16진수 값
MQGA_FIRST	8001	X'00001F41'
MQGA_LAST	9000	X'00002328'

### MQGACF\_\*(명령 형식 그룹 매개변수 유형)

표 172. 상수 값		
이름	10진수 값	16진수 값
MQGACF_FIRST	8001	X'00001F41'
MQGACF_COMMAND_CONTEXT	8001	X'00001F41'



표 172. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQGACF_COMMAND_DATA	8002	X'00001F42'
MQGACF_TRACE_ROUTE	8003	X'00001F43'
MQGACF_OPERATION	8004	X'00001F44'
MQGACF_ACTIVITY	8005	X'00001F45'
MQGACF_EMBEDDED_MQMD	8006	X'00001F46'
MQGACF_MESSAGE	8007	X'00001F47'
MQGACF_MQMD	8008	X'00001F48'
MQGACF_VALUE_NAMING	8009	X'00001F49'
MQGACF_Q_ACCOUNTING_DATA	8010	X'00001F4A'
MQGACF_Q_STATISTICS_DATA	8011	X'00001F4B'
MQGACF_CHL_STATISTICS_DATA	8012	X'00001F4C'
MQGACF_LAST_USED	8012	X'00001F4C'

### MQGI\_\*(그룹 ID)

표 173. 상수 이름 및 값	
이름	값
MQGI_NONE	X'00...00' (24널)
MQGI_NONE_ARRAY	'\0', '\0', ... (24널)

### MQGMO\_\*(메시지 가져오기 옵션 및 구조)

#### 메시지 가져오기 옵션 구조

표 174. 상수 구조	
이름	구조
MQGMO_STRUC_ID	"GMO~"
MQGMO_STRUC_ID_ARRAY	'G', 'M', 'O', '~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 175. 상수 값		
이름	10진수 값	16진수 값
MQGMO_VERSION_1	1	X'00000001'
MQGMO_VERSION_2	2	X'00000002'
MQGMO_VERSION_3	3	X'00000003'
MQGMO_VERSION_4	4	X'00000004'
MQGMO_CURRENT_VERSION	4	X'00000004'

#### 메시지 가져오기 옵션

표 176. 상수 값		
이름	10진수 값	16진수 값
MQGMO_WAIT	1	X'00000001'

표 176. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQGMO_NO_WAIT	0	X'00000000'
MQGMO_SET_SIGNAL	8	X'00000008'
MQGMO_FAIL_IF QUIESCING	8192	X'00002000'
MQGMO_SYNCPOINT	2	X'00000002'
MQGMO_SYNCPOINT_IF _PERSISTENT	4096	X'00001000'
MQGMO_NO_SYNCPOINT	4	X'00000004'
MQGMO_MARK_SKIP _BACKOUT	128	X'00000080'
MQGMO_BROWSE_FIRST	16	X'00000010'
MQGMO_BROWSE_NEXT	32	X'00000020'
MQGMO_BROWSE_MSG_UNDER_CURSOR	2048	X'00008000'
MQGMO_BROWSE_HANDLE	17825808	X'01100010'
MQGMO_BROWSE_CO_OP	18874384	X'01200010'
MQGMO_MSG_UNDER_CURSOR	256	X'00000100'
MQGMO_LOCK	512	X'00000200'
MQGMO_UNLOCK	1024	X'00000400'
MQGMO_ACCEPT_TRUNCATED_MSG	64	X'00000040'
MQGMO_CONVERT	16384	X'00004000'
MQGMO_LOGICAL_ORDER	32768	X'00008000'
MQGMO_COMPLETE_MSG	65536	X'00010000'
MQGMO_ALL_MSGS_AVAILABLE	131072	X'00020000'
MQGMO_ALL_SEGMENTS_AVAILABLE	262144	X'00040000'
MQGMO_MARK_BROWSE_HANDLE	1048576	X'00100000'
MQGMO_MARK_BROWSE_CO_OP	2097152	X'00200000'
MQGMO_UNMARK_BROWSE_CO_OP	4194304	X'00400000'
MQGMO_UNMARK_BROWSE_HANDLE	8388608	X'00800000'
MQGMO_UNMARKED_BROWSE_MSG	16777216	X'01000000'
MQGMO_PROPERTIES_FORCE_MQRFH2	33554432	X'02000000'
MQGMO_NO_PROPERTIES	67108864	X'04000000'
MQGMO_PROPERTIES_IN_HANDLE	134217728	X'08000000'
MQGMO_PROPERTIES_COMPATIBILITY	268435456	X'10000000'
MQGMO_PROPERTIES_AS_Q_DEF	0	X'00000000'
MQGMO_NONE	0	X'00000000'

### MQGS\_\*(그룹 상태)

표 177. 상수 이름 및 값	
이름	값
MQGS_NOT_IN_GROUP	'N'
MQGS_MSG_IN_GROUP	'G'
MQGS_LAST_MSG_IN_GROUP	'L'

참고: ↵ 기호는 단일 공백 문자를 나타냅니다.

### MQHA\_\*(핸들 선택자)

표 178. 상수 값		
이름	10진수 값	16진수 값
MQHA_FIRST	4001	X'00000FA1'
MQHA_BAG_HANDLE	4001	X'00000FA1'
MQHA_LAST_USED	4001	X'00000FA1'
MQHA_LAST	6000	X'00001770'

### MQHB\_\*(백 핸들)

표 179. 상수 값		
이름	10진수 값	16진수 값
MQHB_UNUSABLE_HBAG	-1	X'FFFFFFFF'
MQHB_NONE	-2	X'FFFFFFFE'

### MQHC\_\*(연결 핸들)

표 180. 상수 값		
이름	10진수 값	16진수 값
MQHC_DEF_HCONN	0	X'00000000'
MQHC_UNUSABLE_HCONN	-1	X'FFFFFFFF'
MQHC_UNASSOCIATED_HCONN	-3	X'FFFFFFFD'

### MQHM\_\*(메시지 핸들)

표 181. 상수 값		
이름	10진수 값	16진수 값
MQHM_UNUSABLE_HMSG	-1	X'FFFFFFFF'
MQHM_NONE	0	X'00000000'

### MQHO\_\*(오브젝트 핸들)

표 182. 상수 값		
이름	10진수 값	16진수 값
MQHO_UNUSABLE_HOBJ	-1	X'FFFFFFFF'
MQHO_NONE	0	X'00000000'

### MQHSTATE\_\*(명령 형식 핸들 상태)

표 183. 상수 값		
이름	10진수 값	16진수 값
MQHSTATE_INACTIVE	0	X'00000000'
MQHSTATE_ACTIVE	1	X'00000001'

## MQIA\_\*(정수 속성 선택자)

표 184. 상수 값		
이름	10진수 값	16진수 값
MQIA_ACCOUNTING_CONN_OVERRIDE	136	X'00000088'
MQIA_ACCOUNTING_INTERVAL	135	X'00000087'
MQIA_ACCOUNTING_MQI	133	X'00000085'
MQIA_ACCOUNTING_Q	134	X'00000086'
MQIA_ACTIVE_CHANNELS	100	X'00000064'
MQIA_ACTIVITY_CONN_OVERRIDE	239	X'000000EF'
MQIA_ACTIVITY_RECORDING	138	X'0000008A'
MQIA_ACTIVITY_TRACE	240	X'000000F0'
MQIA_ADOPTNEWMCA_CHECK	102	X'00000066'
MQIA_ADOPTNEWMCA_INTERVAL	104	X'00000068'
MQIA_ADOPTNEWMCA_TYPE	103	X'00000067'
MQIA_ADOPT_CONTEXT	260	X'00000104'
<del>MQ Adv. VUE</del> <del>MQ Adv. VUE</del> MQIA_ADVANCED_CAPABILITY	273	X'00000111'
MQIA_AMQP_CAPABILITY	265	X'00000109'
MQIA_APPL_TYPE	1	X'00000001'
MQIA_ARCHIVE	60	X'0000003C'
MQIA_AUTHENTICATION_FAIL_DELAY	259	X'00000103'
MQIA_AUTHENTICATION_METHOD	266	X'0000010A'
MQIA_AUTH_INFO_TYPE	66	X'00000042'
MQIA_AUTHORITY_EVENT	47	X'0000002F'
MQIA_AUTO_REORG_INTERVAL	174	X'000000AE'
MQIA_AUTO_REORGANIZATION	173	X'000000AD'
MQIA_BACKOUT_THRESHOLD	22	X'00000016'
MQIA_BASE_TYPE	193	X'000000C1'
MQIA_BATCH_INTERFACE_AUTO	86	X'00000056'
MQIA_BRIDGE_EVENT	74	X'0000004A'
MQIA_CF_LEVEL	70	X'00000046'
MQIA_CF_RECOVER	71	X'00000047'
MQIA_CHANNEL_AUTO_DEF	55	X'00000037'
MQIA_CHANNEL_AUTO_DEF_EVENT	56	X'00000038'
MQIA_CHANNEL_EVENT	73	X'00000049'
MQIA_CHECK_CLIENT_BINDING	258	X'00000102'
MQIA_CHECK_LOCAL_BINDING	257	X'00000101'
MQIA_CHINIT_ADAPTERS	101	X'00000065'
MQIA_CHINIT_CONTROL	119	X'00000077'
MQIA_CHINIT_DISPATCHERS	105	X'00000069'
MQIA_CHINIT_TRACE_AUTO_START	117	X'00000075'
MQIA_CHINIT_TRACE_TABLE_SIZE	118	X'00000076'

표 184. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIA_CLUSTER_OBJECT_STATE	256	X'00000100'
MQIA_CLUSTER_PUB_ROUTE	255	X'000000FF'
MQIA_CLUSTER_Q_TYPE	59	X'0000003B'
MQIA_CLUSTER_WORKLOAD_LENGTH	58	X'0000003A'
MQIA_CLWL_MRU_CHANNELS	97	X'00000061'
MQIA_CLWL_Q_RANK	95	X'0000005F'
MQIA_CLWL_Q_PRIORITY	96	X'00000060'
MQIA_CLWL_USEQ	98	X'00000062'
MQIA_CMD_SERVER_AUTO	87	X'00000057'
MQIA_CMD_SERVER_CONTROL	120	X'00000078'
MQIA_CMD_SERVER_CONVERT_MSG	88	X'00000058'
MQIA_CMD_SERVER_DLQ_MSG	89	X'00000059'
MQIA_CODED_CHAR_SET_ID	2	X'00000002'
MQIA_COMM_EVENT	232	X'000000E8'
MQIA_COMMAND_EVENT	99	X'00000063'
MQIA_COMMAND_LEVEL	31	X'0000001F'
MQIA_CONFIGURATION_EVENT	51	X'00000033'
MQIA_CPI_LEVEL	27	X'0000001B'
MQIA_CURRENT_Q_DEPTH	3	X'00000003'
MQIA_DEF_BIND	61	X'0000003D'
MQIA_DEF_CLUSTER_XMIT_Q_TYPE	250	X'000000FA'
MQIA_DEF_INPUT_OPEN_OPTION	4	X'00000004'
MQIA_DEF_PERSISTENCE	5	X'00000005'
MQIA_DEF_PRIORITY	6	X'00000006'
MQIA_DEF_PUT_RESPONSE_TYPE	184	X'000000B8'
MQIA_DEF_READ_AHEAD	188	X'000000BC'
MQIA_DEFINITION_TYPE	7	X'00000007'
MQIA_DISPLAY_TYPE	262	X'00000106'
MQIA_DIST_LISTS	34	X'00000022'
MQIA_DNS_WLM	106	X'0000006A'
MQIA_DURABLE_SUB	175	X'000000AF'
MQIA_EXPIRY_INTERVAL	39	X'00000027'
MQIA_FIRST	1	X'00000001'
 MQIA_GROUP_UR	221	X'000000DD'
MQIA_HARDEN_GET_BACKOUT	8	X'00000008'
MQIA_HIGH_Q_DEPTH	36	X'00000024'
MQIA_IGQ_PUT_AUTHORITY	65	X'00000041'
MQIA_INDEX_TYPE	57	X'00000039'
MQIA_INHIBIT_EVENT	48	X'00000030'

표 184. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIA_INHIBIT_GET	9	X'00000009'
MQIA_INHIBIT_PUB	181	X'000000B5'
MQIA_INHIBIT_PUT	10	X'0000000A'
MQIA_INHIBIT_SUB	182	X'000000B6'
MQIA_INTRA_GROUP_QUEUING	64	X'00000040'
MQIA_IP_ADDRESS_VERSION	93	X'0000005D'
MQIA_KEY_REUSE_COUNT	267	X'0000010B'
MQIA_LAST	2000	X'000007D0'
MQIA_LAST_USED	267	X'0000010B'
MQIA_LDAP_AUTHORMD	263	X'00000107'
MQIA_LDAP_NESTGRP	264	X'00000108'
MQIA_LDAP_SECURE_COMM	261	X'00000105'
MQIA_LISTENER_PORT_NUMBER	85	X'00000055'
MQIA_LISTENER_TIMER	107	X'0000006B'
MQIA_LOGGER_EVENT	94	X'0000005E'
MQIA_LU62_CHANNELS	108	X'0000006C'
MQIA_LOCAL_EVENT	49	X'00000031'
MQIA_MSG_MARK_BROWSE_INTERVAL	68	X'00000044'
MQIA_MAX_CHANNELS	109	X'0000006D'
MQIA_MAX_CLIENTS	172	X'000000AC'
MQIA_MAX_GLOBAL_LOCKS	83	X'00000053'
MQIA_MAX_HANDLES	11	X'0000000B'
MQIA_MAX_LOCAL_LOCKS	84	X'00000054'
MQIA_MAX_MSG_LENGTH	13	X'0000000D'
MQIA_MAX_OPEN_Q	80	X'00000050'
MQIA_MAX_PRIORITY	14	X'0000000E'
MQIA_MAX_PROPERTIES_LENGTH	192	X'000000C0'
MQIA_MAX_Q_DEPTH	15	X'0000000F'
MQIA_MAX_Q_TRIGGERS	90	X'0000005A'
MQIA_MAX_RECOVERY_TASKS	171	X'000000AB'
MQIA_MAX_UNCOMMITTED_MSGS	33	X'00000021'
MQIA_MCAST_BRIDGE	233	X'000000E9'
MQIA_MONITOR_INTERVAL	81	X'00000051'
MQIA_MONITORING_AUTO_CLUSSDR	124	X'0000007C'
MQIA_MONITORING_CHANNEL	122	X'0000007A'
MQIA_MONITORING_Q	123	X'0000007B'
MQIA_MSG_DELIVERY_SEQUENCE	16	X'00000010'
MQIA_MSG_DEQ_COUNT	38	X'00000026'
MQIA_MSG_ENQ_COUNT	37	X'00000025'

표 184. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIA_NAME_COUNT	19	X'00000013'
MQIA_NAMELIST_TYPE	72	X'00000048'
MQIA_NPM_CLASS	78	X'0000004E'
MQIA_NPM_DELIVERY	196	X'000000C4'
MQIA_OPEN_INPUT_COUNT	17	X'00000011'
MQIA_OPEN_OUTPUT_COUNT	18	X'00000012'
MQIA_OUTBOUND_PORT_MAX	140	X'0000008C'
MQIA_OUTBOUND_PORT_MIN	110	X'0000006E'
MQIA_PAGESET_ID	62	X'0000003E'
MQIA_PERFORMANCE_EVENT	53	X'00000035'
MQIA_PLATFORM	32	X'00000020'
MQIA_PM_DELIVERY	195	X'000000C3'
MQIA_PROPERTY_CONTROL	190	X'000000BE'
MQIA_PROT_POLICY_CAPABILITY	251	X'000000FB'
MQIA_PROXY_SUB	199	X'000000C7'
MQIA_PUB_COUNT	215	X'000000D7'
MQIA_PUB_SCOPE	219	X'000000DB'
MQIA_PUBSUB_CLUSTER	249	X'000000F9'
MQIA_PUBSUB_MAXMSG_RETRY_COUNT	206	X'000000CE'
MQIA_PUBSUB_MODE	187	X'000000BB'
MQIA_PUBSUB_NP_MSG	203	X'000000CB'
MQIA_PUBSUB_NP_RESP	205	X'000000CD'
MQIA_PUBSUB_SYNC_PT	207	X'000000CF'
MQIA_Q_DEPTH_HIGH_EVENT	43	X'0000002B'
MQIA_Q_DEPTH_HIGH_LIMIT	40	X'00000028'
MQIA_Q_DEPTH_LOW_EVENT	44	X'0000002C'
MQIA_Q_DEPTH_LOW_LIMIT	41	X'00000029'
MQIA_Q_DEPTH_MAX_EVENT	42	X'0000002A'
MQIA_Q_SERVICE_INTERVAL	54	X'00000036'
MQIA_Q_SERVICE_INTERVAL_EVENT	46	X'0000002E'
MQIA_Q_TYPE	20	X'00000014'
MQIA_Q_USERS	82	X'00000052'
MQIA_QMGR_CFCNLOS	245	X'000000F5'
MQIA_QMOPT_CONS_COMMS_MSGS	155	X'0000009B'
MQIA_QMOPT_CONS_CRITICAL_MSGS	154	X'0000009A'
MQIA_QMOPT_CONS_ERROR_MSGS	153	X'00000099'
MQIA_QMOPT_CONS_INFO_MSGS	151	X'00000097'
MQIA_QMOPT_CONS_REORG_MSGS	156	X'0000009C'
MQIA_QMOPT_CONS_SYSTEM_MSGS	157	X'0000009D'

표 184. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIA_QMOPT_CONS_WARNING_MSGS	152	X'00000098'
MQIA_QMOPT_CSMT_ON_ERROR	150	X'00000096'
MQIA_QMOPT_INTERNAL_DUMP	170	X'000000AA'
MQIA_QMOPT_LOG_COMMS_MSGS	162	X'000000A2'
MQIA_QMOPT_LOG_CRITICAL_MSGS	161	X'000000A1'
MQIA_QMOPT_LOG_ERROR_MSGS	160	X'000000A0'
MQIA_QMOPT_LOG_INFO_MSGS	158	X'0000009E'
MQIA_QMOPT_LOG_REORG_MSGS	163	X'000000A3'
MQIA_QMOPT_LOG_SYSTEM_MSGS	164	X'000000A4'
MQIA_QMOPT_LOG_WARNING_MSGS	159	X'0000009F'
MQIA_QMOPT_TRACE_COMMS	166	X'000000A6'
MQIA_QMOPT_TRACE_CONVERSION	168	X'000000A8'
MQIA_QMOPT_TRACE_REORG	167	X'000000A7'
MQIA_QMOPT_TRACE_MQI_CALLS	165	X'000000A5'
MQIA_QMOPT_TRACE_SYSTEM	169	X'000000A9'
MQIA_QSG_DISP	63	X'0000003F'
MQIA_READ_AHEAD	189	X'000000BD'
MQIA_RECEIVE_TIMEOUT	111	X'0000006F'
MQIA_RECEIVE_TIMEOUT_MIN	113	X'00000071'
MQIA_RECEIVE_TIMEOUT_TYPE	112	X'00000070'
MQIA_REMOTE_EVENT	50	X'00000032'
MQIA_RETENTION_INTERVAL	21	X'00000015'
MQIA_REVERSE_DNS_LOOKUP	254	X'000000FE'
MQIA_SCOPE	45	X'0000002D'
MQIA_SECURITY_CASE	141	X'0000008D'
MQIA_SERVICE_CONTROL	139	X'0000008B'
MQIA_SERVICE_TYPE	121	X'00000079'
MQIA_SHAREABILITY	23	X'00000017'
MQIA_SHARED_Q_Q_MGR_NAME	77	X'0000004D'
MQIA_SSL_EVENT	75	X'0000004B'
MQIA_SSL_FIPS_REQUIRED	92	X'0000005C'
MQIA_SSL_RESET_COUNT	76	X'0000004C'
MQIA_SSL_TASKS	69	X'00000045'
MQIA_START_STOP_EVENT	52	X'00000034'
MQIA_STATISTICS_CHANNEL	129	X'00000081'
MQIA_STATISTICS_AUTO_CLUSSDR	130	X'00000082'
MQIA_STATISTICS_INTERVAL	131	X'00000083'
MQIA_STATISTICS_MQI	127	X'0000007F'
MQIA_STATISTICS_Q	128	X'00000080'



표 184. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIA_SUB_COUNT	204	X'000000CC'
MQIA_SUB_SCOPE	218	X'000000DA'
MQIA_SYNCPOINT	30	X'0000001E'
MQIA_TCP_CHANNELS	114	X'00000072'
MQIA_TCP_KEEP_ALIVE	115	X'00000073'
MQIA_TCP_STACK_TYPE	116	X'00000074'
MQIA_TIME_SINCE_RESET	35	X'00000023'
MQIA_TOPIC_DEF_PERSISTENCE	185	X'000000B9'
MQIA_TOPIC_NODE_COUNT	253	X'000000FD'
MQIA_TOPIC_TYPE	208	X'000000D0'
MQIA_TRACE_ROUTE_RECORDING	137	X'00000089'
MQIA_TREE_LIFE_TIME	183	X'000000B7'
MQIA_TRIGGER_CONTROL	24	X'00000018'
MQIA_TRIGGER_DEPTH	29	X'0000001D'
MQIA_TRIGGER_INTERVAL	25	X'00000019'
MQIA_TRIGGER_MSG_PRIORITY	26	X'0000001A'
MQIA_TRIGGER_TYPE	28	X'0000001C'
MQIA_TRIGGER_RESTART	91	X'0000005B'
MQIA_USAGE	12	X'0000000C'
MQIA_USE_DEAD_LETTER_Q	234	X'000000EA'
MQIA_USER_LIST	2000	X'000007D0'
MQIA_WILDCARD_OPERATION	216	X'000000D8'
MQIA_XR_CAPABILITY	243	X'000000F3'

### MQIACF\_\*(명령 형식 정수 매개변수 유형)

표 185. 상수 값		
이름	10진수 값	16진수 값
MQIACF_FIRST	1001	X'000003E9'
MQIACF_Q_MGR_ATTRS	1001	X'000003E9'
MQIACF_Q_ATTRS	1002	X'000003EA'
MQIACF_PROCESS_ATTRS	1003	X'000003EB'
MQIACF_NAMELIST_ATTRS	1004	X'000003EC'
MQIACF_FORCE	1005	X'000003ED'
MQIACF_REPLACE	1006	X'000003EE'
MQIACF_PURGE	1007	X'000003EF'
MQIACF QUIESCE	1008	X'000003F0'
MQIACF_MODE	1008	X'000003F0'
MQIACF_ALL	1009	X'000003F1'
MQIACF_EVENT_APPL_TYPE	1010	X'000003F2'

표 185. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIACF_EVENT_ORIGIN	1011	X'000003F3'
MQIACF_PARAMETER_ID	1012	X'000003F4'
MQIACF_ERROR_ID	1013	X'000003F5'
MQIACF_ERROR_IDENTIFIER	1013	X'000003F5'
MQIACF_SELECTOR	1014	X'000003F6'
MQIACF_CHANNEL_ATTRS	1015	X'000003F7'
MQIACF_OBJECT_TYPE	1016	X'000003F8'
MQIACF_ESCAPE_TYPE	1017	X'000003F9'
MQIACF_ERROR_OFFSET	1018	X'000003FA'
MQIACF_AUTH_INFO_ATTRS	1019	X'000003FB'
MQIACF_REASON_QUALIFIER	1020	X'000003FC'
MQIACF_COMMAND	1021	X'000003FD'
MQIACF_OPEN_OPTIONS	1022	X'000003FE'
MQIACF_OPEN_TYPE	1023	X'000003FF'
MQIACF_PROCESS_ID	1024	X'00000400'
MQIACF_THREAD_ID	1025	X'00000401'
MQIACF_Q_STATUS_ATTRS	1026	X'00000402'
MQIACF_UNCOMMITTED_MSGS	1027	X'00000403'
MQIACF_HANDLE_STATE	1028	X'00000404'
MQIACF_AUX_ERROR_DATA_INT_1	1070	X'0000042E'
MQIACF_AUX_ERROR_DATA_INT_2	1071	X'0000042F'
MQIACF_CONV_REASON_CODE	1072	X'00000430'
MQIACF_BRIDGE_TYPE	1073	X'00000431'
MQIACF_INQUIRY	1074	X'00000432'
MQIACF_WAIT_INTERVAL	1075	X'00000433'
MQIACF_OPTIONS	1076	X'00000434'
MQIACF_BROKER_OPTIONS	1077	X'00000435'
MQIACF_REFRESH_TYPE	1078	X'00000436'
MQIACF_SEQUENCE_NUMBER	1079	X'00000437'
MQIACF_INTEGER_DATA	1080	X'00000438'
MQIACF_REGISTRATION_OPTIONS	1081	X'00000439'
MQIACF_PUBLICATION_OPTIONS	1082	X'0000043A'
MQIACF_CLUSTER_INFO	1083	X'0000043B'
MQIACF_Q_MGR_DEFINITION_TYPE	1084	X'0000043C'
MQIACF_Q_MGR_TYPE	1085	X'0000043D'
MQIACF_ACTION	1086	X'0000043E'
MQIACF_SUSPEND	1087	X'0000043F'
MQIACF_BROKER_COUNT	1088	X'00000440'
MQIACF_APPL_COUNT	1089	X'00000441'

표 185. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIACF_ANONYMOUS_COUNT	1090	X'00000442'
MQIACF_REG_REG_OPTIONS	1091	X'00000443'
MQIACF_DELETE_OPTIONS	1092	X'00000444'
MQIACF_CLUSTER_Q_MGR_ATTRS	1093	X'00000445'
MQIACF_REFRESH_INTERVAL	1094	X'00000446'
MQIACF_REFRESH_REPOSITORY	1095	X'00000447'
MQIACF_REMOVE_QUEUES	1096	X'00000448'
MQIACF_OPEN_INPUT_TYPE	1098	X'0000044A'
MQIACF_OPEN_OUTPUT	1099	X'0000044B'
MQIACF_OPEN_SET	1100	X'0000044C'
MQIACF_OPEN_INQUIRE	1101	X'0000044D'
MQIACF_OPEN_BROWSE	1102	X'0000044E'
MQIACF_Q_STATUS_TYPE	1103	X'0000044F'
MQIACF_Q_HANDLE	1104	X'00000450'
MQIACF_Q_STATUS	1105	X'00000451'
MQIACF_SECURITY_TYPE	1106	X'00000452'
MQIACF_CONNECTION_ATTRS	1107	X'00000453'
MQIACF_CONNECT_OPTIONS	1108	X'00000454'
MQIACF_CONN_INFO_TYPE	1110	X'00000456'
MQIACF_CONN_INFO_CONN	1111	X'00000457'
MQIACF_CONN_INFO_HANDLE	1112	X'00000458'
MQIACF_CONN_INFO_ALL	1113	X'00000459'
MQIACF_AUTH_PROFILE_ATTRS	1114	X'0000045A'
MQIACF_AUTHORIZATION_LIST	1115	X'0000045B'
MQIACF_AUTH_ADD_AUTHS	1116	X'0000045C'
MQIACF_AUTH_REMOVE_AUTHS	1117	X'0000045D'
MQIACF_ENTITY_TYPE	1118	X'0000045E'
MQIACF_COMMAND_INFO	1120	X'00000460'
MQIACF_CMDSCOPE_Q_MGR_COUNT	1121	X'00000461'
MQIACF_Q_MGR_SYSTEM	1122	X'00000462'
MQIACF_Q_MGR_EVENT	1123	X'00000463'
MQIACF_Q_MGR_DQM	1124	X'00000464'
MQIACF_Q_MGR_CLUSTER	1125	X'00000465'
MQIACF_QSG_DISPS	1126	X'00000466'
MQIACF_UOW_STATE	1128	X'00000468'
MQIACF_SECURITY_ITEM	1129	X'00000469'
MQIACF_CF_STRUC_STATUS	1130	X'0000046A'
MQIACF_UOW_TYPE	1132	X'0000046C'
MQIACF_CF_STRUC_ATTRS	1133	X'0000046D'

표 185. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIACF_EXCLUDE_INTERVAL	1134	X'0000046E'
MQIACF_CF_STATUS_TYPE	1135	X'0000046F'
MQIACF_CF_STATUS_SUMMARY	1136	X'00000470'
MQIACF_CF_STATUS_CONNECT	1137	X'00000471'
MQIACF_CF_STATUS_BACKUP	1138	X'00000472'
MQIACF_CF_STRUC_TYPE	1139	X'00000473'
MQIACF_CF_STRUC_SIZE_MAX	1140	X'00000474'
MQIACF_CF_STRUC_SIZE_USED	1141	X'00000475'
MQIACF_CF_STRUC_ENTRIES_MAX	1142	X'00000476'
MQIACF_CF_STRUC_ENTRIES_USED	1143	X'00000477'
MQIACF_CF_STRUC_BACKUP_SIZE	1144	X'00000478'
MQIACF_MOVE_TYPE	1145	X'00000479'
MQIACF_MOVE_TYPE_MOVE	1146	X'0000047A'
MQIACF_MOVE_TYPE_ADD	1147	X'0000047B'
MQIACF_Q_MGR_NUMBER	1148	X'0000047C'
MQIACF_Q_MGR_STATUS	1149	X'0000047D'
MQIACF_DB2_CONN_STATUS	1150	X'0000047E'
MQIACF_SECURITY_ATTRS	1151	X'0000047F'
MQIACF_SECURITY_TIMEOUT	1152	X'00000480'
MQIACF_SECURITY_INTERVAL	1153	X'00000481'
MQIACF_SECURITY_SWITCH	1154	X'00000482'
MQIACF_SECURITY_SETTING	1155	X'00000483'
MQIACF_STORAGE_CLASS_ATTRS	1156	X'00000484'
MQIACF_USAGE_TYPE	1157	X'00000485'
MQIACF_BUFFER_POOL_ID	1158	X'00000486'
MQIACF_USAGE_TOTAL_PAGES	1159	X'00000487'
MQIACF_USAGE_UNUSED_PAGES	1160	X'00000488'
MQIACF_USAGE_PERSIST_PAGES	1161	X'00000489'
MQIACF_USAGE_NONPERSIST_PAGES	1162	X'0000048A'
MQIACF_USAGE_RESTART_EXTENTS	1163	X'0000048B'
MQIACF_USAGE_EXPAND_COUNT	1164	X'0000048C'
MQIACF_PAGESET_STATUS	1165	X'0000048D'
MQIACF_USAGE_TOTAL_BUFFERS	1166	X'0000048E'
MQIACF_USAGE_DATA_SET_TYPE	1167	X'0000048F'
MQIACF_USAGE_PAGESET	1168	X'00000490'
MQIACF_USAGE_DATA_SET	1169	X'00000491'
MQIACF_USAGE_BUFFER_POOL	1170	X'00000492'
MQIACF_MOVE_COUNT	1171	X'00000493'
MQIACF_EXPIRY_Q_COUNT	1172	X'00000494'

표 185. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIACF_CONFIGURATION_OBJECTS	1173	X'00000495'
MQIACF_CONFIGURATION_EVENTS	1174	X'00000496'
MQIACF_SYSP_TYPE	1175	X'00000497'
MQIACF_SYSP_DEALLOC_INTERVAL	1176	X'00000498'
MQIACF_SYSP_MAX_ARCHIVE	1177	X'00000499'
MQIACF_SYSP_MAX_READ_TAPES	1178	X'0000049A'
MQIACF_SYSP_IN_BUFFER_SIZE	1179	X'0000049B'
MQIACF_SYSP_OUT_BUFFER_SIZE	1180	X'0000049C'
MQIACF_SYSP_OUT_BUFFER_COUNT	1181	X'0000049D'
MQIACF_SYSP_ARCHIVE	1182	X'0000049E'
MQIACF_SYSP_DUAL_ACTIVE	1183	X'0000049F'
MQIACF_SYSP_DUAL_ARCHIVE	1184	X'000004A0'
MQIACF_SYSP_DUAL_BSDS	1185	X'000004A1'
MQIACF_SYSP_MAX_CONNS	1186	X'000004A2'
MQIACF_SYSP_MAX_CONNS_FORE	1187	X'000004A3'
MQIACF_SYSP_MAX_CONNS_BACK	1188	X'000004A4'
MQIACF_SYSP_EXIT_INTERVAL	1189	X'000004A5'
MQIACF_SYSP_EXIT_TASKS	1190	X'000004A6'
MQIACF_SYSP_CHKPOINT_COUNT	1191	X'000004A7'
MQIACF_SYSP_OTMA_INTERVAL	1192	X'000004A8'
MQIACF_SYSP_Q_INDEX_DEFER	1193	X'000004A9'
MQIACF_SYSP_DB2_TASKS	1194	X'000004AA'
MQIACF_SYSP_RESLEVEL_AUDIT	1195	X'000004AB'
MQIACF_SYSP_ROUTING_CODE	1196	X'000004AC'
MQIACF_SYSP_SMF_ACCOUNTING	1197	X'000004AD'
MQIACF_SYSP_SMF_STATS	1198	X'000004AE'
MQIACF_SYSP_SMF_INTERVAL	1199	X'000004AF'
MQIACF_SYSP_TRACE_CLASS	1200	X'000004B0'
MQIACF_SYSP_TRACE_SIZE	1201	X'000004B1'
MQIACF_SYSP_WLM_INTERVAL	1202	X'000004B2'
MQIACF_SYSP_ALLOC_UNIT	1203	X'000004B3'
MQIACF_SYSP_ARCHIVE_RETAIN	1204	X'000004B4'
MQIACF_SYSP_ARCHIVE_WTOR	1205	X'000004B5'
MQIACF_SYSP_BLOCK_SIZE	1206	X'000004B6'
MQIACF_SYSP_CATALOG	1207	X'000004B7'
MQIACF_SYSP_COMPACT	1208	X'000004B8'
MQIACF_SYSP_ALLOC_PRIMARY	1209	X'000004B9'
MQIACF_SYSP_ALLOC_SECONDARY	1210	X'000004BA'
MQIACF_SYSP_PROTECT	1211	X'000004BB'

표 185. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIACF_SYSP_QUIESCE_INTERVAL	1212	X'000004BC'
MQIACF_SYSP_TIMESTAMP	1213	X'000004BD'
MQIACF_SYSP_UNIT_ADDRESS	1214	X'000004BE'
MQIACF_SYSP_UNIT_STATUS	1215	X'000004BF'
MQIACF_SYSP_LOG_COPY	1216	X'000004C0'
MQIACF_SYSP_LOG_USED	1217	X'000004C1'
MQIACF_SYSP_LOG_SUSPEND	1218	X'000004C2'
MQIACF_SYSP_OFFLOAD_STATUS	1219	X'000004C3'
MQIACF_SYSP_TOTAL_LOGS	1220	X'000004C4'
MQIACF_SYSP_FULL_LOGS	1221	X'000004C5'
MQIACF_LISTENER_ATTRS	1222	X'000004C6'
MQIACF_LISTENER_STATUS_ATTRS	1223	X'000004C7'
MQIACF_SERVICE_ATTRS	1224	X'000004C8'
MQIACF_SERVICE_STATUS_ATTRS	1225	X'000004C9'
MQIACF_Q_TIME_INDICATOR	1226	X'000004CA'
MQIACF_OLDEST_MSG_AGE	1227	X'000004CB'
MQIACF_AUTH_OPTIONS	1228	X'000004CC'
MQIACF_Q_MGR_STATUS_ATTRS	1229	X'000004CD'
MQIACF_CONNECTION_COUNT	1230	X'000004CE'
MQIACF_Q_MGR_FACILITY	1231	X'000004CF'
MQIACF_CHINIT_STATUS	1232	X'000004D0'
MQIACF_CMD_SERVER_STATUS	1233	X'000004D1'
MQIACF_ROUTE_DETAIL	1234	X'000004D2'
MQIACF_RECORDED_ACTIVITIES	1235	X'000004D3'
MQIACF_MAX_ACTIVITIES	1236	X'000004D4'
MQIACF_DISCONTINUITY_COUNT	1237	X'000004D5'
MQIACF_ROUTE_ACCUMULATION	1238	X'000004D6'
MQIACF_ROUTE_DELIVERY	1239	X'000004D7'
MQIACF_OPERATION_TYPE	1240	X'000004D8'
MQIACF_BACKOUT_COUNT	1241	X'000004D9'
MQIACF_COMP_CODE	1242	X'000004DA'
MQIACF_ENCODING	1243	X'000004DB'
MQIACF_EXPIRY	1244	X'000004DC'
MQIACF_FEEDBACK	1245	X'000004DD'
MQIACF_MSG_FLAGS	1247	X'000004DF'
MQIACF_MSG_LENGTH	1248	X'000004E0'
MQIACF_MSG_TYPE	1249	X'000004E1'
MQIACF_OFFSET	1250	X'000004E2'
MQIACF_ORIGINAL_LENGTH	1251	X'000004E3'

표 185. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIACF_PERSISTENCE	1252	X'000004E4'
MQIACF_PRIORITY	1253	X'000004E5'
MQIACF_REASON_CODE	1254	X'000004E6'
MQIACF_REPORT	1255	X'000004E7'
MQIACF_VERSION	1256	X'000004E8'
MQIACF_UNRECORDED_ACTIVITIES	1257	X'000004E9'
MQIACF_MONITORING	1258	X'000004EA'
MQIACF_ROUTE_FORWARDING	1259	X'000004EB'
MQIACF_SERVICE_STATUS	1260	X'000004EC'
MQIACF_Q_TYPES	1261	X'000004ED'
MQIACF_USER_ID_SUPPORT	1262	X'000004EE'
MQIACF_INTERFACE_VERSION	1263	X'000004EF'
MQIACF_AUTH_SERVICE_ATTRS	1264	X'000004F0'
MQIACF_USAGE_EXPAND_TYPE	1265	X'000004F1'
MQIACF_SYSP_CLUSTER_CACHE	1266	X'000004F2'
MQIACF_SYSP_DB2_BLOB_TASKS	1267	X'000004F3'
MQIACF_SYSP_WLM_INT_UNITS	1268	X'000004F4'
MQIACF_TOPIC_ATTRS	1269	X'000004F5'
MQIACF_PUBSUB_PROPERTIES	1271	X'000004F7'
MQIACF_DESTINATION_CLASS	1273	X'000004F9'
MQIACF_DURABLE_SUBSCRIPTION	1274	X'000004FA'
MQIACF_SUBSCRIPTION_SCOPE	1275	X'000004FB'
MQIACF_VARIABLE_USER_ID	1277	X'000004FD'
MQIACF_REQUEST_ONLY	1280	X'00000500'
MQIACF_PUB_PRIORITY	1283	X'00000503'
MQIACF_SUB_ATTRS	1287	X'00000507'
MQIACF_WILDCARD_SCHEMA	1288	X'00000508'
MQIACF_SUB_TYPE	1289	X'00000509'
MQIACF_MESSAGE_COUNT	1290	X'0000050A'
MQIACF_Q_MGR_PUBSUB	1291	X'0000050B'
MQIACF_Q_MGR_VERSION	1292	X'0000050C'
MQIACF_SUB_STATUS_ATTRS	1294	X'0000050E'
MQIACF_TOPIC_STATUS	1295	X'0000050F'
MQIACF_TOPIC_SUB	1296	X'00000510'
MQIACF_TOPIC_PUB	1297	X'00000511'
MQIACF_RETAINED_PUBLICATION	1300	X'00000514'
MQIACF_TOPIC_STATUS_ATTRS	1301	X'00000515'
MQIACF_TOPIC_STATUS_TYPE	1302	X'00000516'
MQIACF_SUB_OPTIONS	1303	X'00000517'

표 185. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIACF_PUBLISH_COUNT	1304	X'00000518'
MQIACF_CLEAR_TYPE	1305	X'00000519'
MQIACF_CLEAR_SCOPE	1306	X'0000051A'
MQIACF_SUB_LEVEL	1307	X'0000051B'
MQIACF_ASYNC_STATE	1308	X'0000051C'
MQIACF_SUB_SUMMARY	1309	X'0000051D'
MQIACF_OBSOLETE_MSGS	1310	X'0000051E'
MQIACF_PUBSUB_STATUS	1311	X'0000051F'
MQIACF_PS_STATUS_TYPE	1314	X'00000522'
MQIACF_PUBSUB_STATUS_ATTRS	1318	X'00000526'
MQIACF_SELECTOR_TYPE	1321	X'00000529'
MQIACF_MCAST_REL_INDICATOR	1351	X'00000547'
MQIACF_CHLAUTH_TYPE	1352	X'00000548'
MQXR_DIAGNOSTICS_TYPE	1354	X'0000054A'
MQIACF_CHLAUTH_ATTRS	1355	X'0000054B'
MQIACF_OPERATION_ID	1356	X'0000054C'
MQIACF_API_CALLER_TYPE	1357	X'0000054D'
MQIACF_API_ENVIRONMENT	1358	X'0000054E'
MQIACF_TRACE_DETAIL	1359	X'0000054F'
MQIACF_HOBJ	1360	X'00000550'
MQIACF_CALL_TYPE	1361	X'00000551'
MQIACF_MQCB_OPERATION	1362	X'00000552'
MQIACF_MQCB_TYPE	1363	X'00000553'
MQIACF_MQCB_OPTIONS	1364	X'00000554'
MQIACF_CLOSE_OPTIONS	1365	X'00000555'
MQIACF_CTL_OPERATION	1366	X'00000556'
MQIACF_GET_OPTIONS	1367	X'00000557'
MQIACF_RECS_PRESENT	1368	X'00000558'
MQIACF_KNOWN_DEST_COUNT	1369	X'00000559'
MQIACF_UNKNOWN_DEST_COUNT	1370	X'0000055A'
MQIACF_INVALID_DEST_COUNT	1371	X'0000055B'
MQIACF_RESOLVED_TYPE	1372	X'0000055C'
MQIACF_PUT_OPTIONS	1373	X'0000055D'
MQIACF_BUFFER_LENGTH	1374	X'0000055E'
MQIACF_TRACE_DATA_LENGTH	1375	X'0000055F'
MQIACF_SMDS_EXPANDST	1376	X'00000560'
MQIACF_STRUC_LENGTH	1377	X'00000561'
MQIACF_ITEM_COUNT	1378	X'00000562'
MQIACF_EXPIRY_TIME	1379	X'00000563'



표 185. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIACF_CONNECT_TIME	1380	X'00000564'
MQIACF_DISCONNECT_TIME	1381	X'00000565'
MQIACF_HSUB	1382	X'00000566'
MQIACF_SUBRQ_OPTIONS	1383	X'00000567'
MQIACF_XA_RMID	1384	X'00000568'
MQIACF_XA_FLAGS	1385	X'00000569'
MQIACF_XA_RETCODE	1386	X'0000056A'
MQIACF_XA_HANDLE	1387	X'0000056B'
MQIACF_XA_RETVAL	1388	X'0000056C'
MQIACF_STATUS_TYPE	1389	X'0000056D'
MQIACF_XA_COUNT	1390	X'0000056E'
MQIACF_SELECTOR_COUNT	1391	X'0000056F'
MQIACF_SELECTORS	1392	X'00000570'
MQIACF_INTATTR_COUNT	1393	X'00000571'
MQIACF_INTATTRS	1394	X'00000572'
MQIACF_SUBRQ_ACTION	1395	X'00000573'
MQIACF_NUM_PUBS	1396	X'00000574'
MQIACF_POINTER_SIZE	1397	X'00000575'
MQIACF_REMOVE_AUTHREC	1398	X'00000576'
MQIACF_XR_ATTRS	1399	X'00000577'
MQIACF_APPL_FUNCTION_TYPE	1400	X'00000578'
MQIACF_AMQP_ATTRS	1401	X'00000579'
MQIACF_EXPORT_TYPE	1402	X'0000057A'
MQIACF_EXPORT_ATTRS	1403	X'0000057B'
MQIACF_SYSTEM_OBJECTS	1404	X'0000057C'
MQIACF_CONNECTION_SWAP	1405	X'0000057D'
MQIACF_AMQP_DIAGNOSTICS_TYPE	1406	X'0000057E'
MQIACF_BUFFER_POOL_LOCATION	1408	X'00000580'
MQIACF_LDAP_CONNECTION_STATUS	1409	X'00000581'
MQIACF_SYSP_MAX_ACE_POOL	1410	X'00000582'
MQIACF_PAGECLAS	1411	X'00000583'
MQIACF_AUTH_REC_TYPE	1412	X'00000584'
MQIACF_SYSP_MAX_CONC_OFFLOADS	1413	X'00000585'
MQIACF_SYSP_ZHYPERWRITE	1414	X'00000586'
MQIACF_Q_MGR_STATUS_LOG	1415	X'00000587'
MQIACF_ARCHIVE_LOG_SIZE	1416	X'00000588'
MQIACF_MEDIA_LOG_SIZE	1417	X'00000589'
MQIACF_RESTART_LOG_SIZE	1418	X'0000058A'
MQIACF_REUSABLE_LOG_SIZE	1419	X'0000058B'

표 185. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIACF_LOG_IN_USE	1420	X'0000058C'
MQIACF_LOG_UTILIZATION	1421	X'0000058D'
MQIACF_IGNORE_STATE	1423	X'0000058F'
MQIACF_MOVABLE_APPL_COUNT	1424	X'00000590'
MQIACF_APPL_INFO_ATTRS	1425	X'00000591'
MQIACF_APPL_MOVABLE	1426	X'00000592'
MQIACF_REMOTE_QMGR_ACTIVE	1427	X'00000593'
MQIACF_APPL_INFO_TYPE	1428	X'00000594'
MQIACF_APPL_INFO_APPL	1429	X'00000595'
MQIACF_APPL_INFO_QMGR	1430	X'00000596'
MQIACF_APPL_INFO_LOCAL	1431	X'00000597'
MQIACF_APPL_IMMOVABLE_COUNT	1432	X'00000598'
MQIACF_BALANCED	1433	X'00000599'
MQIACF_BALSTATE	1434	X'0000059A'
MQIACF_APPL_IMMOVABLE_REASON	1435	X'0000059B'
MQIACF_DS_ENCRYPTED	1436	X'0000059C'
MQIACF_CUR_Q_FILE_SIZE	1437	X'0000059D'
MQIACF_CUR_MAX_FILE_SIZE	1438	X'0000059E'
MQIACF_BALANCING_TYPE	1439	X'0000059F'
MQIACF_BALANCING_OPTIONS	1440	X'000005A0'
MQIACF_BALANCING_TIMEOUT	1441	X'000005A1'
MQIACF_SYSP_SMF_STAT_TIME_SECS	1442	X'000005A2'
MQIACF_SYSP_SMF_ACCT_TIME_MINS	1443	X'000005A3'
MQIACF_SYSP_SMF_ACCT_TIME_SECS	1444	X'000005A4'
MQIACF_LAST_USED	1444	X'000005A4'


### MQIACH\_\*(명령 형식 정수 채널 유형)

표 186. 상수 값		
이름	10진수 값	16진수 값
MQIACH_FIRST	1501	X'000005DD'
MQIACH_XMIT_PROTOCOL_TYPE	1501	X'000005DD'
MQIACH_BATCH_SIZE	1502	X'000005DE'
MQIACH_DISC_INTERVAL	1503	X'000005DF'
MQIACH_SHORT_TIMER	1504	X'000005E0'
MQIACH_SHORT_RETRY	1505	X'000005E1'
MQIACH_LONG_TIMER	1506	X'000005E2'
MQIACH_LONG_RETRY	1507	X'000005E3'
MQIACH_PUT_AUTHORITY	1508	X'000005E4'
MQIACH_SEQUENCE_NUMBER_WRAP	1509	X'000005E5'

표 186. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIACH_MAX_MSG_LENGTH	1510	X'000005E6'
MQIACH_CHANNEL_TYPE	1511	X'000005E7'
MQIACH_DATA_COUNT	1512	X'000005E8'
MQIACH_NAME_COUNT	1513	X'000005E9'
MQIACH_MSG_SEQUENCE_NUMBER	1514	X'000005EA'
MQIACH_DATA_CONVERSION	1515	X'000005EB'
MQIACH_IN_DOUBT	1516	X'000005EC'
MQIACH_MCA_TYPE	1517	X'000005ED'
MQIACH_SESSION_COUNT	1518	X'000005EE'
MQIACH_ADAPTER	1519	X'000005EF'
MQIACH_COMMAND_COUNT	1520	X'000005F0'
MQIACH_SOCKET	1521	X'000005F1'
MQIACH_PORT	1522	X'000005F2'
MQIACH_CHANNEL_INSTANCE_TYPE	1523	X'000005F3'
MQIACH_CHANNEL_INSTANCE_ATTRS	1524	X'000005F4'
MQIACH_CHANNEL_ERROR_DATA	1525	X'000005F5'
MQIACH_CHANNEL_TABLE	1526	X'000005F6'
MQIACH_CHANNEL_STATUS	1527	X'000005F7'
MQIACH_INDOUBT_STATUS	1528	X'000005F8'
MQIACH_LAST_SEQ_NUMBER	1529	X'000005F9'
MQIACH_LAST_SEQUENCE_NUMBER	1529	X'000005F9'
MQIACH_CURRENT_MSGS	1531	X'000005FB'
MQIACH_CURRENT_SEQ_NUMBER	1532	X'000005FC'
MQIACH_CURRENT_SEQUENCE_NUMBER	1532	X'000005FC'
MQIACH_SSL_RETURN_CODE	1533	X'000005FD'
MQIACH_MSGS	1534	X'000005FE'
MQIACH_BYTES_SENT	1535	X'000005FF'
MQIACH_BYTES_RCVD	1536	X'00000600'
MQIACH_BYTES_RECEIVED	1536	X'00000600'
MQIACH_BATCHES	1537	X'00000601'
MQIACH_BUFFERS_SENT	1538	X'00000602'
MQIACH_BUFFERS_RCVD	1539	X'00000603'
MQIACH_BUFFERS_RECEIVED	1539	X'00000603'
MQIACH_LONG_RETRIES_LEFT	1540	X'00000604'
MQIACH_SHORT_RETRIES_LEFT	1541	X'00000605'
MQIACH_MCA_STATUS	1542	X'00000606'
MQIACH_STOP_REQUESTED	1543	X'00000607'
MQIACH_MR_COUNT	1544	X'00000608'
MQIACH_MR_INTERVAL	1545	X'00000609'

표 186. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIACH_NPM_SPEED	1562	X'0000061A'
MQIACH_HB_INTERVAL	1563	X'0000061B'
MQIACH_BATCH_INTERVAL	1564	X'0000061C'
MQIACH_NETWORK_PRIORITY	1565	X'0000061D'
MQIACH_KEEP_ALIVE_INTERVAL	1566	X'0000061E'
MQIACH_BATCH_HB	1567	X'0000061F'
MQIACH_SSL_CLIENT_AUTH	1568	X'00000620'
MQIACH_ALLOC_RETRY	1570	X'00000622'
MQIACH_ALLOC_FAST_TIMER	1571	X'00000623'
MQIACH_ALLOC_SLOW_TIMER	1572	X'00000624'
MQIACH_DISC_RETRY	1573	X'00000625'
MQIACH_PORT_NUMBER	1574	X'00000626'
MQIACH_HDR_COMPRESSION	1575	X'00000627'
MQIACH_MSG_COMPRESSION	1576	X'00000628'
MQIACH_CLWL_CHANNEL_RANK	1577	X'00000629'
MQIACH_CLWL_CHANNEL_PRIORITY	1578	X'0000062A'
MQIACH_CLWL_CHANNEL_WEIGHT	1579	X'0000062B'
MQIACH_CHANNEL_DISP	1580	X'0000062C'
MQIACH_INBOUND_DISP	1581	X'0000062D'
MQIACH_CHANNEL_TYPES	1582	X'0000062E'
MQIACH_ADAPS_STARTED	1583	X'0000062F'
MQIACH_ADAPS_MAX	1584	X'00000630'
MQIACH_DISPS_STARTED	1585	X'00000631'
MQIACH_DISPS_MAX	1586	X'00000632'
MQIACH_SSLTASKS_STARTED	1587	X'00000633'
MQIACH_SSLTASKS_MAX	1588	X'00000634'
MQIACH_CURRENT_CHL	1589	X'00000635'
MQIACH_CURRENT_CHL_MAX	1590	X'00000636'
MQIACH_CURRENT_CHL_TCP	1591	X'00000637'
MQIACH_CURRENT_CHL_LU62	1592	X'00000638'
MQIACH_ACTIVE_CHL	1593	X'00000639'
MQIACH_ACTIVE_CHL_MAX	1594	X'0000063A'
MQIACH_ACTIVE_CHL_PAUSED	1595	X'0000063B'
MQIACH_ACTIVE_CHL_STARTED	1596	X'0000063C'
MQIACH_ACTIVE_CHL_STOPPED	1597	X'0000063D'
MQIACH_ACTIVE_CHL_RETRY	1598	X'0000063E'
MQIACH_LISTENER_STATUS	1599	X'0000063F'
MQIACH_SHARED_CHL_RESTART	1600	X'00000640'
MQIACH_LISTENER_CONTROL	1601	X'00000641'

표 186. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIACH_BACKLOG	1602	X'00000642'
MQIACH_XMITQ_TIME_INDICATOR	1604	X'00000644'
MQIACH_NETWORK_TIME_INDICATOR	1605	X'00000645'
MQIACH_EXIT_TIME_INDICATOR	1606	X'00000646'
MQIACH_BATCH_SIZE_INDICATOR	1607	X'00000647'
MQIACH_XMITQ_MSGS_AVAILABLE	1608	X'00000648'
MQIACH_CHANNEL_SUBSTATE	1609	X'00000649'
MQIACH_SSL_KEY_RESETS	1610	X'0000064A'
MQIACH_COMPRESSION_RATE	1611	X'0000064B'
MQIACH_COMPRESSION_TIME	1612	X'0000064C'
MQIACH_MAX_XMIT_SIZE	1613	X'0000064D'
MQIACH_DEF_CHANNEL_DISP	1614	X'0000064E'
MQIACH_SHARING_CONVERSATIONS	1615	X'0000064F'
MQIACH_MAX_SHARING_CONVS	1616	X'00000650'
MQIACH_CURRENT_SHARING_CONVS	1617	X'00000651'
MQIACH_MAX_INSTANCES	1618	X'00000652'
MQIACH_MAX_INSTS_PER_CLIENT	1619	X'00000653'
MQIACH_CLIENT_CHANNEL_WEIGHT	1620	X'00000654'
MQIACH_CONNECTION_AFFINITY	1621	X'00000655'
MQIACH_AUTH_INFO_TYPES	1622	X'00000656'
MQIACH_RESET_REQUESTED	1623	X'00000657'
MQIACH_BATCH_DATA_LIMIT	1624	X'00000658'
MQIACH_MSG_HISTORY	1625	X'00000659'
MQIACH_MULTICAST_PROPERTIES	1626	X'0000065A'
MQIACH_NEW_SUBSCRIBER_HISTORY	1627	X'0000065B'
MQIACH_MC_HB_INTERVAL	1628	X'0000065C'
MQIACH_USE_CLIENT_ID	1629	X'0000065D'
MQIACH_MQTT_KEEP_ALIVE	1630	X'0000065E'
MQIACH_IN_DOUBT_IN	1631	X'0000065F'
MQIACH_IN_DOUBT_OUT	1632	X'00000660'
MQIACH_MSGS_SENT<	1633	X'00000661'
MQIACH_MSGS_RECEIVED	1634	X'00000662'
MQIACH_MSGS_RCVD	1634	X'00000662'
MQIACH_PENDING_OUT	1635	X'00000663'
MQIACH_AVAILABLE_CIPHERSPECS	1636	X'00000664'
MQIACH_MATCH	1637	X'00000665'
MQIACH_USER_SOURCE	1638	X'00000666'
MQIACH_WARNING	1639	X'00000667'
MQIACH_DEF_RECONNECT	1640	X'00000668'

표 186. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIACH_CHANNEL_SUMMARY_ATTRS	1642	X'0000066A'
MQIACH_PROTOCOL	1643	X'0000066B'
MQIACH_AMQPKEEPALIVE	1644	X'0000066C'
MQIACH_SECURITY_PROTOCOL	1645	X'0000066D'
 MQIACH_SPL_PROTECTION	1646	X'0000066E'
MQIACH_LAST_USED	1646	X'0000066E'

### MQIAMO\_\*(명령 형식 정수 모니터링 매개변수 유형)

표 187. 상수 값		
이름	10진수 값	16진수 값
MQIAMO_FIRST	701	X'000002BD'
MQIAMO_AVG_BATCH_SIZE	702	X'000002BE'
MQIAMO_AVG_Q_TIME	703	X'000002BF'
MQIAMO_BACKOUTS	704	X'000002C0'
MQIAMO_BROWSES	705	X'000002C1'
MQIAMO_BROWSE_MAX_BYTES	706	X'000002C2'
MQIAMO_BROWSE_MIN_BYTES	707	X'000002C3'
MQIAMO_BROWSES_FAILED	708	X'000002C4'
MQIAMO_CLOSES	709	X'000002C5'
MQIAMO_COMMITS	710	X'000002C6'
MQIAMO_COMMITS_FAILED	711	X'000002C7'
MQIAMO_CONNS	712	X'000002C8'
MQIAMO_CONNS_MAX	713	X'000002C9'
MQIAMO_DISCS	714	X'000002CA'
MQIAMO_DISCS_IMPLICIT	715	X'000002CB'
MQIAMO_DISC_TYPE	716	X'000002CC'
MQIAMO_EXIT_TIME_AVG	717	X'000002CD'
MQIAMO_EXIT_TIME_MAX	718	X'000002CE'
MQIAMO_EXIT_TIME_MIN	719	X'000002CF'
MQIAMO_FULL_BATCHES	720	X'000002D0'
MQIAMO_GENERATED_MSGS	721	X'000002D1'
MQIAMO_GETS	722	X'000002D2'
MQIAMO_GET_MAX_BYTES	723	X'000002D3'
MQIAMO_GET_MIN_BYTES	724	X'000002D4'
MQIAMO_GETS_FAILED	725	X'000002D5'
MQIAMO_INCOMPLETE_BATCHES	726	X'000002D6'
MQIAMO_INQS	727	X'000002D7'
MQIAMO_MSGS	728	X'000002D8'
MQIAMO_NET_TIME_AVG	729	X'000002D9'

표 187. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIAMO_NET_TIME_MAX	730	X'000002DA'
MQIAMO_NET_TIME_MIN	731	X'000002DB'
MQIAMO_OBJECT_COUNT	732	X'000002DC'
MQIAMO_OPENS	733	X'000002DD'
MQIAMO_PUT1S	734	X'000002DE'
MQIAMO_PUTS	735	X'000002DF'
MQIAMO_PUT_MAX_BYTES	736	X'000002E0'
MQIAMO_PUT_MIN_BYTES	737	X'000002E1'
MQIAMO_PUT_RETRIES	738	X'000002E2'
MQIAMO_Q_MAX_DEPTH	739	X'000002E3'
MQIAMO_Q_MIN_DEPTH	740	X'000002E4'
MQIAMO_Q_TIME_AVG	741	X'000002E5'
MQIAMO_Q_TIME_MAX	742	X'000002E6'
MQIAMO_Q_TIME_MIN	743	X'000002E7'
MQIAMO_SETS	744	X'000002E8'
MQIAMO_CONNS_FAILED	749	X'000002ED'
MQIAMO_OPENS_FAILED	751	X'000002EF'
MQIAMO_INQS_FAILED	752	X'000002F0'
MQIAMO_SETS_FAILED	753	X'000002F1'
MQIAMO_PUTS_FAILED	754	X'000002F2'
MQIAMO_PUT1S_FAILED	755	X'000002F3'
MQIAMO_CLOSES_FAILED	757	X'000002F5'
MQIAMO_MSGS_EXPIRED	758	X'000002F6'
MQIAMO_MSGS_NOT_QUEUED	759	X'000002F7'
MQIAMO_MSGS_PURGED	760	X'000002F8'
MQIAMO_SUBS_DUR	764	X'000002FC'
MQIAMO_SUBS_NDUR	765	X'000002FD'
MQIAMO_SUBS_FAILED	766	X'000002FE'
MQIAMO_SUBRQS	767	X'000002FF'
MQIAMO_SUBRQS_FAILED	768	X'00000300'
MQIAMO_CBS	769	X'00000301'
MQIAMO_CBS_FAILED	770	X'00000302'
MQIAMO_CTLs	771	X'00000303'
MQIAMO_CTLs_FAILED	772	X'00000304'
MQIAMO_STATS	773	X'00000305'
MQIAMO_STATS_FAILED	774	X'00000306'
MQIAMO_SUB_DUR_HIGHWATER	775	X'00000307'
MQIAMO_SUB_DUR_LOWWATER	776	X'00000308'
MQIAMO_SUB_NDUR_HIGHWATER	777	X'00000309'

표 187. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIAMO_SUB_NDUR_LOWWATER	778	X'0000030A'
MQIAMO_TOPIC_PUTS	779	X'0000030B'
MQIAMO_TOPIC_PUTS_FAILED	780	X'0000030C'
MQIAMO_TOPIC_PUT1S	781	X'0000030D'
MQIAMO_TOPIC_PUT1S_FAILED	782	X'0000030E'
MQIAMO_PUBLISH_MSG_COUNT	784	X'00000310'
MQIAMO_UNSUBS_DUR	786	X'00000312'
MQIAMO_UNSUBS_NDUR	787	X'00000313'
MQIAMO_UNSUBS_FAILED	788	X'00000314'
MQIAMO_INTERVAL	789	X'00000315'
MQIAMO_MSGS_SENT	790	X'00000316'
MQIAMO_BYTES_SENT	791	X'00000317'
MQIAMO_REPAIR_BYTES	792	X'00000318'
MQIAMO_FEEDBACK_MODE	793	X'00000319'
MQIAMO_RELIABILITY_TYPE	794	X'0000031A'
MQIAMO_LATE_JOIN_MARK	795	X'0000031B'
MQIAMO_NACKS_RCVD	796	X'0000031C'
MQIAMO_REPAIR_PKTS	797	X'0000031D'
MQIAMO_HISTORY_PKTS	798	X'0000031E'
MQIAMO_PENDING_PKTS	799	X'0000031F'
MQIAMO_PKT_RATE	800	X'00000320'
MQIAMO_MCAST_XMIT_RATE	801	X'00000321'
MQIAMO_MCAST_BATCH_TIME	802	X'00000322'
MQIAMO_MCAST_HEARTBEAT	803	X'00000323'
MQIAMO_DEST_DATA_PORT	804	X'00000324'
MQIAMO_DEST_REPAIR_PORT	805	X'00000325'
MQIAMO_ACKS_RCVD	806	X'00000326'
MQIAMO_ACTIVE_ACKERS	807	X'00000327'
MQIAMO_PKTS_SENT	808	X'00000328'
MQIAMO_TOTAL_REPAIR_PKTS	809	X'00000329'
MQIAMO_TOTAL_PKTS_SENT	810	X'0000032A'
MQIAMO_TOTAL_MSGS_SENT	811	X'0000032B'
MQIAMO_TOTAL_BYTES_SENT	812	X'0000032C'
MQIAMO_NUM_STREAMS	813	X'0000032D'
MQIAMO_ACK_FEEDBACK	814	X'0000032E'
MQIAMO_NACK_FEEDBACK	815	X'0000032F'
MQIAMO_PKTS_LOST	816	X'00000330'
MQIAMO_MSGS_RCVD	817	X'00000331'
MQIAMO_MSG_BYTES_RCVD	818	X'00000332'



표 187. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIAMO_MSGS_DELIVERED	819	X'00000333'
MQIAMO_PKTS_PROCESSED	820	X'00000334'
MQIAMO_PKTS_DLVD	821	X'00000335'
MQIAMO_PKTS_DROPPED	822	X'00000336'
MQIAMO_PKTS_DUPLICATED	823	X'00000337'
MQIAMO_NACKS_CREATED	824	X'00000338'
MQIAMO_NACK_PKTS_SENT	825	X'00000339'
MQIAMO_REPAIR_PKTS_RQSTD	826	X'0000033A'
MQIAMO_REPAIR_PKTS_RCVD	827	X'0000033B'
MQIAMO_PKTS_REPAIRED	828	X'0000033C'
MQIAMO_TOTAL_MSGS_RCVD	829	X'0000033D'
MQIAMO_TOTAL_MSGS_BYTES_RCVD	830	X'0000033E'
MQIAMO_TOTAL_REPAIR_PKTS_RCVD	831	X'0000033F'
MQIAMO_TOTAL_REPAIR_PKTS_RQSTD	832	X'00000340'
MQIAMO_TOTAL_MSGS_PROCESSED	833	X'00000341'
MQIAMO_TOTAL_MSGS_SELECTED	834	X'00000342'
MQIAMO_TOTAL_MSGS_EXPIRED	835	X'00000343'
MQIAMO_TOTAL_MSGS_DELIVERED	836	X'00000344'
MQIAMO_TOTAL_MSGS_RETURNED	837	X'00000345'
MQIAMO_LAST_USED	837	X'00000345'

### MQIAMO64\_\*(명령 형식 64비트 정수 모니터링 매개변수 유형)

표 188. 상수 값		
이름	10진수 값	16진수 값
MQIAMO64_AVG_Q_TIME	703	X'000002BF'
MQIAMO64_Q_TIME_AVG	741	X'000002E5'
MQIAMO64_Q_TIME_MAX	742	X'000002E6'
MQIAMO64_Q_TIME_MIN	743	X'000002E7'
MQIAMO64_BROWSE_BYTES	745	X'000002E9'
MQIAMO64_BYTES	746	X'000002EA'
MQIAMO64_GET_BYTES	747	X'000002EB'
MQIAMO64_PUT_BYTES	748	X'000002EC'
MQIAMO64_TOPIC_PUT_BYTES	783	X'0000030F'
MQIAMO64_PUBLISH_MSG_BYTES	785	X'00000311'

### MQIASY\_\*(정수 시스템 선택자)

표 189. 상수 값		
이름	10진수 값	16진수 값
MQIASY_FIRST	-1	X'FFFFFFFF'
MQIASY_CODED_CHAR_SET_ID	-1	X'FFFFFFFF'

표 189. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQIASY_TYPE	-2	X'FFFFFFFFE'
MQIASY_COMMAND	-3	X'FFFFFFFFD'
MQIASY_MSG_SEQ_NUMBER	-4	X'FFFFFFFFC'
MQIASY_CONTROL	-5	X'FFFFFFFFB'
MQIASY_COMP_CODE	-6	X'FFFFFFFFA'
MQIASY_REASON	-7	X'FFFFFFFF9'
MQIASY_BAG_OPTIONS	-8	X'FFFFFFFF8'
MQIASY_VERSION	-9	X'FFFFFFFF7'
MQIASY_LAST_USED	-9	X'FFFFFFFF7'
MQIASY_LAST	-2000	X'FFFFFF830'

### MQIAUT\_\*(IMS 정보 헤더 인증자)

표 190. 상수 이름 및 값	
이름	값
MQIAUT_NONE	"~~~~~"
MQIAUT_NONE_ARRAY	'~','~','~','~','~','~','~','~','~','~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

### MQIAV\_\*(정수 속성 값)

표 191. 상수 값		
이름	10진수 값	16진수 값
MQIAV_NOT_APPLICABLE	-1	X'FFFFFFFF'
MQIAV_UNDEFINED	-2	X'FFFFFFFFE'

### MQICM\_\*(IMS 정보 헤더 커미트 모드)

표 192. 상수 이름 및 값	
이름	값
MQICM_COMMIT_THEN_SEND	'0'
MQICM_SEND_THEN_COMMIT	'1'

### MQIDO\_\*(명령 형식 인다우트(in-doubt) 옵션)

표 193. 상수 값		
이름	10진수 값	16진수 값
MQIDO_COMMIT	1	X'00000001'
MQIDO_BACKOUT	2	X'00000002'

## MQIEP\_\*(인터페이스 시작점)

### 연결 보안 매개변수 구조

표 194. 상수 구조	
이름	구조
MQIEP_STRUC_ID	"IEP~"
MQIEP_STRUC_ID_ARRAY	'I','E','P','~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 195. 상수 값		
이름	10진수 값	16진수 값
MQIEP_VERSION_1	1	X'00000001'
MQDXP_CURRENT_VERSION	1	X'00000001'

## MQIGQ\_\*(그룹 내 큐잉)

표 196. 상수 값		
이름	10진수 값	16진수 값
MQIGQ_DISABLED	0	X'00000000'
MQIGQ_ENABLED	1	X'00000001'

## MQIGQPA\_\*(그룹 내 큐잉 넣기 권한)

표 197. 상수 값		
이름	10진수 값	16진수 값
MQIGQPA_DEFAULT	1	X'00000001'
MQIGQPA_CONTEXT	2	X'00000002'
MQIGQPA_ONLY_IGQ	3	X'00000003'
MQIGQPA_ALTERNATE_OR_IGQ	4	X'00000004'

## MQIIH\_\*(IMS 정보 헤더 구조 및 플래그)

### IMS 정보 헤더 구조

표 198. 상수 구조	
이름	구조
MQIIH_STRUC_ID	"IIH~"
MQIIH_STRUC_ID_ARRAY	'I','I','H','~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 199. 상수 값		
이름	10진수 값	16진수 값
MQIIH_VERSION_1	1	X'00000001'
MQIIH_CURRENT_VERSION	1	X'00000001'
MQIIH_LENGTH_1	84	X'00000054'

## IMS 정보 헤더 플래그

표 200. 상수 값		
이름	10진수 값	16진수 값
MQIIH_NONE	0	X'00000000'
MQIIH_PASS_EXPIRATION	1	X'00000001'
MQIIH_UNLIMITED_EXPIRATION	0	X'00000000'
MQIIH_REPLY_FORMAT_NONE	8	X'00000008'
MQIIH_IGNORE_PURG	16	X'00000010'
MQIIH_CM0_REQUEST_RESPONSE	32	X'00000020'

## MQIMPO\_\*(메시지 특성 조회 옵션 및 구조)

### 메시지 특성 조회 옵션 구조

표 201. 상수 구조	
이름	구조
MQIMPO_STRUC_ID	"IMPO"
MQIMPO_STRUC_ID_ARRAY	'I','M','P','O'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 202. 상수 값		
이름	10진수 값	16진수 값
MQIMPO_VERSION_1	1	X'00000001'
MQIMPO_CURRENT_VERSION	1	X'00000001'

### 메시지 특성 조회 옵션

표 203. 상수 값		
이름	10진수 값	16진수 값
MQIMPO_CONVERT_TYPE	2	X'00000002'
MQIMPO_QUERY_LENGTH	4	X'00000004'
MQIMPO_INQ_FIRST	0	X'00000000'
MQIMPO_INQ_NEXT	8	X'00000008'
MQIMPO_INQ_PROP_UNDER_CURSOR	16	X'00000010'
MQIMPO_CONVERT_VALUE	32	X'00000020'
MQIMPO_NONE	0	X'00000000'

## MQINBD\_\*(명령 형식 인바운드 배치)

표 204. 상수 값		
이름	10진수 값	16진수 값
MQINBD_Q_MGR	0	X'00000000'
MQINBD_GROUP	3	X'00000003'

### MQIND\_\*(특수 색인 값)

표 205. 상수 값		
이름	10진수 값	16진수 값
MQIND_NONE	-1	X'FFFFFFFF'
MQIND_ALL	-2	X'FFFFFFFE'

### MQIPADDR\_\*(IP 주소 버전)

표 206. 상수 값		
이름	10진수 값	16진수 값
MQIPADDR_IPV4	0	X'00000000'
MQIPADDR_IPV6	1	X'00000001'

### MQISS\_\*(IMS 정보 헤더 보안 범위)

표 207. 상수 이름 및 값	
이름	값
MQISS_CHECK	'C'
MQISS_FULL	'F'

### MQIT\_\*(색인 유형)

표 208. 상수 값		
이름	10진수 값	16진수 값
MQIT_NONE	0	X'00000000'
MQIT_MSG_ID	1	X'00000001'
MQIT_CORREL_ID	2	X'00000002'
MQIT_MSG_TOKEN	4	X'00000004'
MQIT_GROUP_ID	5	X'00000005'

### MQITEM\_\*(mqInquireItemInfo의 항목 유형)

표 209. 상수 값		
이름	10진수 값	16진수 값
MQITEM_INTEGER	1	X'00000001'
MQITEM_STRING	2	X'00000002'
MQITEM_BAG	3	X'00000003'
MQITEM_BYTE_STRING	4	X'00000004'
MQITEM_INTEGER_FILTER	5	X'00000005'
MQITEM_STRING_FILTER	6	X'00000006'
MQITEM_INTEGER64	7	X'00000007'
MQITEM_BYTE_STRING_FILTER	8	X'00000008'

### MQITII\_\*(IMS 정보 헤더 트랜잭션 인스턴스 ID)

표 210. 상수 이름 및 값	
이름	값
MQITII_NONE	X'00...00' (16널)
MQITII_NONE_ARRAY	'\0', '\0', ... (16널)

### MQITS\_\*(IMS 정보 헤더 트랜잭션 상태)

표 211. 상수 이름 및 값	
이름	값
MQITS_IN_CONVERSATION	'C'
MQITS_NOT_IN_CONVERSATION	'-'
MQITS_ARCHITECTED	'A'

참고: - 기호는 단일 공백 문자를 나타냅니다.

### MQKAI\_\*(활성 유지(keepalive) 간격)

표 212. 상수 값		
이름	10진수 값	16진수 값
MQKAI_AUTO	-1	X'FFFFFFFF'

### MQMASTER\_\*(마스터 관리)

표 213. 상수 값		
이름	10진수 값	16진수 값
MQMASTER_NO	0	X'00000000'
MQMASTER_YES	1	X'00000001'

### MQMCAS\_\*(명령 형식 메시지 채널 에이전트 상태)

표 214. 상수 값		
이름	10진수 값	16진수 값
MQMCAS_STOPPED	0	X'00000000'
MQMCAS_RUNNING	3	X'00000003'

### MQMCAT\_\*(MCA 유형)

표 215. 상수 값		
이름	10진수 값	16진수 값
MQMCAT_PROCESS	1	X'00000001'
MQMCAT_THREAD	2	X'00000002'

## MQMCD\_\*(발행/구독 옵션 태그 정보)

### 발행/구독 옵션 태그 메시지 콘텐츠 디스크립터(mcd) 태그

표 216. 상수 값		
이름	10진수 값	16진수 값
MQMCD_FOLDER_VERSION	1	X'00000001'

### 발행/구독 옵션 태그 태그 이름

표 217. 상수 이름 및 값	
이름	값
MQMCD_MSG_DOMAIN	"Msd"
MQMCD_MSG_SET	"Set"
MQMCD_MSG_TYPE	"Type"
MQMCD_MSG_FORMAT	"Fmt"

### 발행/구독 옵션 태그 XML 태그 이름

표 218. 상수 이름 및 값	
이름	값
MQMCD_MSG_DOMAIN_B	"<Msd>"
MQMCD_MSG_DOMAIN_E	"</Msd>"
MQMCD_MSG_SET_B	"<Set>"
MQMCD_MSG_SET_E	"</Set>"
MQMCD_MSG_TYPE_B	"<Type>"
MQMCD_MSG_TYPE_E	"</Type>"
MQMCD_MSG_FORMAT_B	"<Fmt>"
MQMCD_MSG_FORMAT_E	"</Fmt>"

### 발행/구독 옵션 태그 태그 값

표 219. 상수 이름 및 값	
이름	값
MQMCD_DOMAIN_NONE	"none"
MQMCD_DOMAIN_NEON	"neon"
MQMCD_DOMAIN_MRM	"mrm"
MQMCD_DOMAIN_JMS_NONE	"jms_none"
MQMCD_DOMAIN_JMS_TEXT	"jms_text"
MQMCD_DOMAIN_JMS_OBJECT	"jms_object"
MQMCD_DOMAIN_JMS_MAP	"jms_map"
MQMCD_DOMAIN_JMS_STREAM	"jms_stream"
MQMCD_DOMAIN_JMS_BYTES	"jms_bytes"

## MQMD\_\*(메시지 디스크립터 구조)

표 220. 상수 구조	
이름	구조
MQMD_STRUC_ID	"MD↵"
MQMD_STRUC_ID_ARRAY	'M','D','↵','↵'

참고: ↵ 기호는 단일 공백 문자를 나타냅니다.

표 221. 상수 값		
이름	10진수 값	16진수 값
MQMD_VERSION_1	1	X'00000001'
MQMD_VERSION_2	2	X'00000002'
MQMD_CURRENT_VERSION	2	X'00000002'

## MQMDE\_\*(메시지 디스크립터 확장 구조)

표 222. 상수 구조	
이름	구조
MQMDE_STRUC_ID	"MDE↵"
MQMDE_STRUC_ID_ARRAY	'M','D','E','↵'

참고: ↵ 기호는 단일 공백 문자를 나타냅니다.

표 223. 상수 값		
이름	10진수 값	16진수 값
MQMDE_VERSION_2	2	X'00000002'
MQMDE_CURRENT_VERSION	2	X'00000002'
MQMDE_LENGTH_2	72	X'00000048'

## MQMDEF\_\*(메시지 디스크립터 확장 플래그)

표 224. 상수 값		
이름	10진수 값	16진수 값
MQMDEF_NONE	0	X'00000000'

## MQMDS\_\*(메시지 전달 순서)

표 225. 상수 값		
이름	10진수 값	16진수 값
MQMDS_PRIORITY	0	X'00000000'
MQMDS_FIFO	1	X'00000001'

## MQMF\_\*(메시지 플래그)

표 226. 상수 값		
이름	10진수 값	16진수 값
MQMF_SEGMENTATION_INHIBITED	0	X'00000000'
MQMF_SEGMENTATION_ALLOWED	1	X'00000001'



표 226. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQMF_MSG_IN_GROUP	8	X'00000008'
MQMF_LAST_MSG_IN_GROUP	16	X'00000010'
MQMF_SEGMENT	2	X'00000002'
MQMF_LAST_SEGMENT	4	X'00000004'
MQMF_NONE	0	X'00000000'

## MQMHBO\_\*(메시지 핸들 대 버퍼 옵션 구조)

### 메시지 핸들 대 버퍼 옵션 구조

표 227. 상수 구조	
이름	구조
MQMHBO_STRUC_ID	"MHBO"
MQMHBO_STRUC_ID_ARRAY	'M', 'H', 'B', 'O'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 228. 상수 값		
이름	10진수 값	16진수 값
MQMHBO_VERSION_1	1	X'00000001'
MQMHBO_CURRENT_VERSION	1	X'00000001'

### 메시지 핸들 대 버퍼 옵션

표 229. 상수 값		
이름	10진수 값	16진수 값
MQMHBO_PROPERTIES_IN_MQRFH2	1	X'00000001'
MQMHBO_DELETE_PROPERTIES	2	X'00000002'
MQMHBO_NONE	0	X'00000000'

## MQMI\_\*(메시지 ID)

표 230. 상수 이름 및 값	
이름	값
MQMI_NONE	X'00...00' (24널)
MQMI_NONE_ARRAY	'\0', '\0', ... (24널)

## MQMMBI\_\*(메시지 표시-찾아보기 간격)

표 231. 상수 값		
이름	10진수 값	16진수 값
MQMMBI_UNLIMITED	-1	X'FFFFFFFF'

## MQMO\_\*(일치 옵션)

표 232. 상수 값		
이름	10진수 값	16진수 값
MQMO_MATCH_MSG_ID	1	X'00000001'
MQMO_MATCH_CORREL_ID	2	X'00000002'
MQMO_MATCH_GROUP_ID	4	X'00000004'
MQMO_MATCH_MSG_SEQ_NUMBER	8	X'00000008'
MQMO_MATCH_OFFSET	16	X'00000010'
MQMO_MATCH_MSG_TOKEN	32	X'00000020'
MQMO_NONE	0	X'00000000'

## MQMODE\_\*(명령 형식 모드 옵션)

표 233. 상수 값		
이름	10진수 값	16진수 값
MQMODE_FORCE	0	X'00000000'
MQMODE_QUIESCE	1	X'00000001'
MQMODE_TERMINATE	2	X'00000002'

## MQMON\_\*(모니터링 값)

표 234. 상수 값		
이름	10진수 값	16진수 값
MQMON_NOT_AVAILABLE	-1	X'FFFFFFFF'
MQMON_NONE	-1	X'FFFFFFFF'
MQMON_Q_MGR	-3	X'FFFFFFFD'
MQMON_OFF	0	X'00000000'
MQMON_ON	1	X'00000001'
MQMON_DISABLED	0	X'00000000'
MQMON_ENABLED	1	X'00000001'
MQMON_LOW	17	X'00000011'
MQMON_MEDIUM	33	X'00000021'
MQMON_HIGH	65	X'00000041'

## MQMT\_\*(메시지 유형)

표 235. 상수 값		
이름	10진수 값	16진수 값
MQMT_SYSTEM_FIRST	1	X'00000001'
MQMT_REQUEST	1	X'00000001'
MQMT_REPLY	2	X'00000002'
MQMT_DATAGRAM	8	X'00000008'
MQMT_REPORT	4	X'00000004'
MQMT_MQE_FIELDS_FROM_MQE	112	X'00000070'

표 235. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQMT_MQE_FIELDS	113	X'00000071'
MQMT_SYSTEM_LAST	65535	X'0000FFFF'
MQMT_APPL_FIRST	65536	X'00010000'
MQMT_APPL_LAST	99999999	X'3B9AC9FF'

### MQMTOK\_\*(메시지 토큰)

표 236. 상수 이름 및 값	
이름	값
MQMTOK_NONE	X'00...00' (16널)
MQMTOK_NONE_ARRAY	'\0', '\0', ... (16널)

### MQNC\_\*(이름 수)

표 237. 상수 값		
이름	10진수 값	16진수 값
MQNC_MAX_NAMELIST_NAME_COUNT	256	X'00000100'

### MQNPM\_\*(비지속 메시지 클래스)

표 238. 상수 값		
이름	10진수 값	16진수 값
MQNPM_CLASS_NORMAL	0	X'00000000'
MQNPM_CLASS_HIGH	10	X'0000000A'

### MQNPMS\_\*(비지속 메시지 속도)

표 239. 상수 값		
이름	10진수 값	16진수 값
MQNPMS_NORMAL	1	X'00000001'
MQNPMS_FAST	2	X'00000002'

### MQNT\_\*(이름 목록 유형)

표 240. 상수 값		
이름	10진수 값	16진수 값
MQNT_NONE	0	X'00000000'
MQNT_Q	1	X'00000001'
MQNT_CLUSTER	2	X'00000002'
MQNT_AUTH_INFO	4	X'00000004'
MQNT_ALL	1001	X'000003E9'

## MQNVS\_\*(이름/값 문자열의 이름)

표 241. 상수 이름 및 값	
이름	값
MQNVS_APPL_TYPE	"OPT_APP_GRP~"
MQNVS_MSG_TYPE	"OPT_MSG_TYPE~"

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

## MQOA\_\*(오브젝트 속성에 대한 선택자의 한계)

표 242. 상수 값		
이름	10진수 값	16진수 값
MQOA_FIRST	1	X'00000001'
MQOA_LAST	9000	X'00002328'

## MQOD\_\*(오브젝트 디스크립터 구조)

표 243. 상수 구조	
이름	구조
MQOD_STRUC_ID	"OD~"
MQOD_STRUC_ID_ARRAY	'0','D','~','~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 244. 상수 값		
이름	10진수 값	16진수 값
MQOD_VERSION_1	1	X'00000001'
MQOD_VERSION_2	2	X'00000002'
MQOD_VERSION_3	3	X'00000003'
MQOD_VERSION_4	4	X'00000004'
MQOD_CURRENT_VERSION	4	X'00000004'
MQOD_CURRENT_LENGTH	(value differs by platform or version)	(value differs by platform or version)

## MQOII\_\*(오브젝트 인스턴스 ID)

표 245. 상수 이름 및 값	
이름	값
MQOII_NONE	X'00...00' (24년)
MQOII_NONE_ARRAY	'\0','\0',... (24년)

## MQOL\_\*(원래 길이)

표 246. 상수 값		
이름	10진수 값	16진수 값
MQOL_UNDEFINED	-1	X'FFFFFFFF'

### MQOM\_\*(조회 그룹에서 더 이상 사용되지 않는 Db2 메시지 옵션)

표 247. 상수 값		
이름	10진수 값	16진수 값
MQOM_NO	0	X'00000000'
MQOM_YES	1	X'00000001'

### MQOO\_\*(열기 옵션)

표 248. 상수 값		
이름	10진수 값	16진수 값
MQOO_BIND_AS_Q_DEF	0	X'00000000'
MQOO_READ_AHEAD_AS_Q_DEF	0	X'00000000'
MQOO_INPUT_AS_Q_DEF	1	X'00000001'
MQOO_INPUT_SHARED	2	X'00000002'
MQOO_INPUT_EXCLUSIVE	4	X'00000004'
MQOO_BROWSE	8	X'00000008'
MQOO_OUTPUT	16	X'00000010'
MQOO_INQUIRE	32	X'00000020'
MQOO_SET	64	X'00000040'
MQOO_SAVE_ALL_CONTEXT	128	X'00000080'
MQOO_PASS_IDENTITY_CONTEXT	256	X'00000100'
MQOO_PASS_ALL_CONTEXT	512	X'00000200'
MQOO_SET_IDENTITY_CONTEXT	1024	X'00000400'
MQOO_SET_ALL_CONTEXT	2048	X'00000800'
MQOO_ALTERNATE_USER_AUTHORITY	4096	X'00001000'
MQOO_FAIL_IF QUIESCING	8192	X'00002000'
MQOO_BIND_ON_OPEN	16384	X'00004000'
MQOO_BIND_NOT_FIXED	32768	X'00008000'
MQOO_CO_OP	131072	X'00020000'
MQOO_RESOLVE_LOCAL_TOPIC	262144	X'00040000'
MQOO_NO_READ_AHEAD	524288	X'00080000'
MQOO_READ_AHEAD	1048576	X'00100000'
MQOO_BIND_ON_GROUP	4194304	X'00400000'

### MQOO\_\*(다음은 C++에서만 사용됨)

표 249. 상수 값		
이름	10진수 값	16진수 값
MQOO_RESOLVE_NAMES	65536	X'00010000'
MQOO_RESOLVE_LOCAL_Q	262144	X'00040000'

## MQOP\_\*(MQCTL 및 MQCB에 대한 조작 코드)

### MQCTL에 대한 조작 코드

표 250. 상수 값		
이름	10진수 값	16진수 값
MQOP_START	1	X'00000001'
MQOP_START_WAIT	2	X'00000002'
MQOP_STOP	4	X'00000004'

### MQCB에 대한 조작 코드

표 251. 상수 값		
이름	10진수 값	16진수 값
MQOP_REGISTER	256	X'00000100'
MQOP_DEREGISTER	512	X'00000200'

### MQCTL 및 MQCB에 대한 조작 코드

표 252. 상수 값		
이름	10진수 값	16진수 값
MQOP_SUSPEND	65536	X'00010000'
MQOP_RESUME	131072	X'00020000'

### MQOPEN\_\*(MQOPEN\_PRIV 구조 관련 값)

표 253. 상수 값		
이름	10진수 값	16진수 값
MQOPEN_PRIV_VERSION_1	1	X'00000001'
MQOPEN_PRIV_CURRENT_VERSION	1	X'00000001'

### MQOPER\_\*(활동 조작)

표 254. 상수 값		
이름	10진수 값	16진수 값
MQOPER_SYSTEM_FIRST	0	X'00000000'
MQOPER_UNKNOWN	0	X'00000000'
MQOPER_BROWSE	1	X'00000001'
MQOPER_DISCARD	2	X'00000002'
MQOPER_GET	3	X'00000003'
MQOPER_PUT	4	X'00000004'
MQOPER_PUT_REPLY	5	X'00000005'
MQOPER_PUT_REPORT	6	X'00000006'
MQOPER_RECEIVE	7	X'00000007'
MQOPER_SEND	8	X'00000008'
MQOPER_TRANSFORM	9	X'00000009'

표 254. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQOPER_PUBLISH	10	X'0000000A'
MQOPER_EXCLUDED_PUBLISH	11	X'0000000B'
MQOPER_DISCARDED_PUBLISH	12	X'0000000C'
MQOPER_SYSTEM_LAST	65535	X'0000FFFF'
MQOPER_APPL_FIRST	65536	X'00010000'
MQOPER_APPL_LAST	999999999	X'3B9AC9FF'

## MQOT\_\*(오브젝트 유형 및 확장 오브젝트 유형)

### 오브젝트 유형

표 255. 상수 값		
이름	10진수 값	16진수 값
MQOT_NONE	0	X'00000000'
MQOT_Q	1	X'00000001'
MQOT_NAMELIST	2	X'00000002'
MQOT_PROCESS	3	X'00000003'
MQOT_STORAGE_CLASS	4	X'00000004'
MQOT_Q_MGR	5	X'00000005'
MQOT_CHANNEL	6	X'00000006'
MQOT_AUTH_INFO	7	X'00000007'
MQOT_TOPIC	8	X'00000008'
MQOT_CF_STRUC	10	X'0000000A'
MQOT_LISTENER	11	X'0000000B'
MQOT_SERVICE	12	X'0000000C'
MQOT_RESERVED_1	999	X'000003E7'

### 확장 오브젝트 유형

표 256. 상수 값		
이름	10진수 값	16진수 값
MQOT_ALL	1001	X'000003E9'
MQOT_ALIAS_Q	1002	X'000003EA'
MQOT_MODEL_Q	1003	X'000003EB'
MQOT_LOCAL_Q	1004	X'000003EC'
MQOT_REMOTE_Q	1005	X'000003ED'
MQOT_SENDER_CHANNEL	1007	X'000003EF'
MQOT_SERVER_CHANNEL	1008	X'000003F0'
MQOT_REQUESTER_CHANNEL	1009	X'000003F1'
MQOT_RECEIVER_CHANNEL	1010	X'000003F2'
MQOT_CURRENT_CHANNEL	1011	X'000003F3'
MQOT_SAVED_CHANNEL	1012	X'000003F4'

표 256. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQOT_SVRCONN_CHANNEL	1013	X'000003F5'
MQOT_CLNTCONN_CHANNEL	1014	X'000003F6'
MQOT_SHORT_CHANNEL	1015	X'000003F7'
MQOT_CHLAUTH	1016	X'000003F8'
MQOT_REMOTE_Q_MGR_NAME	1017	X'000003F9'
MQOT_PROT_POLICY	1019	X'000003FB'
MQOT_TT_CHANNEL	1020	X'000003FC'
MQOT_AMQP_CHANNEL	1021	X'000003FD'
MQOT_AUTH_REC	1022	X'000003FE'

### MQPA\_\*(넣기 권한)

표 257. 상수 값		
이름	10진수 값	16진수 값
MQPA_DEFAULT	1	X'00000001'
MQPA_CONTEXT	2	X'00000002'
MQPA_ONLY_MCA	3	X'00000003'
MQPA_ALTERNATE_OR_MCA	4	X'00000004'

### MQPD\_\*(특성 디스크립터, 지원 및 컨텍스트)

#### 특성 디스크립터 구조

표 258. 상수 구조	
이름	구조
MQPD_STRUC_ID	"PD<sub>nn</sub>"
MQPD_STRUC_ID_ARRAY	'P','D','<sub>nn</sub>','<sub>nn</sub>'

참고: <sub>nn</sub> 기호는 단일 공백 문자를 나타냅니다.

표 259. 상수 값		
이름	10진수 값	16진수 값
MQPD_VERSION_1	1	X'00000001'
MQPD_CURRENT_VERSION	1	X'00000001'

#### 특성 디스크립터 옵션

표 260. 상수 값		
이름	10진수 값	16진수 값
MQPD_NONE	0	X'00000000'



## 특성 지원 옵션

표 261. 상수 값		
이름	10진수 값	16진수 값
MQPD_SUPPORT_OPTIONAL	1	X'00000001'
MQPD_SUPPORT_REQUIRED	1048576	X'00100000'
MQPD_SUPPORT_REQUIRED_IF_LOCAL	1024	X'00000400'
MQPD_REJECT_UNSUP_MASK	-1048576	X'FFF00000'
MQPD_ACCEPT_UNSUP_IF_XMIT_MASK	1047552	X'000FFC00'
MQPD_ACCEPT_UNSUP_MASK	1023	X'000003FF'

## 특성 컨텍스트

표 262. 상수 값		
이름	10진수 값	16진수 값
MQPD_NO_CONTEXT	0	X'00000000'
MQPD_USER_CONTEXT	1	X'00000001'

## MQPER\_\*(지속성 값)

표 263. 상수 값		
이름	10진수 값	16진수 값
MQPER_PERSISTENCE_AS_PARENT	-1	X'FFFFFFFF'
MQPER_NOT_PERSISTENT	0	X'00000000'
MQPER_PERSISTENT	1	X'00000001'
MQPER_PERSISTENCE_AS_Q_DEF	2	X'00000002'
MQPER_PERSISTENCE_AS_TOPIC_DEF	2	X'00000002'

## MQPL\_\*(플랫폼)

표 264. 상수 값		
이름	10진수 값	16진수 값
MQPL_MVS	1	X'00000001'
MQPL_OS390	1	X'00000001'
MQPL_ZOS	1	X'00000001'
MQPL_OS2	2	X'00000002'
MQPL_AIX	3	X'00000003'
MQPL_UNIX	3	X'00000003'
MQPL_OS400	4	X'00000004'
MQPL_WINDOWS	5	X'00000005'
MQPL_WINDOWS_NT	11	X'0000000B'
MQPL_VMS	12	X'0000000C'
MQPL_NSK	13	X'0000000D'
MQPL_OPEN_TP1	15	X'0000000F'
MQPL_VM	18	X'00000012'

표 264. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQPL_TPF	23	X'00000017'
MQPL_VSE	27	X'0000001B'
MQPL_APPLIANCE	28	X'0000001C'
MQPL_NATIVE	1	X'00000001'

## MQPMO\_\*(공개 마스크에 대한 메시지 넣기 옵션 및 구조)

### 메시지 넣기 옵션 구조

표 265. 상수 구조	
이름	구조
MQPMO_STRUC_ID	"PMO↵"
MQPMO_STRUC_ID_ARRAY	'P', 'M', 'O', '↵'

참고: ↵ 기호는 단일 공백 문자를 나타냅니다.

표 266. 상수 값		
이름	10진수 값	16진수 값
MQPMO_VERSION_1	1	X'00000001'
MQPMO_VERSION_2	2	X'00000002'
MQPMO_VERSION_3	3	X'00000003'
MQPMO_CURRENT_VERSION	3	X'00000003'
MQPMO_CURRENT_LENGTH	(value differs by platform or version)	(value differs by platform or version)

### 메시지 넣기 옵션

표 267. 상수 값		
이름	10진수 값	16진수 값
MQPMO_SYNCPOINT	2	X'00000002'
MQPMO_NO_SYNCPOINT	4	X'00000004'
MQPMO_DEFAULT_CONTEXT	32	X'00000020'
MQPMO_NEW_MSG_ID	64	X'00000040'
MQPMO_NEW_CORREL_ID	128	X'00000080'
MQPMO_PASS_IDENTITY_CONTEXT	256	X'00000100'
MQPMO_PASS_ALL_CONTEXT	512	X'00000200'
MQPMO_SET_IDENTITY_CONTEXT	1024	X'00000400'
MQPMO_SET_ALL_CONTEXT	2048	X'00000800'
MQPMO_ALTERNATE_USER_AUTHORITY	4096	X'00001000'
MQPMO_FAIL_IF QUIESCING	8192	X'00002000'
MQPMO_NO_CONTEXT	16384	X'00004000'
MQPMO_LOGICAL_ORDER	32768	X'00008000'
MQPMO_ASYNC_RESPONSE	65536	X'00010000'

표 267. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQPMO_SYNC_RESPONSE	131072	X'00020000'
MQPMO_RESOLVE_LOCAL_Q	262144	X'00040000'
MQPMO_RETAIN	2097152	X'00200000'
MQPMO_MD_FOR_OUTPUT_ONLY	8388608	X'00800000'
MQPMO_SCOPE_QMGR	67108864	X'04000000'
MQPMO_SUPPRESS_REPLYTO	134217728	X'08000000'
MQPMO_NOT_OWN_SUBS	268435456	X'10000000'
MQPMO_RESPONSE_AS_Q_DEF	0	X'00000000'
MQPMO_RESPONSE_AS_TOPIC_DEF	0	X'00000000'
MQPMO_NONE	0	X'00000000'

### 공개 마스크에 대한 메시지 넣기 옵션

표 268. 상수 값		
이름	10진수 값	16진수 값
MQPMO_PUB_OPTIONS_MASK	2097152	X'00200000'

### MQPMRF\_\*(메시지 넣기 레코드 필드)

표 269. 상수 값		
이름	10진수 값	16진수 값
MQPMRF_MSG_ID	1	X'00000001'
MQPMRF_CORREL_ID	2	X'00000002'
MQPMRF_GROUP_ID	4	X'00000004'
MQPMRF_FEEDBACK	8	X'00000008'
MQPMRF_ACCOUNTING_TOKEN	16	X'00000010'
MQPMRF_NONE	0	X'00000000'

### MQPO\_\*(명령 형식 영구 제거 옵션)

표 270. 상수 값		
이름	10진수 값	16진수 값
MQPO_YES	1	X'00000001'
MQPO_NO	0	X'00000000'

### MQPRI\_\*(우선순위)

표 271. 상수 값		
이름	10진수 값	16진수 값
MQPRI_PRIORITY_AS_Q_DEF	-1	X'FFFFFFFF'
MQPRI_PRIORITY_AS_PARENT	-2	X'FFFFFFFE'
MQPRI_PRIORITY_AS_PUBLISHED	-3	X'FFFFFFFD'
MQPRI_PRIORITY_AS_TOPIC_DEF	-1	X'FFFFFFFF'

## MQPROP\_\*(큐 및 채널 특성 제어 값 및 최대 특성 길이)

### 큐 및 채널 특성 제어 값

표 272. 상수 값		
이름	10진수 값	16진수 값
MQPROP_COMPATIBILITY	0	X'00000000'
MQPROP_NONE	1	X'00000001'
MQPROP_ALL	2	X'00000002'
MQPROP_FORCE_MQRFH2	3	X'00000003'

### 최대 특성 길이

표 273. 상수 값		
이름	10진수 값	16진수 값
MQPROP_UNRESTRICTED_LENGTH	-1	X'FFFFFFFF'

## MQPRT\_\*(넣기 응답 값)

표 274. 상수 값		
이름	10진수 값	16진수 값
MQPRT_RESPONSE_AS_PARENT	0	X'00000000'
MQPRT_SYNC_RESPONSE	1	X'00000001'
MQPRT_ASYNC_RESPONSE	2	X'00000002'

## MQPS\_\*(발행/구독)

### 명령 형식 발행/구독 상태

표 275. 상수 값		
이름	10진수 값	16진수 값
MQPS_STATUS_INACTIVE	0	X'00000000'
MQPS_STATUS_STARTING	1	X'00000001'
MQPS_STATUS_STOPPING	2	X'00000002'
MQPS_STATUS_ACTIVE	3	X'00000003'
MQPS_STATUS_COMPAT	4	X'00000004'
MQPS_STATUS_ERROR	5	X'00000005'
MQPS_STATUS_REFUSED	6	X'00000006'

### 문자열로서의 발행/구독 태그

MQPS_COMMAND	"MQPSCommand"
MQPS_COMP_CODE	"MQPSCompCode"
MQPS_CORREL_ID	"MQPSCorrelId"
MQPS_DELETE_OPTIONS	"MQPSDelOpts"
MQPS_ERROR_ID	"MQPSErrorId"

MQPS_ERROR_POS	"MQPSErrorPos"
MQPS_INTEGER_DATA	"MQPSIntData"
MQPS_PARAMETER_ID	"MQSParmId"
MQPS_PUBLICATION_OPTIONS	"MQSPubOpts"
MQPS_PUBLISH_TIMESTAMP	"MQSPubTime"
MQPS_Q_MGR_NAME	"MQPSQMgrName"
MQPS_Q_NAME	"MQPSQName"
MQPS_REASON	"MQPSReason"
MQPS_REASON_TEXT	"MQPSReasonText"
MQPS_REGISTRATION_OPTIONS	"MQPSRegOpts"
MQPS_SEQUENCE_NUMBER	"MQPSSeqNum"
MQPS_STREAM_NAME	"MQPSStreamName"
MQPS_STRING_DATA	"MQPSStringData"
MQPS_SUBSCRIPTION_IDENTITY	"MQPSSubIdentity"
MQPS_SUBSCRIPTION_NAME	"MQPSSubName"
MQPS_SUBSCRIPTION_USER_DATA	"MQPSSubUserData"
MQPS_TOPIC	"MQPSTopic"
MQPS_USER_ID	"MQPSUserId"

**공백으로 묶인 문자열로서의 발행/구독 태그**

MQPS_COMMAND_B	"-MQPSCommand-"
MQPS_COMP_CODE_B	"-MQPSCompCode-"
MQPS_CORREL_ID_B	"-MQPSCorrelId-"
MQPS_DELETE_OPTIONS_B	"-MQPSDelOpts-"
MQPS_ERROR_ID_B	"-MQPSErrorId-"
MQPS_ERROR_POS_B	"-MQPSErrorPos-"
MQPS_INTEGER_DATA_B	"-MQPSIntData-"
MQPS_PARAMETER_ID_B	"-MQSParmId-"
MQPS_PUBLICATION_OPTIONS_B	"-MQSPubOpts-"
MQPS_PUBLISH_TIMESTAMP_B	"-MQSPubTime-"
MQPS_Q_MGR_NAME_B	"-MQPSQMgrName-"
MQPS_Q_NAME_B	"-MQPSQName-"
MQPS_REASON_B	"-MQPSReason-"
MQPS_REASON_TEXT_B	"-MQPSReasonText-"
MQPS_REGISTRATION_OPTIONS_B	"-MQPSRegOpts-"
MQPS_SEQUENCE_NUMBER_B	"-MQPSSeqNum-"

MQPS_STREAM_NAME_B	"~MQPSStreamName~"
MQPS_STRING_DATA_B	"~MQPSStringData~"
MQPS_SUBSCRIPTION_IDENTITY_B	"~MQPSSubIdentity~"
MQPS_SUBSCRIPTION_NAME_B	"~MQPSSubName~"
MQPS_SUBSCRIPTION_USER_DATA_B	"~MQPSSubUserData~"
MQPS_TOPIC_B	"~MQPSTopic~"
MQPS_USER_ID_B	"~MQPSUserId~"

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

### 문자열로서의 발행/구독 명령 태그 값

MQPS_DELETE_PUBLICATION	"DeletePub"
MQPS_DEREGISTER_PUBLISHER	"DeregPub"
MQPS_DEREGISTER_SUBSCRIBER	"DeregSub"
MQPS_PUBLISH	"Publish"
MQPS_REGISTER_PUBLISHER	"RegPub"
MQPS_REGISTER_SUBSCRIBER	"RegSub"
MQPS_REQUEST_UPDATE	"ReqUpdate"

### 공백으로 묶인 문자열로서의 발행/구독 명령 태그 값

MQPS_DELETE_PUBLICATION_B	"~DeletePub~"
MQPS_DEREGISTER_PUBLISHER_B	"~DeregPub~"
MQPS_DEREGISTER_SUBSCRIBER_B	"~DeregSub~"
MQPS_PUBLISH_B	"~Publish~"
MQPS_REGISTER_PUBLISHER_B	"~RegPub~"
MQPS_REGISTER_SUBSCRIBER_B	"~RegSub~"
MQPS_REQUEST_UPDATE_B	"~ReqUpdate~"

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

### 문자열로서의 발행/구독 옵션 태그 값

MQPS_ADD_NAME	"AddName"
MQPS_ANONYMOUS	"Anon"
MQPS_CORREL_ID_AS_IDENTITY	"CorrelAsId"
MQPS_DEREGISTER_ALL	"DeregAll"
MQPS_DIRECT_REQUESTS	"DirectReq"
MQPS_DUPLICATES_OK	"DupsOK"
MQPS_FULL_RESPONSE	"FullResp"

MQPS_INCLUDE_STREAM_NAME	"InclStreamName"
MQPS_INFORM_IF_RETAINED	"InformIfRet"
MQPS_IS_RETAINED_PUBLICATION	"IsRetainedPub"
MQPS_JOIN_EXCLUSIVE	"JoinExcl"
MQPS_JOIN_SHARED	"JoinShared"
MQPS_LEAVE_ONLY	"LeaveOnly"
MQPS_LOCAL	"Local"
MQPS_LOCKED	"Locked"
MQPS_NEW_PUBLICATIONS_ONLY	"NewPubsOnly"
MQPS_NO_ALTERATION	"NoAlter"
MQPS_NO_REGISTRATION	"NoReg"
MQPS_NON_PERSISTENT	"NonPers"
MQPS_NONE	"None"
MQPS_OTHER_SUBSCRIBERS_ONLY	"OtherSubsOnly"
MQPS_PERSISTENT	"Pers"
MQPS_PERSISTENT_AS_PUBLISH	"PersAsPub"
MQPS_PERSISTENT_AS_Q	"PersAsQueue"
MQPS_PUBLISH_ON_REQUEST_ONLY	"PubOnReqOnly"
MQPS_RETAIN_PUBLICATION	"RetainPub"
MQPS_VARIABLE_USER_ID	"VariableUserId"

**공백으로 묶인 문자열로서의 발행/구독 옵션 태그 값**

MQPS_ADD_NAME_B	"-AddName-"
MQPS_ANONYMOUS_B	"-Anon-"
MQPS_CORREL_ID_AS_IDENTITY_B	"-CorrelAsId-"
MQPS_DEREGISTER_ALL_B	"-DeregAll-"
MQPS_DIRECT_REQUESTS_B	"-DirectReq-"
MQPS_DUPLICATES_OK_B	"-DupsOK-"
MQPS_FULL_RESPONSE_B	"-FullResp-"
MQPS_INCLUDE_STREAM_NAME_B	"-InclStreamName-"
MQPS_INFORM_IF_RETAINED_B	"-InformIfRet-"
MQPS_IS_RETAINED_PUBLICATION_B	"-IsRetainedPub-"
MQPS_JOIN_EXCLUSIVE_B	"-JoinExcl-"
MQPS_JOIN_SHARED_B	"-JoinShared-"
MQPS_LEAVE_ONLY_B	"-LeaveOnly-"
MQPS_LOCAL_B	"-Local-"

MQPS_LOCKED_B	"¬Locked¬"
MQPS_NEW_PUBLICATIONS_ONLY_B	"¬NewPubsOnly¬"
MQPS_NO_ALTERATION_B	"¬NoAlter¬"
MQPS_NO_REGISTRATION_B	"¬NoReg¬"
MQPS_NON_PERSISTENT_B	"¬NonPers¬"
MQPS_NONE_B	"¬None¬"
MQPS_OTHER_SUBSCRIBERS_ONLY_B	"¬OtherSubsOnly¬"
MQPS_PERSISTENT_B	"¬Pers¬"
MQPS_PERSISTENT_AS_PUBLISH_B	"¬PersAsPub¬"
MQPS_PERSISTENT_AS_Q_B	"¬PersAsQueue¬"
MQPS_PUBLISH_ON_REQUEST_ONLY_B	"¬PubOnReqOnly¬"
MQPS_RETAIN_PUBLICATION_B	"¬RetainPub¬"
MQPS_VARIABLE_USER_ID_B	"¬VariableUserId¬"

참고: ¬ 기호는 단일 공백 문자를 나타냅니다.

### MQPSC\_\*(발행/구독 옵션 태그 발행/구독 명령 폴더(psc) 태그)

표 276. 상수 값		
이름	10진수 값	16진수 값
MQPSC_FOLDER_VERSION	1	X'00000001'

### MQPSC\_\*(발행/구독 옵션 태그 태그 이름)

MQPSC_COMMAND	"Command"
MQPSC_REGISTRATION_OPTION	"RegOpt"
MQPSC_PUBLICATION_OPTION	"PubOpt"
MQPSC_DELETE_OPTION	"DelOpt"
MQPSC_TOPIC	"Topic"
MQPSC_SUBSCRIPTION_POINT	"SubPoint"
MQPSC_FILTER	"Filter"
MQPSC_Q_MGR_NAME	"QMgrName"
MQPSC_Q_NAME	"QName"
MQPSC_PUBLISH_TIMESTAMP	"PubTime"
MQPSC_SEQUENCE_NUMBER	"SeqNum"
MQPSC_SUBSCRIPTION_NAME	"SubName"
MQPSC_SUBSCRIPTION_IDENTITY	"SubIdentity"
MQPSC_SUBSCRIPTION_USER_DATA	"SubUserData"
MQPSC_CORREL_ID	"CorrelId"



**MQPSC\_\*(발행/구독 옵션 태그 XML 태그 이름)**

MQPSC_COMMAND_B	"<Command>"
MQPSC_COMMAND_E	"</Command>"
MQPSC_REGISTRATION_OPTION_B	"<RegOpt>"
MQPSC_REGISTRATION_OPTION_E	"</RegOpt>"
MQPSC_PUBLICATION_OPTION_B	"<PubOpt>"
MQPSC_PUBLICATION_OPTION_E	"</PubOpt>"
MQPSC_DELETE_OPTION_B	"<DelOpt>"
MQPSC_DELETE_OPTION_E	"</DelOpt>"
MQPSC_TOPIC_B	"<Topic>"
MQPSC_TOPIC_E	"</Topic>"
MQPSC_SUBSCRIPTION_POINT_B	"<SubPoint>"
MQPSC_SUBSCRIPTION_POINT_E	"</SubPoint>"
MQPSC_FILTER_B	"<Filter>"
MQPSC_FILTER_E	"</Filter>"
MQPSC_Q_MGR_NAME_B	"<QMgrName>"
MQPSC_Q_MGR_NAME_E	"</QMgrName>"
MQPSC_Q_NAME_B	"<QName>"
MQPSC_Q_NAME_E	"</QName>"
MQPSC_PUBLISH_TIMESTAMP_B	"<PubTime>"
MQPSC_PUBLISH_TIMESTAMP_E	"</PubTime>"
MQPSC_SEQUENCE_NUMBER_B	"<SeqNum>"
MQPSC_SEQUENCE_NUMBER_E	"</SeqNum>"
MQPSC_SUBSCRIPTION_NAME_B	"<SubName>"
MQPSC_SUBSCRIPTION_NAME_E	"</SubName>"
MQPSC_SUBSCRIPTION_IDENTITY_B	"<SubIdentity>"
MQPSC_SUBSCRIPTION_IDENTITY_E	"</SubIdentity>"
MQPSC_SUBSCRIPTION_USER_DATA_B	"<SubUserData>"
MQPSC_SUBSCRIPTION_USER_DATA_E	"</SubUserData>"
MQPSC_CORREL_ID_B	"<CorrelId>"
MQPSC_CORREL_ID_E	"</CorrelId>"

**MQPSC\_\*(문자열로서의 발행/구독 옵션 태그 발행자 값)**

MQPSC_DELETE_PUBLICATION	"DeletePub"
MQPSC_DEREGISTER_SUBSCRIBER	"DeregSub"
MQPSC_PUBLISH	"Publish"
MQPSC_REGISTER_SUBSCRIBER	"RegSub"

MQPSC_REQUEST_UPDATE	"ReqUpdate"
----------------------	-------------

**MQPSC\_\*(문자열로서의 발행/구독 옵션 태그 이름 값)**

MQPSC_ADD_NAME	"AddName"
MQPSC_CORREL_ID_AS_IDENTITY	"CorrelAsId"
MQPSC_DEREGISTER_ALL	"DeregAll"
MQPSC_DUPLICATES_OK	"DupsOK"
MQPSC_FULL_RESPONSE	"FullResp"
MQPSC_INFORM_IF_RETAINED	"InformIfRet"
MQPSC_IS_RETAINED_PUB	"IsRetainedPub"
MQPSC_JOIN_SHARED	"JoinShared"
MQPSC_JOIN_EXCLUSIVE	"JoinExcl"
MQPSC_LEAVE_ONLY	"LeaveOnly"
MQPSC_LOCAL	"Local"
MQPSC_LOCKED	"Locked"
MQPSC_NEW_PUBS_ONLY	"NewPubsOnly"
MQPSC_NO_ALTERATION	"NoAlter"
MQPSC_NON_PERSISTENT	"NonPers"
MQPSC_OTHER_SUBS_ONLY	"OtherSubsOnly"
MQPSC_PERSISTENT	"Pers"
MQPSC_PERSISTENT_AS_PUBLISH	"PersAsPub"
MQPSC_PERSISTENT_AS_Q	"PersAsQueue"
MQPSC_NONE	"None"
MQPSC_PUB_ON_REQUEST_ONLY	"PubOnReqOnly"
MQPSC_RETAIN_PUB	"RetainPub"
MQPSC_VARIABLE_USER_ID	"VariableUserId"

**MQPSCR\_\*(발행/구독 옵션)**

**발행/구독 옵션 태그 발행/구독 응답 폴더(pscr) 태그**

표 277. 상수 값		
이름	10진수 값	16진수 값
MQPSCR_FOLDER_VERSION	1	X'00000001'

**발행/구독 옵션 태그 태그 이름**

MQPSCR_COMPLETION	"Completion"
MQPSCR_RESPONSE	"Response"
MQPSCR_REASON	"Reason"

### 발행/구독 옵션 태그 XML 태그 이름

MQPSCR_COMPLETION_B	"<Completion>"
MQPSCR_COMPLETION_E	"</Completion>"
MQPSCR_RESPONSE_B	"<Response>"
MQPSCR_RESPONSE_E	"</Response>"
MQPSCR_REASON_B	"<Reason>"
MQPSCR_REASON_E	"</Reason>"

### 발행/구독 옵션 태그 태그 값

MQPSCR_OK	"ok"
MQPSCR_WARNING	"warning"
MQPSCR_ERROR	"error"

### MQPSM\_\*(발행/구독 모드)

표 278. 상수 값		
이름	10진수 값	16진수 값
MQPSM_DISABLED	0	X'00000000'
MQPSM_COMPAT	1	X'00000001'
MQPSM_ENABLED	2	X'00000002'

### MQPSPROP\_\*(발행/구독 메시지 특성)

표 279. 상수 값		
이름	10진수 값	16진수 값
MQPSPROP_NONE	0	X'00000000'
MQPSPROP_COMPAT	1	X'00000001'
MQPSPROP_RFH2	2	X'00000002'
MQPSPROP_MSGPROP	3	X'00000003'

### MQPSST\_\*(명령 형식 발행/구독 상태 유형)

표 280. 상수 값		
이름	10진수 값	16진수 값
MQPSST_ALL	0	X'00000000'
MQPSST_LOCAL	1	X'00000001'
MQPSST_PARENT	2	X'00000002'
MQPSST_CHILD	3	X'00000003'

### MQPUBO\_\*(발행물 발행/구독 옵션)

표 281. 상수 값		
이름	10진수 값	16진수 값
MQPUBO_NONE	0	X'00000000'

표 281. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQPUBO_CORREL_ID_AS_IDENTITY	1	X'00000001'
MQPUBO_RETAIN_PUBLICATION	2	X'00000002'
MQPUBO_OTHER_SUBSCRIBERS_ONLY	4	X'00000004'
MQPUBO_NO_REGISTRATION	8	X'00000008'
MQPUBO_IS_RETAINED_PUBLICATION	16	X'00000010'

### MQPXP\_\*(발행/구독 라우팅 엑시트 매개변수 구조)

표 282. 상수 구조	
이름	구조
MQPXP_STRUC_ID	"PXP↵"
MQPXP_STRUC_ID_ARRAY	'P','X','P','↵'

참고: ↵ 기호는 단일 공백 문자를 나타냅니다.

표 283. 상수 값		
이름	10진수 값	16진수 값
MQPXP_VERSION_1	1	X'00000001'
MQPXP_CURRENT_VERSION	1	X'00000001'

### MQQA\_\*(큐 속성)

#### 가져오기 금지 값

표 284. 상수 값		
이름	10진수 값	16진수 값
MQQA_GET_INHIBITED	1	X'00000001'
MQQA_GET_ALLOWED	0	X'00000000'

#### 넣기 금지 값

표 285. 상수 값		
이름	10진수 값	16진수 값
MQQA_PUT_INHIBITED	1	X'00000001'
MQQA_PUT_ALLOWED	0	X'00000000'

#### 큐 공유 가능성

표 286. 상수 값		
이름	10진수 값	16진수 값
MQQA_SHAREABLE	1	X'00000001'
MQQA_NOT_SHAREABLE	0	X'00000000'

## 백아웃 경화

표 287. 상수 값		
이름	10진수 값	16진수 값
MQQA_BACKOUT_HARDENED	1	X'00000001'
MQQA_BACKOUT_NOT_HARDENED	0	X'00000000'

## MQQDT\_\*(큐 정의 유형)

표 288. 상수 값		
이름	10진수 값	16진수 값
MQQDT_PREDEFINED	1	X'00000001'
MQQDT_PERMANENT_DYNAMIC	2	X'00000002'
MQQDT_TEMPORARY_DYNAMIC	3	X'00000003'
MQQDT_SHARED_DYNAMIC	4	X'00000004'

## MQQF\_\*(큐 플래그)

표 289. 상수 값		
이름	10진수 값	16진수 값
MQQF_LOCAL_Q	1	X'00000001'
MQQF_CLWL_USEQ_ANY	64	X'00000040'
MQQF_CLWL_USEQ_LOCAL	128	X'00000080'

## MQQMDT\_\*(명령 형식 큐 관리자 정의 유형)

표 290. 상수 값		
이름	10진수 값	16진수 값
MQQMDT_EXPLICIT_CLUSTER_SENDER	1	X'00000001'
MQQMDT_AUTO_CLUSTER_SENDER	2	X'00000002'
MQQMDT_AUTO_EXP_CLUSTER_SENDER	4	X'00000004'
MQQMDT_CLUSTER_RECEIVER	3	X'00000003'

## MQQMF\_\*(큐 관리자 플래그)

표 291. 상수 값		
이름	10진수 값	16진수 값
MQQMF_REPOSITORY_Q_MGR	2	X'00000002'
MQQMF_CLUSSDR_USER_DEFINED	8	X'00000008'
MQQMF_CLUSSDR_AUTO_DEFINED	16	X'00000010'
MQQMF_AVAILABLE	32	X'00000020'

## MQQMFC\_\*(명령 형식 큐 관리자 기능)

표 292. 상수 값		
이름	10진수 값	16진수 값
MQQMFC_IMS_BRIDGE	1	X'00000001'

표 292. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQQMFACT_DB2	2	X'00000002'

### MQQMSTA\_\*(명령 형식 큐 관리자 상태)

표 293. 상수 값		
이름	10진수 값	16진수 값
MQQMSTA_STARTING	1	X'00000001'
MQQMSTA_RUNNING	2	X'00000002'
MQQMSTA QUIESCING	3	X'00000003'

### MQQMT\_\*(명령 형식 큐 관리자 유형)

표 294. 상수 값		
이름	10진수 값	16진수 값
MQQMT_NORMAL	0	X'00000000'
MQQMT_REPOSITORY	1	X'00000001'

### MQQO\_\*(명령 형식 일시정지 옵션)

표 295. 상수 값		
이름	10진수 값	16진수 값
MQQO_YES	1	X'00000001'
MQQO_NO	0	X'00000000'

### MQQSGD\_\*(큐 공유 그룹 속성 지정)

표 296. 상수 값		
이름	10진수 값	16진수 값
MQQSGD_ALL	-1	X'FFFFFFFF'
MQQSGD_Q_MGR	0	X'00000000'
MQQSGD_COPY	1	X'00000001'
MQQSGD_SHARED	2	X'00000002'
MQQSGD_GROUP	3	X'00000003'
MQQSGD_PRIVATE	4	X'00000004'
MQQSGD_LIVE	6	X'00000006'

### MQQSGS\_\*(명령 형식 큐 공유 그룹 상태)

표 297. 상수 값		
이름	10진수 값	16진수 값
MQQSGS_UNKNOWN	0	X'00000000'
MQQSGS_CREATED	1	X'00000001'
MQQSGS_ACTIVE	2	X'00000002'
MQQSGS_INACTIVE	3	X'00000003'
MQQSGS_FAILED	4	X'00000004'

표 297. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQQSGS_PENDING	5	X'00000005'

### MQQSIE\_\*(명령 형식 큐 서비스 간격 이벤트)

표 298. 상수 값		
이름	10진수 값	16진수 값
MQQSIE_NONE	0	X'00000000'
MQQSIE_HIGH	1	X'00000001'
MQQSIE_OK	2	X'00000002'

### MQQSO\_\*(SET, BROWSE, INPUT의 명령 형식 큐 상태 열기 옵션)

표 299. 상수 값		
이름	10진수 값	16진수 값
MQQSO_NO	0	X'00000000'
MQQSO_YES	1	X'00000001'
MQQSO_SHARED	1	X'00000001'
MQQSO_EXCLUSIVE	2	X'00000002'

### MQQSOT\_\*(명령 형식 큐 상태 열기 유형)

표 300. 상수 값		
이름	10진수 값	16진수 값
MQQSOT_ALL	1	X'00000001'
MQQSOT_INPUT	2	X'00000002'
MQQSOT_OUTPUT	3	X'00000003'

### MQQSUM\_\*(명령 형식 큐 상태 언커미트된 메시지)

표 301. 상수 값		
이름	10진수 값	16진수 값
MQQSUM_YES	1	X'00000001'
MQQSUM_NO	0	X'00000000'

### MQQT\_\*(큐 유형 및 확장 큐 유형)

#### 큐 유형

표 302. 상수 값		
이름	10진수 값	16진수 값
MQQT_LOCAL	1	X'00000001'
MQQT_MODEL	2	X'00000002'
MQQT_ALIAS	3	X'00000003'
MQQT_REMOTE	6	X'00000006'
MQQT_CLUSTER	7	X'00000007'

## 확장 큐 유형

표 303. 상수 값		
이름	10진수 값	16진수 값
MQQT_ALL	1001	X'000003E9'

## MQRC\_\* (이유 코드)

표 304. 상수 값		
이름	10진수 값	16진수 값
MQRC_NONE	0	X'00000000'
MQRC_APPL_FIRST	900	X'00000384'
MQRC_APPL_LAST	999	X'000003E7'
MQRC_ALIAS_BASE_Q_TYPE_ERROR	2001	X'000007D1'
MQRC_ALREADY_CONNECTED	2002	X'000007D2'
MQRC_BACKED_OUT	2003	X'000007D3'
MQRC_BUFFER_ERROR	2004	X'000007D4'
MQRC_BUFFER_LENGTH_ERROR	2005	X'000007D5'
MQRC_CHAR_ATTR_LENGTH_ERROR	2006	X'000007D6'
MQRC_CHAR_ATTRS_ERROR	2007	X'000007D7'
MQRC_CHAR_ATTRS_TOO_SHORT	2008	X'000007D8'
MQRC_CONNECTION_BROKEN	2009	X'000007D9'
MQRC_DATA_LENGTH_ERROR	2010	X'000007DA'
MQRC_DYNAMIC_Q_NAME_ERROR	2011	X'000007DB'
MQRC_ENVIRONMENT_ERROR	2012	X'000007DC'
MQRC_EXPIRY_ERROR	2013	X'000007DD'
MQRC_FEEDBACK_ERROR	2014	X'000007DE'
MQRC_GET_INHIBITED	2016	X'000007E0'
MQRC_HANDLE_NOT_AVAILABLE	2017	X'000007E1'
MQRC_HCONN_ERROR	2018	X'000007E2'
MQRC_HOBJ_ERROR	2019	X'000007E3'
MQRC_INHIBIT_VALUE_ERROR	2020	X'000007E4'
MQRC_INT_ATTR_COUNT_ERROR	2021	X'000007E5'
MQRC_INT_ATTR_COUNT_TOO_SMALL	2022	X'000007E6'
MQRC_INT_ATTRS_ARRAY_ERROR	2023	X'000007E7'
MQRC_SYNCPOINT_LIMIT_REACHED	2024	X'000007E8'
MQRC_MAX_CONNS_LIMIT_REACHED	2025	X'000007E9'
MQRC_MD_ERROR	2026	X'000007EA'
MQRC_MISSING_REPLY_TO_Q	2027	X'000007EB'
MQRC_MSG_TYPE_ERROR	2029	X'000007ED'
MQRC_MSG_TOO_BIG_FOR_Q	2030	X'000007EE'
MQRC_MSG_TOO_BIG_FOR_Q_MGR	2031	X'000007EF'
MQRC_NO_MSG_AVAILABLE	2033	X'000007F1'



표 304. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRC_NO_MSG_UNDER_CURSOR	2034	X'000007F2'
MQRC_NOT_AUTHORIZED	2035	X'000007F3'
MQRC_NOT_OPEN_FOR_BROWSE	2036	X'000007F4'
MQRC_NOT_OPEN_FOR_INPUT	2037	X'000007F5'
MQRC_NOT_OPEN_FOR_INQUIRE	2038	X'000007F6'
MQRC_NOT_OPEN_FOR_OUTPUT	2039	X'000007F7'
MQRC_NOT_OPEN_FOR_SET	2040	X'000007F8'
MQRC_OBJECT_CHANGED	2041	X'000007F9'
MQRC_OBJECT_IN_USE	2042	X'000007FA'
MQRC_OBJECT_TYPE_ERROR	2043	X'000007FB'
MQRC_OD_ERROR	2044	X'000007FC'
MQRC_OPTION_NOT_VALID_FOR_TYPE	2045	X'000007FD'
MQRC_OPTIONS_ERROR	2046	X'000007FE'
MQRC_PERSISTENCE_ERROR	2047	X'000007FF'
MQRC_PERSISTENT_NOT_ALLOWED	2048	X'00000800'
MQRC_PRIORITY_EXCEEDS_MAXIMUM	2049	X'00000801'
MQRC_PRIORITY_ERROR	2050	X'00000802'
MQRC_PUT_INHIBITED	2051	X'00000803'
MQRC_Q_DELETED	2052	X'00000804'
MQRC_Q_FULL	2053	X'00000805'
MQRC_Q_NOT_EMPTY	2055	X'00000807'
MQRC_Q_SPACE_NOT_AVAILABLE	2056	X'00000808'
MQRC_Q_TYPE_ERROR	2057	X'00000809'
MQRC_Q_MGR_NAME_ERROR	2058	X'0000080A'
MQRC_Q_MGR_NOT_AVAILABLE	2059	X'0000080B'
MQRC_REPORT_OPTIONS_ERROR	2061	X'0000080D'
MQRC_SECOND_MARK_NOT_ALLOWED	2062	X'0000080E'
MQRC_SECURITY_ERROR	2063	X'0000080F'
MQRC_SELECTOR_COUNT_ERROR	2065	X'00000811'
MQRC_SELECTOR_LIMIT_EXCEEDED	2066	X'00000812'
MQRC_SELECTOR_ERROR	2067	X'00000813'
MQRC_SELECTOR_NOT_FOR_TYPE	2068	X'00000814'
MQRC_SIGNAL_OUTSTANDING	2069	X'00000815'
MQRC_SIGNAL_REQUEST_ACCEPTED	2070	X'00000816'
MQRC_STORAGE_NOT_AVAILABLE	2071	X'00000817'
MQRC_SYNCPOINT_NOT_AVAILABLE	2072	X'00000818'
MQRC_TRIGGER_CONTROL_ERROR	2075	X'0000081B'
MQRC_TRIGGER_DEPTH_ERROR	2076	X'0000081C'
MQRC_TRIGGER_MSG_PRIORITY_ERR	2077	X'0000081D'

표 304. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRC_TRIGGER_TYPE_ERROR	2078	X'0000081E'
MQRC_TRUNCATED_MSG_ACCEPTED	2079	X'0000081F'
MQRC_TRUNCATED_MSG_FAILED	2080	X'00000820'
MQRC_UNKNOWN_ALIAS_BASE_Q	2082	X'00000822'
MQRC_UNKNOWN_OBJECT_NAME	2085	X'00000825'
MQRC_UNKNOWN_OBJECT_Q_MGR	2086	X'00000826'
MQRC_UNKNOWN_REMOTE_Q_MGR	2087	X'00000827'
MQRC_WAIT_INTERVAL_ERROR	2090	X'0000082A'
MQRC_XMIT_Q_TYPE_ERROR	2091	X'0000082B'
MQRC_XMIT_Q_USAGE_ERROR	2092	X'0000082C'
MQRC_NOT_OPEN_FOR_PASS_ALL	2093	X'0000082D'
MQRC_NOT_OPEN_FOR_PASS_IDENT	2094	X'0000082E'
MQRC_NOT_OPEN_FOR_SET_ALL	2095	X'0000082F'
MQRC_NOT_OPEN_FOR_SET_IDENT	2096	X'00000830'
MQRC_CONTEXT_HANDLE_ERROR	2097	X'00000831'
MQRC_CONTEXT_NOT_AVAILABLE	2098	X'00000832'
MQRC_SIGNAL1_ERROR	2099	X'00000833'
MQRC_OBJECT_ALREADY_EXISTS	2100	X'00000834'
MQRC_OBJECT_DAMAGED	2101	X'00000835'
MQRC_RESOURCE_PROBLEM	2102	X'00000836'
MQRC_ANOTHER_Q_MGR_CONNECTED	2103	X'00000837'
MQRC_UNKNOWN_REPORT_OPTION	2104	X'00000838'
MQRC_STORAGE_CLASS_ERROR	2105	X'00000839'
MQRC_COD_NOT_VALID_FOR_XCF_Q	2106	X'0000083A'
MQRC_XWAIT_CANCELED	2107	X'0000083B'
MQRC_XWAIT_ERROR	2108	X'0000083C'
MQRC_SUPPRESSED_BY_EXIT	2109	X'0000083D'
MQRC_FORMAT_ERROR	2110	X'0000083E'
MQRC_SOURCE_CCSID_ERROR	2111	X'0000083F'
MQRC_SOURCE_INTEGER_ENC_ERROR	2112	X'00000840'
MQRC_SOURCE_DECIMAL_ENC_ERROR	2113	X'00000841'
MQRC_SOURCE_FLOAT_ENC_ERROR	2114	X'00000842'
MQRC_TARGET_CCSID_ERROR	2115	X'00000843'
MQRC_TARGET_INTEGER_ENC_ERROR	2116	X'00000844'
MQRC_TARGET_DECIMAL_ENC_ERROR	2117	X'00000845'
MQRC_TARGET_FLOAT_ENC_ERROR	2118	X'00000846'
MQRC_NOT_CONVERTED	2119	X'00000847'
MQRC_CONVERTED_MSG_TOO_BIG	2120	X'00000848'
MQRC_TRUNCATED	2120	X'00000848'

표 304. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRC_NO_EXTERNAL_PARTICIPANTS	2121	X'00000849'
MQRC_PARTICIPANT_NOT_AVAILABLE	2122	X'0000084A'
MQRC_OUTCOME_MIXED	2123	X'0000084B'
MQRC_OUTCOME_PENDING	2124	X'0000084C'
MQRC_BRIDGE_STARTED	2125	X'0000084D'
MQRC_BRIDGE_STOPPED	2126	X'0000084E'
MQRC_ADAPTER_STORAGE_SHORTAGE	2127	X'0000084F'
MQRC_UOW_IN_PROGRESS	2128	X'00000850'
MQRC_ADAPTER_CONN_LOAD_ERROR	2129	X'00000851'
MQRC_ADAPTER_SERV_LOAD_ERROR	2130	X'00000852'
MQRC_ADAPTER_DEFS_ERROR	2131	X'00000853'
MQRC_ADAPTER_DEFS_LOAD_ERROR	2132	X'00000854'
MQRC_ADAPTER_CONV_LOAD_ERROR	2133	X'00000855'
MQRC_BO_ERROR	2134	X'00000856'
MQRC_DH_ERROR	2135	X'00000857'
MQRC_MULTIPLE_REASONS	2136	X'00000858'
MQRC_OPEN_FAILED	2137	X'00000859'
MQRC_ADAPTER_DISC_LOAD_ERROR	2138	X'0000085A'
MQRC_CNO_ERROR	2139	X'0000085B'
MQRC_CICS_WAIT_FAILED	2140	X'0000085C'
MQRC_DLH_ERROR	2141	X'0000085D'
MQRC_HEADER_ERROR	2142	X'0000085E'
MQRC_SOURCE_LENGTH_ERROR	2143	X'0000085F'
MQRC_TARGET_LENGTH_ERROR	2144	X'00000860'
MQRC_SOURCE_BUFFER_ERROR	2145	X'00000861'
MQRC_TARGET_BUFFER_ERROR	2146	X'00000862'
MQRC_IIH_ERROR	2148	X'00000864'
MQRC_PCF_ERROR	2149	X'00000865'
MQRC_DBCS_ERROR	2150	X'00000866'
MQRC_OBJECT_NAME_ERROR	2152	X'00000868'
MQRC_OBJECT_Q_MGR_NAME_ERROR	2153	X'00000869'
MQRC_RECS_PRESENT_ERROR	2154	X'0000086A'
MQRC_OBJECT_RECORDS_ERROR	2155	X'0000086B'
MQRC_RESPONSE_RECORDS_ERROR	2156	X'0000086C'
MQRC_ASID_MISMATCH	2157	X'0000086D'
MQRC_PMO_RECORD_FLAGS_ERROR	2158	X'0000086E'
MQRC_PUT_MSG_RECORDS_ERROR	2159	X'0000086F'
MQRC_CONN_ID_IN_USE	2160	X'00000870'
MQRC_Q_MGR QUIESCING	2161	X'00000871'

표 304. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRC_Q_MGR_STOPPING	2162	X'00000872'
MQRC_DUPLICATE_RECOV_COORD	2163	X'00000873'
MQRC_PMO_ERROR	2173	X'0000087D'
MQRC_API_EXIT_NOT_FOUND	2182	X'00000886'
MQRC_API_EXIT_LOAD_ERROR	2183	X'00000887'
MQRC_REMOTE_Q_NAME_ERROR	2184	X'00000888'
MQRC_INCONSISTENT_PERSISTENCE	2185	X'00000889'
MQRC_GMO_ERROR	2186	X'0000088A'
MQRC_CICS_BRIDGE_RESTRICTION	2187	X'0000088B'
MQRC_STOPPED_BY_CLUSTER_EXIT	2188	X'0000088C'
MQRC_CLUSTER_RESOLUTION_ERROR	2189	X'0000088D'
MQRC_CONVERTED_STRING_TOO_BIG	2190	X'0000088E'
MQRC_TMC_ERROR	2191	X'0000088F'
MQRC_PAGESET_FULL	2192	X'00000890'
MQRC_STORAGE_MEDIUM_FULL	2192	X'00000890'
MQRC_PAGESET_ERROR	2193	X'00000891'
MQRC_NAME_NOT_VALID_FOR_TYPE	2194	X'00000892'
MQRC_UNEXPECTED_ERROR	2195	X'00000893'
MQRC_UNKNOWN_XMIT_Q	2196	X'00000894'
MQRC_UNKNOWN_DEF_XMIT_Q	2197	X'00000895'
MQRC_DEF_XMIT_Q_TYPE_ERROR	2198	X'00000896'
MQRC_DEF_XMIT_Q_USAGE_ERROR	2199	X'00000897'
MQRC_MSG_MARKED_BROWSE_CO_OP	2200	X'00000898'
MQRC_NAME_IN_USE	2201	X'00000899'
MQRC_CONNECTION QUIESCING	2202	X'0000089A'
MQRC_CONNECTION_STOPPING	2203	X'0000089B'
MQRC_ADAPTER_NOT_AVAILABLE	2204	X'0000089C'
MQRC_MSG_ID_ERROR	2206	X'0000089E'
MQRC_CORREL_ID_ERROR	2207	X'0000089F'
MQRC_FILE_SYSTEM_ERROR	2208	X'000008A0'
MQRC_NO_MSG_LOCKED	2209	X'000008A1'
MQRC_SOAP_DOTNET_ERROR	2210	X'000008A2'
MQRC_SOAP_AXIS_ERROR	2211	X'000008A3'
MQRC_SOAP_URL_ERROR	2212	X'000008A4'
MQRC_FILE_NOT_AUDITED	2216	X'000008A8'
MQRC_CONNECTION_NOT_AUTHORIZED	2217	X'000008A9'
MQRC_MSG_TOO_BIG_FOR_CHANNEL	2218	X'000008AA'
MQRC_CALL_IN_PROGRESS	2219	X'000008AB'
MQRC_RMH_ERROR	2220	X'000008AC'

표 304. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRC_Q_MGR_ACTIVE	2222	X'000008AE'
MQRC_Q_MGR_NOT_ACTIVE	2223	X'000008AF'
MQRC_Q_DEPTH_HIGH	2224	X'000008B0'
MQRC_Q_DEPTH_LOW	2225	X'000008B1'
MQRC_Q_SERVICE_INTERVAL_HIGH	2226	X'000008B2'
MQRC_Q_SERVICE_INTERVAL_OK	2227	X'000008B3'
MQRC_RFH_HEADER_FIELD_ERROR	2228	X'000008B4'
MQRC_RAS_PROPERTY_ERROR	2229	X'000008B5'
MQRC_UNIT_OF_WORK_NOT_STARTED	2232	X'000008B8'
MQRC_CHANNEL_AUTO_DEF_OK	2233	X'000008B9'
MQRC_CHANNEL_AUTO_DEF_ERROR	2234	X'000008BA'
MQRC_CFH_ERROR	2235	X'000008BB'
MQRC_CFIL_ERROR	2236	X'000008BC'
MQRC_CFIN_ERROR	2237	X'000008BD'
MQRC_CFSL_ERROR	2238	X'000008BE'
MQRC_CFST_ERROR	2239	X'000008BF'
MQRC_INCOMPLETE_GROUP	2241	X'000008C1'
MQRC_INCOMPLETE_MSG	2242	X'000008C2'
MQRC_INCONSISTENT_CCSIDS	2243	X'000008C3'
MQRC_INCONSISTENT_ENCODINGS	2244	X'000008C4'
MQRC_INCONSISTENT_UOW	2245	X'000008C5'
MQRC_INVALID_MSG_UNDER_CURSOR	2246	X'000008C6'
MQRC_MATCH_OPTIONS_ERROR	2247	X'000008C7'
MQRC_MDE_ERROR	2248	X'000008C8'
MQRC_MSG_FLAGS_ERROR	2249	X'000008C9'
MQRC_MSG_SEQ_NUMBER_ERROR	2250	X'000008CA'
MQRC_OFFSET_ERROR	2251	X'000008CB'
MQRC_ORIGINAL_LENGTH_ERROR	2252	X'000008CC'
MQRC_SEGMENT_LENGTH_ZERO	2253	X'000008CD'
MQRC_UOW_NOT_AVAILABLE	2255	X'000008CF'
MQRC_WRONG_GMO_VERSION	2256	X'000008D0'
MQRC_WRONG_MD_VERSION	2257	X'000008D1'
MQRC_GROUP_ID_ERROR	2258	X'000008D2'
MQRC_INCONSISTENT_BROWSE	2259	X'000008D3'
MQRC_XQH_ERROR	2260	X'000008D4'
MQRC_SRC_ENV_ERROR	2261	X'000008D5'
MQRC_SRC_NAME_ERROR	2262	X'000008D6'
MQRC_DEST_ENV_ERROR	2263	X'000008D7'
MQRC_DEST_NAME_ERROR	2264	X'000008D8'

표 304. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRC_TM_ERROR	2265	X'000008D9'
MQRC_CLUSTER_EXIT_ERROR	2266	X'000008DA'
MQRC_CLUSTER_EXIT_LOAD_ERROR	2267	X'000008DB'
MQRC_CLUSTER_PUT_INHIBITED	2268	X'000008DC'
MQRC_CLUSTER_RESOURCE_ERROR	2269	X'000008DD'
MQRC_NO_DESTINATIONS_AVAILABLE	2270	X'000008DE'
MQRC_CONN_TAG_IN_USE	2271	X'000008DF'
MQRC_PARTIALLY_CONVERTED	2272	X'000008E0'
MQRC_CONNECTION_ERROR	2273	X'000008E1'
MQRC_OPTION_ENVIRONMENT_ERROR	2274	X'000008E2'
MQRC_CD_ERROR	2277	X'000008E5'
MQRC_CLIENT_CONN_ERROR	2278	X'000008E6'
MQRC_CHANNEL_STOPPED_BY_USER	2279	X'000008E7'
MQRC_HCONFIG_ERROR	2280	X'000008E8'
MQRC_FUNCTION_ERROR	2281	X'000008E9'
MQRC_CHANNEL_STARTED	2282	X'000008EA'
MQRC_CHANNEL_STOPPED	2283	X'000008EB'
MQRC_CHANNEL_CONV_ERROR	2284	X'000008EC'
MQRC_SERVICE_NOT_AVAILABLE	2285	X'000008ED'
MQRC_INITIALIZATION_FAILED	2286	X'000008EE'
MQRC_TERMINATION_FAILED	2287	X'000008EF'
MQRC_UNKNOWN_Q_NAME	2288	X'000008F0'
MQRC_SERVICE_ERROR	2289	X'000008F1'
MQRC_Q_ALREADY_EXISTS	2290	X'000008F2'
MQRC_USER_ID_NOT_AVAILABLE	2291	X'000008F3'
MQRC_UNKNOWN_ENTITY	2292	X'000008F4'
MQRC_UNKNOWN_AUTH_ENTITY	2293	X'000008F5'
MQRC_UNKNOWN_REF_OBJECT	2294	X'000008F6'
MQRC_CHANNEL_ACTIVATED	2295	X'000008F7'
MQRC_CHANNEL_NOT_ACTIVATED	2296	X'000008F8'
MQRC_UOW_CANCELED	2297	X'000008F9'
MQRC_FUNCTION_NOT_SUPPORTED	2298	X'000008FA'
MQRC_SELECTOR_TYPE_ERROR	2299	X'000008FB'
MQRC_COMMAND_TYPE_ERROR	2300	X'000008FC'
MQRC_MULTIPLE_INSTANCE_ERROR	2301	X'000008FD'
MQRC_SYSTEM_ITEM_NOT_ALTERABLE	2302	X'000008FE'
MQRC_BAG_CONVERSION_ERROR	2303	X'000008FF'
MQRC_SELECTOR_OUT_OF_RANGE	2304	X'00000900'
MQRC_SELECTOR_NOT_UNIQUE	2305	X'00000901'

표 304. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRC_INDEX_NOT_PRESENT	2306	X'00000902'
MQRC_STRING_ERROR	2307	X'00000903'
MQRC_ENCODING_NOT_SUPPORTED	2308	X'00000904'
MQRC_SELECTOR_NOT_PRESENT	2309	X'00000905'
MQRC_OUT_SELECTOR_ERROR	2310	X'00000906'
MQRC_STRING_TRUNCATED	2311	X'00000907'
MQRC_SELECTOR_WRONG_TYPE	2312	X'00000908'
MQRC_INCONSISTENT_ITEM_TYPE	2313	X'00000909'
MQRC_INDEX_ERROR	2314	X'0000090A'
MQRC_SYSTEM_BAG_NOT_ALTERABLE	2315	X'0000090B'
MQRC_ITEM_COUNT_ERROR	2316	X'0000090C'
MQRC_FORMAT_NOT_SUPPORTED	2317	X'0000090D'
MQRC_SELECTOR_NOT_SUPPORTED	2318	X'0000090E'
MQRC_ITEM_VALUE_ERROR	2319	X'0000090F'
MQRC_HBAG_ERROR	2320	X'00000910'
MQRC_PARAMETER_MISSING	2321	X'00000911'
MQRC_CMD_SERVER_NOT_AVAILABLE	2322	X'00000912'
MQRC_STRING_LENGTH_ERROR	2323	X'00000913'
MQRC_INQUIRY_COMMAND_ERROR	2324	X'00000914'
MQRC_NESTED_BAG_NOT_SUPPORTED	2325	X'00000915'
MQRC_BAG_WRONG_TYPE	2326	X'00000916'
MQRC_ITEM_TYPE_ERROR	2327	X'00000917'
MQRC_SYSTEM_BAG_NOT_DELETABLE	2328	X'00000918'
MQRC_SYSTEM_ITEM_NOT_DELETABLE	2329	X'00000919'
MQRC_CODED_CHAR_SET_ID_ERROR	2330	X'0000091A'
MQRC_MSG_TOKEN_ERROR	2331	X'0000091B'
MQRC_MISSING_WIH	2332	X'0000091C'
MQRC_WIH_ERROR	2333	X'0000091D'
MQRC_RFH_ERROR	2334	X'0000091E'
MQRC_RFH_STRING_ERROR	2335	X'0000091F'
MQRC_RFH_COMMAND_ERROR	2336	X'00000920'
MQRC_RFH_PARM_ERROR	2337	X'00000921'
MQRC_RFH_DUPLICATE_PARM	2338	X'00000922'
MQRC_RFH_PARM_MISSING	2339	X'00000923'
MQRC_CHAR_CONVERSION_ERROR	2340	X'00000924'
MQRC_UCS2_CONVERSION_ERROR	2341	X'00000925'
MQRC_DB2_NOT_AVAILABLE	2342	X'00000926'
MQRC_OBJECT_NOT_UNIQUE	2343	X'00000927'
MQRC_CONN_TAG_NOT_RELEASED	2344	X'00000928'

표 304. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRC_CF_NOT_AVAILABLE	2345	X'00000929'
MQRC_CF_STRUC_IN_USE	2346	X'0000092A'
MQRC_CF_STRUC_LIST_HDR_IN_USE	2347	X'0000092B'
MQRC_CF_STRUC_AUTH_FAILED	2348	X'0000092C'
MQRC_CF_STRUC_ERROR	2349	X'0000092D'
MQRC_CONN_TAG_NOT_USABLE	2350	X'0000092E'
MQRC_GLOBAL_UOW_CONFLICT	2351	X'0000092F'
MQRC_LOCAL_UOW_CONFLICT	2352	X'00000930'
MQRC_HANDLE_IN_USE_FOR_UOW	2353	X'00000931'
MQRC_UOW_ENLISTMENT_ERROR	2354	X'00000932'
MQRC_UOW_MIX_NOT_SUPPORTED	2355	X'00000933'
MQRC_WXP_ERROR	2356	X'00000934'
MQRC_CURRENT_RECORD_ERROR	2357	X'00000935'
MQRC_NEXT_OFFSET_ERROR	2358	X'00000936'
MQRC_NO_RECORD_AVAILABLE	2359	X'00000937'
MQRC_OBJECT_LEVEL_INCOMPATIBLE	2360	X'00000938'
MQRC_NEXT_RECORD_ERROR	2361	X'00000939'
MQRC_BACKOUT_THRESHOLD_REACHED	2362	X'0000093A'
MQRC_MSG_NOT_MATCHED	2363	X'0000093B'
MQRC_JMS_FORMAT_ERROR	2364	X'0000093C'
MQRC_SEGMENTS_NOT_SUPPORTED	2365	X'0000093D'
MQRC_WRONG_CF_LEVEL	2366	X'0000093E'
MQRC_CONFIG_CREATE_OBJECT	2367	X'0000093F'
MQRC_CONFIG_CHANGE_OBJECT	2368	X'00000940'
MQRC_CONFIG_DELETE_OBJECT	2369	X'00000941'
MQRC_CONFIG_REFRESH_OBJECT	2370	X'00000942'
MQRC_CHANNEL_SSL_ERROR	2371	X'00000943'
MQRC_PARTICIPANT_NOT_DEFINED	2372	X'00000944'
MQRC_CF_STRUC_FAILED	2373	X'00000945'
MQRC_API_EXIT_ERROR	2374	X'00000946'
MQRC_API_EXIT_INIT_ERROR	2375	X'00000947'
MQRC_API_EXIT_TERM_ERROR	2376	X'00000948'
MQRC_EXIT_REASON_ERROR	2377	X'00000949'
MQRC_RESERVED_VALUE_ERROR	2378	X'0000094A'
MQRC_NO_DATA_AVAILABLE	2379	X'0000094B'
MQRC_SCO_ERROR	2380	X'0000094C'
MQRC_KEY_REPOSITORY_ERROR	2381	X'0000094D'
MQRC_CRYPTO_HARDWARE_ERROR	2382	X'0000094E'
MQRC_AUTH_INFO_REC_COUNT_ERROR	2383	X'0000094F'



표 304. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRC_AUTH_INFO_REC_ERROR	2384	X'00000950'
MQRC_AIR_ERROR	2385	X'00000951'
MQRC_AUTH_INFO_TYPE_ERROR	2386	X'00000952'
MQRC_AUTH_INFO_CONN_NAME_ERROR	2387	X'00000953'
MQRC_LDAP_USER_NAME_ERROR	2388	X'00000954'
MQRC_LDAP_USER_NAME_LENGTH_ERR	2389	X'00000955'
MQRC_LDAP_PASSWORD_ERROR	2390	X'00000956'
MQRC_SSL_ALREADY_INITIALIZED	2391	X'00000957'
MQRC_SSL_CONFIG_ERROR	2392	X'00000958'
MQRC_SSL_INITIALIZATION_ERROR	2393	X'00000959'
MQRC_Q_INDEX_TYPE_ERROR	2394	X'0000095A'
MQRC_CFBS_ERROR	2395	X'0000095B'
MQRC_SSL_NOT_ALLOWED	2396	X'0000095C'
MQRC_JSSE_ERROR	2397	X'0000095D'
MQRC_SSL_PEER_NAME_MISMATCH	2398	X'0000095E'
MQRC_SSL_PEER_NAME_ERROR	2399	X'0000095F'
MQRC_UNSUPPORTED_CIPHER_SUITE	2400	X'00000960'
MQRC_SSL_CERTIFICATE_REVOKED	2401	X'00000961'
MQRC_SSL_CERT_STORE_ERROR	2402	X'00000962'
MQRC_CLIENT_EXIT_LOAD_ERROR	2406	X'00000966'
MQRC_CLIENT_EXIT_ERROR	2407	X'00000967'
MQRC_UOW_COMMITTED	2408	X'00000968'
MQRC_SSL_KEY_RESET_ERROR	2409	X'00000969'
MQRC_UNKNOWN_COMPONENT_NAME	2410	X'0000096A'
MQRC_LOGGER_STATUS	2411	X'0000096B'
MQRC_COMMAND_MQSC	2412	X'0000096C'
MQRC_COMMAND_PCF	2413	X'0000096D'
MQRC_CFIF_ERROR	2414	X'0000096E'
MQRC_CFSF_ERROR	2415	X'0000096F'
MQRC_CFGR_ERROR	2416	X'00000970'
MQRC_MSG_NOT_ALLOWED_IN_GROUP	2417	X'00000971'
MQRC_FILTER_OPERATOR_ERROR	2418	X'00000972'
MQRC_NESTED_SELECTOR_ERROR	2419	X'00000973'
MQRC_EPH_ERROR	2420	X'00000974'
MQRC_RFH_FORMAT_ERROR	2421	X'00000975'
MQRC_CFBF_ERROR	2422	X'00000976'
MQRC_CLIENT_CHANNEL_CONFLICT	2423	X'00000977'
MQRC_SD_ERROR	2424	X'00000978'
MQRC_TOPIC_STRING_ERROR	2425	X'00000979'

표 304. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRC_STS_ERROR	2426	X'0000097A'
MQRC_NO_SUBSCRIPTION	2428	X'0000097C'
MQRC_SUBSCRIPTION_IN_USE	2429	X'0000097D'
MQRC_STAT_TYPE_ERROR	2430	X'0000097E'
MQRC_SUB_USER_DATA_ERROR	2431	X'0000097F'
MQRC_SUB_ALREADY_EXISTS	2432	X'00000980'
MQRC_IDENTITY_MISMATCH	2434	X'00000982'
MQRC_ALTER_SUB_ERROR	2435	X'00000983'
MQRC_DURABILITY_NOT_ALLOWED	2436	X'00000984'
MQRC_NO_RETAINED_MSG	2437	X'00000985'
MQRC_SRO_ERROR	2438	X'00000986'
MQRC_SUB_NAME_ERROR	2440	X'00000988'
MQRC_OBJECT_STRING_ERROR	2441	X'00000989'
MQRC_PROPERTY_NAME_ERROR	2442	X'0000098A'
MQRC_SEGMENTATION_NOT_ALLOWED	2443	X'0000098B'
MQRC_CBD_ERROR	2444	X'0000098C'
MQRC_CTLO_ERROR	2445	X'0000098D'
MQRC_NO_CALLBACKS_ACTIVE	2446	X'0000098E'
MQRC_CALLBACK_NOT_REGISTERED	2448	X'00000990'
MQRC_OPTIONS_CHANGED	2457	X'00000999'
MQRC_READ_AHEAD_MSGS	2458	X'0000099A'
MQRC_SELECTOR_SYNTAX_ERROR	2459	X'0000099B'
MQRC_HMSG_ERROR	2460	X'0000099C'
MQRC_CMHO_ERROR	2461	X'0000099D'
MQRC_DMHO_ERROR	2462	X'0000099E'
MQRC_SMPO_ERROR	2463	X'0000099F'
MQRC_IMPO_ERROR	2464	X'000009A0'
MQRC_PROPERTY_NAME_TOO_BIG	2465	X'000009A1'
MQRC_PROP_VALUE_NOT_CONVERTED	2466	X'000009A2'
MQRC_PROP_TYPE_NOT_SUPPORTED	2467	X'000009A3'
MQRC_PROPERTY_VALUE_TOO_BIG	2469	X'000009A5'
MQRC_PROP_CONV_NOT_SUPPORTED	2470	X'000009A6'
MQRC_PROPERTY_NOT_AVAILABLE	2471	X'000009A7'
MQRC_PROP_NUMBER_FORMAT_ERROR	2472	X'000009A8'
MQRC_PROPERTY_TYPE_ERROR	2473	X'000009A9'
MQRC_PROPERTIES_TOO_BIG	2478	X'000009AE'
MQRC_PUT_NOT_RETAINED	2479	X'000009AF'
MQRC_ALIAS_TARGTYPE_CHANGED	2480	X'000009B0'
MQRC_DMPO_ERROR	2481	X'000009B1'

표 304. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRC_PD_ERROR	2482	X'000009B2'
MQRC_CALLBACK_TYPE_ERROR	2483	X'000009B3'
MQRC_CBD_OPTIONS_ERROR	2484	X'000009B4'
MQRC_MAX_MSG_LENGTH_ERROR	2485	X'000009B5'
MQRC_CALLBACK_ROUTINE_ERROR	2486	X'000009B6'
MQRC_CALLBACK_LINK_ERROR	2487	X'000009B7'
MQRC_OPERATION_ERROR	2488	X'000009B8'
MQRC_BMHO_ERROR	2489	X'000009B9'
MQRC_UNSUPPORTED_PROPERTY	2490	X'000009BA'
MQRC_PROP_NAME_NOT_CONVERTED	2492	X'000009BC'
MQRC_GET_ENABLED	2494	X'000009BE'
MQRC_MODULE_NOT_FOUND	2495	X'000009BF'
MQRC_MODULE_INVALID	2496	X'000009C0'
MQRC_MODULE_ENTRY_NOT_FOUND	2497	X'000009C1'
MQRC_MIXED_CONTENT_NOT_ALLOWED	2498	X'000009C2'
MQRC_MSG_HANDLE_IN_USE	2499	X'000009C3'
MQRC_HCONN_ASYNC_ACTIVE	2500	X'000009C4'
MQRC_MHBO_ERROR	2501	X'000009C5'
MQRC_PUBLICATION_FAILURE	2502	X'000009C6'
MQRC_SUB_INHIBITED	2503	X'000009C7'
MQRC_SELECTOR_ALWAYS_FALSE	2504	X'000009C8'
MQRC_XEPO_ERROR	2507	X'000009CB'
MQRC_DURABILITY_NOT_ALTERABLE	2509	X'000009CD'
MQRC_TOPIC_NOT_ALTERABLE	2510	X'000009CE'
MQRC_SUBLEVEL_NOT_ALTERABLE	2512	X'000009D0'
MQRC_PROPERTY_NAME_LENGTH_ERR	2513	X'000009D1'
MQRC_DUPLICATE_GROUP_SUB	2514	X'000009D2'
MQRC_GROUPING_NOT_ALTERABLE	2515	X'000009D3'
MQRC_SELECTOR_INVALID_FOR_TYPE	2516	X'000009D4'
MQRC_HOBJ QUIESCED	2517	X'000009D5'
MQRC_HOBJ QUIESCED_NO_MSGS	2518	X'000009D6'
MQRC_SELECTION_STRING_ERROR	2519	X'000009D7'
MQRC_RES_OBJECT_STRING_ERROR	2520	X'000009D8'
MQRC_CONNECTION_SUSPENDED	2521	X'000009D9'
MQRC_INVALID_DESTINATION	2522	X'000009DA'
MQRC_INVALID_SUBSCRIPTION	2523	X'000009DB'
MQRC_SELECTOR_NOT_ALTERABLE	2524	X'000009DC'
MQRC_RETAINED_MSG_Q_ERROR	2525	X'000009DD'
MQRC_RETAINED_NOT_DELIVERED	2526	X'000009DE'

표 304. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRC_RFH_RESTRICTED_FORMAT_ERR	2527	X'000009DF'
MQRC_CONNECTION_STOPPED	2528	X'000009E0'
MQRC_ASYNC_UOW_CONFLICT	2529	X'000009E1'
MQRC_ASYNC_XA_CONFLICT	2530	X'000009E2'
MQRC_PUBSUB_INHIBITED	2531	X'000009E3'
MQRC_MSG_HANDLE_COPY_FAILURE	2532	X'000009E4'
MQRC_DEST_CLASS_NOT_ALTERABLE	2533	X'000009E5'
MQRC_OPERATION_NOT_ALLOWED	2534	X'000009E6'
MQRC_ACTION_ERROR	2535	X'000009E7'
MQRC_CHANNEL_NOT_AVAILABLE	2537	X'000009E9'
MQRC_HOST_NOT_AVAILABLE	2538	X'000009EA'
MQRC_CHANNEL_CONFIG_ERROR	2539	X'000009EB'
MQRC_UNKNOWN_CHANNEL_NAME	2540	X'000009EC'
MQRC_LOOPING_PUBLICATION	2541	X'000009ED'
MQRC_ALREADY_JOINED	2542	X'000009EE'
MQRC_CHANNEL_SSL_WARNING	2552	X'000009F8'
MQRC_OCSP_URL_ERROR	2553	X'000009F9'
MQRC_CIPHER_SPEC_NOT_SUITE_B	2591	X'00000A1F'
MQRC_SUITE_B_ERROR	2592	X'00000A20'
MQRC_PASSWORD_PROTECTION_ERROR	2594	X'00000A22'
MQRC_REOPEN_EXCL_INPUT_ERROR	6100	X'000017D4'
MQRC_REOPEN_INQUIRE_ERROR	6101	X'000017D5'
MQRC_REOPEN_SAVED_CONTEXT_ERR	6102	X'000017D6'
MQRC_REOPEN_TEMPORARY_Q_ERROR	6103	X'000017D7'
MQRC_ATTRIBUTE_LOCKED	6104	X'000017D8'
MQRC_CURSOR_NOT_VALID	6105	X'000017D9'
MQRC_ENCODING_ERROR	6106	X'000017DA'
MQRC_STRUC_ID_ERROR	6107	X'000017DB'
MQRC_NULL_POINTER	6108	X'000017DC'
MQRC_NO_CONNECTION_REFERENCE	6109	X'000017DD'
MQRC_NO_BUFFER	6110	X'000017DE'
MQRC_BINARY_DATA_LENGTH_ERROR	6111	X'000017DF'
MQRC_BUFFER_NOT_AUTOMATIC	6112	X'000017E0'
MQRC_INSUFFICIENT_BUFFER	6113	X'000017E1'
MQRC_INSUFFICIENT_DATA	6114	X'000017E2'
MQRC_DATA_TRUNCATED	6115	X'000017E3'
MQRC_ZERO_LENGTH	6116	X'000017E4'
MQRC_NEGATIVE_LENGTH	6117	X'000017E5'
MQRC_NEGATIVE_OFFSET	6118	X'000017E6'

표 304. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRC_INCONSISTENT_FORMAT	6119	X'000017E7'
MQRC_INCONSISTENT_OBJECT_STATE	6120	X'000017E8'
MQRC_CONTEXT_OBJECT_NOT_VALID	6121	X'000017E9'
MQRC_CONTEXT_OPEN_ERROR	6122	X'000017EA'
MQRC_STRUC_LENGTH_ERROR	6123	X'000017EB'
MQRC_NOT_CONNECTED	6124	X'000017EC'
MQRC_NOT_OPEN	6125	X'000017ED'
MQRC_DISTRIBUTION_LIST_EMPTY	6126	X'000017EE'
MQRC_INCONSISTENT_OPEN_OPTIONS	6127	X'000017EF'
MQRC_WRONG_VERSION	6128	X'000017F0'
MQRC_REFERENCE_ERROR	6129	X'000017F1'

### **MQRCCF\_\*(명령 형식 헤더 이유 코드)**

프로그래머 응답에 대한 자세한 정보는 [PCF 이유 코드](#)를 참조하십시오.

표 305. 상수 값		
이름	10진수 값	16진수 값
MQRCCF_CFH_TYPE_ERROR	3001	X'00000BB9'
MQRCCF_CFH_LENGTH_ERROR	3002	X'00000BBA'
MQRCCF_CFH_VERSION_ERROR	3003	X'00000BBB'
MQRCCF_CFH_MSG_SEQ_NUMBER_ERR	3004	X'00000BBC'
MQRCCF_CFH_CONTROL_ERROR	3005	X'00000BBD'
MQRCCF_CFH_PARM_COUNT_ERROR	3006	X'00000BBE'
MQRCCF_CFH_COMMAND_ERROR	3007	X'00000BBF'
MQRCCF_COMMAND_FAILED	3008	X'00000BC0'
MQRCCF_CFIN_LENGTH_ERROR	3009	X'00000BC1'
MQRCCF_CFST_LENGTH_ERROR	3010	X'00000BC2'
MQRCCF_CFST_STRING_LENGTH_ERR	3011	X'00000BC3'
MQRCCF_FORCE_VALUE_ERROR	3012	X'00000BC4'
MQRCCF_STRUCTURE_TYPE_ERROR	3013	X'00000BC5'
MQRCCF_CFIN_PARM_ID_ERROR	3014	X'00000BC6'
MQRCCF_CFST_PARM_ID_ERROR	3015	X'00000BC7'
MQRCCF_MSG_LENGTH_ERROR	3016	X'00000BC8'
MQRCCF_CFIN_DUPLICATE_PARM	3017	X'00000BC9'
MQRCCF_CFST_DUPLICATE_PARM	3018	X'00000BCA'
MQRCCF_PARM_COUNT_TOO_SMALL	3019	X'00000BCB'
MQRCCF_PARM_COUNT_TOO_BIG	3020	X'00000BCC'
MQRCCF_Q_ALREADY_IN_CELL	3021	X'00000BCD'
MQRCCF_Q_TYPE_ERROR	3022	X'00000BCE'
MQRCCF_MD_FORMAT_ERROR	3023	X'00000BCF'

표 305. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRCCF_CFSL_LENGTH_ERROR	3024	X'00000BD0'
MQRCCF_REPLACE_VALUE_ERROR	3025	X'00000BD1'
MQRCCF_CFIL_DUPLICATE_VALUE	3026	X'00000BD2'
MQRCCF_CFIL_COUNT_ERROR	3027	X'00000BD3'
MQRCCF_CFIL_LENGTH_ERROR	3028	X'00000BD4'
MQRCCF_QUIESCE_VALUE_ERROR	3029	X'00000BD5'
MQRCCF_MODE_VALUE_ERROR	3029	X'00000BD5'
MQRCCF_MSG_SEQ_NUMBER_ERROR	3030	X'00000BD6'
MQRCCF_PING_DATA_COUNT_ERROR	3031	X'00000BD7'
MQRCCF_PING_DATA_COMPARE_ERROR	3032	X'00000BD8'
MQRCCF_CFSL_PARM_ID_ERROR	3033	X'00000BD9'
MQRCCF_CHANNEL_TYPE_ERROR	3034	X'00000BDA'
MQRCCF_PARM_SEQUENCE_ERROR	3035	X'00000BDB'
MQRCCF_XMIT_PROTOCOL_TYPE_ERR	3036	X'00000BDC'
MQRCCF_BATCH_SIZE_ERROR	3037	X'00000BDD'
MQRCCF_DISC_INT_ERROR	3038	X'00000BDE'
MQRCCF_SHORT_RETRY_ERROR	3039	X'00000BDF'
MQRCCF_SHORT_TIMER_ERROR	3040	X'00000BE0'
MQRCCF_LONG_RETRY_ERROR	3041	X'00000BE1'
MQRCCF_LONG_TIMER_ERROR	3042	X'00000BE2'
MQRCCF_SEQ_NUMBER_WRAP_ERROR	3043	X'00000BE3'
MQRCCF_MAX_MSG_LENGTH_ERROR	3044	X'00000BE4'
MQRCCF_PUT_AUTH_ERROR	3045	X'00000BE5'
MQRCCF_PURGE_VALUE_ERROR	3046	X'00000BE6'
MQRCCF_CFIL_PARM_ID_ERROR	3047	X'00000BE7'
MQRCCF_MSG_TRUNCATED	3048	X'00000BE8'
MQRCCF_CCsid_ERROR	3049	X'00000BE9'
MQRCCF_ENCODING_ERROR	3050	X'00000BEA'
MQRCCF_QUEUES_VALUE_ERROR	3051	X'00000BEB'
MQRCCF_DATA_CONV_VALUE_ERROR	3052	X'00000BEC'
MQRCCF_INDOUBT_VALUE_ERROR	3053	X'00000BED'
MQRCCF_ESCAPE_TYPE_ERROR	3054	X'00000BEE'
MQRCCF_REPOS_VALUE_ERROR	3055	X'00000BEF'
MQRCCF_CHANNEL_TABLE_ERROR	3062	X'00000BF6'
MQRCCF_MCA_TYPE_ERROR	3063	X'00000BF7'
MQRCCF_CHL_INST_TYPE_ERROR	3064	X'00000BF8'
MQRCCF_CHL_STATUS_NOT_FOUND	3065	X'00000BF9'
MQRCCF_CFSL_DUPLICATE_PARM	3066	X'00000BFA'
MQRCCF_CFSL_TOTAL_LENGTH_ERROR	3067	X'00000BFB'

표 305. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRCCF_CFSL_COUNT_ERROR	3068	X'00000BFC'
MQRCCF_CFSL_STRING_LENGTH_ERR	3069	X'00000BFD'
MQRCCF_BROKER_DELETED	3070	X'00000BFE'
MQRCCF_STREAM_ERROR	3071	X'00000BFF'
MQRCCF_TOPIC_ERROR	3072	X'00000C00'
MQRCCF_NOT_REGISTERED	3073	X'00000C01'
MQRCCF_Q_MGR_NAME_ERROR	3074	X'00000C02'
MQRCCF_INCORRECT_STREAM	3075	X'00000C03'
MQRCCF_Q_NAME_ERROR	3076	X'00000C04'
MQRCCF_NO_RETAINED_MSG	3077	X'00000C05'
MQRCCF_DUPLICATE_IDENTITY	3078	X'00000C06'
MQRCCF_INCORRECT_Q	3079	X'00000C07'
MQRCCF_CORREL_ID_ERROR	3080	X'00000C08'
MQRCCF_NOT_AUTHORIZED	3081	X'00000C09'
MQRCCF_UNKNOWN_STREAM	3082	X'00000C0A'
MQRCCF_REG_OPTIONS_ERROR	3083	X'00000C0B'
MQRCCF_PUB_OPTIONS_ERROR	3084	X'00000C0C'
MQRCCF_UNKNOWN_BROKER	3085	X'00000C0D'
MQRCCF_Q_MGR_CCSID_ERROR	3086	X'00000C0E'
MQRCCF_DEL_OPTIONS_ERROR	3087	X'00000C0F'
MQRCCF_CLUSTER_NAME_CONFLICT	3088	X'00000C10'
MQRCCF_REPOS_NAME_CONFLICT	3089	X'00000C11'
MQRCCF_CLUSTER_Q_USAGE_ERROR	3090	X'00000C12'
MQRCCF_ACTION_VALUE_ERROR	3091	X'00000C13'
MQRCCF_COMMS_LIBRARY_ERROR	3092	X'00000C14'
MQRCCF_NETBIOS_NAME_ERROR	3093	X'00000C15'
MQRCCF_BROKER_COMMAND_FAILED	3094	X'00000C16'
MQRCCF_CFST_CONFLICTING_PARM	3095	X'00000C17'
MQRCCF_PATH_NOT_VALID	3096	X'00000C18'
MQRCCF_PARM_SYNTAX_ERROR	3097	X'00000C19'
MQRCCF_PWD_LENGTH_ERROR	3098	X'00000C1A'
MQRCCF_FILTER_ERROR	3150	X'00000C4E'
MQRCCF_WRONG_USER	3151	X'00000C4F'
MQRCCF_DUPLICATE_SUBSCRIPTION	3152	X'00000C50'
MQRCCF_SUB_NAME_ERROR	3153	X'00000C51'
MQRCCF_SUB_IDENTITY_ERROR	3154	X'00000C52'
MQRCCF_SUBSCRIPTION_IN_USE	3155	X'00000C53'
MQRCCF_SUBSCRIPTION_LOCKED	3156	X'00000C54'
MQRCCF_ALREADY_JOINED	3157	X'00000C55'

표 305. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRCCF_OBJECT_IN_USE	3160	X'00000C58'
MQRCCF_UNKNOWN_FILE_NAME	3161	X'00000C59'
MQRCCF_FILE_NOT_AVAILABLE	3162	X'00000C5A'
MQRCCF_DISC_RETRY_ERROR	3163	X'00000C5B'
MQRCCF_ALLOC_RETRY_ERROR	3164	X'00000C5C'
MQRCCF_ALLOC_SLOW_TIMER_ERROR	3165	X'00000C5D'
MQRCCF_ALLOC_FAST_TIMER_ERROR	3166	X'00000C5E'
MQRCCF_PORT_NUMBER_ERROR	3167	X'00000C5F'
MQRCCF_CHL_SYSTEM_NOT_ACTIVE	3168	X'00000C60'
MQRCCF_ENTITY_NAME_MISSING	3169	X'00000C61'
MQRCCF_PROFILE_NAME_ERROR	3170	X'00000C62'
MQRCCF_AUTH_VALUE_ERROR	3171	X'00000C63'
MQRCCF_AUTH_VALUE_MISSING	3172	X'00000C64'
MQRCCF_OBJECT_TYPE_MISSING	3173	X'00000C65'
MQRCCF_CONNECTION_ID_ERROR	3174	X'00000C66'
MQRCCF_LOG_TYPE_ERROR	3175	X'00000C67'
MQRCCF_PROGRAM_NOT_AVAILABLE	3176	X'00000C68'
MQRCCF_PROGRAM_AUTH_FAILED	3177	X'00000C69'
MQRCCF_NONE_FOUND	3200	X'00000C80'
MQRCCF_SECURITY_SWITCH_OFF	3201	X'00000C81'
MQRCCF_SECURITY_REFRESH_FAILED	3202	X'00000C82'
MQRCCF_PARM_CONFLICT	3203	X'00000C83'
MQRCCF_COMMAND_INHIBITED	3204	X'00000C84'
MQRCCF_OBJECT_BEING_DELETED	3205	X'00000C85'
MQRCCF_STORAGE_CLASS_IN_USE	3207	X'00000C87'
MQRCCF_OBJECT_NAME_RESTRICTED	3208	X'00000C88'
MQRCCF_OBJECT_LIMIT_EXCEEDED	3209	X'00000C89'
MQRCCF_OBJECT_OPEN_FORCE	3210	X'00000C8A'
MQRCCF_DISPOSITION_CONFLICT	3211	X'00000C8B'
MQRCCF_Q_MGR_NOT_IN_QSG	3212	X'00000C8C'
MQRCCF_ATTR_VALUE_FIXED	3213	X'00000C8D'
MQRCCF_NAMELIST_ERROR	3215	X'00000C8F'
MQRCCF_NO_CHANNEL_INITIATOR	3217	X'00000C91'
MQRCCF_CHANNEL_INITIATOR_ERROR	3218	X'00000C92'
MQRCCF_COMMAND_LEVEL_CONFLICT	3222	X'00000C96'
MQRCCF_Q_ATTR_CONFLICT	3223	X'00000C97'
MQRCCF_EVENTS_DISABLED	3224	X'00000C98'
MQRCCF_COMMAND_SCOPE_ERROR	3225	X'00000C99'
MQRCCF_COMMAND_REPLY_ERROR	3226	X'00000C9A'




표 305. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRCCF_FUNCTION_RESTRICTED	3227	X'00000C9B'
MQRCCF_PARM_MISSING	3228	X'00000C9C'
MQRCCF_PARM_VALUE_ERROR	3229	X'00000C9D'
MQRCCF_COMMAND_LENGTH_ERROR	3230	X'00000C9E'
MQRCCF_COMMAND_ORIGIN_ERROR	3231	X'00000C9F'
MQRCCF_LISTENER_CONFLICT	3232	X'00000CA0'
MQRCCF_LISTENER_STARTED	3233	X'00000CA1'
MQRCCF_LISTENER_STOPPED	3234	X'00000CA2'
MQRCCF_CHANNEL_ERROR	3235	X'00000CA3'
MQRCCF_CF_STRUC_ERROR	3236	X'00000CA4'
MQRCCF_UNKNOWN_USER_ID	3237	X'00000CA5'
MQRCCF_UNEXPECTED_ERROR	3238	X'00000CA6'
MQRCCF_NO_XCF_PARTNER	3239	X'00000CA7'
MQRCCF_CFGR_PARM_ID_ERROR	3240	X'00000CA8'
MQRCCF_CFIF_LENGTH_ERROR	3241	X'00000CA9'
MQRCCF_CFIF_OPERATOR_ERROR	3242	X'00000CAA'
MQRCCF_CFIF_PARM_ID_ERROR	3243	X'00000CAB'
MQRCCF_CFSF_FILTER_VAL_LEN_ERR	3244	X'00000CAC'
MQRCCF_CFSF_LENGTH_ERROR	3245	X'00000CAD'
MQRCCF_CFSF_OPERATOR_ERROR	3246	X'00000CAE'
MQRCCF_CFSF_PARM_ID_ERROR	3247	X'00000CAF'
MQRCCF_TOO_MANY_FILTERS	3248	X'00000CB0'
MQRCCF_LISTENER_RUNNING	3249	X'00000CB1'
MQRCCF_LSTR_STATUS_NOT_FOUND	3250	X'00000CB2'
MQRCCF_SERVICE_RUNNING	3251	X'00000CB3'
MQRCCF_SERV_STATUS_NOT_FOUND	3252	X'00000CB4'
MQRCCF_SERVICE_STOPPED	3253	X'00000CB5'
MQRCCF_CFBS_DUPLICATE_PARM	3254	X'00000CB6'
MQRCCF_CFBS_LENGTH_ERROR	3255	X'00000CB7'
MQRCCF_CFBS_PARM_ID_ERROR	3256	X'00000CB8'
MQRCCF_CFBS_STRING_LENGTH_ERR	3257	X'00000CB9'
MQRCCF_CFGR_LENGTH_ERROR	3258	X'00000CBA'
MQRCCF_CFGR_PARM_COUNT_ERROR	3259	X'00000CBB'
MQRCCF_CONN_NOT_STOPPED	3260	X'00000CBC'
MQRCCF_SERVICE_REQUEST_PENDING	3261	X'00000CBD'
MQRCCF_NO_START_CMD	3262	X'00000CBE'
MQRCCF_NO_STOP_CMD	3263	X'00000CBF'
MQRCCF_CFBF_LENGTH_ERROR	3264	X'00000CC0'
MQRCCF_CFBF_PARM_ID_ERROR	3265	X'00000CC1'

표 305. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRCCF_CFBF_OPERATOR_ERROR	3266	X'00000CC2'
MQRCCF_CFBF_FILTER_VAL_LEN_ERR	3267	X'00000CC3'
MQRCCF_LISTENER_STILL_ACTIVE	3268	X'00000CC4'
MQRCCF_DEF_XMIT_Q_CLUS_ERROR	3269	X'00000CC5'
MQRCCF_TOPICSTR_ALREADY_EXISTS	3300	X'00000CE4'
MQRCCF_SHARING_CONVS_ERROR	3301	X'00000CE5'
MQRCCF_SHARING_CONVS_TYPE	3302	X'00000CE6'
MQRCCF_SECURITY_CASE_CONFLICT	3303	X'00000CE7'
MQRCCF_TOPIC_TYPE_ERROR	3305	X'00000CE9'
MQRCCF_MAX_INSTANCES_ERROR	3306	X'00000CEA'
MQRCCF_MAX_INSTS_PER_CLNT_ERR	3307	X'00000CEB'
MQRCCF_TOPIC_STRING_NOT_FOUND	3308	X'00000CEC'
MQRCCF_SUBSCRIPTION_POINT_ERR	3309	X'00000CED'
MQRCCF_SUB_ALREADY_EXISTS	3311	X'00000CEF'
MQRCCF_UNKNOWN_OBJECT_NAME	3312	X'00000CF0'
MQRCCF_REMOTE_Q_NAME_ERROR	3313	X'00000CF1'
MQRCCF_DURABILITY_NOT_ALLOWED	3314	X'00000CF2'
MQRCCF_HOBJ_ERROR	3315	X'00000CF3'
MQRCCF_DEST_NAME_ERROR	3316	X'00000CF4'
MQRCCF_INVALID_DESTINATION	3317	X'00000CF5'
MQRCCF_PUBSUB_INHIBITED	3318	X'00000CF6'
MQRCCF_CHLAUTH_TYPE_ERROR	3326	X'00000CFE'
MQRCCF_CHLAUTH_ACTION_ERROR	3327	X'00000CFF'
MQRCCF_CHLAUTH_USERSRC_ERROR	3335	X'00000D07'
MQRCCF_WRONG_CHLAUTH_TYPE	3336	X'00000D08'
MQRCCF_CHLAUTH_ALREADY_EXISTS	3337	X'00000D09'
MQRCCF_CHLAUTH_NOT_FOUND	3338	X'00000D0A'
MQRCCF_WRONG_CHLAUTH_ACTION	3339	X'00000D0B'
MQRCCF_WRONG_CHLAUTH_USERSRC	3340	X'00000D0C'
MQRCCF_CHLAUTH_WARN_ERROR	3341	X'00000D0D'
MQRCCF_WRONG_CHLAUTH_MATCH	3342	X'00000D0E'
MQRCCF_IPADDR_RANGE_CONFLICT	3343	X'00000D0F'
MQRCCF_CHLAUTH_MAX_EXCEEDED	3344	X'00000D10'
MQRCCF_IPADDR_ERROR	3345	X'00000D11'
MQRCCF_IPADDR_RANGE_ERROR	3346	X'00000D12'
MQRCCF_PROFILE_NAME_MISSING	3347	X'00000D13'
MQRCCF_CHLAUTH_CLNTUSER_ERROR	3348	X'00000D14'
MQRCCF_CHLAUTH_NAME_ERROR	3349	X'00000D15'
MQRCCF_SUITE_B_ERROR	3353	X'00000D19'

표 305. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRCCF_PSCLUS_DISABLED_TOPDEF	3359	X'00000D1F'
MQRCCF_PSCLUS_TOPIC_EXISTS	3360	X'00000D20'
MQRCCF_INVALID_PROTOCOL	3365	X'00000D25'
MQRCCF_ACCESS_BLOCKED	3382	X'00000D36'
MQRCCF_OBJECT_ALREADY_EXISTS	4001	X'00000FA1'
MQRCCF_OBJECT_WRONG_TYPE	4002	X'00000FA2'
MQRCCF_LIKE_OBJECT_WRONG_TYPE	4003	X'00000FA3'
MQRCCF_OBJECT_OPEN	4004	X'00000FA4'
MQRCCF_ATTR_VALUE_ERROR	4005	X'00000FA5'
MQRCCF_UNKNOWN_Q_MGR	4006	X'00000FA6'
MQRCCF_Q_WRONG_TYPE	4007	X'00000FA7'
MQRCCF_OBJECT_NAME_ERROR	4008	X'00000FA8'
MQRCCF_ALLOCATE_FAILED	4009	X'00000FA9'
MQRCCF_HOST_NOT_AVAILABLE	4010	X'00000FAA'
MQRCCF_CONFIGURATION_ERROR	4011	X'00000FAB'
MQRCCF_CONNECTION_REFUSED	4012	X'00000FAC'
MQRCCF_ENTRY_ERROR	4013	X'00000FAD'
MQRCCF_SEND_FAILED	4014	X'00000FAE'
MQRCCF_RECEIVED_DATA_ERROR	4015	X'00000FAF'
MQRCCF_RECEIVE_FAILED	4016	X'00000FB0'
MQRCCF_CONNECTION_CLOSED	4017	X'00000FB1'
MQRCCF_NO_STORAGE	4018	X'00000FB2'
MQRCCF_NO_COMMS_MANAGER	4019	X'00000FB3'
MQRCCF_LISTENER_NOT_STARTED	4020	X'00000FB4'
MQRCCF_BIND_FAILED	4024	X'00000FB8'
MQRCCF_CHANNEL_INDOUBT	4025	X'00000FB9'
MQRCCF_MQCONN_FAILED	4026	X'00000FBA'
MQRCCF_MQOPEN_FAILED	4027	X'00000FBB'
MQRCCF_MQGET_FAILED	4028	X'00000FBC'
MQRCCF_MQPUT_FAILED	4029	X'00000FBD'
MQRCCF_PING_ERROR	4030	X'00000FBE'
MQRCCF_CHANNEL_IN_USE	4031	X'00000FBF'
MQRCCF_CHANNEL_NOT_FOUND	4032	X'00000FC0'
MQRCCF_UNKNOWN_REMOTE_CHANNEL	4033	X'00000FC1'
MQRCCF_REMOTE_QM_UNAVAILABLE	4034	X'00000FC2'
MQRCCF_REMOTE_QM_TERMINATING	4035	X'00000FC3'
MQRCCF_MQINQ_FAILED	4036	X'00000FC4'
MQRCCF_NOT_XMIT_Q	4037	X'00000FC5'
MQRCCF_CHANNEL_DISABLED	4038	X'00000FC6'

표 305. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRCCF_USER_EXIT_NOT_AVAILABLE	4039	X'00000FC7'
MQRCCF_COMMIT_FAILED	4040	X'00000FC8'
MQRCCF_WRONG_CHANNEL_TYPE	4041	X'00000FC9'
MQRCCF_CHANNEL_ALREADY_EXISTS	4042	X'00000FCA'
MQRCCF_DATA_TOO_LARGE	4043	X'00000FCB'
MQRCCF_CHANNEL_NAME_ERROR	4044	X'00000FCC'
MQRCCF_XMIT_Q_NAME_ERROR	4045	X'00000FCD'
MQRCCF_MCA_NAME_ERROR	4047	X'00000FCF'
MQRCCF_SEND_EXIT_NAME_ERROR	4048	X'00000FD0'
MQRCCF_SEC_EXIT_NAME_ERROR	4049	X'00000FD1'
MQRCCF_MSG_EXIT_NAME_ERROR	4050	X'00000FD2'
MQRCCF_RCV_EXIT_NAME_ERROR	4051	X'00000FD3'
MQRCCF_XMIT_Q_NAME_WRONG_TYPE	4052	X'00000FD4'
MQRCCF_MCA_NAME_WRONG_TYPE	4053	X'00000FD5'
MQRCCF_DISC_INT_WRONG_TYPE	4054	X'00000FD6'
MQRCCF_SHORT_RETRY_WRONG_TYPE	4055	X'00000FD7'
MQRCCF_SHORT_TIMER_WRONG_TYPE	4056	X'00000FD8'
MQRCCF_LONG_RETRY_WRONG_TYPE	4057	X'00000FD9'
MQRCCF_LONG_TIMER_WRONG_TYPE	4058	X'00000FDA'
MQRCCF_PUT_AUTH_WRONG_TYPE	4059	X'00000FDB'
MQRCCF_KEEP_ALIVE_INT_ERROR	4060	X'00000FDC'
MQRCCF_MISSING_CONN_NAME	4061	X'00000FDD'
MQRCCF_CONN_NAME_ERROR	4062	X'00000FDE'
MQRCCF_MQSET_FAILED	4063	X'00000FDF'
MQRCCF_CHANNEL_NOT_ACTIVE	4064	X'00000FE0'
MQRCCF_TERMINATED_BY_SEC_EXIT	4065	X'00000FE1'
MQRCCF_DYNAMIC_Q_SCOPE_ERROR	4067	X'00000FE3'
MQRCCF_CELL_DIR_NOT_AVAILABLE	4068	X'00000FE4'
MQRCCF_MR_COUNT_ERROR	4069	X'00000FE5'
MQRCCF_MR_COUNT_WRONG_TYPE	4070	X'00000FE6'
MQRCCF_MR_EXIT_NAME_ERROR	4071	X'00000FE7'
MQRCCF_MR_EXIT_NAME_WRONG_TYPE	4072	X'00000FE8'
MQRCCF_MR_INTERVAL_ERROR	4073	X'00000FE9'
MQRCCF_MR_INTERVAL_WRONG_TYPE	4074	X'00000FEA'
MQRCCF_NPM_SPEED_ERROR	4075	X'00000FEB'
MQRCCF_NPM_SPEED_WRONG_TYPE	4076	X'00000FEC'
MQRCCF_HB_INTERVAL_ERROR	4077	X'00000FED'
MQRCCF_HB_INTERVAL_WRONG_TYPE	4078	X'00000FEE'
MQRCCF_CHAD_ERROR	4079	X'00000FEF'

표 305. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRCCF_CHAD_WRONG_TYPE	4080	X'00000FF0'
MQRCCF_CHAD_EVENT_ERROR	4081	X'00000FF1'
MQRCCF_CHAD_EVENT_WRONG_TYPE	4082	X'00000FF2'
MQRCCF_CHAD_EXIT_ERROR	4083	X'00000FF3'
MQRCCF_CHAD_EXIT_WRONG_TYPE	4084	X'00000FF4'
MQRCCF_SUPPRESSED_BY_EXIT	4085	X'00000FF5'
MQRCCF_BATCH_INT_ERROR	4086	X'00000FF6'
MQRCCF_BATCH_INT_WRONG_TYPE	4087	X'00000FF7'
MQRCCF_NET_PRIORITY_ERROR	4088	X'00000FF8'
MQRCCF_NET_PRIORITY_WRONG_TYPE	4089	X'00000FF9'
MQRCCF_CHANNEL_CLOSED	4090	X'00000FFA'
MQRCCF_Q_STATUS_NOT_FOUND	4091	X'00000FFB'
MQRCCF_SSL_CIPHER_SPEC_ERROR	4092	X'00000FFC'
MQRCCF_SSL_PEER_NAME_ERROR	4093	X'00000FFD'
MQRCCF_SSL_CLIENT_AUTH_ERROR	4094	X'00000FFE'
MQRCCF_RETAINED_NOT_SUPPORTED	4095	X'00000FFF'
 MQRCCF_KWD_VALUE_WRONG_TYPE	4096	X'00001000'

### MQR CN\_\*(클라이언트 재연결 상수)

표 306. 상수 값		
이름	10진수 값	16진수 값
MQR CN_NO	0	X'00000000'
MQR CN_YES	1	X'00000001'
MQR CN_Q_MGR	2	X'00000002'
MQR CN_DISABLED	3	X'00000003'

### MQR CVTIME\_\*(수신 제한시간 유형)

표 307. 상수 값		
이름	10진수 값	16진수 값
MQR CVTIME_MULTIPLY	0	X'00000000'
MQR CVTIME_ADD	1	X'00000001'
MQR CVTIME_EQUAL	2	X'00000002'

### MQR READA\_\*(미리 읽기 값)

표 308. 상수 값		
이름	10진수 값	16진수 값
MQR READA_NO	0	X'00000000'
MQR READA_YES	1	X'00000001'
MQR READA_DISABLED	2	X'00000002'

표 308. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQREADA_INHIBITED	3	X'00000003'
MQREADA_BACKLOG	4	X'00000004'

### MQRECORDING\_\*(레코딩 옵션)

표 309. 상수 값		
이름	10진수 값	16진수 값
MQRECORDING_DISABLED	0	X'00000000'
MQRECORDING_Q	1	X'00000001'
MQRECORDING_MSG	2	X'00000002'

### MQREGO\_\*(발행/구독 등록 옵션)

표 310. 상수 값		
이름	10진수 값	16진수 값
MQREGO_NONE	0	X'00000000'
MQREGO_CORREL_ID_AS_IDENTITY	1	X'00000001'
MQREGO_ANONYMOUS	2	X'00000002'
MQREGO_LOCAL	4	X'00000004'
MQREGO_DIRECT_REQUESTS	8	X'00000008'
MQREGO_NEW_PUBLICATIONS_ONLY	16	X'00000010'
MQREGO_PUBLISH_ON_REQUEST_ONLY	32	X'00000020'
MQREGO_DEREGISTER_ALL	64	X'00000040'
MQREGO_INCLUDE_STREAM_NAME	128	X'00000080'
MQREGO_INFORM_IF_RETAINED	256	X'00000100'
MQREGO_DUPLICATES_OK	512	X'00000200'
MQREGO_NON_PERSISTENT	1024	X'00000400'
MQREGO_PERSISTENT	2048	X'00000800'
MQREGO_PERSISTENT_AS_PUBLISH	4096	X'00001000'
MQREGO_PERSISTENT_AS_Q	8192	X'00002000'
MQREGO_ADD_NAME	16384	X'00004000'
MQREGO_NO_ALTERATION	32768	X'00008000'
MQREGO_FULL_RESPONSE	65536	X'00010000'
MQREGO_JOIN_SHARED	131072	X'00020000'
MQREGO_JOIN_EXCLUSIVE	262144	X'00040000'
MQREGO_LEAVE_ONLY	524288	X'00080000'
MQREGO_VARIABLE_USER_ID	1048576	X'00100000'
MQREGO_LOCKED	2097152	X'00200000'

## MQRFH\_\*(규칙 및 형식화 헤더 구조 및 플래그)

### 규칙 및 형식화 헤더 구조

표 311. 상수 구조	
이름	구조
MQRFH_STRUC_ID	"RFH↵"
MQRFH_STRUC_ID_ARRAY	'R', 'F', 'H', '↵'

참고: ↵ 기호는 단일 공백 문자를 나타냅니다.

표 312. 상수 값		
이름	10진수 값	16진수 값
MQRFH_VERSION_1	1	X'00000001'
MQRFH_VERSION_2	2	X'00000002'
MQRFH_STRUC_LENGTH_FIXED	32	X'00000020'
MQRFH_STRUC_LENGTH_FIXED_2	36	X'00000024'

### 규칙 및 형식화 헤더 플래그

표 313. 상수 값		
이름	10진수 값	16진수 값
MQRFH_NONE	0	X'00000000'
MQRFH_NO_FLAGS	0	X'00000000'

## MQRFH2\_\*(발행/구독 옵션 태그 RFH2 최상위 레벨 폴더 태그)

표 314. 상수 값		
이름	10진수 값	16진수 값
MQRFH2_NAME_VALUE_VERSION	1	X'00000001'

### MQRFH2\_\*(발행/구독 옵션 태그 태그 이름)

MQRFH2_PUBSUB_CMD_FOLDER	"psc"
MQRFH2_PUBSUB_RESP_FOLDER	"pscr"
MQRFH2_MSG_CONTENT_FOLDER	"mcd"
MQRFH2_USER_FOLDER	"usr"

### MQRFH2\_\*(발행/구독 옵션 태그 XML 태그 이름)

MQRFH2_PUBSUB_CMD_FOLDER_B	"<psc>"
MQRFH2_PUBSUB_CMD_FOLDER_E	"</psc>"
MQRFH2_PUBSUB_RESP_FOLDER_B	"<pscr>"
MQRFH2_PUBSUB_RESP_FOLDER_E	"</pscr>"
MQRFH2_MSG_CONTENT_FOLDER_B	"<mcd>"
MQRFH2_MSG_CONTENT_FOLDER_E	"</mcd>"
MQRFH2_USER_FOLDER_B	"<usr>"

MQRFH2_USER_FOLDER_E	"</usr>"
----------------------	----------

### MQRL\_\*(리턴된 길이)

표 315. 상수 값		
이름	10진수 값	16진수 값
MQRL_UNDEFINED	-1	X'FFFFFFFF'

### MQRMH\_\*(참조 메시지 헤더 구조)

표 316. 상수 구조	
이름	구조
MQRMH_STRUC_ID	"RMH↵"
MQRMH_STRUC_ID_ARRAY	'R', 'M', 'H', '↵'

참고: ↵ 기호는 단일 공백 문자를 나타냅니다.

표 317. 상수 값		
이름	10진수 값	16진수 값
MQRMH_VERSION_1	1	X'00000001'
MQRMH_CURRENT_VERSION	1	X'00000001'

### MQRMHF\_\*(참조 메시지 헤더 플래그)

표 318. 상수 값		
이름	10진수 값	16진수 값
MQRMHF_LAST	1	X'00000001'
MQRMHF_NOT_LAST	0	X'00000000'

### MQRO\_\*(보고서 옵션)

표 319. 상수 값		
이름	10진수 값	16진수 값
MQRO_EXCEPTION	16777216	X'01000000'
MQRO_EXCEPTION_WITH_DATA	50331648	X'03000000'
MQRO_EXCEPTION_WITH_FULL_DATA	117440512	X'07000000'
MQRO_EXPIRATION	2097152	X'00200000'
MQRO_EXPIRATION_WITH_DATA	6291456	X'00600000'
MQRO_EXPIRATION_WITH_FULL_DATA	14680064	X'00E00000'
MQRO_COA	256	X'00000100'
MQRO_COA_WITH_DATA	768	X'00000300'
MQRO_COA_WITH_FULL_DATA	1792	X'00000700'
MQRO_COD	2048	X'00000800'
MQRO_COD_WITH_DATA	6144	X'00001800'
MQRO_COD_WITH_FULL_DATA	14336	X'00003800'
MQRO_PAN	1	X'00000001'
MQRO_NAN	2	X'00000002'



표 319. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRO_ACTIVITY	4	X'00000004'
MQRO_NEW_MSG_ID	0	X'00000000'
MQRO_PASS_MSG_ID	128	X'00000080'
MQRO_COPY_MSG_ID_TO_CORREL_ID	0	X'00000000'
MQRO_PASS_CORREL_ID	64	X'00000040'
MQRO_DEAD_LETTER_Q	0	X'00000000'
MQRO_DISCARD_MSG	134217728	X'08000000'
MQRO_PASS_DISCARD_AND_EXPIRY	16384	X'00004000'
MQRO_NONE	0	X'00000000'

### **MQRO\_\*(보고서 옵션 마스크)**

표 320. 상수 값		
이름	10진수 값	16진수 값
MQRO_REJECT_UNSUP_MASK	270270464	X'101C0000'
MQRO_ACCEPT_UNSUP_MASK	-270532353	X'EFE000FF'
MQRO_ACCEPT_UNSUP_IF_XMIT_MASK	261888	X'0003FF00'

### **MQROUTE\_\*(추적-라우트)**

#### **추적-라우트 최대 활동(MQIACF\_MAX\_ACTIVITIES)**

표 321. 상수 값		
이름	10진수 값	16진수 값
MQROUTE_UNLIMITED_ACTIVITIES	0	X'00000000'

#### **추적-라우트 세부사항(MQIACF\_ROUTE\_DETAIL)**

표 322. 상수 값		
이름	10진수 값	16진수 값
MQROUTE_DETAIL_LOW	2	X'00000002'
MQROUTE_DETAIL_MEDIUM	8	X'00000008'
MQROUTE_DETAIL_HIGH	32	X'00000020'

#### **추적 라우트 포워딩(MQIACF\_ROUTE\_FORWARDING)**

표 323. 상수 값		
이름	10진수 값	16진수 값
MQROUTE_FORWARD_ALL	256	X'00000100'
MQROUTE_FORWARD_IF_SUPPORTED	512	X'00000200'
MQROUTE_FORWARD_REJ_UNSUP_MASK	-65536	X'FFFF0000'

### 추적 라우트 전달(MQIACF\_ROUTE\_DELIVERY)

표 324. 상수 값		
이름	10진수 값	16진수 값
MQROUTE_DELIVER_YES	4096	X'00001000'
MQROUTE_DELIVER_NO	8192	X'00002000'
MQROUTE_DELIVER_REJ_UNSUP_MASK	-65536	X'FFFF0000'

### 추적 라우트 누적(MQIACF\_ROUTE\_ACCUMULATION)

표 325. 상수 값		
이름	10진수 값	16진수 값
MQROUTE_ACCUMULATE_NONE	65539	X'00010003'
MQROUTE_ACCUMULATE_IN_MSG	65540	X'00010004'
MQROUTE_ACCUMULATE_AND_REPLY	65541	X'00010005'

### MQRP\_\*(명령 형식 대체 옵션)

표 326. 상수 값		
이름	10진수 값	16진수 값
MQRP_YES	1	X'00000001'
MQRP_NO	0	X'00000000'

### MQRQ\_\*(명령 형식 이유 규정자)

표 327. 상수 값		
이름	10진수 값	16진수 값
MQRQ_CONN_NOT_AUTHORIZED	1	X'00000001'
MQRQ_OPEN_NOT_AUTHORIZED	2	X'00000002'
MQRQ_CLOSE_NOT_AUTHORIZED	3	X'00000003'
MQRQ_CMD_NOT_AUTHORIZED	4	X'00000004'
MQRQ_Q_MGR_STOPPING	5	X'00000005'
MQRQ_Q_MGR_QUIESCING	6	X'00000006'
MQRQ_CHANNEL_STOPPED_OK	7	X'00000007'
MQRQ_CHANNEL_STOPPED_ERROR	8	X'00000008'
MQRQ_CHANNEL_STOPPED_RETRY	9	X'00000009'
MQRQ_CHANNEL_STOPPED_DISABLED	10	X'0000000A'
MQRQ_BRIDGE_STOPPED_OK	11	X'0000000B'
MQRQ_BRIDGE_STOPPED_ERROR	12	X'0000000C'
MQRQ_SSL_HANDSHAKE_ERROR	13	X'0000000D'
MQRQ_SSL_CIPHER_SPEC_ERROR	14	X'0000000E'
MQRQ_SSL_CLIENT_AUTH_ERROR	15	X'0000000F'
MQRQ_SSL_PEER_NAME_ERROR	16	X'00000010'
MQRQ_SUB_NOT_AUTHORIZED	17	X'00000011'
MQRQ_SUB_DEST_NOT_AUTHORIZED	18	X'00000012'

표 327. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQRQ_SSL_UNKNOWN_REVOCATION	19	X'00000013'
MQRQ_SYS_CONN_NOT_AUTHORIZED	20	X'00000014'
MQRQ_CHANNEL_BLOCKED_ADDRESS	21	X'00000015'
MQRQ_CHANNEL_BLOCKED_USERID	22	X'00000016'
MQRQ_CHANNEL_BLOCKED_NOACCESS	23	X'00000017'
MQRQ_MAX_ACTIVE_CHANNELS	24	X'00000018'
MQRQ_MAX_CHANNELS	25	X'00000019'
MQRQ_SVRCONN_INST_LIMIT	26	X'0000001A'
MQRQ_CLIENT_INST_LIMIT!	27	X'0000001B'
MQRQ_CAF_NOT_INSTALLED	28	X'0000001C'
MQRQ_CSP_NOT_AUTHORIZED	29	X'0000001D'
MQRQ_FAILOVER_PERMITTED	30	X'0000001E'
MQRQ_FAILOVER_NOT_PERMITTED	31	X'0000001F'
MQRQ_STANDBY_ACTIVATED	32	X'00000020'
MQRQ_REPLICA_ACTIVATED	33	X'00000021'

### **MQRT\_\*(명령 형식 새로 고치기 유형)**

표 328. 상수 값		
이름	10진수 값	16진수 값
MQRT_CONFIGURATION	1	X'00000001'
MQRT_EXPIRY	2	X'00000002'
MQRT_NSPROC	3	X'00000003'
MQRT_PROXYSUB	4	X'00000004'

### **MQRU\_\*(요청 전용)**

표 329. 상수 값		
이름	10진수 값	16진수 값
MQRU_PUBLISH_ON_REQUEST	1	X'00000001'
MQRU_PUBLISH_ALL	2	X'00000002'

### **MQSCA\_\*(TLS 클라이언트 인증)**

표 330. 상수 값		
이름	10진수 값	16진수 값
MQSCA_REQUIRED	0	X'00000000'
MQSCA_OPTIONAL	1	X'00000001'

## MQSCO\_\*(TLS 구성 옵션)

### TLS 구성 옵션 구조

표 331. 상수 구조	
이름	구조
MQSCO_STRUC_ID	"SCO~"
MQSCO_STRUC_ID_ARRAY	'S','C','O','~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 332. 상수 값		
이름	10진수 값	16진수 값
MQSCO_VERSION_1	1	X'00000001'
MQSCO_VERSION_2	2	X'00000002'
MQSCO_VERSION_3	3	X'00000003'
MQSCO_VERSION_4	4	X'00000004'
MQSCO_CURRENT_VERSION	4	X'00000004'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

### TLS 구성 옵션 키 재설정 수

표 333. 상수 값		
이름	10진수 값	16진수 값
MQSCO_RESET_COUNT_DEFAULT	0	X'00000000'

### 명령 형식 큐 정의 범위

표 334. 상수 값		
이름	10진수 값	16진수 값
MQSCO_Q_MGR	1	X'00000001'
MQSCO_CELL	2	X'00000002'

### MQSCOPE\_\*(발행 범위)

표 335. 상수 값		
이름	10진수 값	16진수 값
MQSCOPE_ALL	0	X'00000000'
MQSCOPE_AS_PARENT	1	X'00000001'
MQSCOPE_QMGR	4	X'00000004'

### MQSCYC\_\*(보안 케이스)

표 336. 상수 값		
이름	10진수 값	16진수 값
MQSCYC_UPPER	0	X'00000000'
MQSCYC_MIXED	1	X'00000001'

## MQSD\_\*(오브젝트 디스크립터 구조)

표 337. 상수 이름 및 구조	
이름	구조
MQSD_STRUC_ID	"SD--"
MQSD_STRUC_ID_ARRAY	'S', 'D', '-', '-'

참고: - 기호는 단일 공백 문자를 나타냅니다.

표 338. 상수 값		
이름	10진수 값	16진수 값
MQSD_VERSION_1	1	X'00000001'
MQSD_CURRENT_VERSION	1	X'00000001'

## MQSECITEM\_\*(명령 형식 보안 항목)

표 339. 상수 값		
이름	10진수 값	16진수 값
MQSECITEM_ALL	0	X'00000000'
MQSECITEM_MQADMIN	1	X'00000001'
MQSECITEM_MQNLIST	2	X'00000002'
MQSECITEM_MQPROC	3	X'00000003'
MQSECITEM_MQQUEUE	4	X'00000004'
MQSECITEM_MQCONN	5	X'00000005'
MQSECITEM_MQCMDS	6	X'00000006'
MQSECITEM_MXADMIN	7	X'00000007'
MQSECITEM_MXNLIST	8	X'00000008'
MQSECITEM_MXPROC	9	X'00000009'
MQSECITEM_MXQUEUE	10	X'0000000A'
MQSECITEM_MXTOPIC	11	X'0000000B'

## MQSECPROT\_\*(보안 프로토콜 유형)

표 340. 상수 값		
이름	10진수 값	16진수 값
MQSECPROT_NONE	0	X'00000000'
MQSECPROT_SSLV30	1	X'00000001'
MQSECPROT_TL SV10	2	X'00000002'
MQSECPROT_TL SV12	4	X'00000004'

## MQSECSW\_\*(명령 형식 보안 스위치 및 스위치 상태)

### 명령 형식 보안 스위치

표 341. 상수 값		
이름	10진수 값	16진수 값
MQSECSW_PROCESS	1	X'00000001'

표 341. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQSECSW_NAMELIST	2	X'00000002'
MQSECSW_Q	3	X'00000003'
MQSECSW_TOPIC	4	X'00000004'
MQSECSW_CONTEXT	6	X'00000006'
MQSECSW_ALTERNATE_USER	7	X'00000007'
MQSECSW_COMMAND	8	X'00000008'
MQSECSW_CONNECTION	9	X'00000009'
MQSECSW_SUBSYSTEM	10	X'0000000A'
MQSECSW_COMMAND_RESOURCES	11	X'0000000B'
MQSECSW_Q_MGR	15	X'0000000F'
MQSECSW_QSG	16	X'00000010'

### 명령 형식 보안 스위치 상태

표 342. 상수 값		
이름	10진수 값	16진수 값
MQSECSW_OFF_FOUND	21	X'00000015'
MQSECSW_ON_FOUND	22	X'00000016'
MQSECSW_OFF_NOT_FOUND	23	X'00000017'
MQSECSW_ON_NOT_FOUND	24	X'00000018'
MQSECSW_OFF_ERROR	25	X'00000019'
MQSECSW_ON_OVERRIDDEN	26	X'0000001A'

### MQSECTYPE\_\*(명령 형식 보안 유형)

표 343. 상수 값		
이름	10진수 값	16진수 값
MQSECTYPE_AUTHSERV	1	X'00000001'
MQSECTYPE_SSL	2	X'00000002'
MQSECTYPE_CLASSES	3	X'00000003'

### MQSEG\_\*(세그먼트화)

표 344. 상수 이름 및 값	
이름	값
MQSEG_INHIBITED	'-'
MQSEG_ALLOWED	'A'

참고: - 기호는 단일 공백 문자를 나타냅니다.

## MQSEL\_\*(특수 선택자 값)

표 345. 상수 값		
이름	10진수 값	16진수 값
MQSEL_ANY_SELECTOR	-30001	X'FFFF8ACF'
MQSEL_ANY_USER_SELECTOR	-30002	X'FFFF8ACE'
MQSEL_ANY_SYSTEM_SELECTOR	-30003	X'FFFF8ACD'
MQSEL_ALL_SELECTORS	-30001	X'FFFF8ACF'
MQSEL_ALL_USER_SELECTORS	-30002	X'FFFF8ACE'
MQSEL_ALL_SYSTEM_SELECTORS	-30003	X'FFFF8ACD'

## MQSELTYPE\_\*(선택자 유형)

표 346. 상수 값		
이름	10진수 값	16진수 값
MQSELTYPE_NONE	0	X'00000000'
MQSELTYPE_STANDARD	1	X'00000001'
MQSELTYPE_EXTENDED	2	X'00000002'

## MQSID\_\*(보안 ID)

표 347. 상수 이름 및 값	
이름	값
MQSID_NONE	X'00...00' (40년)
MQSID_NONE_ARRAY	'\0', '\0', ... (40년)

## MQSIDT\_\*(보안 ID 유형)

표 348. 상수 이름 및 값	
이름	16진수 값
MQSIDT_NONE	X'00'
MQSIDT_NT_SECURITY_ID	X'01'
MQSIDT_WAS_SECURITY_ID	X'02'

## MQSMPO\_\*(메시지 특성 설정 옵션 및 구조)

### 메시지 특성 설정 옵션 구조

표 349. 상수 구조	
이름	구조
MQSMPO_STRUC_ID	"SMPO"
MQSMPO_STRUC_ID_ARRAY	'S', 'M', 'P', 'O'

참고: ㄴ 기호는 단일 공백 문자를 나타냅니다.

표 350. 상수 값		
이름	10진수 값	16진수 값
MQSMPO_VERSION_1	1	X'00000001'

표 350. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQSMPO_CURRENT_VERSION	1	X'00000001'

### 메시지 특성 설정 옵션

표 351. 상수 값		
이름	10진수 값	16진수 값
MQSMPO_SET_FIRST	0	X'00000000'
MQSMPO_SET_PROP_UNDER_CURSOR	1	X'00000001'
MQSMPO_SET_PROP_AFTER_CURSOR	2	X'00000002'
MQSMPO_APPEND_PROPERTY	4	X'00000004'
MQSMPO_SET_PROP_BEFORE_CURSOR	8	X'00000008'
MQSMPO_NONE	0	X'00000000'

### MQSO\_\*(구독 옵션)

표 352. 상수 값		
이름	10진수 값	16진수 값
MQSO_NONE	0	X'00000000'
MQSO_NON_DURABLE	0	X'00000000'
MQSO_READ_AHEAD_AS_Q_DEF	0	X'00000000'
MQSO_ALTER	1	X'00000001'
MQSO_CREATE	2	X'00000002'
MQSO_RESUME	4	X'00000004'
MQSO_DURABLE	8	X'00000008'
MQSO_GROUP_SUB	16	X'00000010'
MQSO_MANAGED	32	X'00000020'
MQSO_SET_IDENTITY_CONTEXT	64	X'00000040'
MQSO_FIXED_USERID	256	X'00000100'
MQSO_ANY_USERID	512	X'00000200'
MQSO_PUBLICATIONS_ON_REQUEST	2048	X'00000800'
MQSO_NEW_PUBLICATIONS_ONLY	4096	X'00001000'
MQSO_FAIL_IF QUIESCING	8192	X'00002000'
MQSO_ALTERNATE_USER_AUTHORITY	262144	X'00040000'
MQSO_WILDCARD_CHAR	1048576	X'00100000'
MQSO_WILDCARD_TOPIC	2097152	X'00200000'
MQSO_SET_CORREL_ID	4194304	X'00400000'
MQSO_SCOPE_QMGR	67108864	X'04000000'
MQSO_NO_READ_AHEAD	134217728	X'08000000'
MQSO_READ_AHEAD	268435456	X'10000000'



### MQSP\_\*(동기점 가용성)

표 353. 상수 값		
이름	10진수 값	16진수 값
MQSP_AVAILABLE	1	X'00000001'
MQSP_NOT_AVAILABLE	0	X'00000000'

### MQSPL\_\*(보안 정책 보호 옵션)

표 354. 상수 값		
이름	10진수 값	16진수 값
MQSPL_PASSTHRU	0	X'00000000'
MQSPL_REMOVE	1	X'00000001'
MQSPL_AS_POLICY	2	X'00000002'

### MQSQQM\_\*(공유 큐 큐 관리자 이름)

표 355. 상수 값		
이름	10진수 값	16진수 값
MQSQQM_USE	0	X'00000000'
MQSQQM_IGNORE	1	X'00000001'

### MQSR\_\*(조치)

표 356. 상수 값		
이름	10진수 값	16진수 값
MQSR_ACTION_PUBLICATION	1	X'00000001'

### MQSRO\_\*(구독 요청 옵션 구조)

표 357. 상수 구조	
이름	구조
MQSRO_STRUC_ID	"SRO↵"
MQSRO_STRUC_ID_ARRAY	'S', 'R', 'O', '↵'

참고: ↵ 기호는 단일 공백 문자를 나타냅니다.

표 358. 상수 값		
이름	10진수 값	16진수 값
MQSRO_VERSION_1	1	X'00000001'
MQSRO_CURRENT_VERSION	1	X'00000001'
MQSRO_NONE	0	X'00000000'
MQSRO_FAIL_IF QUIESCING	8192	X'00002000'

### MQSS\_\*(세그먼트 상태)

표 359. 상수 이름 및 구조	
이름	구조
MQSS_NOT_A_SEGMENT	'↵'

표 359. 상수 이름 및 구조 (계속)	
이름	구조
MQSS_SEGMENT	'S'
MQSS_LAST_SEGMENT	'L'

**참고:** - 기호는 단일 공백 문자를 나타냅니다.

### MQSSL\_\*(TLS FIPS 요구사항)

**참고:** AIX, Linux, and Windows에서 IBM MQ 는 IBM Crypto for C (ICC) 암호화 모듈을 통해 FIPS 140-2준수를 제공합니다. 이 모듈의 인증서가 히스토리 상태로 이동되었습니다. 고객은 IBM Crypto for C (ICC) 인증서를 보고 NIST에서 제공하는 조언을 알고 있어야 합니다. 대체 FIPS 140-2모듈이 현재 진행 중이며 해당 상태는 프로세스 목록의 NIST CMVP 모듈에서 검색하여 볼 수 있습니다.

IBM MQ Operator 3.2.0 및 큐 관리자 컨테이너 이미지 9.4.0.0 이상은 UBI 9를 기반으로 합니다. FIPS 140-2 준수가 현재 보류 중이며 해당 상태는 프로세스 목록의 NIST CMVP 모듈에서 "Red Hat Enterprise Linux 9-OpenSSL FIPS 제공자" 를 검색하여 볼 수 있습니다.

표 360. 상수 값		
이름	10진수 값	16진수 값
MQSSL_FIPS_NO	0	X'00000000'
MQSSL_FIPS_YES	1	X'00000001'

### MQSTAT\_\*(Stat 옵션)

표 361. 상수 값		
이름	10진수 값	16진수 값
MQSTAT_TYPE_ASYNC_ERROR	0	X'00000000'
MQSTAT_TYPE_RECONNECTION	0	X'00000000'
MQSTAT_TYPE_RECONNECTION_ERROR	0	X'00000000'

### MQSTS\_\*(상태 보고 구조 구조)

표 362. 상수 구조	
이름	구조
MQSTS_STRUC_ID	"STAT"
MQSTS_STRUC_ID_ARRAY	'S', 'T', 'A', 'T'

**참고:** - 기호는 단일 공백 문자를 나타냅니다.

표 363. 상수 값		
이름	10진수 값	16진수 값
MQSTS_VERSION_1	1	X'00000001'
MQSTS_CURRENT_VERSION	1	X'00000001'

## MQSUB\_\*(지속 가능 구독)

### 지속 가능 허용 구독

표 364. 상수 값		
이름	10진수 값	16진수 값
MQSUB_DURABLE_AS_PARENT	0	X'00000000'
MQSUB_DURABLE_ALLOWED	1	X'00000001'
MQSUB_DURABLE_INHIBITED	2	X'00000002'

### 지속 가능 구독 범위

표 365. 상수 값		
이름	10진수 값	16진수 값
MQSUB_DURABLE_ALL	-1	X'FFFFFFFF'
MQSUB_DURABLE_YES	1	X'00000001'
MQSUB_DURABLE_NO	2	X'00000002'

## MQSUBTYPE\_\*(명령 형식 구독 유형)

표 366. 상수 값		
이름	10진수 값	16진수 값
MQSUBTYPE_API	1	X'00000001'
MQSUBTYPE_ADMIN	2	X'00000002'
MQSUBTYPE_PROXY	3	X'00000003'
MQSUBTYPE_ALL	-1	X'FFFFFFFF'
MQSUBTYPE_USER	-2	X'FFFFFFFE'

## MQSUS\_\*(명령 형식 일시중단 상태)

표 367. 상수 값		
이름	10진수 값	16진수 값
MQSUS_YES	1	X'00000001'
MQSUS_NO	0	X'00000000'

## MQSVC\_\*(서비스)

### 서비스 유형

표 368. 상수 값		
이름	10진수 값	16진수 값
MQSVC_TYPE_COMMAND	0	X'00000000'
MQSVC_TYPE_SERVER	1	X'00000001'

## 서비스 제어

표 369. 상수 값		
이름	10진수 값	16진수 값
MQSVC_CONTROL_Q_MGR	0	X'00000000'
MQSVC_CONTROL_Q_MGR_START	1	X'00000001'
MQSVC_CONTROL_MANUAL	2	X'00000002'

## 서비스 상태

표 370. 상수 값		
이름	10진수 값	16진수 값
MQSVC_STATUS_STOPPED	0	X'00000000'
MQSVC_STATUS_STARTING	1	X'00000001'
MQSVC_STATUS_RUNNING	2	X'00000002'
MQSVC_STATUS_STOPPING	3	X'00000003'
MQSVC_STATUS_RETRYING	4	X'00000004'

## MQSYNCPOINT\_\*(발행/구독 마이그레이션에 대한 명령 형식 동기점 값)

표 371. 상수 값		
이름	10진수 값	16진수 값
MQSYNCPOINT_YES	0	X'00000000'
MQSYNCPOINT_IFPER	1	X'00000001'

## MQSYSP\_\*(명령 형식 시스템 매개변수 값)

표 372. 상수 값		
이름	10진수 값	16진수 값
MQSYSP_NO	0	X'00000000'
MQSYSP_YES	1	X'00000001'
MQSYSP_EXTENDED	2	X'00000002'
MQSYSP_TYPE_INITIAL	10	X'0000000A'
MQSYSP_TYPE_SET	11	X'0000000B'
MQSYSP_TYPE_LOG_COPY	12	X'0000000C'
MQSYSP_TYPE_LOG_STATUS	13	X'0000000D'
MQSYSP_TYPE_ARCHIVE_TAPE	14	X'0000000E'
MQSYSP_ALLOC_BLK	20	X'00000014'
MQSYSP_ALLOC_TRK	21	X'00000015'
MQSYSP_ALLOC_CYL	22	X'00000016'
MQSYSP_STATUS_BUSY	30	X'0000001E'
MQSYSP_STATUS_PREMOUNT	31	X'0000001F'
MQSYSP_STATUS_AVAILABLE	32	X'00000020'
MQSYSP_STATUS_UNKNOWN	33	X'00000021'
MQSYSP_STATUS_ALLOC_ARCHIVE	34	X'00000022'

표 372. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQSYSP_STATUS_COPYING_BSDS	35	X'00000023'
MQSYSP_STATUS_COPYING_LOG	36	X'00000024'

## MQTA\_\*(토픽 속성)

### 와일드카드

표 373. 상수 값		
이름	10진수 값	16진수 값
MQTA_BLOCK	1	X'00000001'
MQTA_PASSTHRU	2	X'00000002'

### 구독 허용

표 374. 상수 값		
이름	10진수 값	16진수 값
MQTA_SUB_AS_PARENT	0	X'00000000'
MQTA_SUB_INHIBITED	1	X'00000001'
MQTA_SUB_ALLOWED	2	X'00000002'

### 프록시 하위 전파

표 375. 상수 값		
이름	10진수 값	16진수 값
MQTA_PROXY_SUB_FORCE	1	X'00000001'
MQTA_PROXY_SUB_FIRSTUSE	2	X'00000002'

### 발행 허용

표 376. 상수 값		
이름	10진수 값	16진수 값
MQTA_PUB_AS_PARENT	0	X'00000000'
MQTA_PUB_INHIBITED	1	X'00000001'
MQTA_PUB_ALLOWED	2	X'00000002'

## MQTC\_\*(트리거 제어)

표 377. 상수 값		
이름	10진수 값	16진수 값
MQTC_OFF	0	X'00000000'
MQTC_ON	1	X'00000001'

## MQTCPKEEP\_\*(TCP 활성 유지(keepalive))

표 378. 상수 값		
이름	10진수 값	16진수 값
MQTCPKEEP_NO	0	X'00000000'
MQTCPKEEP_YES	1	X'00000001'

## MQTCPSTACK\_\*(TCP 스택 유형)

표 379. 상수 값		
이름	10진수 값	16진수 값
MQTCPSTACK_SINGLE	0	X'00000000'
MQTCPSTACK_MULTIPLE	1	X'00000001'

## MQTIME\_\*(명령 형식 시간 단위)

표 380. 상수 값		
이름	10진수 값	16진수 값
MQTIME_UNIT_MINS	0	X'00000000'
MQTIME_UNIT_SECS	1	X'00000001'

## MQTM\_\*(트리거 메시지 구조)

표 381. 상수 구조	
이름	구조
MQTM_STRUC_ID	"TM--"
MQTM_STRUC_ID_ARRAY	'T','M','-', '-'

참고: - 기호는 단일 공백 문자를 나타냅니다.

표 382. 상수 값		
이름	10진수 값	16진수 값
MQTM_VERSION_1	1	X'00000001'
MQTM_CURRENT_VERSION	1	X'00000001'

## MQTMC\_\*(트리거 메시지 문자 형식 구조)

표 383. 상수 구조	
이름	구조
MQTMC_STRUC_ID	"TMC--"
MQTMC_STRUC_ID_ARRAY	'T','M','C','-', '-'
MQTMC_VERSION_1	"---1"
MQTMC_VERSION_2	"---2"
MQTMC_CURRENT_VERSION	"---2"
MQTMC_VERSION_1_ARRAY	'-', '-', '-', '1'
MQTMC_VERSION_2_ARRAY	'-', '-', '-', '2'
MQTMC_CURRENT_VERSION_ARRAY	'-', '-', '-', '2'

### MQTOPT\_\*(토픽 유형)

표 384. 상수 값		
이름	10진수 값	16진수 값
MQTOPT_LOCAL	0	X'00000000'
MQTOPT_CLUSTER	1	X'00000001'
MQTOPT_ALL	2	X'00000002'

### MQTRAXSTR\_\*(채널 시작기 추적 자동시작)

표 385. 상수 값		
이름	10진수 값	16진수 값
MQTRAXSTR_NO	0	X'00000000'
MQTRAXSTR_YES	1	X'00000001'

### MQTSCOPE\_\*(구독 범위)

표 386. 상수 값		
이름	10진수 값	16진수 값
MQTSCOPE_QMGR	1	X'00000001'
MQTSCOPE_ALL	2	X'00000002'

### MQTT\_\*(트리거 유형)

표 387. 상수 값		
이름	10진수 값	16진수 값
MQTT_NONE	0	X'00000000'
MQTT_FIRST	1	X'00000001'
MQTT_EVERY	2	X'00000002'
MQTT_DEPTH	3	X'00000003'

### MQTYPE\_\*(특성 데이터 유형)

표 388. 상수 값		
이름	10진수 값	16진수 값
MQTYPE_AS_SET	0	X'00000000'
MQTYPE_NULL	2	X'00000002'
MQTYPE_BOOLEAN	4	X'00000004'
MQTYPE_BYTE_STRING	8	X'00000008'
MQTYPE_INT8	16	X'00000010'
MQTYPE_INT16	32	X'00000020'
MQTYPE_INT32	64	X'00000040'
MQTYPE_LONG	64	X'00000040'
MQTYPE_INT64	128	X'00000080'
MQTYPE_FLOAT32	256	X'00000100'
MQTYPE_FLOAT64	512	X'00000200'

표 388. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQTYPE_STRING	1024	X'00000400'

### MQUA\_\*(발행/구독 사용자 속성 선택자)

표 389. 상수 값		
이름	10진수 값	16진수 값
MQUA_FIRST	65536	X'00010000'
MQUA_LAST	999999999	X'3B9AC9FF'

### MQUIDSUPP\_\*(명령 형식 사용자 ID 지원)

표 390. 상수 값		
이름	10진수 값	16진수 값
MQUIDSUPP_NO	0	X'00000000'
MQUIDSUPP_YES	1	X'00000001'

### MQUNDELIVERED\_\*(명령 형식 발행/구독 마이그레이션의 미배달 값)

표 391. 상수 값		
이름	10진수 값	16진수 값
MQUNDELIVERED_NORMAL	0	X'00000000'
MQUNDELIVERED_SAFE	1	X'00000001'
MQUNDELIVERED_DISCARD	2	X'00000002'
MQUNDELIVERED_KEEP	3	X'00000003'

### MQUOWST\_\*(명령 형식 UOW 상태)

표 392. 상수 값		
이름	10진수 값	16진수 값
MQUOWST_NONE	0	X'00000000'
MQUOWST_ACTIVE	1	X'00000001'
MQUOWST_PREPARED	2	X'00000002'
MQUOWST_UNRESOLVED	3	X'00000003'

### MQUOWT\_\*(명령 형식 UOW 유형)

표 393. 상수 값		
이름	10진수 값	16진수 값
MQUOWT_Q_MGR	0	X'00000000'
MQUOWT_CICS	1	X'00000001'
MQUOWT_RRS	2	X'00000002'
MQUOWT_IMS	3	X'00000003'
MQUOWT_XA	4	X'00000004'



## MQUS\_\*(큐 용도)

표 394. 상수 값		
이름	10진수 값	16진수 값
MQUS_NORMAL	0	X'00000000'
MQUS_TRANSMISSION	1	X'00000001'

## MQUSAGE\_\*(명령 형식 페이지 세트 사용 값 및 데이터 세트 사용 값)

### 명령 형식 페이지 세트 사용 값

표 395. 상수 값		
이름	10진수 값	16진수 값
MQUSAGE_PS_AVAILABLE	0	X'00000000'
MQUSAGE_PS_DEFINED	1	X'00000001'
MQUSAGE_PS_OFFLINE	2	X'00000002'
MQUSAGE_PS_NOT_DEFINED	3	X'00000003'
MQUSAGE_PS_SUSPENDED	4	X'00000004'
MQUSAGE_EXPAND_USER	1	X'00000001'
MQUSAGE_EXPAND_SYSTEM	2	X'00000002'
MQUSAGE_EXPAND_NONE	3	X'00000003'

### 명령 형식 데이터 세트 사용 값

표 396. 상수 값		
이름	10진수 값	16진수 값
MQUSAGE_DS_OLDEST_ACTIVE_UOW	10	X'0000000A'
MQUSAGE_DS_OLDEST_PS_RECOVERY	11	X'0000000B'
MQUSAGE_DS_OLDEST_CF_RECOVERY	12	X'0000000C'

## MQVL\_\*(값 길이)

표 397. 상수 값		
이름	10진수 값	16진수 값
MQVL_NULL_TERMINATED	-1	X'FFFFFFFF'
MQVL_EMPTY_STRING	0	X'00000000'

## MQVU\_\*(가변 사용자 ID)

표 398. 상수 값		
이름	10진수 값	16진수 값
MQVU_FIXED_USER	1	X'00000001'
MQVU_ANY_USER	2	X'00000002'

## MQWDR\_\*(클러스터 워크로드 엑시트 목적지 레코드 구조)

표 399. 상수 구조	
이름	구조
MQWDR_STRUC_ID	"WDR~"
MQWDR_STRUC_ID_ARRAY	'W', 'D', 'R', '~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 400. 상수 값		
이름	10진수 값	16진수 값
MQWDR_VERSION_1	1	X'00000001'
MQWDR_VERSION_2	2	X'00000002'
MQWDR_CURRENT_VERSION	2	X'00000002'
MQWDR_LENGTH_1	124	X'0000007C'
MQWDR_LENGTH_2	136	X'00000088'
MQWDR_CURRENT_LENGTH	136	X'00000088'

## MQWI\_\*(대기 간격)

표 401. 상수 값		
이름	10진수 값	16진수 값
MQWI_UNLIMITED	-1	X'FFFFFFFF'

## MQWIH\_\*(워크로드 정보 헤더 구조 및 플래그)

### 워크로드 정보 헤더 구조

표 402. 상수 구조	
이름	구조
MQWIH_STRUC_ID	"WIH~"
MQWIH_STRUC_ID_ARRAY	'W', 'I', 'H', '~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 403. 상수 값		
이름	10진수 값	16진수 값
MQWIH_VERSION_1	1	X'00000001'
MQWIH_CURRENT_VERSION	1	X'00000001'
MQWIH_LENGTH_1	120	X'00000078'
MQWIH_CURRENT_LENGTH	120	X'00000078'

### 워크로드 정보 헤더 플래그

표 404. 상수 값		
이름	10진수 값	16진수 값
MQWIH_NONE	0	X'00000000'

## MQWQR\_\*(클러스터 워크로드 엑시트 큐 레코드 구조)

표 405. 상수 구조	
이름	구조
MQWQR_STRUC_ID	"WQR~"
MQWQR_STRUC_ID_ARRAY	'W', 'Q', 'R', '~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 406. 상수 값		
이름	10진수 값	16진수 값
MQWQR_VERSION_1	1	X'00000001'
MQWQR_VERSION_2	2	X'00000002'
MQWQR_VERSION_3	3	X'00000003'
MQWQR_CURRENT_VERSION	3	X'00000003'
MQWQR_LENGTH_1	200	X'000000C8'
MQWQR_LENGTH_2	208	X'000000D0'
MQWQR_LENGTH_3	212	X'000000D4'
MQWQR_CURRENT_LENGTH	212	X'000000D4'

## MQWS\_\*(와일드카드 스키마)

표 407. 상수 값		
이름	10진수 값	16진수 값
MQWS_DEFAULT	0	X'00000000'
MQWS_CHAR	1	X'00000001'
MQWS_TOPIC	2	X'00000002'

## MQWXP\_\*(클러스터 워크로드 엑시트 매개변수 구조)

### MQWXP\_\*(클러스터 워크로드 엑시트 매개변수 구조)

표 408. 상수 구조	
이름	구조
MQWXP_STRUC_ID	"WXP~"
MQWXP_STRUC_ID_ARRAY	'W', 'X', 'P', '~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 409. 상수 값		
이름	10진수 값	16진수 값
MQWXP_VERSION_1	1	X'00000001'
MQWXP_VERSION_2	2	X'00000002'
MQWXP_VERSION_3	3	X'00000003'
MQWXP_VERSION_4	4	X'00000004'
MQWXP_CURRENT_VERSION	4	X'00000004'

## MQWXP\_\*(클러스터 워크로드 플래그)

표 410. 상수 값		
이름	10진수 값	16진수 값
MQWXP_PUT_BY_CLUSTER_CHL	2	X'00000002'

### 관련 참조

1470 페이지의 『MQWXP의 필드 - 클러스터 워크로드 엑시트 매개변수 구조』  
MQWXP(클러스터 워크로드 엑시트 매개변수 구조)의 필드 설명

## MQXACT\_\*(API 호출자 유형)

표 411. 상수 값		
이름	10진수 값	16진수 값
MQXACT_EXTERNAL	1	X'00000001'
MQXACT_INTERNAL	2	X'00000002'

## MQXC\_\*(엑시트 명령)

표 412. 상수 값		
이름	10진수 값	16진수 값
MQXC_MQOPEN	1	X'00000001'
MQXC_MQCLOSE	2	X'00000002'
MQXC_MQGET	3	X'00000003'
MQXC_MQPUT	4	X'00000004'
MQXC_MQPUT1	5	X'00000005'
MQXC_MQINQ	6	X'00000006'
MQXC_MQSET	8	X'00000008'
MQXC_MQBACK	9	X'00000009'
MQXC_MQCMIT	10	X'0000000A'

## MQXCC\_\*(엑시트 응답)

표 413. 상수 값		
이름	10진수 값	16진수 값
MQXCC_OK	0	X'00000000'
MQXCC_SUPPRESS_FUNCTION	-1	X'FFFFFFFF'
MQXCC_SKIP_FUNCTION	-2	X'FFFFFFFE'
MQXCC_SEND_AND_REQUEST_SEC_MSG	-3	X'FFFFFFFD'
MQXCC_SEND_SEC_MSG	-4	X'FFFFFFFC'
MQXCC_SUPPRESS_EXIT	-5	X'FFFFFFFB'
MQXCC_CLOSE_CHANNEL	-6	X'FFFFFFFA'
MQXCC_REQUEST_ACK	-7	X'FFFFFFF9'
MQXCC_FAILED	-8	X'FFFFFFF8'

## MQXDR\_\*(엑시트 응답)

표 414. 상수 값		
이름	10진수 값	16진수 값
MQXDR_OK	0	X'00000000'
MQXDR_CONVERSION_FAILED	1	X'00000001'

## MQXE\_\*(환경)

표 415. 상수 값		
이름	10진수 값	16진수 값
MQXE_OTHER	0	X'00000000'
MQXE_MCA	1	X'00000001'
MQXE_MCA_SVRCONN	2	X'00000002'
MQXE_COMMAND_SERVER	3	X'00000003'
MQXE_MQSC	4	X'00000004'

## MQXEPO\_\*(시작점 등록 옵션 구조 및 엑시트 옵션)

### 시작점 등록 옵션 구조

표 416. 상수 구조	
이름	구조
MQXEPO_STRUC_ID	"XEPO"
MQXEPO_STRUC_ID_ARRAY	'X', 'E', 'P', 'O'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 417. 상수 값		
이름	10진수 값	16진수 값
MQXEPO_VERSION_1	1	X'00000001'
MQXEPO_CURRENT_VERSION	1	X'00000001'

### 엑시트 옵션

표 418. 상수 값		
이름	10진수 값	16진수 값
MQXEPO_NONE	0	X'00000000'

## MQXF\_\*(API 함수 ID)

표 419. 상수 값		
이름	10진수 값	16진수 값
MQXF_INIT	1	X'00000001'
MQXF_TERM	2	X'00000002'
MQXF_CONN	3	X'00000003'
MQXF_CONNX	4	X'00000004'
MQXF_DISC	5	X'00000005'

표 419. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQXF_OPEN	6	X'00000006'
MQXF_CLOSE	7	X'00000007'
MQXF_PUT1	8	X'00000008'
MQXF_PUT	9	X'00000009'
MQXF_GET	10	X'0000000A'
MQXF_DATA_CONV_ON_GET	11	X'0000000B'
MQXF_INQ	12	X'0000000C'
MQXF_SET	13	X'0000000D'
MQXF_BEGIN	14	X'0000000E'
MQXF_CMIT	15	X'0000000F'
MQXF_BACK	16	X'00000010'
MQXF_STAT	18	X'00000012'
MQXF_CB	19	X'00000013'
MQXF_CTL	20	X'00000014'
MQXF_CALLBACK	21	X'00000015'
MQXF_SUB	22	X'00000016'
MQXF_SUBRQ	23	X'00000017'
MQXF_XACLOSE	24	X'00000018'
MQXF_XACOMMIT	25	X'00000019'
MQXF_XACOMplete	26	X'0000001A'
MQXF_XAEND	27	X'0000001B'
MQXF_XAFORGET	28	X'0000001C'
MQXF_XAOPEN	29	X'0000001D'
MQXF_XAPREPARE	30	X'0000001E'
MQXF_XARECOVER	31	X'0000001F'
MQXF_XAROLLBACK	32	X'00000020'
MQXF_XASTART	33	X'00000021'
MQXF_AXREG	34	X'00000022'
MQXF_AXUNREG	35	X'00000023'

### MQXP\_\*(API 교차 엑시트 매개변수 구조)

표 420. 상수 구조	
이름	구조
MQXP_STRUC_ID	"XP~"
MQXP_STRUC_ID_ARRAY	'X', 'P', '~', '~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 421. 상수 값		
이름	10진수 값	16진수 값
MQXP_VERSION_1	1	X'00000001'

## MQXPDA\_\*(문제점 판별 영역)

표 422. 상수 이름 및 값	
이름	값
MQXPDA_NONE	X'00...00' (48널)
MQXPDA_NONE_ARRAY	'\0', '\0', ... (48널)

## MQXPT\_\*(전송 유형)

표 423. 상수 값		
이름	10진수 값	16진수 값
MQXPT_ALL	-1	X'FFFFFFFF'
MQXPT_LOCAL	0	X'00000000'
MQXPT_LU62	1	X'00000001'
MQXPT_TCP	2	X'00000002'
MQXPT_NETBIOS	3	X'00000003'
MQXPT_SPX	4	X'00000004'
MQXPT_DECNET	5	X'00000005'
MQXPT_UDP	6	X'00000006'

## MQXQH\_\*(전송 큐 헤더 구조)

표 424. 상수 구조	
이름	구조
MQXQH_STRUC_ID	"XQH~"
MQXQH_STRUC_ID_ARRAY	'X', 'Q', 'H', '~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 425. 상수 값		
이름	10진수 값	16진수 값
MQXQH_VERSION_1	1	X'00000001'
MQXQH_CURRENT_VERSION	1	X'00000001'

## MQXR\_\*(엑시트 이유)

표 426. 상수 값		
이름	10진수 값	16진수 값
MQXR_BEFORE	1	X'00000001'
MQXR_AFTER	2	X'00000002'
MQXR_CONNECTION	3	X'00000003'
MQXR_INIT	11	X'0000000B'
MQXR_TERM	12	X'0000000C'
MQXR_MSG	13	X'0000000D'
MQXR_XMIT	14	X'0000000E'
MQXR_SEC_MSG	15	X'0000000F'

표 426. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQXR_INIT_SEC	16	X'00000010'
MQXR_RETRY	17	X'00000011'
MQXR_AUTO_CLUSSDR	18	X'00000012'
MQXR_AUTO_RECEIVER	19	X'00000013'
MQXR_CLWL_OPEN	20	X'00000014'
MQXR_CLWL_PUT	21	X'00000015'
MQXR_CLWL_MOVE	22	X'00000016'
MQXR_CLWL_REPOS	23	X'00000017'
MQXR_CLWL_REPOS_MOVE	24	X'00000018'
MQXR_END_BATCH	25	X'00000019'
MQXR_ACK_RECEIVED	26	X'0000001A'
MQXR_AUTO_SVRCONN	27	X'0000001B'
MQXR_AUTO_CLUSRCVR	28	X'0000001C'
MQXR_SEC_PARMS	29	X'0000001D'

### MQXR2\_\*(엑시트 응답 2)

표 427. 상수 값		
이름	10진수 값	16진수 값
MQXR2_PUT_WITH_DEF_ACTION	0	X'00000000'
MQXR2_PUT_WITH_DEF_USERID	1	X'00000001'
MQXR2_PUT_WITH_MSG_USERID	2	X'00000002'
MQXR2_USE_AGENT_BUFFER	0	X'00000000'
MQXR2_USE_EXIT_BUFFER	4	X'00000004'
MQXR2_DEFAULT_CONTINUATION	0	X'00000000'
MQXR2_CONTINUE_CHAIN	8	X'00000008'
MQXR2_SUPPRESS_CHAIN	16	X'00000010'
MQXR2_STATIC_CACHE	0	X'00000000'
MQXR2_DYNAMIC_CACHE	32	X'00000020'

### MQXT\_\*(엑시트 ID)

표 428. 상수 값		
이름	10진수 값	16진수 값
MQXT_API_CROSSING_EXIT	1	X'00000001'
MQXT_API_EXIT	2	X'00000002'
MQXT_CHANNEL_SEC_EXIT	11	X'0000000B'
MQXT_CHANNEL_MSG_EXIT	12	X'0000000C'
MQXT_CHANNEL_SEND_EXIT	13	X'0000000D'
MQXT_CHANNEL_RCV_EXIT	14	X'0000000E'
MQXT_CHANNEL_MSG_RETRY_EXIT	15	X'0000000F'
MQXT_CHANNEL_AUTO_DEF_EXIT	16	X'00000010'



표 428. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQXT_CLUSTER_WORKLOAD_EXIT	20	X'00000014'
MQXT_PUBSUB_ROUTING_EXIT	21	X'00000015'

### MQXUA\_\*(엑시트 사용자 영역 값)

표 429. 상수 이름 및 값	
이름	값
MQXUA_NONE	X'00...00' (16널)
MQXUA_NONE_ARRAY	'\0', '\0', ... (16널)

### MQXWD\_\*(엑시트 대기 디스크립터 구조)

표 430. 상수 구조	
이름	구조
MQXWD_STRUC_ID	"XWD~"
MQXWD_STRUC_ID_ARRAY	'X', 'W', 'D', '~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 431. 상수 값		
이름	10진수 값	16진수 값
MQXWD_VERSION_1	1	X'00000001'

### MQZAC\_\*(애플리케이션 컨텍스트 구조)

표 432. 상수 구조	
이름	구조
MQZAC_STRUC_ID	"ZAC~"
MQZAC_STRUC_ID_ARRAY	'Z', 'A', 'C', '~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 433. 상수 값		
이름	10진수 값	16진수 값
MQZAC_VERSION_1	1	X'00000001'
MQZAC_CURRENT_VERSION	1	X'00000001'

### MQZAD\_\*(권한 데이터 구조)

표 434. 상수 구조	
이름	구조
MQZAD_STRUC_ID	"ZAD~"
MQZAD_STRUC_ID_ARRAY	'Z', 'A', 'D', '~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 435. 상수 값		
이름	10진수 값	16진수 값
MQZAD_VERSION_1	1	X'00000001'
MQZAD_VERSION_2	2	X'00000002'
MQZAD_CURRENT_VERSION	2	X'00000002'

### MQZAET\_\*(설치 가능 서비스 엔티티 유형)

표 436. 상수 값		
이름	10진수 값	16진수 값
MQZAET_NONE	0	X'00000000'
MQZAET_PRINCIPAL	1	X'00000001'
MQZAET_GROUP	2	X'00000002'
MQZAET_UNKNOWN	3	X'00000003'

### MQZAO\_\*(설치 가능 서비스 권한 부여)

표 437. 상수 값		
이름	10진수 값	16진수 값
MQZAO_CONNECT	1	X'00000001'
MQZAO_BROWSE	2	X'00000002'
MQZAO_INPUT	4	X'00000004'
MQZAO_OUTPUT	8	X'00000008'
MQZAO_INQUIRE	16	X'00000010'
MQZAO_SET	32	X'00000020'
MQZAO_PASS_IDENTITY_CONTEXT	64	X'00000040'
MQZAO_PASS_ALL_CONTEXT	128	X'00000080'
MQZAO_SET_IDENTITY_CONTEXT	256	X'00000100'
MQZAO_SET_ALL_CONTEXT	512	X'00000200'
MQZAO_ALTERNATE_USER_AUTHORITY	1024	X'00000400'
MQZAO_PUBLISH	2048	X'00000800'
MQZAO_SUBSCRIBE	4096	X'00001000'
MQZAO_RESUME	8192	X'00002000'
MQZAO_ALL_MQI	16383	X'00003FFF'
MQZAO_CREATE	65536	X'00010000'
MQZAO_DELETE	131072	X'00020000'
MQZAO_DISPLAY	262144	X'00040000'
MQZAO_CHANGE	524288	X'00080000'
MQZAO_CLEAR	1048576	X'00100000'
MQZAO_CONTROL	2097152	X'00200000'
MQZAO_CONTROL_EXTENDED	4194304	X'00400000'
MQZAO_AUTHORIZE	8388608	X'00800000'
MQZAO_ALL_ADMIN	16646144	X'00FE0000'
MQZAO_ALL	16662527	X'00FE3FFF'

표 437. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQZAO_REMOVE	16777216	X'01000000'
MQZAO_NONE	0	X'00000000'

### MQZAS\_\*(설치 가능 서비스 서비스 인터페이스 버전)

표 438. 상수 값		
이름	10진수 값	16진수 값
MQZAS_VERSION_1	1	X'00000001'
MQZAS_VERSION_2	2	X'00000002'
MQZAS_VERSION_3	3	X'00000003'
MQZAS_VERSION_4	4	X'00000004'
MQZAS_VERSION_5	5	X'00000005'
MQZAS_VERSION_6	6	X'00000006'

### MQZAT\_\*(인증 유형)

표 439. 상수 값		
이름	10진수 값	16진수 값
MQZAT_INITIAL_CONTEXT	0	X'00000000'
MQZAT_CHANGE_CONTEXT	1	X'00000001'

### MQZCI\_\*(설치 가능 서비스 연속 표시기)

표 440. 상수 값		
이름	10진수 값	16진수 값
MQZCI_DEFAULT	0	X'00000000'
MQZCI_CONTINUE	0	X'00000000'
MQZCI_STOP	1	X'00000001'

### MQZED\_\*(엔티티 데이터 구조)

표 441. 상수 구조	
이름	구조
MQZED_STRUC_ID	"ZED↵"
MQZED_STRUC_ID_ARRAY	'Z','E','D','↵'

참고: ↵ 기호는 단일 공백 문자를 나타냅니다.

표 442. 상수 값		
이름	10진수 값	16진수 값
MQZED_VERSION_1	1	X'00000001'
MQZED_VERSION_2	2	X'00000002'
MQZED_CURRENT_VERSION	2	X'00000002'

## MQZFP\_\*(프리 매개변수 구조)

표 443. 상수 구조	
이름	구조
MQZFP_STRUC_ID	"ZFP~"
MQZFP_STRUC_ID_ARRAY	'Z','F','P','~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 444. 상수 값		
이름	10진수 값	16진수 값
MQZFP_VERSION_1	1	X'00000001'
MQZFP_CURRENT_VERSION	1	X'00000001'

## MQZIC\_\*(ID 컨텍스트 구조)

표 445. 상수 구조	
이름	구조
MQZIC_STRUC_ID	"ZIC~"
MQZIC_STRUC_ID_ARRAY	'Z','I','C','~'

참고: ~ 기호는 단일 공백 문자를 나타냅니다.

표 446. 상수 값		
이름	10진수 값	16진수 값
MQZIC_VERSION_1	1	X'00000001'
MQZIC_CURRENT_VERSION	1	X'00000001'

## MQZID\_\*(서비스의 함수 id)

### 모든 서비스에 공용인 함수 id

표 447. 상수 값		
이름	10진수 값	16진수 값
MQZID_INIT	0	X'00000000'
MQZID_TERM	1	X'00000001'

### 권한 서비스의 함수 id

표 448. 상수 값		
이름	10진수 값	16진수 값
MQZID_INIT_AUTHORITY	0	X'00000000'
MQZID_TERM_AUTHORITY	1	X'00000001'
MQZID_CHECK_AUTHORITY	2	X'00000002'
MQZID_COPY_ALL_AUTHORITY	3	X'00000003'
MQZID_DELETE_AUTHORITY	4	X'00000004'
MQZID_SET_AUTHORITY	5	X'00000005'
MQZID_GET_AUTHORITY	6	X'00000006'

표 448. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQZID_GET_EXPLICIT_AUTHORITY	7	X'00000007'
MQZID_REFRESH_CACHE	8	X'00000008'
MQZID_ENUMERATE_AUTHORITY_DATA	9	X'00000009'
MQZID_AUTHENTICATE_USER	10	X'0000000A'
MQZID_FREE_USER	11	X'0000000B'
MQZID_INQUIRE	12	X'0000000C'
MQZID_CHECK_PRIVILEGED	13	X'0000000D'

### 이름 서비스의 함수 id

표 449. 상수 값		
이름	10진수 값	16진수 값
MQZID_INIT_NAME	0	X'00000000'
MQZID_TERM_NAME	1	X'00000001'
MQZID_LOOKUP_NAME	2	X'00000002'
MQZID_INSERT_NAME	3	X'00000003'
MQZID_DELETE_NAME	4	X'00000004'

### 사용자 ID 서비스의 함수 id

표 450. 상수 값		
이름	10진수 값	16진수 값
MQZID_INIT_USERID	0	X'00000000'
MQZID_TERM_USERID	1	X'00000001'
MQZID_FIND_USERID	2	X'00000002'

### MQZIO\_\*(설치 가능 서비스 초기화 옵션)

표 451. 상수 값		
이름	10진수 값	16진수 값
MQZIO_PRIMARY	0	X'00000000'
MQZIO_SECONDARY	1	X'00000001'

### MQZNS\_\*(이름 서비스 인터페이스 버전)

표 452. 상수 값		
이름	10진수 값	16진수 값
MQZNS_VERSION_1	1	X'00000001'

### MQZSE\_\*(설치 가능 서비스 시작-열거 표시기)

표 453. 상수 값		
이름	10진수 값	16진수 값
MQZSE_START	1	X'00000001'

표 453. 상수 값 (계속)		
이름	10진수 값	16진수 값
MQZSE_CONTINUE	0	X'00000000'

### MQZSL\_\*(설치 가능 서비스 선택자 표시기)

표 454. 상수 값		
이름	10진수 값	16진수 값
MQZSL_NOT_RETURNED	0	X'00000000'
MQZSL_RETURNED	1	X'00000001'

### MQZTO\_\*(설치 가능 서비스 종료 옵션)

표 455. 상수 값		
이름	10진수 값	16진수 값
MQZTO_PRIMARY	0	X'00000000'
MQZTO_SECONDARY	1	X'00000001'

### MQZUS\_\*(사용자 ID 인터페이스 버전)

표 456. 상수 값		
이름	10진수 값	16진수 값
MQZUS_VERSION_1	1	X'00000001'

## MQI에서 사용되는 데이터 유형

MQI(Message Queue Interface)에서 사용될 수 있는 데이터 유형에 대한 정보입니다. 각 데이터 유형에서 관련 언어에 대한 설명, 필드 및 언어 선언입니다.

### MQI의 데이터 유형 및 프로그래밍

기본 및 구조 데이터 유형과 C 프로그래밍, COBOL 프로그래밍 또는 High Level Assembler 프로그래밍을 통해 MQI를 사용하는 방법을 소개합니다.

#### 기본 데이터 유형

MQI 또는 엑시트 함수에서 사용되는 데이터 유형에 대한 정보입니다. 지원되는 프로그래밍 언어로 기본 데이터 유형을 선언하는 방법을 표시하는 예제가 뒤에 자세히 설명되어 있습니다.

MQI 또는 엑시트 함수에서 사용되는 데이터 유형은 다음 중 하나입니다.

- 요소 데이터 유형, 또는
- 요소 데이터 유형(배열 또는 구조)의 집합

다음 기본 데이터 유형은 MQI 또는 엑시트 함수에서 사용됩니다.

표 457. 요소 데이터 유형 이름, 유형 및 설명		
요소 데이터 유형 이름	데이터 유형	설명
MQBOOL	부울	MQBOOL 데이터 유형은 부울 값을 나타냅니다. 값 0은 false로 나타냅니다. 기타 값은 true를 나타냅니다. MQBOOL은 MQLONG 데이터 유형에 대해서와 같이 맞추어야 합니다.

표 457. 요소 데이터 유형 이름, 유형 및 설명 (계속)

요소 데이터 유형 이름	데이터 유형	설명
MQBYTE	Byte	<p>MQBYTE 데이터 유형은 1바이트의 데이터를 나타냅니다. 바이트에는 특별한 해석이 적용되지 않습니다. 2진 숫자 또는 문자가 아닌 비트 문자열로 간주됩니다. 특별하게 정렬하지 않아도 됩니다.</p> <p>다른 문자 세트 또는 인코딩을 사용하는 큐 관리자 간에 MQBYTE 데이터를 송신하는 경우, MQBYTE 데이터는 어떤 식으로든 변환되지 않습니다. MQMD 구조의 <i>MsgId</i> 및 <i>CorrelId</i> 필드가 이와 같습니다.</p> <p>MQBYTE의 배열이 때때로 큐 관리자에 인식되지 않는 주기억장치의 영역을 표시하는 데 사용됩니다. 예를 들면, 영역은 애플리케이션 메시지 데이터 또는 구조를 포함할 수 있습니다. 이 영역의 경계 맞추기는 포함된 데이터의 네이처와 호환 가능해야 합니다.</p> <p>C 프로그래밍 언어에서 MQBYTE 배열로 표시되는 함수 매개변수에는 모든 데이터 유형을 사용할 수 있습니다. 이는 이러한 매개변수가 항상 주소를 통해 전달되기 때문이며, C에서는 함수 매개변수가 pointer-to-void로 선언됩니다.</p>
MQBYTEn	n바이트 문자열	<p>각 MQBYTEn 데이터 유형은 n바이트의 문자열을 나타내며, 여기서 n은 8, 16, 24, 32, 40 또는 128 값 중 하나를 사용할 수 있습니다. 각 바이트는 MQBYTE 데이터 유형으로 설명됩니다. 특별하게 정렬하지 않아도 됩니다.</p> <p>바이트 문자열의 데이터가 문자열의 정의된 길이보다 짧으면 문자열을 채우기 위해 데이터가 널로 채워져야 합니다.</p> <p>큐 관리자가 바이트 문자열을 애플리케이션으로 되돌리면(예: MQGET 호출에서), 큐 관리자는 문자열의 정의된 길이에 맞게 널로 채웁니다.</p> <p>이름 지정된 상수를 사용하여 바이트 문자열 필드의 길이를 정의할 수 있습니다. 이는 61 페이지의 『Constants』에 나열되어 있습니다.</p>

표 457. 요소 데이터 유형 이름, 유형 및 설명 (계속)		
요소 데이터 유형 이름	데이터 유형	설명
MQCHAR	문자	<p>MQCHAR 데이터 유형은 1바이트 문자 또는 2바이트 또는 다중 바이트 문자의 1바이트를 나타냅니다. 특별하게 정렬하지 않아도 됩니다.</p> <p>다른 문자 세트 또는 인코딩을 사용하는 큐 관리자 간에 MQCHAR 데이터를 송신하는 경우, 올바른 데이터 해석을 위해 일반적으로 MQCHAR 데이터는 변환되어야 합니다. 큐 관리자는 MQMD 구조에서 MQCHAR 데이터에 대해 이 작업을 자동으로 수행합니다. 애플리케이션 메시지 데이터에서 MQCHAR 데이터의 변환은 MQGET 호출에 지정된 MQGMO_CONVERT 옵션에 의해 제어됩니다. 자세한 정보는 362 페이지의 『MQGMO - 메시지 가져오기 옵션』에서 이 옵션에 대한 설명을 참조하십시오.</p>
MQCHARn	n자 문자열	<p>각 MQCHARn 데이터 유형은 n자 문자열을 나타내며, 여기서 n은 4, 8, 12, 20, 28, 32, 48, 64, 128 또는 256 값 중 하나를 사용할 수 있습니다. 각 문자는 MQCHAR 데이터 유형으로 설명됩니다. 특별하게 정렬하지 않아도 됩니다.</p> <p>문자열의 데이터가 정의된 길이의 문자열보다 짧으면 문자열을 채우기 위해 데이터가 공백으로 채워집니다. 일부 경우에 널 문자는 공백으로 채우는 대신 중간에 문자열을 끝내는 데 사용될 수 있습니다. 뒤에 오는 널 문자 및 문자는 문자열의 정의된 길이까지 공백으로 처리됩니다. 널이 사용될 수 있는 위치는 호출 및 데이터 유형 설명에서 식별됩니다.</p> <p>큐 관리자가 문자열을 애플리케이션으로 리턴하면(예: MQGET 호출의 경우) 큐 관리자는 항상 문자열의 정의된 길이에 공백으로 채워집니다. 큐 관리자는 널 문자를 사용하여 문자열을 구분하지 않습니다.</p> <p>문자열 필드의 길이를 정의하는 이름 지정된 상수를 사용할 수 있으며, 이 상수는 61 페이지의 『Constants』에 나와 있습니다.</p>



표 457. 요소 데이터 유형 이름, 유형 및 설명 (계속)		
요소 데이터 유형 이름	데이터 유형	설명
MQFLOAT32	32비트 부동 소수점 숫자	<p>MQFLOAT32 데이터 유형은 표준 IEEE 부동 소수점 형식을 사용하여 표시되는 32비트 부동 소수점 숫자입니다. MQFLOAT32는 4바이트 경계에 맞춰야 합니다.</p> <p>z/OS에서 C의 MQFLOAT32를 사용하려면 FLOAT(IEEE) 컴파일러 플래그를 사용해야 합니다.</p> <p>COBOL에서 MQFLOAT32 사용은 IEEE 형식의 부동 소수점 숫자를 지원하는 컴파일러로 제한됩니다. 이는 FLOAT(NATIVE) 컴파일러 플래그의 사용을 요구할 수 있습니다.</p>
MQFLOAT64	64비트 부동 소수점 숫자	<p>MQFLOAT64 데이터 유형은 표준 IEEE 부동 소수점 형식을 사용하여 표시되는 64비트 부동 소수점 숫자입니다. MQFLOAT64는 8바이트 경계에 맞춰야 합니다.</p> <p>z/OS에서 C의 MQFLOAT64를 사용하려면 FLOAT(IEEE) 컴파일러 플래그를 사용해야 합니다.</p> <p>COBOL에서 MQFLOAT64 사용은 IEEE 형식의 부동 소수점 숫자를 지원하는 컴파일러로 제한됩니다. 이는 FLOAT(NATIVE) 컴파일러 플래그의 사용을 요구할 수 있습니다.</p>
MQHCONFIG	구성 핸들	<p>MQHCONFIG 데이터 유형은 특정 설치 가능 서비스에 대해 구성 중인 컴포넌트인 구성 핸들을 표시합니다. 구성 핸들은 자연적인 경계에 맞춰야 합니다.</p> <p>애플리케이션이 이 핸들 안에 저장된 데이터의 형식에 의존하지 않아야 합니다. 적절한 경우, 이 값은 이후 MQI 호출에 사용할 수 있지만, 해당 목적 외에 어떤 의미를 의도하지는 않습니다.</p>
MQHCONN	연결 핸들	<p>MQHCONN 데이터 유형은 연결 핸들, 즉 특정 큐 관리자에 대한 연결을 표시합니다. 연결 핸들은 4바이트 경계에 맞춰야 합니다.</p> <p>애플리케이션이 이 핸들 안에 저장된 데이터의 형식에 의존하지 않아야 합니다. 적절한 경우, 이 값은 이후 MQI 호출에 사용할 수 있지만, 해당 목적 외에 어떤 의미를 의도하지는 않습니다.</p>

표 457. 요소 데이터 유형 이름, 유형 및 설명 (계속)		
요소 데이터 유형 이름	데이터 유형	설명
MQHMSG	메시지 핸들	<p>MQHMSG 데이터 유형은 메시지에 액세스할 수 있는 메시지 핸들을 나타냅니다. 메시지 핸들은 8바이트 경계에 맞춰야 합니다.</p> <p>애플리케이션이 이 핸들 안에 저장된 데이터의 형식에 의존하지 않아야 합니다. 적절한 경우, 이 값은 이후 MQI 호출에 사용할 수 있지만, 해당 목적 외에 어떤 의미를 의도하지는 않습니다.</p>
MQHOBJ	오브젝트 핸들	<p>MQHOBJ 데이터 유형은 오브젝트에 액세스할 수 있는 오브젝트 핸들을 나타냅니다. 오브젝트 핸들은 4바이트 경계에 맞춰야 합니다.</p> <p>애플리케이션이 이 핸들 안에 저장된 데이터의 형식에 의존하지 않아야 합니다. 적절한 경우, 이 값은 이후 MQI 호출에 사용할 수 있지만, 해당 목적 외에 어떤 의미를 의도하지는 않습니다.</p>
MQINT8	8비트 부호있는 정수	컨텍스트에 의해 제한되지 않는 한 MQINT8 데이터 유형은 -128에서 +127까지 범위의 값을 가져올 수 있는 8비트 부호 있는 정수입니다.
MQINT16	16비트 부호 있는 정수	컨텍스트에 달리 제한되어 있지 않는 한, MQINT16 데이터 유형은 -32 768에서 +32 767까지 범위의 아무 값을 사용할 수 있는 16비트 부호 있는 정수입니다. MQINT16은 2바이트 경계에 맞춰야 합니다.
MQINT32	32비트 부호 있는 정수	<p>컨텍스트에 달리 제한되어 있지 않는 한, MQINT32 데이터 유형은 -2 147 483 648에서 +2 147 483 647까지 범위의 아무 값을 사용할 수 있는 32비트 부호 있는 2진 정수입니다.</p> <p>MQLONG의 정의를 참조하십시오.</p>
MQINT64	64비트 부호 있는 정수	<p>컨텍스트에 의해 제한되지 않는 한 MQINT64 데이터 유형은 -9 223 372 036 854 775 808에서 +9 223 372 036 854 775 807까지 범위의 값을 가져올 수 있는 64비트 부호 있는 정수입니다.</p> <p>COBOL의 경우, 올바른 범위는 -999 999 999 999 999 999에서 +999 999 999 999 999 999로 제한됩니다. 64비트 정수는 8바이트 경계에 맞춰야 합니다.</p>

표 457. 요소 데이터 유형 이름, 유형 및 설명 (계속)		
요소 데이터 유형 이름	데이터 유형	설명
MQLONG	32비트 부호 있는 정수	컨텍스트에 달리 제한되어 있지 않는 한, MQLONG 데이터 유형은 -2 147 483 648 ~ +2 147 483 647 범위의 값을 사용할 수 있는 32비트 부호 있는 2진 정수입니다.  COBOL의 경우, 올바른 범위는 -999 999 999에서 +999 999 999로 제한됩니다. MQLONG는 4바이트 경계에 맞춰야 합니다.
MQPID	프로세스 ID	IBM MQ 프로세스 ID입니다.  이 ID는 MQ 추적 및 FFST™ 덤프에서 사용되는 것과 동일하지만, 운영 체제 프로세스 ID와 다를 수 있습니다.
MQPTR	포인터	MQPTR 데이터 유형은 임의 유형의 데이터의 주소입니다. 포인터는 일반 경계에 맞춰야 하며, IBM i에서 16바이트 경계이고 다른 플랫폼에서는 8바이트 경계입니다.  일부 프로그래밍 언어는 입력된 포인터를 지원합니다. MQI은 또한 드물지만 이 포인터를 사용합니다(예: C 프로그래밍 언어에서 PMQCHAR 및 PMQLONG).
MQTID	스레드 ID	IBM MQ 스레드 ID.  이 ID는 MQ 추적 및 FFST™ 덤프에서 사용되는 것과 동일하지만, 운영 체제 스레드 ID와 다를 수 있습니다.
MQUINT8	8비트 부호 없는 정수	컨텍스트에 의해 제한되지 않는 한 MQUINT8 데이터 유형은 0에서 +255까지 범위의 값을 가져올 수 있는 8비트 부호 없는 정수입니다.
MQUINT16	16비트 부호 없는 정수	컨텍스트에 달리 제한되어 있지 않는 한, MQUINT16 데이터 유형은 범위 0 ~ +65 535 범위의 값을 사용할 수 있는 16비트 부호 없는 정수입니다. MQUINT16은 2바이트 경계에 맞춰야 합니다.
MQUINT32	32비트 부호 없는 정수	MQUINT32 데이터 유형은 32비트 부호 없는 2진 정수입니다.  MQULONG의 정의를 참조하십시오.

표 457. 요소 데이터 유형 이름, 유형 및 설명 (계속)		
요소 데이터 유형 이름	데이터 유형	설명
MQUINT64	64비트 부호 없는 정수	컨텍스트에 달리 제한되어 있지 않는 한, MQUINT64 데이터 유형은 0 ~ +18 446 744 073 709 551 615 범위의 값을 사용할 수 있는 64비트 부호 없는 정수입니다.  COBOL의 경우, 올바른 범위는 0에서 +999 999 999 999 999 999로 제한됩니다. 64비트 정수는 8바이트 경계에 맞춰야 합니다.
MQULONG	32비트 부호 없는 정수	컨텍스트에 의해 제한되지 않는 한 MQULONG 데이터 유형은 0에서 +4 294 967 294까지 범위의 값을 가져올 수 있는 32비트 부호 없는 경계 정수입니다.  COBOL의 경우, 올바른 범위는 0에서 +999 999 999로 제한됩니다. MQULONG은 4바이트 경계에 맞춰야 합니다.
PMQACH	포인터	MQACH 유형 데이터 구조에 대한 포인터
PMQAIR	포인터	MQAIR 유형 데이터 구조에 대한 포인터
PMQAXC	포인터	MQAXC 유형 데이터 구조에 대한 포인터
PMQAXP	포인터	MQAXP 유형 데이터 구조에 대한 포인터
PMQBMHO	포인터	MQBMHO 유형 데이터 구조에 대한 포인터
PMQBO	포인터	MQBO 유형 데이터 구조에 대한 포인터
PMQBOOL	포인터	MQBOOL 유형 데이터에 대한 포인터
PMQBYTE	포인터	MQBYTE 유형 데이터에 대한 포인터
PMQBYTEn	포인터	MQBYTEn 유형 데이터에 대한 포인터로, 여기서 n은 8, 16, 24, 32, 40, 128입니다.
PMQCBC	포인터	MQCBC 유형 데이터 구조에 대한 포인터
PMQCBD	포인터	MQCBD 유형 데이터 구조에 대한 포인터
PMQCHAR	포인터	MQCHAR 유형 데이터에 대한 포인터
PMQCHARN	포인터	MQCHARN 데이터 유형에 대한 포인터로, 여기서 n은 4, 8, 12, 20, 28, 32, 48, 64, 128, 256, 264입니다.

표 457. 요소 데이터 유형 이름, 유형 및 설명 (계속)		
요소 데이터 유형 이름	데이터 유형	설명
PMQCHARV	포인터	MQCHARV 유형 데이터 구조에 대한 포인터
PMQCIH	포인터	MQCIH 유형 데이터 구조에 대한 포인터
PMQCMHO	포인터	MQCMHO 유형 데이터 구조에 대한 포인터
PMQCNO	포인터	MQCNO 유형 데이터 구조에 대한 포인터
PMQCSP	포인터	MQCSP 유형 데이터 구조에 대한 포인터
PMQCTLO	포인터	MQCTLO 유형 데이터 구조에 대한 포인터
PMQDH	포인터	MQDH 유형 데이터 구조에 대한 포인터
PMQDHO	포인터	MQDHO 유형 데이터 구조에 대한 포인터
PMQDLH	포인터	MQDLH 유형 데이터 구조에 대한 포인터
PMQDMHO	포인터	MQDMHO 유형 데이터 구조에 대한 포인터
PMQDMPO	포인터	MQDMPO 유형 데이터 구조에 대한 포인터
PMQEPH	포인터	MQEPH 유형 데이터 구조에 대한 포인터
PMQFLOAT32	포인터	MQFLOAT32 유형 데이터 구조에 대한 포인터
PMQFLOAT64	포인터	MQFLOAT64 유형 데이터 구조에 대한 포인터
PMQFUNC	포인터	함수에 대한 포인터
PMQGMO	포인터	MQGMO 유형 데이터 구조에 대한 포인터
PMQHCONFIG	포인터	MQHCONFIG 유형 데이터에 대한 포인터
PMQHCONN	포인터	MQHCONN 유형 데이터에 대한 포인터
PMQHMSG	포인터	MQHMSG 유형 데이터에 대한 포인터
PMQHOBJ	포인터	MQHOBJ 유형 데이터에 대한 포인터
PMQIIH	포인터	MQIIH 유형 데이터 구조에 대한 포인터
PMQIMPO	포인터	MQIMPO 유형 데이터 구조에 대한 포인터

표 457. 요소 데이터 유형 이름, 유형 및 설명 (계속)		
요소 데이터 유형 이름	데이터 유형	설명
PMQINT8	포인터	MQINT8 유형 데이터에 대한 포인터
PMQINT16	포인터	MQINT16 유형 데이터에 대한 포인터
PMQINT32	포인터	MQINT32 유형 데이터에 대한 포인터
PMQINT64	포인터	MQINT64 유형 데이터에 대한 포인터
PMQLONG	포인터	MQLONG 유형 데이터에 대한 포인터
PMQMD	포인터	MQMD 유형 구조에 대한 포인터
PMQMDE	포인터	MQMDE 유형 데이터 구조에 대한 포인터
PMQMD1	포인터	MQMD1 유형 데이터 구조에 대한 포인터
PMQMD2	포인터	MQMD2 유형 데이터 구조에 대한 포인터
PMQMHBO	포인터	MQMHBO 유형 데이터 구조에 대한 포인터
PMQOD	포인터	MQOD 유형 데이터 구조에 대한 포인터
PMQOR	포인터	MQOR 유형 데이터 구조에 대한 포인터
PMQPD	포인터	MQPD 유형 데이터 구조에 대한 포인터
PMQPID	포인터	프로세스 ID에 대한 포인터
PMQMD	포인터	MQMD 유형 데이터 구조에 대한 포인터
PMQPMO	포인터	MQPMO 유형 데이터 구조에 대한 포인터
PMQPTR	포인터	MQPTR 유형 데이터에 대한 포인터
PMQRFH	포인터	MQRFH 유형 데이터 구조에 대한 포인터
PMQRFH2	포인터	MQRFH2 유형 데이터 구조에 대한 포인터
PMQRMH	포인터	MQRMH 유형 데이터 구조에 대한 포인터
PMQRR	포인터	MQRR 유형 데이터 구조에 대한 포인터
PMQSCO	포인터	MQSCO 유형 데이터 구조에 대한 포인터
PMQSD	포인터	MQSD 유형 데이터 구조에 대한 포인터
PMQSMPO	포인터	MQSMPO 유형 데이터 구조에 대한 포인터

표 457. 요소 데이터 유형 이름, 유형 및 설명 (계속)		
요소 데이터 유형 이름	데이터 유형	설명
PMQSRO	포인터	MQSRO 유형 데이터 구조에 대한 포인터
PMSSTS	포인터	MQSTS 유형 데이터 구조에 대한 포인터
PMQTID	포인터	스레드 ID에 대한 포인터
PMQTM	포인터	MQTM 유형 데이터 구조에 대한 포인터
PMQTM2	포인터	MQTM2 유형 데이터 구조에 대한 포인터
PMQUINT8	포인터	MQUINT8 데이터 유형에 대한 포인터
PMQUINT16	포인터	MQUINT16 데이터 유형에 대한 포인터
PMQUINT32	포인터	MQUINT32 데이터 유형에 대한 포인터
PMQUINT64	포인터	MQUINT64 데이터 유형에 대한 포인터
PMQULONG	포인터	MQULONG 데이터 유형에 대한 포인터
PMQVOID	포인터	
PMQWIH	포인터	MQWIH 유형 데이터 구조에 대한 포인터
PMQXQH	포인터	MQXQH 유형 데이터 구조에 대한 포인터

### C 데이터 유형 선언

표 458. C 데이터 유형 이름 및 표현	
데이터 유형	표현
MQBOOL	<pre>typedef MQLONG MQBOOL;</pre>
MQBYTE	<pre>typedef unsigned char MQBYTE;</pre>
MQBYTE8	<pre>typedef MQBYTE MQBYTE8[8];</pre>
MQBYTE16	<pre>typedef MQBYTE MQBYTE16[16];</pre>
MQBYTE24	<pre>typedef MQBYTE MQBYTE24[24];</pre>

표 458. C 데이터 유형 이름 및 표현 (계속)	
데이터 유형	표현
MQBYTE32	<code>typedef MQBYTE MQBYTE32[32];</code>
MQBYTE40	<code>typedef MQBYTE MQBYTE40[40];</code>
MQCHAR	<code>typedef char MQCHAR;</code>
MQCHAR4	<code>typedef MQCHAR MQCHAR4[4];</code>
MQCHAR8	<code>typedef MQCHAR MQCHAR8[8];</code>
MQCHAR12	<code>typedef MQCHAR MQCHAR12[12];</code>
MQCHAR20	<code>typedef MQCHAR MQCHAR20[20];</code>
MQCHAR28	<code>typedef MQCHAR MQCHAR28[28];</code>
MQCHAR32	<code>typedef MQCHAR MQCHAR32[32];</code>
MQCHAR48	<code>typedef MQCHAR MQCHAR48[48];</code>
MQCHAR64	<code>typedef MQCHAR MQCHAR64[64];</code>
MQCHAR128	<code>typedef MQCHAR MQCHAR128[128];</code>
MQCHAR256	<code>typedef MQCHAR MQCHAR256[256];</code>
MQFLOAT32	<code>typedef float MQFLOAT32;</code>
MQFLOAT64	<code>typedef double MQFLOAT64;</code>
MQHCONFIG	<code>typedef void MQPOINTER MQHCONFIG;</code>



표 458. C 데이터 유형 이름 및 표현 (계속)	
데이터 유형	표현
MQHCONN	<pre>typedef MQLONG MQHCONN;</pre>
MQHOBJ	<pre>typedef MQLONG MQHOBJ;</pre>
MQINT8	<pre>typedef signed char MQINT8;</pre>
MQINT16	<pre>typedef short MQINT16;</pre>
MQINT64	<p><b>UNIX</b> 64비트 UNIX의 경우:</p> <pre>typedef long;</pre> <p><b>AIX</b> 32비트 AIX의 경우:</p> <pre>typedef int64_t;</pre> <p><b>IBM i</b> <b>z/OS</b> <b>Linux</b> Linux, IBM i 및 z/OS의 경우:</p> <pre>typedef long long;</pre> <p><b>Windows</b> Windows의 경우:</p> <pre>typedef _int64;</pre>
MQLONG	<p><b>IBM i</b> IBM i의 경우:</p> <pre>typedef long MQLONG;</pre> <p><b>z/OS</b> <b>ALW</b> 기타 플랫폼의 경우:</p> <pre>if defined(MQ_64_BIT)     typedef int MQLONG; else     typedef long MQLONG;</pre>
MQPID	<pre>typedef MQLONG MQPID;</pre>
MQPTR	<pre>typedef void MQPOINTER MQPTR;</pre>

표 458. C 데이터 유형 이름 및 표현 (계속)	
데이터 유형	표현
MQTID	<pre>typedef MQLONG MQTID;</pre>
MQUINT8	<pre>typedef unsigned char MQUINT8;</pre>
MQUINT16	<pre>typedef unsigned short MQUINT16;</pre>
MQUINT64	<p><b>UNIX</b> 64비트 UNIX의 경우:</p> <pre>typedef unsigned long;</pre> <p><b>AIX</b> 32비트 AIX의 경우:</p> <pre>typedef uint64_t;</pre> <p><b>IBM i</b> <b>z/OS</b> <b>Linux</b> Linux, IBM i 및 z/OS의 경우:</p> <pre>typedef unsigned long long;</pre> <p><b>Windows</b> Windows의 경우:</p> <pre>typedef unsigned _int64;</pre>
MQLONG	<p><b>IBM i</b> IBM i의 경우:</p> <pre>typedef unsigned long MQLONG;</pre> <p><b>z/OS</b> <b>ALW</b> 기타 플랫폼의 경우:</p> <pre>if defined(MQ_64_BIT)     typedef unsigned int MQLONG; else     typedef unsigned long MQLONG;</pre>
PMQBO	<pre>typedef MQBO MQPOINTER PMQBO;</pre>
PMQBOOL	<pre>typedef MQBOOL MQPOINTER PMQBOOL;</pre>
PMQBYTE	<pre>typedef MQBYTE MQPOINTER PMQBYTE;</pre>

표 458. C 데이터 유형 이름 및 표현 (계속)	
데이터 유형	표현
PMQBYTE8	<code>typedef MQBYTE8[8] MQPOINTER PMQBYTE8[8];</code>
PMQBYTE16	<code>typedef MQBYTE16[16] MQPOINTER PMQBYTE16[16];</code>
PMQBYTE24	<code>typedef MQBYTE24[24] MQPOINTER PMQBYTE24[24];</code>
PMQBYTE32	<code>typedef MQBYTE32[32] MQPOINTER PMQBYTE32[32];</code>
PMQBYTE40	<code>typedef MQBYTE40[40] MQPOINTER PMQBYTE40[40];</code>
PMQBYTE128	<code>typedef MQBYTE128[128] MQPOINTER PMQBYTE128[128];</code>
PMQCHAR	<code>typedef MQCHAR MQPOINTER PMQCHAR;</code>
PMQCHAR4	<code>typedef MQCHAR4[4] MQPOINTER PMQCHAR4[4];</code>
PMQCHAR8	<code>typedef MQCHAR8[8] MQPOINTER PMQCHAR8[8];</code>
PMQCHAR12	<code>typedef MQCHAR12[12] MQPOINTER PMQCHAR12[12];</code>
PMQCHAR20	<code>typedef MQCHAR20[20] MQPOINTER PMQCHAR20[20];</code>
PMQCHAR28	<code>typedef MQCHAR28[28] MQPOINTER PMQCHAR28[28];</code>
PMQCHAR32	<code>typedef MQCHAR32[32] MQPOINTER PMQCHAR32[32];</code>
PMQCHAR48	<code>typedef MQCHAR48[48] MQPOINTER PMQCHAR48[48];</code>
PMQCHAR64	<code>typedef MQCHAR64[64] MQPOINTER PMQCHAR64[64];</code>
PMQCHAR128	<code>typedef MQCHAR128[128] MQPOINTER PMQCHAR128[128];</code>

표 458. C 데이터 유형 이름 및 표현 (계속)	
데이터 유형	표현
PMQCHAR256	<code>typedef MQCHAR256[256] MQPOINTER PMQCHAR256[256];</code>
PMQCHAR264	<code>typedef MQCHAR264[264] MQPOINTER PMQCHAR264[264];</code>
PMQCIH	<code>typedef MQCIH MQPOINTER PMQCIH;</code>
PMQCNO	<code>typedef MQCNO MQPOINTER PMQCNO;</code>
PMQDLH	<code>typedef MQDLH MQPOINTER PMQDLH;</code>
PMQFUNC	<code>typedef void MQPOINTER PMQFUNC;</code>
PMQFLOAT32	<code>typedef MQFLOAT32 MQPOINTER PMQFLOAT32;</code>
PMQFLOAT64	<code>typedef MQFLOAT64 MQPOINTER PMQFLOAT64;</code>
PMQGMO	<code>typedef MQGMO MQPOINTER PMQGMO;</code>
PMQHCONFIG	<code>typedef MQHCONFIG MQPOINTER PMQHCONFIG;</code>
PMQHCONN	<code>typedef MQHCONN MQPOINTER PMQHCONN;</code>
PMQHOBJ	<code>typedef MQHOBJ MQPOINTER PMQHOBJ;</code>
PMQIIH	<code>typedef MQIIH MQPOINTER PMQIIH;</code>
PMQINT8	<code>typedef MQINT8 MQPOINTER PMQINT8;</code>
PMQINT16	<code>typedef MQINT16 MQPOINTER PMQINT16;</code>
PMQLONG	<code>typedef MQLONG MQPOINTER PMQLONG;</code>

표 458. C 데이터 유형 이름 및 표현 (계속)	
데이터 유형	표현
PMQMD	<code>typedef MQMD MQPOINTER PMQMD;</code>
PMQMD1	<code>typedef MQMD1[1] MQPOINTER PMQMD1[1];</code>
PMQMDE	<code>typedef MQMDE MQPOINTER PMQMDE;</code>
PMQOD	<code>typedef MQOD MQPOINTER PMQOD;</code>
PMQPMO	<code>typedef MQPMO MQPOINTER PMQPMO;</code>
PMQPTR	<code>typedef MQPTR MQPOINTER PMQPTR;</code>
PMQRFH	<code>typedef MQRFH MQPOINTER PMQRFH;</code>
PMQRFH2	<code>typedef MQRFH2[2] MQPOINTER PMQRFH2[2];</code>
PMQRMH	<code>typedef MQRMH MQPOINTER PMQRMH;</code>
PMQTM	<code>typedef MQTM MQPOINTER PMQTM;</code>
PMQTM2	<code>typedef MQTM2[2] MQPOINTER PMQTM2[2];</code>
PMQUINT8	<code>typedef MQUINT8 MQPOINTER PMQUINT8;</code>
PMQUINT16	<code>typedef MQUINT16 MQPOINTER PMQUINT16;</code>
PMQULONG	<code>typedef MQULONG MQPOINTER PMQULONG;</code>
PMQVOID	<code>typedef void MQPOINTER PMQVOID;</code>
PMQWIH	<code>typedef MQWIH MQPOINTER PMQWIH;</code>

표 458. C 데이터 유형 이름 및 표현 (계속)	
데이터 유형	표현
PMQXQH	<code>typedef MQXQH MQPOINTER PMQXQH;</code>
PPMQBO	<code>typedef PMQBO MQPOINTER PPMQBO;</code>
PPMQBYTE	<code>typedef PMQBYTE MQPOINTER PPMQBYTE;</code>
PPMQCHAR	<code>typedef PMQCHAR MQPOINTER PPMQCHAR;</code>
PPMQCNO	<code>typedef PMQCNO MQPOINTER PPMQCNO;</code>
PPMQGMO	<code>typedef PMQGMO MQPOINTER PPMQGMO;</code>
PPMQHCONN	<code>typedef PMQHCONN MQPOINTER PPMQHCONN;</code>
PPMQHOBJ	<code>typedef PMQHOBJ MQPOINTER PPMQHOBJ;</code>
PPMQLONG	<code>typedef PMQLONG MQPOINTER PPMQLONG;</code>
PPMQMD	<code>typedef PMQMD MQPOINTER PPMQMD;</code>
PPMQOD	<code>typedef PMQOD MQPOINTER PPMQOD;</code>
PPMQPMO	<code>typedef PMQPMO MQPOINTER PPMQPMO;</code>
PPMQULONG	<code>typedef PMQULONG MQPOINTER PPMQULONG;</code>
PPMQVOID	<code>typedef PMQVOID MQPOINTER PPMQVOID;</code>

여기서 `defined(MQ_64_BIT)`은 64비트 플랫폼을 의미합니다.

MQPOINTER 매크로 변수에 대한 설명은 [261 페이지의 『데이터 유형』](#)의 내용을 참조하십시오.

COBOL 데이터 유형 선언

표 459. COBOL 데이터 유형 이름 및 표현	
데이터 유형	표현
MQBOOL	PIC S9(9) BINARY
MQBYTE	PIC X
MQBYTE8	PIC X(8)
MQBYTE16	PIC X(16)
MQBYTE24	PIC X(24)
MQBYTE32	PIC X(32)
MQBYTE40	PIC X(40)
MQCHAR	PIC X
MQCHAR4	PIC X(4)
MQCHAR8	PIC X(8)
MQCHAR12	PIC X(12)
MQCHAR20	PIC X(20)
MQCHAR28	PIC X(28)
MQCHAR32	PIC X(32)
MQCHAR48	PIC X(48)
MQCHAR64	PIC X(64)

표 459. COBOL 데이터 유형 이름 및 표현 (계속)	
데이터 유형	표현
MQCHAR128	PIC X(128)
MQCHAR256	PIC X(256)
MQFLOAT32	USAGE COMP-1
MQFLOAT64	USAGE COMP-2
MQHCONN	z/OS의 경우 PIC S9(9) COMP-5 기타 플랫폼 PIC S9(9) BINARY
MQHOBJ	PIC S9(9) BINARY
MQINT8	PIC S9(2) BINARY
MQINT16	PIC S9(4) BINARY
MQINT64	PIC S9(18) BINARY
MQLONG	PIC S9(9) BINARY
MQPTR	POINTER
MQUINT8	PIC 9(2) BINARY
MQUINT16	PIC 9(4) BINARY
MQUINT64	PIC 9(18) BINARY



표 459. COBOL 데이터 유형 이름 및 표현 (계속)	
데이터 유형	표현
MQULONG	PIC 9(9) BINARY

PL/I 데이터 유형 선언

표 460. PL/I 데이터 유형 이름 및 표현	
데이터 유형	표현
MQBOOL	fixed bin(31)
MQBYTE	char(1)
MQBYTE8	char(8)
MQBYTE16	char(16)
MQBYTE24	char(24)
MQBYTE32	char(32)
MQBYTE40	char(40)
MQCHAR	char(1)
MQCHAR4	char(4)
MQCHAR8	char(8)
MQCHAR12	char(12)
MQCHAR20	char(20)
MQCHAR28	char(28)

표 460. PL/I 데이터 유형 이름 및 표현 (계속)	
데이터 유형	표현
MQCHAR32	char(32)
MQCHAR48	char(48)
MQCHAR64	char(64)
MQCHAR128	char(128)
MQCHAR256	char(256)
MQFLOAT32	binary float(21) ieee
MQFLOAT64	binary float(52) ieee
MQHCONN	fixed bin(31)
MQHOBJ	fixed bin(31)
MQINT8	fixed bin(7)
MQINT16	fixed bin(15)
MQINT64	fixed bin(63)
MQLONG	fixed bin(31)
MQPTR	pointer
MQUINT8	fixed bin(8)
MQUINT16	fixed bin(16)

표 460. PL/I 데이터 유형 이름 및 표현 (계속)	
데이터 유형	표현
MQUINT64	fixed bin(64)
MQULONG	fixed bin(32)

High Level Assembler 데이터 유형 선언

표 461. System/390 어셈블러 데이터 유형 이름 및 표현	
데이터 유형	표현
MQBOOL	DS F
MQBYTE	DS XL1
MQBYTE8	DS XL8
MQBYTE16	DS XL16
MQBYTE24	DS XL24
MQBYTE32	DS XL32
MQBYTE40	DS XL40
MQCHAR	DS CL1
MQCHAR4	DS CL4
MQCHAR8	DS CL8
MQCHAR12	DS CL12
MQCHAR20	DS CL20

표 461. System/390 어셈블러 데이터 유형 이름 및 표현 (계속)	
데이터 유형	표현
MQCHAR28	DS CL28
MQCHAR32	DS CL32
MQCHAR48	DS CL48
MQCHAR64	DS CL64
MQCHAR128	DS CL128
MQCHAR256	DS CL256
MQFLOAT32	DS EB
MQFLOAT64	DS DB
MQHCONN	DS F
MQHOBJ	DS F
MQINT8	DS XL1
MQINT16	DS H
MQINT64	DS D
MQLONG	DS F
MQPTR	DS F
MQUINT8	DS XL1

표 461. System/390 어셈블러 데이터 유형 이름 및 표현 (계속)	
데이터 유형	표현
MQUINT16	DS H
MQUINT64	DS D
MQULONG	DS F

### 구조 데이터 유형

구조 데이터 유형의 요약, MQI 구조를 일관되게 맵핑하기 위한 규칙 및 각 구조 데이터 유형 설명에서 사용되는 규칙입니다.

- 257 페이지의 『MQI 호출 또는 엑시트 함수에서 사용되는 구조 데이터 유형의 요약』
- 258 페이지의 『메시지 데이터에서 사용되는 구조 데이터 유형의 요약』
- 258 페이지의 『MQI 구조를 일관되게 맵핑하기 위한 규칙』
- 259 페이지의 『각 구조 데이터 유형 설명에 사용되는 규칙』

### MQI 호출 또는 엑시트 함수에서 사용되는 구조 데이터 유형의 요약

표 462. MQI 호출 또는 엑시트 함수에서 사용되는 구조 데이터 유형		
구조	설명	사용 시 호출
MQACH	API 엑시트 체인 헤더	
<u>MQAIR</u>	인증 정보 레코드	<u>MQCONN</u>
MQAXC	API 엑시트 컨텍스트	
MQAXP	API 엑시트 매개변수	
<u>MQBMHO</u>	메시지 핸들 버퍼 옵션	<u>MQBUF</u>
<u>MQBO</u>	시작 옵션	<u>MQBEGIN</u>
<u>MQCBD</u>	콜백 디스크립터	<u>MQCB</u>
MQCBO	Create-bag 옵션	mqCreateBag
<u>MQCHARV</u>	가변 길이 문자열	<u>MQINQMP</u>
<u>MQCNO</u>	연결 옵션	<u>MQCONN</u>
<u>MQCSP</u>	보안 매개변수	<u>MQCONN</u>
<u>MQCTLO</u>	콜백 옵션	<u>MQCTL</u>
<u>MQDMPO</u>	메시지 특성 삭제 옵션	<u>MQDLTMP</u>
<u>MQGMO</u>	메시지 가져오기 옵션	<u>MQGET</u>
<u>MQIMPO</u>	메시지 특성 조회 옵션	<u>MQINQMP</u>
<u>MQMD</u>	메시지 디스크립터	<u>MQBUF</u> , <u>MQMHB</u> , <u>MQCB</u> , <u>MQGET</u> , <u>MQPUT</u> , <u>MQPUT1</u>
<u>MQMHBO</u>	메시지 핸들 버퍼 옵션	<u>MQMHB</u>

표 462. MQI 호출 또는 엑시트 함수에서 사용되는 구조 데이터 유형 (계속)		
구조	설명	사용 시 호출
<a href="#">MQOD</a>	오브젝트 디스크립터	<a href="#">MQOPEN</a> , <a href="#">MQPUT1</a>
<a href="#">MQOR</a>	오브젝트 레코드	<a href="#">MQOPEN</a> , <a href="#">MQPUT1</a>
<a href="#">MQPD</a>	특성 디스크립터	<a href="#">MQSETMP</a>
<a href="#">MQPMO</a>	메시지 넣기 옵션	<a href="#">MQPUT</a> , <a href="#">MQPUT1</a>
<a href="#">MQPMR</a>	메시지 넣기 레코드	<a href="#">MQPUT</a> , <a href="#">MQPUT1</a>
<a href="#">MQRR</a>	응답 레코드	<a href="#">MQOPEN</a> , <a href="#">MQPUT</a> , <a href="#">MQPUT1</a>
<a href="#">MQSCO</a>	TLS 구성 옵션	<a href="#">MQCONN</a>
<a href="#">MQSD</a>	구독 디스크립터	<a href="#">MQSUB</a>
<a href="#">MQSMPO</a>	메시지 특성 설정 옵션	<a href="#">MQSETMP</a>
<a href="#">MQSRO</a>	구독 요청 옵션	<a href="#">MQSUBRQ</a>
<a href="#">MQSTS</a>	상태 보고 구조	<a href="#">MQSTAT</a>

### 메시지 데이터에서 사용되는 구조 데이터 유형의 요약

표 463. 메시지 데이터에 사용된 구조 데이터 유형	
구조	설명
<a href="#">MQCIH</a>	CICS 정보 헤더
<a href="#">MQCFH</a>	PCF 헤더
<a href="#">MQEPH</a>	임베드된 PCF 헤더
<a href="#">MQDH</a>	분배 헤더
<a href="#">MQDLH</a>	데드 레터(미배달 메시지) 헤더
<a href="#">MQIIH</a>	IMS 정보 헤더
<a href="#">MQMDE</a>	메시지 디스크립터 확장
<a href="#">MQRFH</a>	규칙 및 형식화 헤더
<a href="#">MQRFH2</a>	규칙 및 형식화 헤더 2
<a href="#">MQRMH</a>	참조 메시지 헤더
<a href="#">MQTM</a>	트리거 메시지
<a href="#">MQTMC2</a>	트리거 메시지(문자 형식 2)
<a href="#">MQWIH</a>	작업 정보 헤더
<a href="#">MQXQH</a>	전송 큐 헤더

**참고:** MQDXP 구조(데이터 변환 엑시트 매개변수) 및 연관된 데이터 변환 호출에 대한 설명은 [869 페이지의 『데이터 변환 엑시트』](#)에 나와 있습니다.

### MQI 구조를 일관되게 맵핑하기 위한 규칙

프로그래밍 언어는 구조에 대한 자체 지원 레벨에 있어서 다양하며, 각 프로그래밍 언어에서 일관적으로 MQI 구조를 맵핑할 수 있도록 특정 규칙 및 규약이 채택됩니다.

1. 구조는 자체 자연 경계에서 정렬되어야 합니다.
  - 대부분의 MQI 구조는 4바이트 정렬을 요구합니다.
  - IBM i에서, 포인터가 포함된 구조는 16바이트 맞추기를 요구합니다(예: MQCNO, MQOD, MQPMO).
2. 구조의 각 필드는 자체 자연 경계에서 정렬되어야 합니다.
  - MQLONG에 해당하는 데이터 유형의 필드는 4바이트 경계에서 맞추기가 필요합니다.
  - MQPTR에 해당하는 데이터 유형의 필드는 IBM i의 16바이트 경계에서 및 기타 환경의 4바이트 경계에서 맞추기가 필요합니다.
  - 기타 필드는 1바이트 경계에서 정렬됩니다.
3. 구조의 길이는 자체 경계 정렬의 배수여야 합니다.
  - 대부분의 MQI 구조의 길이는 4바이트의 배수입니다.
  - IBM i에서, 포인터가 포함된 구조는 16바이트의 배수인 길이를 보유합니다.
4. 필요한 경우, 채우기 바이트 또는 필드는 선행 규칙의 준수를 보장하도록 추가되어야 합니다.

## 각 구조 데이터 유형 설명에 사용되는 규칙

각 구조 데이터 유형의 설명에는 다음이 포함됩니다.

- 구조의 용도 및 사용 개요
- 프로그래밍 언어와는 독립적인 양식으로, 구조의 필드에 대한 설명
- 각각의 지원되는 프로그래밍 언어에서 구조가 선언되는 방법의 예제

각각의 구조 데이터 유형의 설명에는 다음 절이 포함되어 있습니다.

### 구조 이름

구조의 이름 및 구조에서 필드의 요약.

### 개요

구조의 용도 및 사용에 대한 간략한 설명.

### 필드

필드의 설명. 필드마다 필드의 이름 다음에 괄호()로 묶인 기본 데이터 유형이 옵니다. 텍스트에서 필드 이름은 기울임체 글자체를 사용하여 표시됩니다(예: *Version*).

또한 필드가 취할 수 있는 값의 목록과 함께 필드의 용도에 대한 설명도 있습니다. 상수 이름은 대문자로 표시됩니다(예: MQGMO\_STRUC\_ID). 동일한 접두부를 가진 상수 세트는 \* 문자를 사용하여 표시됩니다(예: MQIA\_\*).

필드의 설명에서는 다음 용어가 사용됩니다.

### 입력

호출을 작성할 때 필드에 정보를 제공합니다.

### 출력

호출이 완료되거나 실패할 때 큐 관리자가 필드의 정보를 리턴합니다.

### 입출력(I/O)

호출을 작성할 때 필드에 정보를 제공하며, 호출이 완료되거나 실패할 때 큐 관리자가 정보를 변경합니다.

### 초기값

MQI에서 제공하는 데이터 정의 파일의 각 필드에 대한 초기값을 표시하는 테이블.

### C 선언

C에서 구조의 일반 선언.

### COBOL 선언

COBOL에서 구조의 일반 선언.

### PL/I 선언

PL/I에서 구조의 일반 선언.

## 상위 레벨 어셈블러 선언

System/390 어셈블러 언어에서 구조의 일반 선언.

## Visual Basic 선언



Visual Basic에서 구조의 일반 선언.

## C 프로그래밍

C 프로그래밍 언어에서 MQI를 사용하는 데 도움이 되는 정보입니다.

- [260 페이지의 『헤더 파일』](#)
- [260 페이지의 『함수』](#)
- [261 페이지의 『데이터 유형이 정의되지 않은 매개변수』](#)
- [261 페이지의 『데이터 유형』](#)
- [261 페이지의 『2진 문자열 조작』](#)
- [261 페이지의 『문자열 조작』](#)
- [261 페이지의 『구조의 초기값』](#)
- [262 페이지의 『동적 구조의 초기값』](#)
- [262 페이지의 『C++에서 사용』](#)
- [263 페이지의 『표기법 규칙』](#)

## 헤더 파일

표 464. C 헤더 파일	
파일	컨텐츠
CMQC	기본 MQI에 대한 함수 프로토타입, 데이터 유형 및 이름 지정된 상수.
CMQXC	데이터 변환 엑시트에 대한 함수 프로토타입, 데이터 유형 및 이름 지정된 상수.
CMQEC	기본 MQI, 데이터 변환 엑시트 및 인터페이스 시작점 구조에 대한 함수 프로토타입, 데이터 유형 및 이름 지정된 상수(CMQEC에 CMQXC 및 CMQC 포함)
CMQSTRC	MQI 상수 정의를 텍스트 등가물로 변환하는 기능.  <b>주의:</b>  IBM MQ 9.1에서 z/OS에 적용 가능합니다. 이 헤더 파일을 사용하는 프로그램은 LONGNAME 컴파일러 옵션으로 컴파일되어야 합니다.

애플리케이션의 이식성을 향상시키려면 #include 프리프로세서 지시문에서 소문자로 헤더 파일의 이름을 코딩하십시오.

```
#include "cmqec.h"
```

## 함수

함수를 호출할 때마다 주소에 의해 전달되는 모든 매개변수를 지정할 필요가 없습니다.

- 값에 의해 MQHCONN, MQHOBJ 또는 MQLONG 유형의 입력 전용 매개변수를 전달합니다.
- 주소에 의해 기타 모든 매개변수를 전달합니다.

특정 매개변수가 필요하지 않은 경우, 매개변수 데이터의 주소 대신 함수 호출에서 널 포인터를 매개변수로서 사용하십시오. 이러한 매개변수는 호출 설명에서 식별됩니다.

함수의 값으로 리턴되는 매개변수는 없습니다. C 용어로 이는 모든 함수가 void를 리턴한다는 것을 의미합니다.

함수의 속성은 MQENTRY 매크로 변수에서 정의합니다. 이 매크로 변수의 값은 환경에 따라 달라집니다.



## 데이터 유형이 정의되지 않은 매개변수

MQGET, MQPUT 및 MQPUT1 함수의 **Buffer** 매개변수에 정의되지 않은 데이터 유형이 있습니다. 이 매개변수는 애플리케이션의 메시지 데이터를 송신하고 수신하는 데 사용됩니다.

이러한 종류의 매개변수는 MQBYTE의 배열로 C 예제에 표시됩니다. 이 방식으로 매개변수를 선언할 수 있지만 일반적으로 메시지의 데이터 레이아웃을 설명하는 특정 구조로 선언하는 것이 보다 편리합니다. 실제 함수 매개변수를 pointer-to-void로 선언하고, 임의의 종류인 데이터의 주소를 함수 호출의 매개변수로서 지정하십시오.

## 데이터 유형

C typedef 명령문을 사용하여 모든 데이터 유형을 정의합니다. 각 데이터 유형마다 해당되는 포인터 데이터 유형도 정의합니다. 포인터 데이터 유형의 이름은 포인터를 표시하기 위해 문자 P가 접두부로 사용되는 기본 또는 구조 데이터 유형의 이름입니다. MQPOINTER 매크로 변수를 사용하여 포인터의 속성을 정의하십시오. 이 매크로 변수의 값은 환경에 따라 다릅니다. 다음은 포인터 데이터 유형을 선언하는 방법을 보여줍니다.

```
#define MQPOINTER * /* depends on environment */
...
typedef MQLONG MQPOINTER PMQLONG; /* pointer to MQLONG */
typedef MQMD MQPOINTER PMQMD; /* pointer to MQMD */
```

## 2진 문자열 조작

MQBYTEn 데이터 유형 중 하나로 2진 데이터의 문자열을 선언합니다.

이 유형의 필드를 복사, 비교하거나 설정할 때마다 C 함수 **memcpy**, **memcmp** 또는 **memset**를 사용하십시오. 예를 들어, 다음과 같습니다.

```
#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));
```

문자열 함수 **strcpy**, **strcmp**, **strncpy** 또는 **strncmp**를 사용하지 마십시오. 이러한 함수가 MQBYTEn 데이터 유형으로 선언된 데이터에 대해 올바르게 작동하지 않기 때문입니다.

## 문자열 조작

큐 관리자가 문자 데이터를 애플리케이션에 리턴할 때, 큐 관리자는 항상 정의된 필드 길이만큼 공백으로 문자 데이터를 채웁니다. 큐 관리자는 널 종료 문자열을 리턴하지 않습니다.

따라서 해당 문자열을 복사, 비교하거나 연결할 때는 문자열 함수 **strncpy**, **strncmp** 또는 **strncat**를 사용하십시오.

문자열이 널 (**strcpy**, **strcmp**, **strcat**) 로 종료되어야 하는 문자열 함수를 사용하지 마십시오. 또한 **strlen** 함수를 사용하여 문자열의 길이를 판별하지 마십시오. 대신 **sizeof** 함수를 사용하여 필드의 길이를 판별하십시오.

## 구조의 초기값

헤더 파일은 해당 구조의 인스턴스를 선언할 때 MQ 구조에 대해 초기값을 제공하기 위해 사용할 수 있는 다양한 매크로 변수를 정의합니다.

이러한 매크로 변수에 MQxxx\_DEFAULT 양식의 이름이 있으며 여기서 MQxxx는 구조의 이름을 나타냅니다. 이는 다음 방법으로 사용됩니다.

```
MQMD MyMsgDesc = {MQMD_DEFAULT};
MQPMO MyPutOpts = {MQPMO_DEFAULT};
```

일부 문자 필드의 경우(예: 대부분의 구조에서 발생하는 *StrucId* 필드 또는 MQMD에서 발생하는 *Format* 필드), MQI는 유효한 특정 값을 정의합니다. 올바른 값에 대해 각각 2개의 매크로 변수가 제공됩니다.

- 하나의 매크로 변수는 정의된 필드 길이와 정확히 일치하는(포함된 널 일치는 제외) 길이의 문자열로서 값을 정의합니다. 예를 들어, MQMD의 *Format* 필드의 경우 다음 매크로 변수가 제공됩니다 (↵는 단일 공백 문자를 나타냄).

```
#define MQFMT_STRING "MQSTR↵↵↵"
```

이 양식을 memcpy 및 memcmp 함수와 함께 사용하십시오.

- 기타 매크로 변수는 값을 문자 배열로서 정의합니다. 이 매크로 변수의 이름은 접미부가 *\_ARRAY*인 문자열 양식의 이름입니다. 예를 들면, 다음과 같습니다.

```
#define MQFMT_STRING_ARRAY 'M','Q','S','T','R','↵','↵','↵'
```

MQMD\_DEFAULT 매크로 변수에서 제공하는 것과는 다른 값으로 구조의 인스턴스를 선언할 때는 이 양식을 사용하여 필드를 초기화하십시오. (이는 항상 필요하지는 않습니다. 일부 환경에서는 두 상황에서 값의 문자열 양식을 사용할 수 있습니다. 그러나 C++ 프로그래밍 언어와의 호환성을 위해 필요하므로 선언의 배열 양식을 사용할 수 있습니다.)

## 동적 구조의 초기값

구조의 가변 인스턴스 수가 필요한 경우, 인스턴스는 일반적으로 calloc 또는 malloc 함수를 사용하여 동적으로 확보한 주 기억장치에서 작성됩니다. 해당 구조에서 필드를 초기화하려면 다음 기술을 고려하십시오.

1. 구조를 초기화하기 위해 적절한 MQxxx\_DEFAULT 매크로 변수를 사용하여 구조의 인스턴스를 선언하십시오. 이 인스턴스는 기타 인스턴스의 모델이 됩니다.

```
MQMD Model = {MQMD_DEFAULT}; /* declare model instance */
```

필요에 따라 모델 인스턴스에 정적 또는 동적 수명을 제공하기 위해, static 또는 auto 키워드를 선언에서 코드화할 수 있습니다.

2. calloc 또는 malloc 함수를 사용하여 구조의 동적 인스턴스에 대한 스토리지를 확보하십시오.

```
PMQMD Instance;
Instance = malloc(sizeof(MQMD)); /* get storage for dynamic instance */
```

3. memcpy 함수를 사용하여 모델 인스턴스를 동적 인스턴스에 복사하십시오.

```
memcpy(Instance,&Model,sizeof(MQMD)); /* initialize dynamic instance */
```

## C++에서 사용

C++ 프로그래밍 언어의 경우, 헤더 파일에는 C++ 컴파일러를 사용할 때만 포함되는 다음의 추가 명령문이 포함됩니다.

```
#ifdef __cplusplus
extern "C" {
#endif

/* rest of header file */

#ifdef __cplusplus
```

```
}  
#endif
```

## 표기법 규칙

이 정보에서는 함수를 호출하고 매개변수를 선언하는 방법을 표시합니다.

일부 경우, 매개변수는 고정되지 않은 크기의 배열입니다. 이 경우에는 숫자 상수를 표시하기 위해 소문자 n이 사용됩니다. 해당 매개변수에 대한 선언을 코드화하는 경우, n을 필요한 숫자 값으로 바꾸십시오.

## COBOL 프로그래밍

COBOL 프로그래밍 언어에서 MQI를 사용하는 데 도움이 되는 정보입니다.

- [263 페이지의 『COPY 파일』](#)
- [264 페이지의 『구조』](#)
- [264 페이지의 『포인터』](#)
- [265 페이지의 『이름 지정된 상수』](#)
- [265 페이지의 『표기법 규칙』](#)

## COPY 파일

다양한 COPY 파일이 MQI를 사용하는 COBOL 애플리케이션 프로그램의 작성에 도움이 되도록 제공됩니다. 이름 지정된 상수가 포함된 2개의 파일이 있으며, 각 구조마다 2개의 파일이 있습니다.

각 구조는 두 가지 양식(초기값이 있는 양식 및 초기값이 없는 양식)으로 제공됩니다.

- COBOL 프로그램의 WORKING-STORAGE SECTION에서 초기값으로 구조를 사용하십시오. 이는 접미부가 문자 V("Value"를 의미함)인 이름의 COPY 파일에 포함되어 있습니다.
- COBOL 프로그램의 LINKAGE SECTION에서 초기값 없이 구조를 사용하십시오. 이는 접미부가 문자 L("Linkage"를 의미함)인 이름의 COPY 파일에 포함되어 있습니다.

COPY 파일은 다음 표에 요약되어 있습니다. 나열된 모든 파일을 모든 환경에서 사용할 수는 없습니다.

표 465. COBOL COPY 파일		
파일(초기값이 있는)	파일(초기값이 없는)	컨텐츠
CMQAIRV	CMQAIRL	인증 정보 레코드
CMQBOV	CMQBOL	시작 옵션 구조
CMQCIHV	CMQCIHL	CICS 정보 헤더 구조
CMQCNOV	CMQCNOV	연결 옵션 구조
CMQDHV	CMQDHL	분산 헤더 구조
CMQDLHV	CMQDLHL	데드-레터 헤더 구조
CMQDXPV	CMQDXPL	데이터 변환 엑시트 매개변수 구조
CMQGMOV	CMQGMOL	메시지 가져오기 옵션 구조
CMQIIHV	CMQIIHL	IMS 정보 헤더 구조
CMQMDV	CMQMDL	메시지 디스크립터 구조
CMQMDEV	CMQMDEL	메시지 디스크립터 확장 구조
CMQMD1V	CMQMD1L	메시지 디스크립터 구조 버전 1
CMQODV	CMQODL	오브젝트 디스크립터 구조
CMQORV	CMQORL	오브젝트 레코드 구조

표 465. COBOL COPY 파일 (계속)		
파일(초기값이 있는)	파일(초기값이 없는)	컨텐츠
CMQPMOV	CMQPMOL	메시지 넣기 옵션 구조
CMQRFHV	CMQRFHL	규칙 및 형식화 헤더 구조
CMQRFH2V	CMQRFH2L	규칙 및 형식화 헤더 구조 버전 2
CMQRMHV	CMQRMHL	참조 메시지 헤더 구조
CMQRRV	CMQRRLL	응답 레코드 구조
CMQSCOV	CMQSCOL	TLS 구성 옵션
CMQTMV	CMQTML	트리거 메시지 구조
CMQTMCV	CMQTMCL	트리거 메시지 구조(문자 형식)
CMQTM2V	CMQTM2L	트리거 메시지 구조(문자 형식) 버전 2
CMQWIHV	CMQWIHL	작업 정보 헤더 구조
CMQXQHV	CMQXQHL	전송 큐 헤더 구조
CMQV	-	기본 MQI의 이름 지정된 상수
CMQXV	-	데이터 변환 엑시트에 대한 이름 지정된 상수
CMQMD2V	CMQMD2L	메시지 디스크립터 구조 버전 2

## 구조

COPY 파일에서, 각 구조 선언은 레벨-10 항목으로 시작합니다. 따라서 사용자는 레벨-01 선언을 코드화한 후에 구조 선언의 나머지에서 복사하기 위한 COPY 명령문을 사용하여 구조의 여러 인스턴스를 선언할 수 있습니다. 적절한 인스턴스를 참조하려면 IN 키워드를 사용하십시오.

```
* Declare two instances of MQMD
01 MY-MQMD.
   COPY CMQMDV.
01 MY-OTHER-MQMD.
   COPY CMQMDV.
*
* Set MSGTYPE field in MY-OTHER-MQMD
MOVE MQMT-REQUEST TO MQMD-MSGTYPE IN MY-OTHER-MQMD.
```

적절한 경계에서 구조를 맞추십시오. COPY 명령문을 사용하여 레벨-01 항목이 아닌 항목 다음의 구조를 포함하는 경우, 구조가 레벨-01 항목의 시작으로 부터 적절한 오프셋에서 시작하는지 확인하십시오. 대부분의 MQI 구조에서는 4바이트 맞추기가 필요합니다. 이에 대한 예외는 MQCNO, MQOD 및 MQPMO이며, 이는 IBM i에서 16바이트 맞추기가 필요합니다.

이 절에서는 구조의 필드 이름이 접두부 없이 표시됩니다. COBOL에서 필드 이름에는 구조 이름 및 하이픈이 결합된 접두부가 있습니다. 그러나 구조 이름이 숫자로 종료되는 경우(구조가 원래 구조의 두 번째 또는 후속 버전임을 표시함), 숫자가 접두부에서 생략됩니다. COBOL의 필드 이름은 대문자로 표시됩니다(필요하면 소문자 또는 대소문자 혼합을 사용할 수 있지만). 예를 들어, 412 페이지의 『MQMD - 메시지 디스크립터』에 설명된 *MsgType* 필드는 COBOL에서 MQMD-MSGTYPE이 됩니다.

V-접미부 구조는 모든 필드의 초기값으로 선언됩니다. 제공된 초기값과 다른 값을 원하는 해당 필드만 설정해야 합니다.

## 포인터

일부 구조는 해당 구조와 인접하지 않을 수 있는 선택적 데이터를 처리해야 합니다(예: MQOD 구조가 처리하는 MQOR 및 MQRR 레코드).

이 선택적 데이터를 처리하기 위해, 구조에는 포인터 데이터 유형으로 선언된 필드가 포함됩니다. 그러나 COBOL은 모든 환경에서 포인터 데이터 유형을 지원하지 않습니다. 이로 인해, 선택적 데이터는 구조의 시작으로부터의 데이터 오프셋을 포함하는 필드를 사용하여 처리될 수도 있습니다.

환경 간에 애플리케이션을 포팅하려면 의도된 모든 환경에서 포인터 데이터 유형이 사용 가능한지 확인하십시오. 그렇지 않으면, 포인터 필드 대신 오프셋 필드를 사용하여 애플리케이션이 선택적 데이터를 처리해야 합니다.

포인터가 지원되지 않는 해당 환경에서 포인터 필드를 적절한 길이의 바이트 문자열로서 선언하십시오(여기서 초기값은 모두 널인 바이트 문자열임). 오프셋 필드를 사용 중이면 이 초기값을 대체하지 마십시오.

## 이름 지정된 상수

이 정보에서는 이름의 일부로 밑줄 문자 (\_) 를 포함하는 상수의 이름이 표시됩니다. COBOL에서는 밑줄 대신 하이픈 문자(-)를 사용하십시오.

문자열 값이 있는 상수는 작은따옴표(')을 문자열 구분 기호로 사용합니다. 일부 환경에서는 컴파일러가 큰따옴표 대신 작은따옴표를 문자열 구분 기호로 허용하도록 적절한 컴파일러 옵션을 지정해야 할 수도 있습니다.

이름 지정된 상수는 COPY 파일에서 레벨-10 항목으로서 선언됩니다. 상수를 사용하려면 레벨-01 항목을 명시적으로 선언한 후에 COPY 명령문을 사용하여 상수의 선언에 복사하십시오.

```
* Declare a structure to hold the constants
01 MY-MQ-CONSTANTS.
   COPY CMQV.
```

선행 메소드는 참조되지 않은 경우에도 상수가 프로그램의 스토리지를 점유하도록 합니다. 동일한 실행 단위 내의 많은 개별 프로그램에 상수를 포함하는 경우에는 상수의 다중 사본이 존재하며, 주 기억장치를 불필요하게 사용합니다. 다음 기술 중 하나를 사용하여 이러한 영향을 피하십시오.

- GLOBAL 절을 레벨-01 선언에 추가하십시오.

```
* Declare a global structure to hold the constants
01 MY-MQ-CONSTANTS GLOBAL.
   COPY CMQV.
```

이는 실행 단위 내에서 하나의 상수 세트에 대해서만 스토리지를 할당합니다. 그러나 상수는 레벨-01 선언이 포함된 프로그램이 아닌 실행 단위 내의 프로그램에 의해 참조될 수 있습니다.

**참고:** GLOBAL 절이 모든 환경에서 지원되지는 않습니다.

- 해당 프로그램이 참조하는 해당 상수만 각 프로그램 내에 수동으로 복사하십시오. COPY 명령문을 사용하여 모든 상수를 프로그램에 복사하지 마십시오.

## 표기법 규칙

이 주제의 이전 절에서는 호출을 호출하고 매개변수를 선언하는 방법을 보여줍니다. 일부 경우, 매개변수는 해당 크기가 고정되지 않은 문자열 또는 테이블입니다. 이 경우에는 숫자 상수를 표시하기 위해 소문자 n이 사용됩니다. 해당 매개변수에 대한 선언을 코드화하는 경우, n을 필요한 숫자 값으로 바꾸십시오.

### High Level Assembler 프로그래밍

System/390 어셈블러 프로그래밍 언어에서 MQI를 사용하는 데 도움이 되는 정보.

- [265 페이지의 『매크로』](#)
- [266 페이지의 『구조』](#)
- [266 페이지의 『CMQVERA 매크로』](#)
- [267 페이지의 『표기법 규칙』](#)

## 매크로

이름 지정된 상수에 대해 두 개의 매크로가 있으며, 각각의 구조에 대해 하나의 매크로가 있습니다. 이러한 파일은 다음 표에 요약되어 있습니다.

표 466. 어셈블러 매크로

파일	컨텐츠
CMQA	기본 MQI의 이름 지정된 상수(equates)
CMQCIHA	CICS 정보 헤더 구조
CMQCNOA	연결 옵션 구조
CMQDLHA	데드-레터 헤더 구조
CMQDXPA	데이터 변환 엑시트 매개변수 구조
CMQGMOA	메시지 가져오기 옵션 구조
CMQIIHA	IMS 정보 헤더 구조
CMQMDA	메시지 디스크립터 구조
CMQMDEA	메시지 디스크립터 확장 구조
CMQODA	오브젝트 디스크립터 구조
CMQPMOA	메시지 넣기 옵션 구조
CMQRFHA	규칙 및 형식화 헤더 구조
CMQRFH2A	규칙 및 형식화 헤더 구조 버전 2
CMQRMHA	참조 메시지 헤더 구조
CMQTMA	트리거 메시지 구조
CMQTMCA	트리거 메시지 구조(문자 형식) 버전 2
CMQVERA	구조 버전 제어
CMQWIHA	작업 정보 헤더 구조
CMQXA	데이터 변환 엑시트에 대한 이름 지정된 상수
CMQXPA	API 교차 엑시트 매개변수 구조
CMQXQHA	전송 큐 헤더 구조

## 구조

구조는 매크로의 조치를 제어하기 위해 다양한 매개변수를 갖는 매크로에 의해 생성됩니다. [267 페이지의 『구조』](#)의 내용을 참조하십시오.

### CMQVERA 매크로

이 매크로를 사용하면 구조 매크로의 DCLVER 매개변수에 사용되는 기본값을 설정할 수 있습니다.

구조 매크로의 호출에서 DCLVER 매개변수를 생략한 경우에만 CMQVERA에서 지정한 값이 구조 매크로에 의해 사용됩니다. 기본값은 DCLVER 매개변수로 CMQVERA 매크로를 코딩하여 설정됩니다.

#### DCLVER=CURRENT

기본 버전이 현재(최신) 버전으로 설정됩니다.

#### DCLVER=SPECIFIED

기본 버전이 VERSION 매개변수에서 지정하는 버전으로 설정됩니다.

**DCLVER** 매개변수를 지정해야 하며, 값은 대문자여야 합니다. CMQVERA에서 설정하는 값은 CMQVERA의 다음 호출까지 또는 어셈블리의 끝까지 기본값을 유지합니다. CMQVERA를 생략하는 경우, 기본값은 DCLVER=CURRENT입니다.

## 표기법 규칙

기타 주제에서는 호출을 호출하고 매개변수를 선언하는 방법을 보여줍니다. 일부 경우, 매개변수는 고정되지 않은 크기의 문자열 또는 배열입니다. 이 경우에 소문자 n은 숫자 상수의 표시에 사용됩니다. 해당 매개변수에 대한 선언을 코드화하는 경우, n을 필요한 숫자 값으로 바꾸십시오.

### 구조

구조는 매크로의 조치를 제어하기 위해 다양한 매개변수를 갖는 매크로에 의해 생성됩니다.

**참고:** 때때로 IBM MQ 구조의 새 버전이 도입됩니다. 새 버전의 추가 필드는 이전에 256바이트보다 작은 구조가 256바이트보다 크게 되도록 할 수 있습니다. 이 때문에 IBM MQ 구조를 복사하거나 IBM MQ 구조를 널 (null) 로 설정하여 256바이트보다 큰 구조에 대해 올바르게 작동하도록 어셈블러 명령어를 작성하십시오. 또는 DCLVER 매크로 매개변수 또는 CMQVERA 매크로를 VERSION 매개변수와 함께 사용하여 구조의 특정 버전을 선언할 수 있습니다.

- [267 페이지의 『구조의 이름 지정』](#)
- [267 페이지의 『구조의 양식 지정』](#)
- [267 페이지의 『구조의 버전 제어』](#)
- [268 페이지의 『한 구조를 다른 구조 내에 임베드하여 선언』](#)
- [268 페이지의 『필드의 초기값 지정』](#)
- [268 페이지의 『목록 제어』](#)

## 구조의 이름 지정

구조의 둘 이상의 인스턴스를 선언하기 위해, 매크로는 구조의 각 필드의 이름에 사용자 특정 문자열 및 밑줄이 있는 접두부를 지정합니다.

사용된 문자열은 매크로의 호출에 지정된 레이블입니다. 레이블이 지정되지 않으면, 구조의 이름은 접두부를 구성하는 데 사용됩니다.

```
* Declare two object descriptors
      CMQODA ,          Prefix used="MQOD_" (the default)
MY_MQOD CMQODA ,          Prefix used="MY_MQOD_"
```

이 절에서 표시된 구조 선언은 기본 접두부를 사용합니다.

## 구조의 양식 지정

구조 선언은 DSECT 매개변수에 의해 제어되어 두 양식 중 하나로 매크로에 의해 생성될 수 있습니다,

### DSECT=YES

어셈블러 DSECT 명령어는 새 데이터 섹션을 시작하는 데 사용됩니다. 구조 정의 바로 다음에는 DSECT 명령어가 나옵니다. 매크로 호출의 레이블은 데이터 섹션의 이름으로 사용됩니다. 레이블이 지정되지 않은 경우 구조의 이름이 사용됩니다.

### DSECT=NO

어셈블러 DC 명령어는 루틴의 현재 위치에서 구조를 정의하는 데 사용됩니다. 필드는 매크로 호출에서 관련 매개변수를 코드화하여 지정될 수 있는 값으로 초기화됩니다. 매크로 호출에서 값이 지정되지 않은 필드는 기본값을 사용하여 초기화됩니다.

지정된 값은 대문자여야 합니다. DSECT 매개변수가 지정되지 않으면 DSECT=NO라고 가정합니다.

## 구조의 버전 제어

기본적으로, 매크로는 항상 각 구조의 최신 버전을 선언합니다.

VERSION 매크로 매개변수를 사용하여 구조의 *Version* 필드에 대한 값을 지정할 수 있음에도 불구하고, 해당 매개변수는 *Version* 필드의 초기값을 정의하며 실제로 선언된 구조의 버전을 제어하지 않습니다. 선언된 구조의 버전을 제어하려면 DCLVER 매개변수를 사용하십시오.

## DCLVER=CURRENT

선언된 버전이 현재(최신) 버전입니다.

## DCLVER=SPECIFIED

버전이 VERSION 매개변수에서 지정하는 버전입니다. VERSION 매개변수를 생략하면 기본값은 버전 1입니다.

VERSION 매개변수를 지정하는 경우, 값은 자체 정의하는 숫자 상수이거나 필요한 버전의 이름 지정된 상수여야 합니다(예: MQCNO\_VERSION\_3). 일부 기타 값을 지정하는 경우, VERSION의 값이 올바른 값으로 해석되어도 DCLVER=CURRENT가 지정된 것처럼 구조가 선언됩니다.

지정된 값은 대문자여야 합니다. DCLVER 매개변수를 생략하는 경우, 사용된 값은 MQDCLVER 글로벌 매크로 변수에서 가져옵니다. CMQVERA 매크로를 사용하여 이 변수를 설정할 수 있습니다.

## 한 구조를 다른 구조 내에 임베드하여 선언

한 구조를 다른 구조의 컴포넌트로 선언하려면 NESTED 매개변수를 사용하십시오.

### NESTED=YES

구조 선언이 다른 구조 선언 내에 중첩됩니다.

### NESTED=NO

구조 선언이 다른 구조 선언 내에 중첩되지 않습니다.

지정된 값은 대문자여야 합니다. NESTED 매개변수를 생략하면 NESTED=NO라고 가정됩니다.

## 필드의 초기값 지정

필수 값과 함께 매크로 호출의 매개변수로서 해당 필드의 이름(점두부 없는)을 코드화하여 구조의 필드를 초기화하는 데 사용되는 값을 지정하십시오.

예를 들어, *MsgType* 필드가 MQMT\_REQUEST로 초기화되고 *ReplyToQ* 필드가 문자열 "MY\_REPLY\_TO\_QUEUE"로 초기화된 메시지 디스크립터 구조를 선언하려면 다음을 사용하십시오.

```
MY_MQMD  CMQMDA  MSGTYPE=MQMT_REQUEST,          X
          REPLYTOQ=MY_REPLY_TO_QUEUE
```

매크로 호출의 값으로서 이름 지정된 상수(동일)를 지정하는 경우에는 CMQA 매크로를 사용하여 이름 지정된 상수를 정의하십시오. 문자열 값을 작은따옴표로 묶지 마십시오.

## 목록 제어

LIST 매개변수를 사용하여 어셈블러 목록에서 구조 선언의 모양을 제어합니다.

### LIST=YES

구조 선언이 어셈블러 목록에 나타납니다.

### LIST=NO

구조 선언이 어셈블러 목록에 나타나지 않습니다.


지정된 값은 대문자여야 합니다. LIST 매개변수를 생략하면 LIST=NO라고 가정됩니다.

## MQAIR - 인증 정보 레코드

MQAIR 구조를 사용하면 IBM MQ MQI client로서 실행 중인 애플리케이션이 클라이언트 연결에 사용되는 인증자에 대한 정보를 지정할 수 있습니다. 이 구조는 MQCONNX 호출의 입력 매개변수입니다.

## 가용성

MQAIR 구조는 다음 클라이언트에 사용 가능합니다.

-  AIX



-  Linux
-  Windows

## 문자 세트 및 인코딩

MQAIR의 데이터는 로컬 큐 관리자의 문자 세트 및 인코딩에 있어야 합니다. 이는 **CodedCharSetId** 큐 관리자 속성 및 MQENC\_NATIVE에서 제공됩니다.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 467. MQAIR의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQAIR_STRUC_ID	'AIR~'
버전 (구조 버전 번호)	MQAIR_VERSION_1	1
AuthInfo유형 (인증 정보 유형)	MQAIT_CRL_LDAP	1
AuthInfoConnName (LDAP CRL 서버의 연결 이름)	없음	널 문자열 또는 공백
LDAPUserNamePtr (LDAP 사용자 이름의 주소)	없음	널 포인터 또는 널 바이트
LDAPUserNameOffset (MQSCO의 시작에서 LDAP 사용자 이름의 오프셋)	없음	0
LDAPUserName길이 (LDAP 사용자 이름의 길이)	없음	0
LDAPPassword (LDAP 서버에 액세스하기 위한 암호)	없음	널 문자열 또는 공백
<b>참고:</b> <i>Version</i> 이 MQAIR_VERSION_2 미만인 경우 나머지 필드는 무시됩니다.		
OCSPResponderURL (OCSP 응답자가 접속할 수 있는 URL)	없음	널 문자열 또는 공백
<b>참고:</b> 1. ~ 기호는 단일 공백 문자를 나타냅니다. 2. C 프로그래밍 언어의 매크로 변수MQAIR_DEFAULT에는 표에 나열된 값이 포함됩니다. 구조의 필드에 초기 값을 제공하기 위해 다음 방법으로 사용하십시오.		
<pre>MQAIR MyAIR = {MQAIR_DEFAULT};</pre>		

## 언어 선언

MQAIR의 C 선언

```
typedef struct tagMQAIR MQAIR;
struct tagMQAIR {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG    Version;           /* Structure version number */
    MQLONG    AuthInfoType;      /* Type of authentication
                                information */
    MQCHAR264  AuthInfoConnName; /* Connection name of CRL LDAP
                                server */
    PMQCHAR    LDAPUserNamePtr;  /* Address of LDAP user name */
    MQLONG    LDAPUserNameOffset; /* Offset of LDAP user name from start
                                of MQAIR structure */
    MQLONG    LDAPUserNameLength; /* Length of LDAP user name */
    MQCHAR32   LDAPPassword;     /* Password to access LDAP server */
};
```

```

MQCHAR256 OCSPResponderURL; /* URL of OCSP responder */
};

```

## MQAIR의 COBOL 선언

```

** MQAIR structure
10 MQAIR.
** Structure identifier
15 MQAIR-STRUCID PIC X(4).
** Structure version number
15 MQAIR-VERSION PIC S9(9) BINARY.
** Type of authentication information
15 MQAIR-AUTHINFOTYPE PIC S9(9) BINARY.
** Connection name of CRL LDAP server
15 MQAIR-AUTHINFOCONNNAME PIC X(264).
** Address of LDAP user name
15 MQAIR-LDAPUSERNAMEPTR POINTER.
** Offset of LDAP user name from start of MQAIR structure
15 MQAIR-LDAPUSERNAMEOFFSET PIC S9(9) BINARY.
** Length of LDAP user name
15 MQAIR-LDAPUSERNAMELENGTH PIC S9(9) BINARY.
** Password to access LDAP server
15 MQAIR-LDAPPASSWORD PIC X(32).
** URL of OCSP responder
15 MQAIR-OCSPRESPONDERURL PIC X(256).

```

## MQAIR의 Visual Basic 선언

```

Type MQAIR
  StrucId          As String*4   'Structure identifier'
  Version          As Long       'Structure version number'
  AuthInfoType    As Long       'Type of authentication information'
  AuthInfoConnName As String*264 'Connection name of CRL LDAP server'
  LDAPUserNamePtr As MQPTR      'Address of LDAP user name'
  LDAPUserNameOffset As Long     'Offset of LDAP user name from start'
                                'of MQAIR structure'
  LDAPUserNameLength As Long     'Length of LDAP user name'
  LDAPPASSWORD    As String*32   'Password to access LDAP server'
End Type

```

### MQAIR의 경우 *StrucId* (MQCHAR4)

인증 정보 레코드 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQAIR\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

#### MQAIR\_STRUC\_ID

인증 정보 레코드의 ID.

C 프로그래밍 언어의 경우 상수 MQAIR\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQAIR\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### MQAIR의 버전 (MQLONG)

인증 정보 레코드 구조의 버전 번호입니다. 항상 입력 필드입니다.

값은 다음 중 하나여야 합니다.

#### MQAIR\_VERSION\_1

버전-1 인증 정보 레코드.

이는 이 필드의 초기값입니다.

#### MQAIR\_VERSION\_2

버전-2 인증 정보 레코드.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### MQAIR\_CURRENT\_VERSION

인증 정보 레코드의 현재 버전.

## **MQAIR의 AuthInfo유형 (MQLONG)**

레코드에 포함된 인증 정보의 유형입니다.

값은 두 개의 다음 매개변수 중 하나일 수 있습니다.

### **MQAIT\_CRL\_LDAP**

LDAP 서버를 사용한 인증서 폐기 검사.

### **MQAIT\_OCSP**

OCSP를 사용한 인증서 폐기 검사.

값이 올바르지 않으면 이유 코드 MQRC\_AUTH\_INFO\_TYPE\_ERROR로 호출이 실패합니다.

입력 필드입니다. 이 필드의 초기값은 MQAIT\_CRL\_LDAP입니다.

## **MQAIR의 AuthInfoConnName (MQCHAR264)**

이는 LDAP 서버가 실행 중인 호스트의 호스트 이름 또는 네트워크 주소입니다. 이 뒤에는 선택적인 포트 번호를 괄호로 묶어 표시할 수 있습니다. 기본 포트 번호는 389입니다.

값이 필드의 길이보다 짧으면 널 문자로 값을 종료하거나 필드 길이에 맞게 공백으로 채웁니다. 값이 올바르지 않으면 이유 코드 MQRC\_AUTH\_INFO\_CONN\_NAME\_ERROR로 호출이 실패합니다.

입력 필드입니다. 이 필드의 길이는 MQ\_AUTH\_INFO\_CONN\_NAME\_LENGTH에서 제공됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 공백 문자입니다.

## **MQAIR의 경우 LDAPUserNamePtr (PMQCHAR)**

LDAP 사용자 이름입니다.

LDAP CRL 서버에 액세스 시도 중인 사용자의 고유 이름으로 구성됩니다. 값이 *LDAPUserNameLength*에서 지정한 값보다 짧은 경우에는 값을 널 문자로 종료하거나 이를 *LDAPUserNameLength* 길이까지 공백으로 채우십시오. *LDAPUserNameLength*가 0인 경우 필드가 무시됩니다.

다음 두 가지 방법 중 하나에 LDAP 사용자 이름을 제공할 수 있습니다.

- 포인터 필드 *LDAPUserNamePtr*를 사용하여

이 경우에 애플리케이션은 MQAIR 구조와는 별도로 문자열을 선언할 수 있으며, *LDAPUserNamePtr*를 문자열의 주소로 설정할 수 있습니다.

상이한 환경(예: C 프로그래밍 언어)에 포팅이 가능한 방식으로 포인터 데이터 유형을 지원하는 프로그래밍 언어의 경우에는 *LDAPUserNamePtr*의 사용을 고려하십시오.

- 오프셋 필드 *LDAPUserNameOffset*를 사용하여

이 경우에 애플리케이션은 MQSCO 구조, MQAIR 레코드의 배열 및 LDAP 사용자 이름 문자열을 차례로 포함하는 복합 구조를 선언해야 하며, *LDAPUserNameOffset*를 MQAIR 구조의 시작으로부터 적절한 이름 문자열의 오프셋으로 설정해야 합니다. 이 값이 정확하고 MQLONG 내에 수용할 수 있는 값이 있는지 확인하십시오 (가장 제한적 프로그래밍 언어가 COBOL이며 올바른 범위는 -999 999 999 - +999 999 999임).

포인터 데이터 유형을 지원하지 않거나 상이한 환경에 포팅이 가능하지 않은 방식으로 포인터 데이터 유형을 구현하는 프로그래밍 언어(예: COBOL 프로그래밍 언어)의 경우에는 *LDAPUserNameOffset*의 사용을 고려하십시오.

어떤 기술을 선택하든 *LDAPUserNamePtr* 및 *LDAPUserNameOffset* 중 하나만 사용하십시오. 둘 모두가 0이 아니면 호출이 이유 코드 MQRC\_LDAP\_USER\_NAME\_ERROR로 실패합니다.

입력 필드입니다. 이 필드의 초기값은 포인터를 지원하는 프로그래밍 언어로 된 널 포인터이며, 그렇지 않은 경우 모두 널 바이트 문자열입니다.

**참고:** 프로그래밍 언어가 포인터 데이터 유형을 지원하지 않는 플랫폼에서 이 필드는 적절한 길이의 바이트 문자열로 선언됩니다.

## **LDAPUserNameMQAIR의 오프셋 (MQLONG)**

MQAIR 구조의 시작부터 LDAP 사용자 이름의 오프셋(바이트)입니다.

오프셋은 양수 또는 음수일 수 있습니다. *LDAPUserNameLength*이(가) 0인 경우 필드는 무시됩니다.

*LDAPUserNamePtr* 또는 *LDAPUserNameOffset*을 사용하여 LDAP 사용자 이름을 지정할 수 있지만 둘 다 지정할 수 없습니다. 세부사항은 *LDAPUserNamePtr* 필드에 대한 설명을 참조하십시오.

입력 필드입니다. 이 필드의 초기값은 0입니다.

### **MQAIR의 LDAPUserName길이 (MQLONG)**

이는 *LDAPUserNamePtr* 또는 *LDAPUserNameOffset* 필드에서 처리하는 LDAP 사용자 이름의 길이(바이트)입니다.

값은 0 - MQ\_DISTINGUISHED\_NAME\_LENGTH 범위에 있어야 합니다. 값이 올바르지 않으면 이유 코드 MQRC\_LDAP\_USER\_NAME\_LENGTH\_ERR로 호출이 실패합니다.

포함된 LDAP 서버에 사용자 이름이 필요하지 않은 경우 이 필드를 0으로 설정하십시오.

입력 필드입니다. 이 필드의 초기값은 0입니다.

### **MQAIR의 LDAPPASSWORD (MQCHAR32)**

이는 LDAP CRL 서버에 액세스하는 데 필요한 비밀번호입니다. 값이 필드 길이보다 짧으면 널 문자로 값을 종료하거나 필드 길이를 공백으로 채웁니다.

LDAP 서버에서 비밀번호가 필요하지 않거나 LDAP 사용자 이름이 생략된 경우, *LDAPPASSWORD*는 널 또는 공백이어야 합니다. LDAP 사용자 이름이 생략되거나 *LDAPPASSWORD*가 널 또는 공백이 아닌 경우, 이유 코드 MQRC\_LDAP\_PASSWORD\_ERROR로 호출이 실패합니다.

입력 필드입니다. 이 필드의 길이는 MQ\_LDAP\_PASSWORD\_LENGTH에서 제공됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 공백 문자입니다.

### **MQAIR의 경우 OCSPResponderURL (MQCHAR256)**

OCSP 응답자에 대한 연결 세부사항을 표시하는 MQAIR 구조의 경우, 이 필드에는 응답자가 연결될 수 있는 URL이 포함됩니다.

이 필드의 값은 HTTP URL입니다. 이 필드는 AuthorityInfoAccess(AIA) 인증서 확장의 URL에 우선합니다.

다음 두 문장 모두가 참인 경우가 아니면 값이 무시됩니다.

- MQAIR 구조가 버전 2 이상입니다(버전 필드가 MQAIR\_VERSION\_2 이상으로 설정됨).
- AuthInfoType 필드가 MQAIT\_OCSP로 설정됩니다.

필드에 올바른 형식의 HTTP URL이 포함되지 않은 경우(그리고 무시되지 않는 경우), MQCONNX 호출이 이유 코드 MQRC\_OCSP\_URL\_ERROR로 실패합니다.

이 필드는 대소문자를 구분합니다. 이는 소문자로 된 http:// 문자열로 시작해야 합니다. URL의 나머지는 OCSP 서버 구현에 따라 대소문자가 구분될 수 있습니다.

이 필드는 데이터 변환에 종속되지 않습니다.

### **MQBMHO - 버퍼 대 메시지 핸들 옵션**

MQBMHO 구조를 사용하여 애플리케이션이 메시지 핸들이 버퍼에서 생성되는 방법을 제어하는 옵션을 지정할 수 있습니다. 구조는 MQBUFMH 호출의 입력 매개변수입니다.

### **문자 세트 및 인코딩**

MQBMHO의 데이터는 애플리케이션의 문자 세트 및 애플리케이션의 인코딩 (MQENC\_NATIVE)에 있어야 합니다.

### **필드**

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 468. MQBMHO의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQBMHO_STRUC_ID	'BMHO'
버전 (구조 버전 번호)	MQBMHO_VERSION_1	1
옵션 (MQBMHO의 조치를 제어하는 옵션)	MQBMHO_NONE	0
<p><b>참고:</b></p> <p>1. C 프로그래밍 언어의 매크로 변수MQBMHO_DEFAULT에는 표에 나열된 값이 포함됩니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오.</p> <pre>MQBMHO MyBMHO = {MQBMHO_DEFAULT};</pre>		

## 언어 선언

### MQBMHO의 C 선언

```
typedef struct tagMQBMHO MQBMHO;
struct tagMQBMHO {
    MQCHAR4  StrucId;          /* Structure identifier */
    MQLONG   Version;         /* Structure version number */
    MQLONG   Options;         /* Options that control the action of
                               MQBUFMH */
};
```

### MQBMHO의 COBOL 선언

```
** MQBMHO structure
10 MQBMHO.
** Structure identifier
15 MQBMHO-STRUCID PIC X(4).
** Structure version number
15 MQBMHO-VERSION PIC S9(9) BINARY.
** Options that control the action of MQBUFMH
15 MQBMHO-OPTIONS PIC S9(9) BINARY.
```

### MQBMHO의 PL/I 선언

```
Dcl
1 MQBMHO based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 Options fixed bin(31), /* Options that control the action
of MQBUFMH */
```

### MQBMHO의 상위 레벨 어셈블러 선언

```
MQBMHO DSECT
MQBMHO_STRUCID DS CL4 Structure identifier
MQBMHO_VERSION DS F Structure version number
MQBMHO_OPTIONS DS F Options that control the
action of MQBUFMH
*
MQBMHO_LENGTH EQU *-MQBMHO
MQBMHO_AREA DS CL(MQBMHO_LENGTH)
```

## MQBMHO의 MQBMHO에 대한 StrucId (MQCHAR4)

메시지 핸들 구조에 대한 버퍼의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQBMHO\_STRUC\_ID입니다. 값은 다음과 같아야 합니다.

## MQBMHO\_STRUC\_ID

버퍼 대 메시지 핸들 구조의 ID.

C 프로그래밍 언어의 경우 상수 MQBMHO\_STRUC\_ID\_ARRAY도 정의됩니다. 이 값은 MQBMHO\_STRUC\_ID와 동일하지만 문자열이 아닌 문자의 배열입니다.

## MQBMHO용 MQBMHO의 버전 (MQLONG)

메시지 핸들 구조에 대한 버퍼의 버전 번호입니다. 항상 입력 필드입니다.

값은 다음과 같아야 합니다.

## MQBMHO\_VERSION\_1

메시지 핸들 구조에 대한 버퍼의 버전 번호입니다.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

## MQBMHO\_CURRENT\_VERSION

메시지 핸들 구조에 대한 버퍼의 현재 버전.

## MQBMHO의 옵션 (MQLONG)

버퍼 대 메시지 핸들 구조 - 옵션 필드

가능한 값은 다음과 같습니다.

## MQBMHO\_DELETE\_PROPERTIES

메시지 핸들에 추가되는 특성이 메시지 버퍼에서 삭제됩니다. 호출이 실패하면 어떤 특성도 삭제되지 않습니다.

기본 옵션: 설명한 옵션 중 필요한 옵션이 없는 경우 다음 옵션을 사용하십시오.

## MQBMHO\_NONE

옵션이 지정되지 않았습니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQBMHO\_DELETE\_PROPERTIES입니다.

## MQBNO - 밸런싱 옵션

다음 표에는 구조의 필드가 요약되어 있습니다.

## 필드

참고: 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 469. MQBNO의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQBNO_STRUC_ID	'BNO-'
버전 (구조 버전 번호)	MQBNO_VERSION_1	1
ApplicationType (구조에 설정된 밸런싱 옵션의 유형)	MQBNO_VALTYPE_SIMPLE	0
제한시간 (리밸런싱이 애플리케이션 활동을 인터럽트할 수 있는 제한시간)	MQBNO_TIMEOUT_AS_DEFAULT	0
BalanceOptions (발행 애플리케이션에서 설정한 밸런싱 옵션)	MQBNO_OPTIONS_NONE	0

표 469. MQBNO의 필드 (계속)

필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
<p><b>참고:</b></p> <ol style="list-style-type: none"> <li>↵ 기호는 단일 공백 문자를 나타냅니다.</li> <li>C 프로그래밍 언어에서 매크로 변수 MQBNO_DEFAULT에는 테이블에 나열된 값이 포함됩니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오.</li> </ol> <pre>MQBNO MyBNO = {MQBNO_DEFAULT};</pre>		

## 언어 선언

### MQBNO에 대한 C 선언

```
typedef struct tagMQBNO MQBNO;
struct tagMQBNO {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;          /* Structure version number */
    MQLONG     Type;             /* Type of balancing options set in the
                                structure */
    MQLONG     Timeout;          /* Timeout after which re-balancing might
                                interrupt application activity */
    MQLONG     BalanceOptions;   /* Balancing options set by the issuing
                                application */
};
```

### MQBNO에 대한 COBOL 선언

```
** MQBNO structure
10 MQBNO.
** Structure identifier
15 MQBNO-STRUCID PIC X(4).
** Structure version number
15 MQBNO-VERSION PIC S9(9) BINARY.
** Type of balancing options set in the structure
15 MQBNO-TYPE PIC S9(9) BINARY.
** Timeout after which re-balancing might interrupt application activity
15 MQBNO-TIMEOUT PIC S9(9) BINARY.
** Balancing options set by the issuing application
15 MQBNO-BALANCEOPTIONS PIC S9(9) BINARY.
```

### MQBNO에 대한 PL/I 선언

```
dcl
1 MQBNO based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 Type fixed bin(31), /* Type of balancing options set in the
                      structure*/
3 Timeout fixed bin(31), /* Timeout after which re-balancing might
                          interrupt application activity */
3 BalanceOptions fixed bin(31), /* Balancing options set by the issuing
                                application*/
```

## 관련 참조

312 페이지의 『MQCNO - 연결 옵션』

MQCNO 구조를 사용하면 애플리케이션이 큐 관리자에 대한 연결과 관련된 옵션을 지정할 수 있습니다. 구조는 MQCONNХ 호출의 입/출력 매개변수입니다.

### MQBNO의 경우 StrucId (MQCHAR4)

이는 밸런싱 옵션 구조의 구조 ID입니다. 항상 입력 필드입니다. 초기 값은 BNO입니다.

값은 다음과 같아야 합니다.

#### **BNO**

밸런싱 옵션 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQBNO\_STRUC\_ID\_ARRAY도 정의됩니다. 이 상수는 BNO와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

**StrucId**에 올바른 값을 제공해야 합니다. 그렇지 않으면 MQRC\_BNO\_ERROR가 리턴됩니다.

#### **MQBNO의 버전 (MQLONG)**

이는 밸런싱 옵션 구조의 버전 번호입니다. 항상 입력 필드입니다.

값은 다음과 같아야 합니다.

#### **MQBNO\_VERSION\_1**

밸런싱 옵션 구조의 버전 번호입니다.

**Version** 또는 MQRC\_BNO\_ERROR에 유효한 값을 제공해야 합니다.

#### **MQBNO의 경우 ApplicationType (MQLONG)**

구조에 설정된 밸런싱 옵션의 유형입니다.

가능한 값은 다음과 같습니다.

#### **MQBNO\_BALTYPE\_SIMPLE**

단순 밸런싱. 균일한 클러스터에서 애플리케이션 리밸런싱에 미치는 영향에 설명된 규칙 외에 특정 규칙이 적용되지 않습니다.

#### **MQBNO\_BALTYPE\_REQREP**

요청-응답 밸런싱. 각 MQPUT 호출 후 응답 메시지에 대해 일치하는 MQGET 호출이 예상됩니다. 이러한 메시지가 수신되거나 요청 메시지 EXPIRY가 초과될 때까지 밸런싱이 지연됩니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQBNO\_BALTYPE\_SIMPLE입니다.

**ApplicationType** 필드에 대해 값 하나만 제공해야 합니다. 그렇지 않으면 MQRC\_BNO\_ERROR가 리턴됩니다.

**참고:** 이 MQBNO\_BALTYPE\_RA\_MANAGED 필드의 추가 값은 IBM MQ Resource Adapter for JEE 환경에서 사용하도록 예약되어 있습니다. 애플리케이션에서 이 값을 직접 제공하는 것은 오류이지만, 예를 들어 애플리케이션 상태를 조회할 때 보고될 수 있습니다.

#### **MQBNO에 대한 제한시간 (MQLONG)**

**Timeout** 이후에는 재조정이 애플리케이션 활동을 인터럽트할 수 있습니다.

가능한 값은 다음과 같습니다.

#### **MQBNO\_TIMEOUT\_AS\_DEFAULT**

설정된 기본 제한시간 값입니다.

#### **MQBNO\_TIMEOUT\_IMMEDIATE**

즉시 제한시간이 초과됩니다.

#### **MQBNO\_TIMEOUT\_NEVER**

제한시간이 초과되지 않습니다.

이 필드의 초기값은 MQBNO\_TIMEOUT\_AS\_DEFAULT입니다.

**Timeout** 필드에 대해 정의된 값에서 하나의 값만 제공하거나 0-999999999초의 값을 제공해야 합니다. 그렇지 않으면 MQRC\_BNO\_ERROR가 리턴됩니다.

#### **MQBNO의 경우 BalanceOptions (MQLONG)**

발행 애플리케이션에서 설정한 밸런싱 옵션

가능한 값은 다음과 같습니다.

#### **MQBNO\_OPTIONS\_NONE**

옵션이 설정되지 않음



## MQBNO\_OPTIONS\_IGNORE\_TRANS

이 옵션을 설정하면 트랜잭션 중간에 있는 경우에도 애플리케이션을 다시 밸런싱할 수 있습니다.

이 필드의 초기값은 MQBNO\_OPTIONS\_NONE입니다.

**BalanceOptions** 필드의 논리 또는 문자를 사용하여 정의된 값의 조합을 제공할 수 있습니다. 값이 올바르지 않으면 MQRC\_BNO\_ERROR가 리턴됩니다.

## MQBO - 시작 옵션

MQBO 구조를 사용하여 애플리케이션이 작업 단위의 작성에 관하여 옵션을 지정할 수 있습니다. 구조는 MQBEGIN 호출의 입/출력 매개변수입니다.

## 가용성

MQBO 구조는 다음 플랫폼에서 사용 가능합니다.

- ▶ **AIX** AIX
- ▶ **IBM i** IBM i
- ▶ **Linux** Linux
- ▶ **Windows** Windows

IBM MQ MQI clients의 경우 MQBO 구조를 사용할 수 없습니다.

## 문자 세트 및 인코딩

MQBO의 데이터는 MQENC\_NATIVE로 지정된 로컬 큐 관리자의 인코딩 및 **CodedCharSetId** 큐 관리자 속성으로 지정된 문자 세트에 있어야 합니다. 그러나 애플리케이션이 MQ MQI 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트와 인코딩으로 작성되어야 합니다.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 470. MQBO에 대한 MQBO의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQBO_STRUC_ID	'BO~'
버전 (구조 버전 번호)	MQBO_VERSION_1	1
옵션 (MQBEGIN의 조치를 제어하는 옵션)	MQBO_NONE	0

**참고:**

- ~ 기호는 단일 공백 문자를 나타냅니다.
- C 프로그래밍 언어의 매크로 변수MQBO\_DEFAULT에는 표에 나열된 값이 포함됩니다. 구조의 필드에 초기 값을 제공하기 위해 다음 방법으로 사용하십시오.

```
MQBO MyBO = {MQBO_DEFAULT};
```

## 언어 선언

MQBO의 C 선언

```
typedef struct tagMQBO MQBO;  
struct tagMQBO {
```

```

MQCHAR4  StrucId; /* Structure identifier */
MQLONG   Version; /* Structure version number */
MQLONG   Options; /* Options that control the action of MQBEGIN */
};

```

## MQBO의 COBOL 선언

```

** MQBO structure
10 MQBO.
** Structure identifier
15 MQBO-STRUCID PIC X(4).
** Structure version number
15 MQBO-VERSION PIC S9(9) BINARY.
** Options that control the action of MQBEGIN
15 MQBO-OPTIONS PIC S9(9) BINARY.

```

## MQBO의 PL/I 선언

```

dcl
1 MQBO based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 Options fixed bin(31); /* Options that control the action of
MQBEGIN */

```

## MQBO의 Visual Basic 선언

```

Type MQBO
StrucId As String*4 'Structure identifier'
Version As Long 'Structure version number'
Options As Long 'Options that control the action of MQBEGIN'
End Type

```

### MQBO의 경우 *StrucId* (MQCHAR4)

시작 옵션 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQBO\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

#### MQBO\_STRUC\_ID

시작 옵션 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQBO\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQBO\_STRUC\_ID와 동일한 값을 갖지만 문자열이 아닌 문자의 배열입니다.

### MQBO의 버전 (MQLONG)

시작 옵션 구조의 버전 번호입니다. 항상 입력 필드입니다.

값은 다음과 같아야 합니다.

#### MQBO\_VERSION\_1

시작 옵션 구조의 버전 번호입니다.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### MQBO\_CURRENT\_VERSION

시작 옵션 구조의 현재 버전입니다.

### MQBO에 대한 옵션 (MQLONG)

이 필드는 항상 입력 필드입니다. 초기값은 MQBO\_NONE입니다.

값은 다음과 같아야 합니다.

#### MQBO\_NONE

옵션이 지정되지 않았습니니다.

## MQCBC - 콜백 컨텍스트

MQCBC 구조는 콜백 함수에 전달되는 컨텍스트 정보를 지정하는 데 사용됩니다. 이 구조는 메시지 이용자 루틴에 대한 호출의 입출력(I/O) 매개변수입니다.

### 가용성

MQCBC 구조는 다음 플랫폼에서 사용 가능합니다.

-  AIX
-  IBM i
-  Linux
-  Windows
-  z/OS

및 해당 시스템에 연결된 IBM MQ MQI clients

### 버전

MQCBC의 현재 버전은 MQCBC\_VERSION\_2입니다.

### 문자 세트 및 인코딩

MQCBC의 데이터는 MQENC\_NATIVE로 지정된 로컬 큐 관리자의 인코딩 및 **CodedCharSetId** 큐 관리자 속성으로 지정된 문자 세트에 있어야 합니다. 그러나 애플리케이션이 MQ MQI 클라이언트로 실행 중인 경우에는 구조가 클라이언트의 문자 세트 및 인코딩에 있습니다.

### 필드

**MQCBC** 구조의 초기값은 없습니다. 구조는 콜백 루틴에 매개변수로서 전달됩니다. 큐 관리자는 구조를 초기화합니다. 애플리케이션은 이를 절대 초기화하지 않습니다.

#### 참고:

- 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.
- MQCBC 구조에 대한 초기값은 없습니다. 구조는 콜백 루틴에 매개변수로서 전달됩니다. 큐 관리자는 구조를 초기화합니다. 애플리케이션은 이를 절대 초기화하지 않습니다.

표 471. MQCBC의 필드	
필드	설명
<a href="#">StrucID</a>	구조 ID
<a href="#">Version</a>	구조 버전 번호
<a href="#">CallType</a>	함수가 호출된 이유
<a href="#">Hobj</a>	오브젝트 핸들
<a href="#">CallbackArea</a>	사용할 콜백 함수의 필드
<a href="#">ConnectionArea</a>	사용할 콜백 함수의 필드
<a href="#">CompCode</a>	완료 코드
<a href="#">Reason</a>	이유 코드
<a href="#">상태</a>	현재 이용자의 상태의 표시
<a href="#">DataLength</a>	메시지 길이

표 471. MQCBC의 필드 (계속)	
필드	설명
BufferLength	메시지 버퍼의 길이(바이트)
플래그	일반 플래그
<b>참고:</b> 나머지 필드는 버전이 MQCBC_VERSION_2 미만인 경우 무시됨	
ReconnectDelay	재연결 시도 전의 시간(밀리초)

## 언어 선언

### MQCBC의 C 선언

```
typedef struct tagMQCBC MQCBC;
struct tagMQCBC {
    MQCHAR4    StructId;           /* Structure identifier */
    MQLONG     Version;           /* Structure version number */
    MQLONG     CallType;         /* Why Function was called */
    MQHOBJS    Hobj;            /* Object Handle */
    MQPTR      CallbackArea;     /* Callback data passed to the function */
    MQPTR      ConnectionArea;   /* MQCTL data area passed to the function */
    MQLONG     CompCode;        /* Completion Code */
    MQLONG     Reason;          /* Reason Code */
    MQLONG     State;           /* Consumer State */
    MQLONG     DataLength;      /* Message Data Length */
    MQLONG     BufferLength;     /* Buffer Length */
    MQLONG     Flags;           /* Flags containing information about
                                this consumer */
    /* Ver:1 */
    MQLONG     ReconnectDelay;   /* Number of milliseconds before */
    /* Ver:2 */ };              /* reconnect attempt */
```

### MQCBC의 COBOL 선언

```
** MQCBC structure
10  MQCBC.
** Structure Identifier
15  MQCBC-STRUCID                PIC X(4).
** Structure Version
15  MQCBC-VERSION              PIC S9(9) BINARY.
** Call Type
15  MQCBC-CALLTYPE             PIC S9(9) BINARY.
** Object Handle
15  MQCBC-HOBJ                 PIC S9(9) BINARY.
** Callback User Area
15  MQCBC-CALLBACKAREA        POINTER
** Connection Area
15  MQCBC-CONNECTIONAREA      POINTER
** Completion Code
15  MQCBC-COMPCODE            PIC S9(9) BINARY.
** Reason Code
15  MQCBC-REASON              PIC S9(9) BINARY.
** Consumer State
15  MQCBC-STATE               PIC S9(9) BINARY.
** Data Length
15  MQCBC-DATALENGTH          PIC S9(9) BINARY.
** Buffer Length
15  MQCBC-BUFFERLENGTH        PIC S9(9) BINARY.
** Flags
15  MQCBC-FLAGS               PIC S9(9) BINARY.
** Ver:1 **
** Number of milliseconds before reconnect attempt
15  MQCBC-RECONNECTDELAY      PIC S9(9) BINARY.
** Ver:2 **
```

### MQCBC의 PL/I 선언

```
dc1
```

```

1 MQCBC based,
3 StructId          char(4),          /* Structure identifier */
3 Version           fixed bin(31), /* Structure version */
3 CallType          fixed bin(31), /* Callback type */
3 Hobj              fixed bin(31), /* Object Handle */
3 CallbackArea      pointer,         /* User area passed to the function */
3 ConnectionArea    pointer,         /* Connection User Area */
3 CompCode          fixed bin(31); /* Completion Code */
3 Reason            fixed bin(31); /* Reason Code */
3 State             fixed bin(31); /* Consumer State */
3 DataLength        fixed bin(31); /* Message Data Length */
3 BufferLength       fixed bin(31); /* Message Buffer length */
3 Flags             fixed bin(31); /* Consumer Flags */
/* Ver:1 */
3 ReconnectDelay    fixed bin(31); /* Number of milliseconds before */
/* Ver:2 */
/* reconnect attempt */

```

## MQCBC의 상위 레벨 어셈블러 선언

```

MQCBC          DSECT
MQCBC          DS 0F      Force fullword alignment
MQCBC_STRUCID  DS CL4    Structure identifier
MQCBC_VERSION  DS F      Structure version number
MQCBC_CALLTYPE DS F      Why Function was called
MQCBC_HOBJ     DS F      Object Handle
MQCBC_CALLBACKAREA DS A  Callback data passed to the function
MQCBC_CONNECTIONAREA DS A  MQCTL Data area passed to the function
MQCBC_COMPCODE DS F      Completion Code
MQCBC_REASON   DS F      Reason Code
MQCBC_STATE    DS F      Consumer State
MQCBC_DATALENGTH DS F    Message Data Length
MQCBC_BUFFERLENGTH DS F  Buffer Length
MQCBC_FLAGS    DS F      Flags containing information about this consumer
MQCBC_RECONNECTDELAY DS F  Number of milliseconds before reconnect
MQCBC_LENGTH   EQU *-MQCBC
               ORG      MQCBC
MQCBC_AREA     DS CL(MQCBC_LENGTH)

```

### MQCBC의 경우 StructId (MQCHAR4)

콜백 컨텍스트 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQCBC\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

#### MQCBC\_STRUC\_ID

콜백 컨텍스트 구조의 ID.

C 프로그래밍 언어의 경우 상수 MQCBC\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQCBC\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### MQCBC의 버전 (MQLONG)

콜백 컨텍스트 구조의 버전 번호입니다. 항상 입력 필드입니다.

값은 다음과 같아야 합니다.

#### MQCBC\_VERSION\_1

버전 1 콜백 컨텍스트 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### MQCBC\_CURRENT\_VERSION

콜백 컨텍스트 구조의 현재 버전.

콜백 함수에는 항상 구조의 최신 버전이 전달됩니다.

### MQCBC의 경우 CallType (MQLONG)

이 기능이 호출된 이유에 대한 정보가 포함된 필드이며, 다음 값이 정의됩니다.

메시지 전달 호출 유형: 이러한 호출 유형은 메시지에 대한 정보를 포함합니다. **DataLength** 및 **BufferLength** 매개변수는 이러한 호출 유형에 대해 유효합니다.

### **MQCBCT\_MSG\_REMOVED**

메시지 이용자 함수가 오브젝트 핸들에서 파괴적으로 제거된 메시지로 호출되었습니다.

*CompCode*의 값이 MQCC\_WARNING인 경우, *Reason* 필드의 값은 MQRC\_TRUNCATED\_MSG\_ACCEPTED 또는 데이터 변환 문제점을 표시하는 코드 중 하나입니다.

### **MQCBCT\_MSG\_NOT\_REMOVED**

메시지 이용자 함수가 오브젝트 핸들에서 아직 파괴적으로 제거되지 않은 메시지로 호출되었습니다. 메시지는 *MsgToken*을 사용하여 오브젝트 핸들에서 파괴적으로 제거될 수 있습니다.

다음 이유 때문에 메시지가 제거되지 않았을 수 있습니다.

- MQGMO 옵션이 찾아보기 조작, MQGMO\_BROWSE\_\*를 요청했습니다.
- 메시지가 사용 가능한 버퍼보다 크며, MQGMO 옵션이 MQGMO\_ACCEPT\_TRUNCATED\_MSG를 지정하지 않습니다.

*CompCode*의 값이 MQCC\_WARNING인 경우, *Reason* 필드의 값은 MQRC\_TRUNCATED\_MSG\_FAILED 또는 데이터 변환 문제점을 표시하는 코드 중 하나입니다.

콜백 제어 호출 유형: 이러한 호출 유형은 콜백의 제어에 대한 정보를 포함하고 메시지에 대한 세부사항은 포함하지 않습니다. 이러한 호출 유형은 MQCBD 구조에서 [Options](#)을 사용하여 요청됩니다.

**DataLength** 및 **BufferLength** 매개변수는 이러한 호출 유형에 대해 유효하지 않습니다.

### **MQCBCT\_REGISTER\_CALL**

이 호출 유형의 목적은 콜백 함수가 일부 초기 설정을 수행할 수 있도록 하는 것입니다.

콜백 함수는 콜백이 등록된 후에 즉시 호출됩니다(즉, MQOP\_REGISTER의 *Operation* 필드에 대한 값을 사용하여 MQCB 호출에서 리턴될 때).

이 호출 유형은 메시지 이용자와 이벤트 핸들러 모두에 사용됩니다.

요청받은 경우 이는 콜백 함수의 첫 번째 호출입니다.

*Reason* 필드의 값은 MQRC\_NONE입니다.

### **MQCBCT\_START\_CALL**

이 호출 유형의 목적은 시작할 때 콜백 함수가 일부 설정을 수행할 수도록 허용하는 것입니다(예를 들어, 이전에 중지되었을 때 정리한 자원 복원).

콜백 함수는 MQOP\_START 또는 MQOP\_START\_WAIT 중 하나를 사용하여 연결이 시작될 때 호출됩니다.

콜백 함수가 다른 콜백 함수 내에 등록되는 경우 콜백이 리턴할 때 이 호출 유형이 호출됩니다.

이 호출 유형은 메시지 이용자에만 사용됩니다.

*Reason* 필드의 값은 MQRC\_NONE입니다.

### **MQCBCT\_STOP\_CALL**

이 호출 유형의 목적은 잠시 중지되었을 때 콜백 함수가 일부 설정을 수행하도록 허용하는 것입니다(예를 들어, 메시지 이용 중 가져온 추가 자원 정리).

콜백 함수는 MQOP\_STOP의 *Operation* 필드에 대한 값을 사용하여 MQCTL 호출이 실행될 때 호출됩니다.

이 호출 유형은 메시지 이용자에만 사용됩니다.

*Reason* 필드의 값은 중지의 이유를 표시하기 위해 설정됩니다.

### **MQCBCT\_DEREGISTER\_CALL**

이 호출 유형의 목적은 이용 프로세스의 마지막 부분에서 콜백 함수가 마지막 정리를 수행할 수 있도록 하는 것입니다. 콜백 함수는 다음일 때 호출됩니다.

- 콜백 함수는 MQOP\_DEREGISTER의 MQCB 호출을 사용하여 등록 취소됩니다.
- 큐가 닫혀서 암시적 등록 취소가 발생합니다. 이 인스턴스에서 콜백 함수에는 오브젝트 핸들로서 MQHO\_UNUSABLE\_HOBJ가 전달됩니다.
- MQDISC 호출 완료 - 암시적 닫기를 유발하므로 등록 취소가 발생합니다. 이 경우에는 연결이 즉시 끊기지 않으며, 진행 중인 트랜잭션은 아직 커밋되지 않습니다.

이러한 조치를 콜백 함수 자체 내에서 가져온 경우 콜백이 리턴하면 조치가 호출됩니다.

이 호출 유형은 메시지 이용자와 이벤트 핸들러 모두에 사용됩니다.

요청받은 경우 이는 콜백 함수의 마지막 호출입니다.

*Reason* 필드의 값은 중지의 이유를 표시하기 위해 설정됩니다.

### **MQCBCT\_EVENT\_CALL**

#### **이벤트 핸들러 함수**

큐 관리자 또는 연결이 중지되거나 정지될 때 이벤트 핸들러 함수가 메시지 없이 호출되었습니다.

이 호출은 모든 콜백 함수에 대한 적합한 조치를 취하는 데 사용될 수 있습니다.

#### **메시지 이용자 함수**

오브젝트 핸들에 특정한 오류(*CompCode* = MQCC\_FAILED)가 감지될 때 메시지 이용자 함수가 메시지 없이 호출되었습니다(예: *Reason code* = MQRC\_GET\_INHIBITED).

*Reason* 필드의 값은 호출의 이유를 표시하기 위해 설정됩니다.

### **MQCBCT\_MC\_EVENT\_CALL**

이벤트 핸들러 함수가 멀티캐스트 이벤트에 대해 호출되었습니다. 이벤트 핸들러에는 '정상' IBM MQ 이벤트 대신 IBM MQ 멀티캐스트 이벤트가 송신됩니다.

MQCBCT\_MC\_EVENT\_CALL에 대한 자세한 정보는 [멀티캐스트 예외 보고서를 참조하십시오](#).

### **MQCBC의 Hobj (MQHOBJ)**

이는 메시지 이용자에 대한 호출의 오브젝트 핸들입니다.

이벤트 핸들러의 경우, 이 값은 MQHO\_NONE입니다.

애플리케이션은 이 핸들 및 가져오기 메시지 옵션 블록의 토큰을 사용하여 메시지가 큐에서 제거되면 메시지를 가져올 수 있습니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQHO\_UNUSABLE\_HOBJ입니다.

### **MQCBC의 경우 CallbackArea (MQPTR)**

이 필드는 콜백 함수가 사용할 수 있습니다.

큐 관리자는 이 필드의 콘텐츠를 기반으로 의사결정을 내리지 않으며, 이는 콜백 함수를 정의하는 데 사용된 MQCB 호출의 매개변수인 MQCBD 구조의 *CallbackArea* 필드에서 변경 없이 전달됩니다.

*CallbackArea*에 대한 변경사항은 *HObj*에 대한 콜백 함수의 호출에서 유지됩니다. 이 필드는 다른 핸들의 콜백 함수와 공유되지 않습니다.

이는 콜백 함수의 입출력(I/O) 필드입니다. 이 필드의 초기값은 널 포인터이거나 널 바이트입니다.

### **MQCBC의 경우 ConnectionArea (MQPTR)**

이 필드는 콜백 함수가 사용할 수 있습니다.

큐 관리자는 이 필드의 콘텐츠를 기반으로 결정하지 않으며 MQCTLO 구조의 *ConnectionArea* 필드에서 변경하지 않은 채 전달되며 이 필드는 콜백 함수를 제어하는 데 사용되는 MQCTL 호출의 매개변수입니다.

콜백 함수에 의해 이 필드로 작성된 변경은 콜백 함수의 호출 전체에 유지됩니다. 이 영역은 모든 콜백 함수에 의해 공유되는 정보를 전달하는 데 사용될 수 있습니다. *CallbackArea*와 달리 이 영역은 연결 핸들의 모든 콜백에서 공통입니다.

이는 입력 및 출력 필드입니다. 이 필드의 초기값은 널 포인터이거나 널 바이트입니다.

### **MQCBC의 경우 CompCode (MQLONG)**

이 필드는 완료 코드입니다. 이는 메시지 이용에 문제점이 있었는지 여부를 표시합니다.

값은 다음 중 하나입니다.

#### **MQCC\_OK**

성공적인 완료

#### **MQCC\_WARNING**

경고(일부 완료)

#### **MQCC\_FAILED**

호출에 실패했습니다.

입력 필드입니다. 이 필드의 초기값은 MQCC\_OK입니다.

### **MQCBC의 이유 (MQLONG)**

이는 CompCode를 규정하는 이유 코드입니다.

입력 필드입니다. 이 필드의 초기값은 MQRC\_NONE입니다.

### **MQCBC의 상태 (MQLONG)**

현재 이용자의 상태에 대한 표시입니다. 이 필드는 0이 아닌 이유 코드가 이용자 함수에 전달될 때 애플리케이션에 가장 유용합니다.

각 이유 코드의 작동을 코드화할 필요가 없기 때문에 애플리케이션 프로그래밍을 단순화하기 위해 이 필드를 사용할 수 있습니다.

입력 필드입니다. 이 필드의 초기값은 MQCS\_NONE입니다.

표 472.		
상태	큐 관리자 조치	상수의 값
<b>MQCS_NONE</b> 이 이유 코드는 추가 이유 정보 없이 정상 호출을 나타냅니다	이는 정상 조작입니다.	0
<b>MQCS_SUSPENDED_TEMPORARY</b> 이 이유 코드는 임시 조건을 나타냅니다.	콜백 루틴은 조건을 보고하도록 호출된 후 일시 중단됩니다. 일정 기간 후에 시스템이 조작을 다시 시도할 수 있으며, 이에 따라 동일한 상태가 다시 발생할 수 있습니다.	1
<b>MQCS_SUSPENDED_USER_ACTION</b> 이러한 이유 코드는 콜백이 상태를 해결하기 위한 조치를 취해야 하는 상태를 표시합니다.	이용자는 일시중단되며 콜백 루틴은 조건을 보고하기 위해 호출됩니다. 가능한 경우 콜백 루틴은 조건을 해결하고 연결을 계속하거나 닫아야 합니다.	2
<b>MQCS_SUSPENDED</b> 이 이유 코드는 추가 메시지 콜백을 방지하는 실패를 표시합니다.	큐 관리자는 자동으로 콜백 함수를 일시중단합니다. 콜백 함수가 재개되는 경우 동일한 이유 코드를 다시 수신하게 됩니다.	3
<b>MQCS_STOPPED</b> 이 이유 코드는 메시지 이용의 끝을 표시합니다.	MQCBDO_STOP_CALL을 지정한 콜백 및 예외 핸들러에 전달됩니다. 추가적인 메시지가 이용될 수 없습니다.	4

### **MQCBC의 경우 DataLength (MQLONG)**

이는 메시지에 있는 애플리케이션 데이터의 길이(바이트)입니다. 값이 0이면 메시지에 애플리케이션 데이터가 없다는 의미입니다.



DataLength 필드에는 메시지의 길이가 포함되지만, 이용자에 전달된 메시지 데이터의 길이가 반드시 포함되지는 않습니다. 메시지가 잘렸을 수 있습니다. MQGMO의 ReturnedLength를 사용하여 이용자에 실제로 전달된 데이터의 양을 판별할 수 있습니다.

이유 코드에서 메시지가 잘렸음을 표시하는 경우, DataLength 필드를 사용하여 실제 메시지의 크기를 판별할 수 있습니다. 이를 사용하면 메시지 데이터를 수용하는 데 필요한 버퍼의 크기를 판별한 후에 MQCB 호출을 실행하여 적절한 값으로 MaxMsgLength를 업데이트할 수 있습니다.

MQGMO\_CONVERT 옵션이 지정된 경우, 변환 메시지는 DataLength에 대해 리턴된 값보다 클 수 있습니다. 해당 경우에, 애플리케이션은 MQCB 호출을 실행하여 DataLength에 대해 큐 관리자가 리턴한 값보다 크도록 MaxMsgLength를 업데이트해야 할 수 있습니다.

메시지 잘림 문제점을 피하려면 MaxMsgLength를 MQCBD\_FULL\_MSG\_LENGTH로서 지정하십시오. 이로 인해 큐 관리자가 데이터 변환 후 전체 메시지 길이에 버퍼를 할당하게 됩니다. 그러나 이 옵션이 지정된 경우라도 요청을 올바르게 처리하기 위해 충분한 스토리지를 사용할 수 없는 경우도 있습니다. 애플리케이션은 항상 리턴된 이유 코드를 확인해야 합니다. 예를 들어, 메시지를 변환하기 위해 충분한 스토리지를 할당하는 것이 불가능한 경우 메시지는 변환되지 않은 애플리케이션으로 리턴됩니다.

이 필드는 메시지 사용자 함수의 입력 필드입니다. 이벤트 핸들러 함수와는 관련이 없습니다.

### **MQCBC의 경우 BufferLength (MQLONG)**

이 필드는 이 함수에 전달된 메시지 버퍼의 길이(바이트)입니다.

버퍼는 MQGMO의 ReturnedLength 값 및 이용자에 대해 정의된 MaxMsgLength 값 모두보다 클 수 있습니다.

실제 메시지 길이는 DataLength 필드에 제공됩니다.

애플리케이션은 콜백 함수의 지속 기간동안 해당 목적을 위해 전체 버퍼를 사용할 수 있습니다.

이 필드는 메시지 사용자 함수의 입력 필드입니다. 예외 핸들러 함수와는 관련이 없습니다.

### **MQCBC에 대한 플래그 (MQLONG)**

이 이용자에 대한 정보를 포함하는 플래그.

다음 옵션이 정의됩니다.

#### **MQCBCF\_READA\_BUFFER\_EMPTY**

MQCO\_QUIESCE 옵션을 사용 중인 이전 MQCLOSE 호출이 MQRC\_READ\_AHEAD\_MSGS의 이유 코드로 실패하면 이 플래그가 리턴될 수 있습니다.

이 코드는 마지막 미리 읽기 메시지가 리턴되고 버퍼가 현재 비어 있음을 나타냅니다. 애플리케이션이 MQCO\_QUIESCE 옵션을 사용하여 다른 MQCLOSE 호출을 실행하면 이는 성공합니다.

현재 선택 기준과 일치하지 않는 미리 읽기 버퍼에 여전히 메시지가 있을 수 있으므로 이 플래그가 설정된 메시지가 애플리케이션에 제공되지 않을 수 있습니다. 이 인스턴스에서 사용자 함수는 이유 코드 MQRC\_HOBJ\_QUIESCED로 호출됩니다.

미리 읽기 버퍼가 모두 비워지는 경우, 이용자는 MQCBCF\_READA\_BUFFER\_EMPTY 플래그 및 이유 코드 MQRC\_HOBJ\_QUIESCED\_NO\_MSGS로 호출됩니다.

이 필드는 메시지 사용자 함수의 입력 필드입니다. 이벤트 핸들러 함수와는 관련이 없습니다.

### **MQCBC의 경우 ReconnectDelay (MQLONG)**

ReconnectDelay는 재연결을 시도하기 전에 큐 관리자가 대기하는 기간을 표시합니다. 이 필드는 지연을 변경하거나 다시 연결을 중지하기 위해 이벤트 핸들러로 수정할 수 있습니다.

콜백 컨텍스트의 Reason 필드의 값이 MQRC\_RECONNECTING인 경우에만 ReconnectDelay 필드를 사용할 수 있습니다.

이벤트 핸들러를 시작할 때 ReconnectDelay의 값은 재연결 시도를 수행하기 전에 큐 관리자가 대기하는 시간(밀리초)입니다. 286 페이지의 표 473에서는 이벤트 핸들러에서 리턴할 때 큐 관리자의 작동을 수정하도록 설정할 수 있는 값을 나열합니다.

표 473. <i>ReconnectDelay</i> 값		
이름	값	설명
MQRD_NO_RECONNECT	-1	더 이상 다시 연결을 시도하지 마십시오. 오류가 애플리케이션에 리턴됩니다.
MQRD_NO_DELAY	0	즉시 다시 연결을 시도합니다.
<i>Milliseconds</i>	>0	연결을 재시도하기 전에 이 시간(밀리초) 동안 대기합니다.

## MQCBD - 콜백 디스크립터

MQCBD 구조는 큐 관리자의 사용을 제어하는 콜백 함수 및 옵션을 지정하는 데 사용됩니다. 구조는 MQCB 호출의 입력 매개변수입니다.

### 가용성

MQCBD 구조는 다음 플랫폼에서 사용 가능합니다.

-  AIX
-  IBM i
-  Linux
-  Windows
-  z/OS

및 해당 시스템에 연결된 IBM MQ MQI clients

### 버전

MQCBD의 현재 버전은 MQCBD\_VERSION\_1입니다.

### 문자 세트 및 인코딩

MQCBD의 데이터는 MQENC\_NATIVE로 지정된 로컬 큐 관리자의 인코딩 및 **CodedCharSetId** 큐 관리자 속성으로 지정된 문자 세트에 있어야 합니다. 그러나 애플리케이션이 MQ MQI 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트와 인코딩으로 작성되어야 합니다.

### 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 474. MQCBD의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
<u>StrucID</u> (구조 ID)	MQCBD_STRUC_ID	'CBD~'
<u>버전</u> (구조 버전 번호)	MQCBD_VERSION_1	1
<u>CallbackType</u> (콜백 함수의 유형)	MQCBT_MESSAGE_CONSUMER	1
<u>옵션</u> (메시지 이동을 제어하는 옵션)	MQCBDO_NONE	0
<u>CallbackArea</u> (사용할 콜백 함수의 필드)	없음	널 포인터 또는 널 공백
<u>CallbackFunction</u> (함수가 API 호출로 호출되는지 여부)	없음	널 포인터 또는 널 공백

표 474. MQCBD의 필드 (계속)

필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
CallbackName (함수가 동적으로 링크된 프로그램으로 호출되는지 여부)	없음	널 문자열 또는 공백
MaxMsg길이 (읽을 수 있는 가장 긴 메시지의 길이)	MQCBD_FULL_MSG_LENGTH	-1

**참고:**

1. ~ 기호는 단일 공백 문자를 나타냅니다.
2. 널 문자열 또는 공백 값은 C 프로그래밍 언어의 널 문자열과 기타 프로그래밍 언어의 공백 문자를 나타냅니다.
3. C 프로그래밍 언어의 매크로 변수MQCBD\_DEFAULT에는 표에 나열된 값이 포함됩니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오.

```
MQCBD MyCBD = {MQCBD_DEFAULT};
```

## 언어 선언

### MQCBD의 C 선언

```
typedef struct tagMQCBD MQCBD;
struct tagMQCBD {
    MQCHAR4    StructId;           /* Structure identifier */
    MQLONG     Version;           /* Structure version number */
    MQLONG     CallbackType;     /* Callback function type */
    MQLONG     Options;          /* Options controlling message
                                consumption */
    MQPTR      CallbackArea;     /* User data passed to the function */
    MQPTR      CallbackFunction; /* Callback function pointer */
    MQCHAR128  CallbackName;     /* Callback name */
    MQLONG     MaxMsgLength;     /* Maximum message length */
};
```

### MQCBD의 COBOL 선언

```
** MQCBCD structure
10  MQCBD.
** Structure Identifier
15  MQCBD-STRUCID                PIC X(4).
** Structure Version
15  MQCBD-VERSION                PIC S9(9) BINARY.
** Callback Type
15  MQCBD-CALLBACKTYPE          PIC S9(9) BINARY.
** Options
15  MQCBD-OPTIONS                PIC S9(9) BINARY.
** Callback User Area
15  MQCBD-CALLBACKAREA          POINTER
** Callback Function Pointer
15  MQCBD-CALLBACKFUNCTION      FUNCTION-POINTER
** Callback Program Name
15  MQCBD-CALLBACKNAME          PIC X(128)
** Maximum Message Length
15  MQCDB-MAXMSGLength          PIC S9(9) BINARY.
```

### MQCBD의 PL/I 선언

```
dcl
1  MQCBD based,
3  StructId          char(4),          /* Structure identifier*/
3  Version           fixed bin(31),   /* Structure version*/
3  CallbackType     fixed bin(31),   /* Callback function type */
3  Options          fixed bin(31),   /* Options */
```

```

3 CallbackArea      pointer,          /* User area passed to the function */
3 CallbackFunction  pointer,          /* Callback Function Pointer */
3 CallbackName      char(128),        /* Callback Program Name */
3 MaxMsgLength     fixed bin(31); /* Maximum Message Length */

```

### **MQCBD의 경우 StrucId (MQCHAR4)**

콜백 디스크립터 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQCBD\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

#### **MQCBD\_STRUC\_ID**

콜백 디스크립터 구조의 ID.

C 프로그래밍 언어의 경우 상수 MQCBD\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQCBD\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### **MQCBD의 버전 (MQLONG)**

콜백 디스크립터 구조의 버전 번호입니다. 항상 입력 필드입니다.

값은 다음과 같아야 합니다.

#### **MQCBD\_VERSION\_1**

버전 1 콜백 디스크립터 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### **MQCBD\_CURRENT\_VERSION**

콜백 디스크립터 구조의 현재 버전.

### **MQCBD의 경우 CallbackType (MQLONG)**

콜백 디스크립터 구조 - CallbackType 필드

이는 콜백 함수의 유형입니다. 값은 다음 중 하나여야 합니다.

#### **MQCBT\_MESSAGE\_CONSUMER**

메시지 사용자 함수로 이 콜백을 정의합니다.

지정된 선택 기준을 충족하는 메시지를 오브젝트 핸들에서 사용할 수 있고 연결이 시작되면 메시지 사용자 콜백 함수가 호출됩니다.

#### **MQCBT\_EVENT\_HANDLER**

비동기 이벤트 루틴으로 이 콜백을 정의합니다. 핸들에 대한 메시지를 이용하지 않습니다.

Hobj는 이벤트 핸들러를 정의하는 MQCB 호출에 필요하지 않으며 지정된 경우 무시됩니다.

이벤트 핸들러는 전체 메시지 사용자 환경에 영향을 주는 조건에 호출됩니다. 이벤트 시 사용자 함수는 메시지 없이 호출됩니다(예를 들어, 큐 관리자나 연결 중지 또는 일시 정지 발생). 이는 단일 메시지 이용자에 특정한 조건의 경우 호출됩니다(예: MQRC\_GET\_INHIBITED).

이벤트는 연결이 다음 환경을 제외하고 시작 또는 중지되었는지 여부에 관계없이 애플리케이션에 전달됩니다.

- CICS on z/OS 환경
- 스레드되지 않은 애플리케이션

호출자가 이러한 값 중 하나를 전달하지 않으면 MQRC\_CALLBACK\_TYPE\_ERROR의 Reason 코드로 호출이 실패합니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQCBT\_MESSAGE\_CONSUMER입니다.

### **MQCBD에 대한 옵션 (MQLONG)**

콜백 디스크립터 구조 - 옵션 필드

이러한 옵션을 하나 이상 지정할 수 있습니다. 둘 이상의 옵션을 지정하려면 값을 함께 추가하거나(동일한 상수를 두 번 이상 추가하지 않음), 비트 단위 OR 연산을 사용하여 값을 결합하십시오(프로그래밍 언어가 비트 연산을 지원하는 경우).

## **MQCBDO\_FAIL\_IF QUIESCING**

큐 관리자가 정지 중인 경우 MQCB 호출에 실패합니다.

z/OS에서 이 옵션은 또한 연결(CICS 또는 IMS 애플리케이션의 경우)이 일시정지 상태에 있는 경우 MQCB 호출 강제 실행에 실패합니다.

정지 상태인 경우에 메시지 이용자에 대한 알림이 발생하도록 하려면, MQCB 호출에서 전달되는 MQGMO 옵션에서 MQGMO\_FAIL\_IF QUIESCING을 지정하십시오.

**제어 옵션:** 다음 옵션은 이용자의 상태가 변경될 때 메시지 없이 콜백 함수가 호출되는지 여부를 제어합니다.

## **MQCBDO\_REGISTER\_CALL**

콜백 함수가 호출 유형 MQCBCT\_REGISTER\_CALL을 사용하여 호출됩니다.

## **MQCBDO\_START\_CALL**

콜백 함수가 호출 유형 MQCBCT\_START\_CALL을 사용하여 호출됩니다.

## **MQCBDO\_STOP\_CALL**

콜백 함수가 호출 유형 MQCBCT\_STOP\_CALL을 사용하여 호출됩니다.

## **MQCBDO\_DEREGISTER\_CALL**

콜백 함수가 호출 유형 MQCBCT\_DEREGISTER\_CALL을 사용하여 호출됩니다.

## **MQCBDO\_EVENT\_CALL**

콜백 함수가 호출 유형 MQCBCT\_EVENT\_CALL을 사용하여 호출됩니다.

## **MQCBDO\_MC\_EVENT\_CALL**

콜백 함수가 호출 유형 MQCBCT\_MC\_EVENT\_CALL을 사용하여 호출됩니다.

이러한 호출 유형에 대한 자세한 내용은 CallType을 참조하십시오.

**기본 옵션:** 설명한 옵션이 필요하지 않은 경우에는 다음 옵션을 사용하십시오.

## **MQCBDO\_NONE**

기타 옵션이 지정되지 않았음을 표시하려면 이 값을 사용하십시오. 모든 옵션은 기본 값을 가정합니다.

MQCBDO\_NONE은 프로그램 문서화를 지원하도록 정의됩니다. 이는 이 옵션을 다른 옵션과 함께 사용하기 위한 용도가 아니지만, 해당 값이 0이므로 해당 사용을 감지할 수 없습니다.

입력 필드입니다. *Options* 필드의 초기값은 MQCBDO\_NONE입니다.

## **MQCBD의 경우 CallbackArea (MQPTR)**

콜백 디스크립터 구조 - CallbackArea 필드

이 필드는 콜백 함수에서 사용할 수 있는 필드입니다.

큐 관리자는 이 필드의 콘텐츠를 기반으로 의사결정을 내리지 않으며, 이는 콜백 함수 선언의 매개변수인 MQCBC 구조의 CallbackArea 필드에서 변경 없이 전달됩니다.

값은 현재 정의된 콜백 없이 MQOP\_REGISTER 값을 갖는 *Operation*에서만 사용됩니다. 이는 이전 정의를 대체하지 않습니다.

이는 콜백 함수의 입력 및 출력 필드입니다. 이 필드의 초기값은 널 포인터이거나 널 바이트입니다.

## **MQCBD의 경우 CallbackFunction (MQPTR)**

콜백 디스크립터 구조 - CallbackFunction 필드

콜백 함수는 함수 호출로서 호출됩니다.

콜백 함수에 포인터를 지정하려면 이 필드를 사용하십시오.

*CallbackFunction* 또는 *CallbackName*을 지정해야 합니다. 둘 모두를 지정하면 이유 코드 MQRC\_CALLBACK\_ROUTINE\_ERROR가 리턴됩니다.

*CallbackName* 또는 *CallbackFunction* 중 어느 것도 설정되지 않으면, 이유 코드 MQRC\_CALLBACK\_ROUTINE\_ERROR로 호출이 실패합니다.

이 옵션이 지원되지 않는 환경: 함수-포인터 참조를 지원하지 않는 프로그래밍 언어 및 컴파일러. 이 경우에는 이 이유 코드 MQRC\_CALLBACK\_ROUTINE\_ERROR로 호출이 실패합니다.

**z/OS** z/OS에서, 함수는 OS 연계 규약으로 호출됨을 예상해야 합니다. 예를 들면, C 프로그래밍 언어에서 다음을 지정하십시오.

```
#pragma linkage(MQCB_FUNCTION,OS)
```

입력 필드입니다. 이 필드의 초기값은 널 포인터이거나 널 바이트입니다.

**참고:** IBM WebSphere® MQ 7.0.1와 함께 CICS 를 사용하는 경우 다음과 같은 경우 비동기 이용이 지원됩니다.

- Apar PK66866은 CICS TS 3.2에 적용됩니다.
- Apar PK89844는 CICS TS 4.1에 적용됩니다.

### **MQCBD의 경우 CallbackName (MQCHAR128)**

콜백 디스크립터 구조 - CallbackName 필드

콜백 함수는 동적으로 링크된 프로그램으로서 호출됩니다.

*CallbackFunction* 또는 *CallbackName*을 지정해야 합니다. 둘 모두를 지정하면 이유 코드 MQRC\_CALLBACK\_ROUTINE\_ERROR가 리턴됩니다.

*CallbackName* 또는 *CallbackFunction* 중 어느 것도 설정되지 않으면 이유 코드 MQRC\_CALLBACK\_ROUTINE\_ERROR로 호출이 실패합니다.

이 모듈은 사용할 첫 번째 콜백 루틴이 등록될 때 로드되고 사용할 마지막 콜백 루틴이 등록 취소될 때 로드 해제됩니다.

다음 텍스트에서 언급한 경우를 제외하면, 이름은 임베드된 공백 없이 필드 내에서 왼쪽으로 정렬됩니다. 이름은 필드의 길이까지 공백으로 채워집니다. 다음 설명에서, 대괄호([ ])는 옵션 정보를 나타냅니다.

#### **IBM i**

콜백 이름은 다음 형식 중 하나일 수 있습니다.

- Library "/" Program
- Library "/" ServiceProgram ("FunctionName")

예를 들어, MyLibrary/MyProgram(MyFunction)입니다.

라이브러리 이름은 \*LIBL일 수 있습니다. 라이브러리 및 프로그램 이름은 모두 최대 10자로 제한됩니다.

#### **AIX and Linux**

콜백 이름은 동적으로 로드할 수 있는 모듈 또는 라이브러리의 이름이며, 해당 라이브러리에 상주하는 함수의 이름이 접미부로 지정됩니다. 함수 이름은 괄호로 묶어야 합니다. 라이브러리 이름 앞에 선택적으로 디렉토리 경로를 지정할 수 있습니다.

```
[path]library(function)
```

경로가 지정되지 않으면 시스템 검색 경로가 사용됩니다.

이름은 최대 128자로 제한됩니다.

#### **Windows**

콜백 이름은 동적-링크 라이브러리의 이름으로 해당 라이브러리에 상주하는 함수의 이름이 후미에 첨부됩니다. 함수 이름은 괄호로 묶어야 합니다. 라이브러리 이름에는 선택적으로 디렉토리 경로 및 드라이브가 접두부로 지정될 수 있습니다.

```
[d:][path]library(function)
```

드라이브 및 경로가 지정되지 않은 경우 시스템 검색 경로가 사용됩니다.

이름은 최대 128자로 제한됩니다.

#### **z/OS**

LINK 또는 LOAD 매크로의 EP 매개변수에서 스펙에 유효한 로드 모듈의 콜백 이름입니다.

이름은 최대 8자로 제한됩니다.

### **z/OS CICS**

콜백 이름은 EXEC CICS LINK 명령 매크로의 PROGRAM 매개변수에서 스펙에 유효한 로드 모듈의 이름입니다.

이름은 최대 8자로 제한됩니다.

프로그램은 설치된 PROGRAM 정의의 REMOTESYTEM 옵션 또는 동적 라우팅 프로그램을 사용하여 원격으로 정의할 수 있습니다.

프로그램이 IBM MQ API 호출을 사용하는 경우 원격 CICS 리전은 IBM MQ에 연결되어야 합니다. 그러나 MQCBC 구조의 *Hobj* 필드가 원격 시스템에서는 올바르지 않음을 유념하십시오.

*CallbackName*을 로드하는 데 실패하는 경우 다음 오류 코드 중 하나가 애플리케이션으로 리턴됩니다.

- MQRC\_MODULE\_NOT\_FOUND
- MQRC\_MODULE\_INVALID
- MQRC\_MODULE\_ENTRY\_NOT\_FOUND

또한 로드가 시도된 모듈의 이름 및 운영 체제의 실패 이유 코드를 포함하는 오류 로그에도 메시지가 기록됩니다.

입력 필드입니다. 이 필드의 초기값은 널 문자열 또는 공백입니다.

### **MaxMsgMQCBD의 길이 (MQLONG)**

이는 핸들에서 읽을 수 있고 콜백 루틴에 지정할 수 있는 가장 긴 메시지의 길이(바이트)입니다. 콜백 디스크립터 구조 - *MaxMsgLength* 필드

메시지에 더 긴 길이가 있는 경우 콜백 루틴은 메시지의 *MaxMsgLength* 바이트 및 이유 코드를 수신합니다.

- MQRC\_TRUNCATED\_MSG\_FAILED 또는
- MQRC\_TRUNCATED\_MSG\_ACCEPTED(MQGMO\_ACCEPT\_TRUNCATED\_MSG를 지정한 경우).

실제 메시지 길이는 MQCBC 구조의 *DataLength* 필드에서 제공됩니다.

다음 특수 값이 정의됩니다.

### **MQCBD\_FULL\_MSG\_LENGTH**

버퍼의 길이는 잘림 없이 메시지를 리턴하기 위해 시스템에서 조정됩니다.

메시지 수신을 위해 버퍼를 할당하기에는 메모리가 부족한 경우, 시스템은 MQRC\_STORAGE\_NOT\_AVAILABLE 이유 코드로 콜백 함수를 호출합니다.

예를 들어, 데이터 변환을 요청하고 메시지 데이터를 변환할 수 있는 메모리 부족이 있는 경우 변환되지 않는 메시지는 콜백 함수에 전달됩니다.

입력 필드입니다. *MaxMsgLength* 필드의 초기값은 MQCBD\_FULL\_MSG\_LENGTH입니다.

### **MQCHARV - 가변 길이 문자열**

MQCHARV 구조를 사용하여 가변 길이 문자열을 설명하십시오.

### **가용성**

MQCHARV 구조는 다음 플랫폼에서 사용 가능합니다.

-  AIX
-  IBM i
-  Linux
-  Windows

및 해당 시스템에 연결된 IBM MQ MQI clients



## 문자 세트 및 인코딩

MQCHARV의 데이터는 MQENC\_NATIVE에서 제공하는 로컬 큐 관리자의 인코딩 및 구조 내 VSCCSID 필드의 문자 세트에 있어야 합니다. 그러나 애플리케이션이 MQ 클라이언트로서 실행 중인 경우, 구조는 클라이언트의 인코딩에 있어야 합니다. 일부 문자 세트에는 인코딩에 의존하는 표현이 있습니다. VSCCSID가 이러한 문자 세트 중 하나인 경우, 사용된 인코딩은 MQCHARV의 기타 필드의 경우와 동일한 인코딩입니다. VSCCSID에 의해 식별된 문자 세트는 2바이트 문자 세트(DBCS)일 수 있습니다.

## 사용법

MQCHARV 구조는 이를 포함하는 구조와 인접하지 않을 수 있는 데이터를 처리합니다. 이 데이터를 처리하려면 포인터 데이터 유형으로 선언된 필드가 사용될 수 있습니다. 참고로, COBOL은 모든 환경에서 포인터 데이터 유형을 지원하지 않습니다. 이로 인해, 데이터는 MQCHARV가 포함된 구조의 시작으로부터의 데이터 오프셋을 포함하는 필드를 사용하여 처리될 수도 있습니다.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 475. MQCHARV의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
<u>VSPtr</u> (가변 길이 문자열에 대한 포인터)	없음	널 포인터 또는 널 바이트.
<u>VSOffset</u> (이 MQCHARV 구조를 포함하는 구조의 시작부터 가변 길이 문자열의 오프셋 (바이트))	없음	0
<u>VSBufSize</u> (VSPtr 또는 VSOffset 필드에서 주소 지정된 버퍼의 크기 (바이트))	MQVS_USE_VSLENGTH	0
<u>VSLength</u> (VSPtr 또는 VSOffset 필드에서 지정되는 가변 길이 문자열의 길이 (바이트))	없음	0
<u>VSCCSID</u> (VSPtr 또는 VSOffset 필드에서 주소 지정된 가변 길이 문자열의 문자 세트 ID)	MQCCSI_APPL	-3

**참고:** C 프로그래밍 언어에서 매크로 변수 MQCHARV\_DEFAULT에는 표에 나열된 값이 포함됩니다. 즉, 구조 내의 필드에 대한 초기값을 제공하기 위해 다음과 같은 방법으로 사용될 수 있습니다.

```
MQCHARV MyVarStr = {MQCHARV_DEFAULT};
```

## 언어 선언

### MQCHARV의 C 선언

```
typedef struct tagMQCHARV MQCHARV;
struct tagMQCHARV {
    MQPTR    VSPtr;                /* Address of variable length string */
    MQLONG   VSOffset;            /* Offset of variable length string */
    MQLONG   VSBufSize;          /* Size of buffer */
    MQLONG   VSLength;           /* Length of variable length string */
    MQLONG   VSCCSID;            /* CCSID of variable length string */
};
```

### MQCHARV의 COBOL 선언

```
** MQCHARV structure
10  MQCHARV.
** Address of variable length string
15  MQCHARV-VSPTR          POINTER.
** Offset of variable length string
```



```

15 MQCHARV-VSOFFSET PIC S9(9) BINARY.
** Size of buffer
15 MQCHARV-VSBUFSIZE PIC S9(9) BINARY.
** Length of variable length string
15 MQCHARV-VSLENGTH PIC S9(9) BINARY.
** CCSID of variable length string
15 MQCHARV-VSCCSID PIC S9(9) BINARY.

```

**참고:** 환경 간에 COBOL 애플리케이션을 포팅하려면 모든 의도된 환경에서 포인터 데이터 유형을 사용할 수 있는지 확인해야 합니다. 그렇지 않으면, 포인터 필드 대신 오프셋 필드를 사용하여 애플리케이션이 데이터를 처리해야 합니다. 포인터가 지원되지 않는 환경에서는 포인터 필드를 적절한 길이의 바이트 문자열로 선언할 수 있으며, 초기값은 all-null 바이트 문자열입니다. 오프셋 필드를 사용 중이면 이 초기값을 대체하지 마십시오. 제공된 사본을 변경하지 않고 이를 수행하는 한 가지 방법은 다음을 사용하는 것입니다.

```
COPY CMQCHRVV REPLACING POINTER BY ==BINARY PIC S9(9)==.
```

여기서 CMQCHRVV는 사용될 사본으로 교환될 수 있습니다.

### MQCHARV의 PL/I 선언

```

dcl
1 MQCHARV based,
3 VSPtr pointer, /* Address of variable length string */
3 VSOffset fixed bin(31), /* Offset of variable length string */
3 VSBufSize fixed bin(31), /* Size of buffer */
3 VSLength fixed bin(31), /* Length of variable length string */
3 VSCCSID fixed bin(31); /* CCSID of variable length string */

```

### MQCHARV의 상위 레벨 어셈블러 선언

```

MQCHARV          DSECT
MQCHARV_VSPTR    DS F      Address of variable length string
MQCHARV_VSOFFSET DS F      Offset of variable length string
MQCHARV_VSBUFSIZE DS F      Size of buffer
MQCHARV_VSLENGTH DS F      Length of variable length string
MQCHARV_VSCCSID DS F      CCSID of variable length string
*
MQCHARV_LENGTH   EQU *-MQCHARV
                  ORG MQCHARV
MQCHARV_AREA     DS CL(MQCHARV_LENGTH)

```

### MQCHARV용 VSPtr (MQPTR)

가변 길이 문자열에 대한 포인터입니다.

VSPtr 또는 VSOffset 필드 중 하나를 사용하여 가변 길이 문자열을 지정할 수 있지만, 둘 모두는 아닙니다.

이 필드의 초기값은 널 포인터이거나 널 바이트입니다.

### MQCHARV용 VSOffset (MQLONG)

오프셋은 양수 또는 음수일 수 있습니다. VSPtr 또는 VSOffset 필드 중 하나를 사용하여 가변 길이 문자열을 지정할 수 있지만, 둘 모두는 아닙니다. MQCHARV 또는 이를 포함하는 구조의 시작부터 가변 길이 문자열의 오프셋(바이트)입니다.

MQCHARV 구조가 다른 구조 내에 임베드될 때 이 값은 이 MQCHARV 구조를 포함하는 구조의 시작으로부터의 가변 길이 문자열의 오프셋(바이트)입니다. MQCHARV 구조가 다른 구조 내에 임베드되지 않은 경우(예: 함수 호출의 매개변수로서 지정된 경우), 오프셋은 MQCHARV 구조의 시작에 대해 상대적입니다.

이 필드의 초기값은 0입니다.

### MQCHARV용 VSBufSize (MQLONG)

이는 VSPtr 또는 VSOffset 필드에서 처리하는 버퍼의 크기(바이트)입니다.

MQCHARV 구조가 함수 호출의 출력 필드로 사용될 때 이 필드는 제공된 버퍼의 길이로 초기화되어야 합니다. VSLength의 값이 VSBufSize보다 큰 경우, 데이터의 VSBufSize 바이트만 버퍼의 호출자에 리턴됩니다.

이 값은 0 이상의 값이거나 인식된 다음 특수 값이어야 합니다.

## MQVS\_USE\_VSLENGTH

지정된 경우, 버퍼의 길이는 MQCHARV 구조의 VSLength 필드에서 가져옵니다. 구조를 출력 버퍼로 사용 중이며 버퍼가 제공될 때는 이 값을 사용하지 마십시오.

이는 이 필드의 초기값입니다.

## MQCHARV의 VSLength (MQLONG)

VSPtr 또는 VSOOffset 필드에서 처리하는 가변 길이 문자열의 길이(바이트)입니다.

이 필드의 초기값은 0입니다. 값은 0보다 크거나 같아야 하거나 인식되는 다음 특수 값이어야 합니다.

## MQVS\_NULL\_TERMINATED

MQVS\_NULL\_TERMINATED가 지정되지 않은 경우에는 VSLength 바이트가 문자열의 일부로서 포함됩니다. 널 문자가 존재하는 경우 문자열을 구분하지 않습니다.

MQVS\_NULL\_TERMINATED를 지정하면 문자열이 문자열에 나타난 첫 번째 널로 구분됩니다. 널 자체는 그 문자열의 일부로 포함되지 않습니다.

**참고:** MQVS\_NULL\_TERMINATED가 지정된 경우 문자열 종료에 사용된 널 문자는 VSCCSID에서 지정한 코드셋의 널입니다.


예를 들어 UTF-16(CCSIDs 1200, 13488 및 17584)에서 이는 2바이트 유니코드 인코딩입니다. 여기서 널은 모두 0인 16비트 숫자로 표시됩니다. UTF-16에서는 문자(예: 7비트 ASCII 문자)의 일부인 모두 0으로 설정된 단일 바이트를 찾는 게 일반적입니다. 그러나 문자열은 2개의 '영(0)' 바이트가 짝수 바이트 경계에서 발견될 때만 널로 종료됩니다. 유효한 문자의 각 부분일 때 홀수 경계에 두 개의 '0'바이트를 가져올 수 있습니다. 예를 들어, x'01' x'00 x'00' x'30'은 2개의 올바른 유니코드 문자를 표시하며 문자열을 널(Null)로 종료하지 않습니다.

## MQCHARV용 VSCCSID (MQLONG)

이는 VSPtr 또는 VSOOffset 필드에서 처리하는 가변 길이 문자열의 문자 세트 ID입니다.

이 필드의 초기값은 MQCCSI\_APPL이며, 이는 현재 프로세스의 실제 문자 세트 ID로 변경되어야 함을 표시하기 위해 MQ에 의해 정의됩니다. 따라서 MQCCSI\_APPL 값은 가변 길이 문자열과 연관되지 않습니다.

이 필드의 초기값은 컴파일 단위의 MQCCSI\_APPL 상수에 대해 다른 값을 정의하여 변경될 수 있습니다. 이를 수행하는 방법은 애플리케이션의 프로그래밍 언어에 따라 다릅니다.

 z/OS 시스템에서 MQCCSI\_APPL 에서 사용되는 기본 애플리케이션 CCSID 는 다음과 같이 정의됩니다.

- DLL 인터페이스를 사용하는 배치 LE 애플리케이션의 경우 기본값은 MQCONN이 실행될 때 현재 로케일과 연관된 CODESET입니다(기본값이 1047임).
- 배치 MQ 스텝 중 하나와 바인딩된 LE 애플리케이션의 경우 기본값은 MQCONN 뒤에 MQI 호출이 처음 실행될 때 현재 로케일과 연관된 CODESET입니다(기본값은 1047임).
- z/OS UNIX System Services 스레드에서 실행 중인 배치 비LE 애플리케이션의 경우, 기본값은 MQCONN 이후에 첫 번째 MQI 호출이 발행될 때 THLICCSID 의 값입니다(기본값은 1047).
- 다른 배치 애플리케이션의 경우 기본값은 큐 관리자의 CCSID입니다.

## MQCCSI\_APPL의 재정의

다음 예제는 다양한 프로그래밍 언어에서 MQCCSI\_APPL의 값을 대체할 수 있는 방법을 보여줍니다. MQCCSI\_APPL의 값을 변경할 수 있으며, 각각의 가변 길이 문자열의 VSCCSID를 개별적으로 설정할 필요가 없어집니다. 이 예제에서 CCSID는 1208로 설정됩니다. 이를 필요한 값으로 변경하십시오. 이는 기본값이 되며, MQCHARV의 특정 인스턴스에서 VSCCSID를 설정하여 이를 대체할 수 있습니다.

### C 사용법

```
#define MQCCSI_APPL 1208
#include <cmqc.h>
```

## COBOL 사용법

```
COPY CMQXYZV REPLACING -3 BY 1208.
```

## PL/I 사용법

```
%MQCCSI_APPL = '1208';  
%include syslib(cmqp);
```

## High Level Assembler 사용법

```
MQCCSI_APPL EQU 1208  
CMQA LIST=NO
```

## MQCIH - CICS bridge 헤더

MQCIH 구조는 CICS bridge 전체에서 CICS에 송신된 메시지에 대한 헤더 정보를 설명합니다.

IBM MQ 지원 플랫폼의 경우에는 MQCIH 구조가 포함된 메시지를 작성하고 전송할 수 있지만, IBM MQ for z/OS 큐 관리자만 CICS bridge를 사용할 수 있습니다. 따라서 메시지가 비z/OS 큐 관리자에서 CICS에 도달하려면 큐 관리자 네트워크에 메시지를 라우팅할 수 있는 하나 이상의 z/OS 큐 관리자가 포함되어야 합니다.

IBM MQ 9.0.0 이상에서 지원되는 모든 CICS 버전은 브릿지의 CICS 제공 버전을 사용합니다. IBM MQ CICS 어댑터 및 IBM MQ CICS bridge 컴포넌트 구성에 대한 자세한 정보는 CICS 문서의 [MQ 절](#)을 참조하십시오.

## 가용성

MQCIH 구조는 다음 플랫폼에서 사용 가능합니다.

- ▶ **AIX** AIX
- ▶ **Linux** Linux
- ▶ **Windows** Windows
- ▶ **z/OS** z/OS

및 해당 시스템에 연결된 IBM MQ MQI clients

## 형식 이름

MQFMT\_CICS

## 버전

MQCIH의 현재 버전은 MQCIH\_VERSION\_2입니다. 구조의 최신 버전에만 있는 필드는 다음 설명에서와 같이 식별됩니다.

지원된 프로그래밍 언어에 대해 제공된 헤더, COPY 및 INCLUDE 파일에는 MQCIH의 최신 버전이 포함되어 있으며, *Version* 필드의 초기값은 MQCIH\_VERSION\_2로 설정되어 있습니다.

## 문자 세트 및 인코딩

특수 조건은 MQCIH 구조 및 애플리케이션 메시지 데이터에 사용되는 문자 세트 및 인코딩에 적용됩니다.

- CICS bridge 큐를 소유하는 큐 관리자에 연결되는 애플리케이션은 큐 관리자의 문자 세트 및 인코딩에 있는 MQCIH 구조를 제공해야 합니다. 이는 MQCIH 구조의 데이터 변환이 이 경우에 수행되지 않기 때문입니다.
- 기타 큐 관리자에 연결하는 애플리케이션은 지원되는 문자 세트 및 인코딩에 있는 MQCIH 구조를 제공할 수 있습니다. CICS bridge 큐를 소유하는 큐 관리자에 연결된 수신 메시지 채널 에이전트는 MQCIH 구조를 변환합니다.

- MQCIH 구조 뒤에 오는 애플리케이션 메시지 데이터는 MQCIH 구조와 동일한 문자 세트 및 인코딩에 있어야 합니다. MQCIH 구조의 *CodedCharSetId* 및 *Encoding* 필드를 사용하여 애플리케이션 메시지 데이터의 문자 세트 및 인코딩을 지정할 수는 없습니다.

데이터가 큐 관리자가 지원하는 내장 형식 중 하나가 아닌 경우 애플리케이션 메시지 데이터를 변환하기 위해 데이터 변환 엑시트를 제공해야 합니다.

## 사용법

애플리케이션에 296 페이지의 표 477에 표시된 초기값과 동일한 값이 필요하고 브릿지가 AUTH=LOCAL 또는 AUTH=IDENTIFY로 실행 중인 경우 메시지에서 MQCIH 구조를 생략할 수 있습니다. 다른 모든 경우에 이 구조는 존재해야 합니다.

브릿지는 버전-1 또는 버전-2 MQCIH 구조를 허용하지만, 3270 트랜잭션의 경우에는 버전-2 구조를 사용해야 합니다.

애플리케이션은 요청 필드로서 문서화된 필드가 브릿지에 송신된 메시지의 적절한 값을 갖도록 보장해야 합니다. 해당 필드는 브릿지에 대한 입력입니다.

응답 필드로서 문서화된 필드는 브릿지가 애플리케이션에 송신한 응답 메시지의 CICS bridge에 의해 설정됩니다. 오류 정보는 *ReturnCode*, *Function*, *CompCode*, *Reason* 및 *AbendCode* 필드에서 리턴되지만, 이들 모두가 모든 경우에 설정되지는 않습니다. 다음 표는 *ReturnCode*의 다른 값에 대해 설정되는 필드를 표시합니다.

<i>ReturnCode</i>	<i>Function</i>	<i>CompCode</i>	<i>Reason</i>	<i>AbendCode</i>
MQCRC_OK	-	-	-	-
MQCRC_BRIDGE_ERROR	-	-	MQFB_CICS_*	-
MQCRC_MQ_API_ERROR MQCRC_BRIDGE_TIMEOUT	MQ 호출 이름	MQ <i>CompCode</i>	MQ <i>Reason</i>	-
MQCRC_CICS_EXEC_ERROR MQCRC_SECURITY_ERROR MQCRC_PROGRAM_NOT_AVAILABLE MQCRC_TRANSID_NOT_AVAILABLE	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	-
MQCRC_BRIDGE_ABEND MQCRC_APPLICATION_ABEND	-	-	-	CICS ABCODE

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
<u>StrucId</u> (구조 ID)	MQCIH_STRUC_ID	'CIH~'
<u>버전</u> (구조 버전 번호)	MQCIH_VERSION_2	2
<u>StrucLength</u> (MQCIH 구조의 길이)	MQCIH_LENGTH_2	180
<u>인코딩</u> (예약됨)	없음	0
<u>CodedCharSetId</u> (예약됨)	없음	0
<u>형식</u> (MQCIH뒤에 오는 데이터의MQ 형식 이름)	MQFMT_NONE	공백
<u>플래그</u> (플래그)	MQCIH_NONE	0
<u>ReturnCode</u> (브릿지의 리턴 코드)	MQCRC_OK	0

표 477. MQCIH에서 MQCIH의 필드 (계속)		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
CompCode (MQ 완료 코드 또는 CICS EIBRESP)	MQCC_OK	0
이유 (MQ 이유 또는 피드백 코드 또는 CICS EIBRESP2)	MQRC_NONE	0
UOWControl (작업 단위 제어)	MQCUOWC_ONLY	273
GetWait간격 (브릿지 태스크가 발행한 MQGET 호출의 대기 간격)	MQCGWI_DEFAULT	-2
LinkType (링크 유형)	MQCLT_PROGRAM	1
OutputData길이 (출력 COMMAREA 데이터 길이)	MQCODL_AS_INPUT	-1
FacilityKeepTime (브릿지 기능 릴리스 시간)	없음	0
ADSDescriptor (ADS 디스크립터 송신/수신)	MQCADSD_NONE	0
ConversationalTask (태스크가 대화식일 수 있는지 여부)	MQCCT_NO	0
TaskEnd상태 (태스크 종료 시 상태)	MQCTES_NOSYNC	0
기능 (브릿지 기능 토큰)	MQCFAC_NONE	널
함수 (MQ 호출 이름 또는 CICS EIBFN 함수)	MQCFUNC_NONE	공백
AbendCode (이상 종료 코드)	없음	공백
인증자 (비밀번호 또는 패스티켓)	없음	공백
Reserved1 (예약됨)	없음	공백
ReplyTo형식 (응답 메시지의 MQ 형식 이름)	MQFMT_NONE	공백
RemoteSysId (사용할 원격 CICS 시스템 ID)	없음	공백
RemoteTransID (사용할 CICS RTRANSID)	없음	공백
TransactionId (접속할 트랜잭션)	없음	공백
FacilityLike (터미널 에뮬레이트 속성)	없음	공백
AttentionId (AID키)	없음	공백
StartCode (트랜잭션 시작 코드)	MQCSC_NONE	공백
CancelCode (이상 종료 트랜잭션 코드)	없음	공백
NextTransactionId (연결할 다음 트랜잭션)	없음	공백
Reserved2 (예약됨)	없음	공백
Reserved3 (예약됨)	없음	공백
<b>참고:</b> <i>Version</i> 이 MQCIH_VERSION_2 미만이면 나머지 필드가 없습니다.		
CursorPosition (커서 위치)	없음	0
ErrorOffset (메시지의 오류 오프셋)	없음	0
InputItem (입력 항목)	없음	0
Reserved4 (예약됨)	없음	0

표 477. MQCIH에서 MQCIH의 필드 (계속)

필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
<p><b>참고:</b></p> <p>1. ~ 기호는 단일 공백 문자를 나타냅니다.</p> <p>2. C 프로그래밍 언어의 매크로 변수MQCIH_DEFAULT에는 표에 나열된 값이 포함됩니다. 구조의 필드에 초기 값을 제공하기 위해 다음 방법으로 사용하십시오.</p> <pre style="background-color: #f0f0f0; padding: 5px;">MQCIH MyCIH = {MQCIH_DEFAULT};</pre>		

## 언어 선언

### MQCIH의 C 선언

```
typedef struct tagMQCIH MQCIH;
struct tagMQCIH {
    MQCHAR4  StrucId;           /* Structure identifier */
    MQLONG   Version;         /* Structure version number */
    MQLONG   StrucLength;     /* Length of MQCIH structure */
    MQLONG   Encoding;       /* Reserved */
    MQLONG   CodedCharSetId;  /* Reserved */
    MQCHAR8  Format;          /* MQ format name of data that follows
    MQCIH */
    MQLONG   Flags;          /* Flags */
    MQLONG   ReturnCode;     /* Return code from bridge */
    MQLONG   CompCode;       /* MQ completion code or CICS EIBRESP */
    MQLONG   Reason;        /* MQ reason or feedback code, or CICS
    EIBRESP2 */
    MQLONG   UOWControl;     /* Unit-of-work control */
    MQLONG   GetWaitInterval; /* Wait interval for MQGET call issued
    by bridge task */
    MQLONG   LinkType;      /* Link type */
    MQLONG   OutputDataLength; /* Output COMMAREA data length */
    MQLONG   FacilityKeepTime; /* Bridge facility release time */
    MQLONG   ADSDescriptor;  /* Send/receive ADS descriptor */
    MQLONG   ConversationalTask; /* Whether task can be conversational */
    MQLONG   TaskEndStatus;  /* Status at end of task */
    MQBYTE8  Facility;      /* Bridge facility token */
    MQCHAR4  Function;      /* MQ call name or CICS EIBFN
    function */
    MQCHAR4  AbendCode;     /* Abend code */
    MQCHAR8  Authenticator; /* Password or passticket */
    MQCHAR8  Reserved1;     /* Reserved */
    MQCHAR8  ReplyToFormat; /* MQ format name of reply message */
    MQCHAR4  RemoteSysId;   /* Reserved */
    MQCHAR4  RemoteTransId; /* Reserved */
    MQCHAR4  TransactionId; /* Transaction to attach */
    MQCHAR4  FacilityLike;  /* Terminal emulated attributes */
    MQCHAR4  AttentionId;  /* AID key */
    MQCHAR4  StartCode;     /* Transaction start code */
    MQCHAR4  CancelCode;   /* Abend transaction code */
    MQCHAR4  NextTransactionId; /* Next transaction to attach */
    MQCHAR8  Reserved2;    /* Reserved */
    MQCHAR8  Reserved3;    /* Reserved */
    MQLONG   CursorPosition; /* Cursor position */
    MQLONG   ErrorOffset;   /* Offset of error in message */
    MQLONG   InputItem;    /* Reserved */
    MQLONG   Reserved4;    /* Reserved */
};
```

### MQCIH의 COBOL 선언

```
** MQCIH structure
   10 MQCIH.
**   Structure identifier
   15 MQCIH-STRUCID          PIC X(4).
**   Structure version number
   15 MQCIH-VERSION        PIC S9(9) BINARY.
```

```

** Length of MQCIH structure
15 MQCIH-STRUCLLENGTH PIC S9(9) BINARY.
** Reserved
15 MQCIH-ENCODING PIC S9(9) BINARY.
** Reserved
15 MQCIH-CODEDCHARSETID PIC S9(9) BINARY.
** MQ format name of data that follows MQCIH
15 MQCIH-FORMAT PIC X(8).
** Flags
15 MQCIH-FLAGS PIC S9(9) BINARY.
** Return code from bridge
15 MQCIH-RETURNCODE PIC S9(9) BINARY.
** MQ completion code or CICS EIBRESP
15 MQCIH-COMPCODE PIC S9(9) BINARY.
** MQ reason or feedback code, or CICS EIBRESP2
15 MQCIH-REASON PIC S9(9) BINARY.
** Unit-of-work control
15 MQCIH-UOWCONTROL PIC S9(9) BINARY.
** Wait interval for MQGET call issued by bridge task
15 MQCIH-GETWAITINTERVAL PIC S9(9) BINARY.
** Link type
15 MQCIH-LINKTYPE PIC S9(9) BINARY.
** Output COMMAREA data length
15 MQCIH-OUTPUTDATALENGTH PIC S9(9) BINARY.
** Bridge facility release time
15 MQCIH-FACILITYKEEPTIME PIC S9(9) BINARY.
** Send/receive ADS descriptor
15 MQCIH-ADSDESCRIPTOR PIC S9(9) BINARY.
** Whether task can be conversational
15 MQCIH-CONVERSATIONALTASK PIC S9(9) BINARY.
** Status at end of task
15 MQCIH-TASKENDSTATUS PIC S9(9) BINARY.
** Bridge facility token
15 MQCIH-FACILITY PIC X(8).
** MQ call name or CICS EIBFN function
15 MQCIH-FUNCTION PIC X(4).
** Abend code
15 MQCIH-ABENDCODE PIC X(4).
** Password or passticket
15 MQCIH-AUTHENTICATOR PIC X(8).
** Reserved
15 MQCIH-RESERVED1 PIC X(8).
** MQ format name of reply message
15 MQCIH-REPLYTOFORMAT PIC X(8).
** Reserved
15 MQCIH-REMOTESYSID PIC X(4).
** Reserved
15 MQCIH-REMOTETRANSID PIC X(4).
** Transaction to attach
15 MQCIH-TRANSACTIONID PIC X(4).
** Terminal emulated attributes
15 MQCIH-FACILITYLIKE PIC X(4).
** AID key
15 MQCIH-ATTENTIONID PIC X(4).
** Transaction start code
15 MQCIH-STARTCODE PIC X(4).
** Abend transaction code
15 MQCIH-CANCELCODE PIC X(4).
** Next transaction to attach
15 MQCIH-NEXTTRANSACTIONID PIC X(4).
** Reserved
15 MQCIH-RESERVED2 PIC X(8).
** Reserved
15 MQCIH-RESERVED3 PIC X(8).
** Cursor position
15 MQCIH-CURSORPOSITION PIC S9(9) BINARY.
** Offset of error in message
15 MQCIH-ERROROFFSET PIC S9(9) BINARY.
** Reserved
15 MQCIH-INPUTITEM PIC S9(9) BINARY.
** Reserved
15 MQCIH-RESERVED4 PIC S9(9) BINARY.

```

## MQCIH의 PL/I 선언

```

dcl
  1 MQCIH based,
  3 StrucId          char(4),          /* Structure identifier */
  3 Version          fixed bin(31), /* Structure version number */

```

```

3 StrucLength      fixed bin(31), /* Length of MQCIH structure */
3 Encoding         fixed bin(31), /* Reserved */
3 CodedCharSetId  fixed bin(31), /* Reserved */
3 Format           char(8), /* MQ format name of data that
                          follows MQCIH */
3 Flags           fixed bin(31), /* Flags */
3 ReturnCode      fixed bin(31), /* Return code from bridge */
3 CompCode       fixed bin(31), /* MQ completion code or CICS
                          EIBRESP */
3 Reason         fixed bin(31), /* MQ reason or feedback code, or
                          CICS EIBRESP2 */
3 UOWControl     fixed bin(31), /* Unit-of-work control */
3 GetWaitInterval fixed bin(31), /* Wait interval for MQGET call
                          issued by bridge task */
3 LinkType       fixed bin(31), /* Link type */
3 OutputDataLength fixed bin(31), /* Output COMMAREA data length */
3 FacilityKeepTime fixed bin(31), /* Bridge facility release time */
3 ADSDescriptor  fixed bin(31), /* Send/receive ADS descriptor */
3 ConversationalTask fixed bin(31), /* Whether task can be
                          conversational */
3 TaskEndStatus  fixed bin(31), /* Status at end of task */
3 Facility       char(8), /* Bridge facility token */
3 Function       char(4), /* MQ call name or CICS EIBFN
                          function */
3 AbendCode     char(4), /* Abend code */
3 Authenticator  char(8), /* Password or passticket */
3 Reserved1     char(8), /* Reserved */
3 ReplyToFormat  char(8), /* MQ format name of reply
                          message */
3 RemoteSysId   char(4), /* Reserved */
3 RemoteTransId char(4), /* Reserved */
3 TransactionId char(4), /* Transaction to attach */
3 FacilityLike  char(4), /* Terminal emulated attributes */
3 AttentionId   char(4), /* AID key */
3 StartCode     char(4), /* Transaction start code */
3 CancelCode    char(4), /* Abend transaction code */
3 NextTransactionId char(4), /* Next transaction to attach */
3 Reserved2     char(8), /* Reserved */
3 Reserved3     char(8), /* Reserved */
3 CursorPosition fixed bin(31), /* Cursor position */
3 ErrorOffset   fixed bin(31), /* Offset of error in message */
3 InputItem     fixed bin(31), /* Reserved */
3 Reserved4     fixed bin(31); /* Reserved */

```

## MQCIH의 상위 레벨 어셈블러 선언

```

MQCIH          DSECT
MQCIH_STRUCID  DS    CL4  Structure identifier
MQCIH_VERSION DS    F    Structure version number
MQCIH_STRULENGTH DS    F    Length of MQCIH structure
MQCIH_ENCODING DS    F    Reserved
MQCIH_CODEDCHARSETID DS    F    Reserved
MQCIH_FORMAT   DS    CL8  MQ format name of data that follows
*              MQCIH
MQCIH_FLAGS    DS    F    Flags
MQCIH_RETURNCODE DS    F    Return code from bridge
MQCIH_COMPCODE DS    F    MQ completion code or CICS EIBRESP
MQCIH_REASON   DS    F    MQ reason or feedback code, or CICS
*              EIBRESP2
MQCIH_UOWCONTROL DS    F    Unit-of-work control
MQCIH_GETWAITINTERVAL DS    F    Wait interval for MQGET call issued
*              by bridge task
MQCIH_LINKTYPE DS    F    Link type
MQCIH_OUTPUTDATALENGTH DS    F    Output COMMAREA data length
MQCIH_FACILITYKEEPTIME DS    F    Bridge facility release time
MQCIH_ADSDSCRIPTOR DS    F    Send/receive ADS descriptor
MQCIH_CONVERSATIONALTASK DS    F    Whether task can be conversational
MQCIH_TASKENDSTATUS DS    F    Status at end of task
MQCIH_FACILITY DS    XL8  Bridge facility token
MQCIH_FUNCTION DS    CL4  MQ call name or CICS EIBFN function
MQCIH_ABENDCODE DS    CL4  Abend code
MQCIH_AUTHENTICATOR DS    CL8  Password or passticket
MQCIH_RESERVED1 DS    CL8  Reserved
MQCIH_REPLYTOFORMAT DS    CL8  MQ format name of reply message
MQCIH_REMOTESYSID DS    CL4  Reserved
MQCIH_REMOTETRANSID DS    CL4  Reserved
MQCIH_TRANSACTIONID DS    CL4  Transaction to attach
MQCIH_FACILITYLIKE DS    CL4  Terminal emulated attributes
MQCIH_ATTENTIONID DS    CL4  AID key

```



MQCIH_STARTCODE	DS	CL4	Transaction start code
MQCIH_CANCELCODE	DS	CL4	Abend transaction code
MQCIH_NEXTTRANSACTIONID	DS	CL4	Next transaction to attach
MQCIH_RESERVED2	DS	CL8	Reserved
MQCIH_RESERVED3	DS	CL8	Reserved
MQCIH_CURSORPOSITION	DS	F	Cursor position
MQCIH_ERROROFFSET	DS	F	Offset of error in message
MQCIH_INPUTITEM	DS	F	Reserved
MQCIH_RESERVED4	DS	F	Reserved
*			
MQCIH_LENGTH	EQU	*-MQCIH	
	ORG	MQCIH	
MQCIH_AREA	DS	CL(MQCIH_LENGTH)	

## MQCIH의 Visual Basic 선언

```

Type MQCIH
  StrucId      As String*4 'Structure identifier'
  Version     As Long      'Structure version number'
  StrucLength  As Long      'Length of MQCIH structure'
  Encoding    As Long      'Reserved'
  CodedCharSetId As Long    'Reserved'
  Format      As String*8  'MQ format name of data that follows'
  'MQCIH'
  Flags      As Long      'Flags'
  ReturnCode As Long      'Return code from bridge'
  CompCode   As Long      'MQ completion code or CICS EIBRESP'
  Reason     As Long      'MQ reason or feedback code, or CICS'
  'EIBRESP2'
  UOWControl As Long      'Unit-of-work control'
  GetWaitInterval As Long  'Wait interval for MQGET call issued'
  'by bridge task'
  LinkType   As Long      'Link type'
  OutputDataLength As Long  'Output COMMAREA data length'
  FacilityKeepTime As Long  'Bridge facility release time'
  ADSDescriptor As Long    'Send/receive ADS descriptor'
  ConversationalTask As Long  'Whether task can be conversational'
  TaskEndStatus As Long    'Status at end of task'
  Facility      As MQBYTE8  'Bridge facility token'
  Function     As String*4  'MQ call name or CICS EIBFN function'
  AbendCode   As String*4  'Abend code'
  Authenticator As String*8  'Password or passticket'
  Reserved1   As String*8  'Reserved'
  ReplyToFormat As String*8  'MQ format name of reply message'
  RemoteSysId As String*4  'Reserved'
  RemoteTransId As String*4  'Reserved'
  TransactionId As String*4  'Transaction to attach'
  FacilityLike As String*4  'Terminal emulated attributes'
  AttentionId  As String*4  'AID key'
  StartCode   As String*4  'Transaction start code'
  CancelCode  As String*4  'Abend transaction code'
  NextTransactionId As String*4  'Next transaction to attach'
  Reserved2   As String*8  'Reserved'
  Reserved3   As String*8  'Reserved'
  CursorPosition As Long    'Cursor position'
  ErrorOffset  As Long      'Offset of error in message'
  InputItem   As Long      'Reserved'
  Reserved4   As Long      'Reserved'
End Type

```

### MQCIH용 StrucId (MQCHAR4)

CICS 정보 헤더 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQCIH\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

### MQCIH\_STRUC\_ID

CICS 정보 헤더 구조의 ID.

C 프로그래밍 언어의 경우 상수 MQCIH\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQCIH\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### MQCIH의 버전 (MQLONG)

이 필드는 요청 필드입니다. 초기값은 MQCIH\_VERSION\_2입니다.

값은 다음 중 하나여야 합니다.

**MQCIH\_VERSION\_1**

버전-1 CICS 정보 헤더 구조.

**MQCIH\_VERSION\_2**

버전-2 CICS 정보 헤더 구조.

최신 버전의 구조에만 있는 필드는 필드의 설명에서 최신 필드로 식별됩니다. 다음 상수는 현재 버전의 버전 번호를 지정합니다.

**MQCIH\_CURRENT\_VERSION**

CICS 정보 헤더 구조의 현재 버전.

**MQCIH용 StrucLength (MQLONG)**

이 필드는 초기값이 MQCIH\_LENGTH\_2인 요청 필드입니다.

값은 다음 중 하나여야 합니다.

**MQCIH\_LENGTH\_1**

버전-1 CICS 정보 헤더 구조의 길이.

**MQCIH\_LENGTH\_2**

버전-2 CICS 정보 헤더 구조의 길이.

다음 상수는 현재 버전의 길이를 지정합니다.

**MQCIH\_CURRENT\_LENGTH**

CICS 정보 헤더 구조의 현재 버전 길이.

**MQCIH용 인코딩 (MQLONG)**

이 필드는 예약된 필드이며, 해당 값은 중요하지 않습니다. 초기 값은 0입니다.

MQCIH 구조를 따르는 지원 구조에 대한 인코딩은 MQCIH 구조 자체의 인코딩과 동일하며, 선행 IBM MQ 헤더에서 가져옵니다.

**MQCIH용 CodedCharSetId (MQLONG)**

CodedCharSetId는 예약된 필드이며, 해당 값은 중요하지 않습니다. 이 필드의 초기값은 0입니다.

MQCIH 구조를 따르는 지원 구조에 대한 문자 세트 ID는 MQCIH 구조 자체의 문자 세트 ID와 동일하며, 선행 IBM MQ 헤더에서 가져옵니다.

**MQCIH의 형식 (MQCHAR8)**

이 필드는 MQCIH 구조를 따르는 데이터의 IBM MQ 형식 이름을 표시합니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 이 필드를 코드화하는 규칙은 MQMD의 *Format* 필드를 코드화하는 규칙과 동일합니다.

*ReplyToFormat* 필드의 값이 MQFMT\_NONE인 경우, 이 형식 이름은 응답 메시지에도 사용됩니다.

- DPL 요청의 경우 *Format*은 COMMAREA의 형식 이름이어야 합니다.
- 3270 요청의 경우 *Format*은 CSQCBDCI이어야 하며, 브릿지는 응답 메시지에 대해 CSQCBDCO로 형식을 설정합니다.

해당 형식에 대한 데이터 변환 엑시트는 실행할 큐 관리자에 설치되어야 합니다.

요청 메시지가 오류 응답 메시지를 생성하는 경우, 오류 응답 메시지의 형식 이름은 MQFMT\_STRING입니다.

이 필드는 요청 필드입니다. 이 필드의 길이는 MQ\_FORMAT\_LENGTH에 지정되어 있습니다. 이 필드의 초기값은 MQFMT\_NONE입니다.

**MQCIH의 플래그 (MQLONG)**

이 필드는 요청 필드입니다. 이 필드의 초기값은 MQCIH\_NONE입니다.

값은 다음과 같아야 합니다.

#### **MQCIH\_NONE**

플래그가 없습니다.

#### **MQCIH\_PASS\_EXPIRATION**

응답 메시지는 다음을 포함합니다.

- 요청 메시지와 동일한 만기 보고서 옵션.
- 브릿지의 처리 시간에 대해 조정되지 않은 요청 메시지의 나머지 만기 시간.

이 값을 생략하면 만기 시간이 무제한으로 설정됩니다.

#### **MQCIH\_REPLY\_WITHOUT\_NULLS**

CICS DPL 프로그램 요청의 응답 메시지 길이는 DPL 프로그램이 리턴한 COMMAREA의 끝에서 후미 문자 널 (X'00')을 제외하도록 조정됩니다. 이 값이 설정되지 않은 경우, 널이 중요할 수 있으며 전체 COMMAREA가 리턴됩니다.

#### **MQCIH\_SYNC\_ON\_RETURN**

DPL 요청의 CICS 링크가 SYNCONRETURN 옵션을 사용하므로, CICS는 다른 CICS 리전에 제공된 경우 프로그램이 완료될 때 동기점을 취합니다. 브릿지는 요청을 제공할 CICS 리전을 지정하지 않습니다. 이는 CICS 프로그램 정의 또는 워크로드 밸런싱 기능의 제어를 받습니다.

#### **MQCIH의 경우 ReturnCode (MQLONG)**

이 필드의 값은 브릿지가 수행하는 처리의 결과를 설명하는 CICS bridge의 리턴 코드입니다. 이 필드는 응답 필드이며 초기값은 MQCRC\_OK입니다.

*Function, CompCode, Reason* 및 *AbendCode* 필드에는 추가 정보가 포함될 수 있습니다([296 페이지의 표 476](#) 참조). 값은 다음 중 하나입니다.

#### **MQCRC\_APPLICATION\_ABEND**

(5, X'005') 애플리케이션이 비정상적으로 종료되었습니다.

#### **MQCRC\_BRIDGE\_ABEND**

(4, X'004') CICS bridge가 비정상적으로 종료되었습니다.

#### **MQCRC\_BRIDGE\_ERROR**

(3, X'003') CICS bridge가 오류를 감지했습니다.

#### **MQCRC\_BRIDGE\_TIMEOUT**

(8, X'008') 지정된 시간 내에 수신된 현재 작업 단위 내에 있는 두 번째 또는 이후 버전의 메시지입니다.

#### **MQCRC\_CICS\_EXEC\_ERROR**

(1, X'001') EXEC CICS 명령문이 오류를 감지했습니다.

#### **MQCRC\_MQ\_API\_ERROR**

(2, X'002') MQ 호출에서 오류를 감지했습니다.

#### **MQCRC\_OK**

(0, X'000') 오류가 없습니다.

#### **MQCRC\_PROGRAM\_NOT\_AVAILABLE**

(7, X'007') 프로그램을 사용할 수 없습니다.

#### **MQCRC\_SECURITY\_ERROR**

(6, X'006') 보안 오류가 발생했습니다.

#### **MQCRC\_TRANSID\_NOT\_AVAILABLE**

(9, X'009') 트랜잭션을 사용할 수 없습니다.

#### **MQCIH용 CompCode (MQLONG)**

이 필드는 응답 필드입니다. 초기값은 MQCC\_OK입니다.

이 필드에서 리턴된 값은 *ReturnCode*에 의존합니다. [296 페이지의 표 476](#)를 참조하십시오.

#### **MQCIH에 대한 이유 (MQLONG)**

이 필드는 응답 필드입니다. 초기값은 MQRC\_NONE입니다.

이 필드에서 리턴된 값은 *ReturnCode*에 의존합니다. [296 페이지의 표 476](#)를 참조하십시오.

### **MQCIH용 UOWControl (MQLONG)**

이 필드는 CICS bridge에서 수행하는 작업 단위 처리를 제어하는 요청 필드입니다. 이 필드의 초기값은 MQCUOWC\_ONLY입니다.

단일 트랜잭션 또는 작업 단위 내에 있는 하나 이상의 프로그램을 실행하기 위해 브릿지를 요청할 수 있습니다. 이 필드는 CICS bridge가 작업 단위를 시작하고 현재 작업 단위 내에서 요청된 함수를 수행하거나 이를 커밋 또는 백아웃하여 작업 단위를 종료하는지 여부를 표시합니다. 다양한 결합은 데이터 전송 플로우를 최적화하기 위해 지원됩니다.

값은 다음 중 하나여야 합니다.

#### **MQCUOWC\_ONLY**

작업 단위를 시작하고 함수를 수행한 후에 작업 단위를 커밋합니다.

#### **MQCUOWC\_CONTINUE**

현재 작업 단위에 대한 추가 데이터입니다(3270 전용).

#### **MQCUOWC\_FIRST**

작업 단위를 시작하고 함수를 수행합니다.

#### **MQCUOWC\_MIDDLE**

현재 작업 단위 내에서 함수를 수행합니다.

#### **MQCUOWC\_LAST**

함수를 수행한 후에 작업 단위를 커밋합니다.

#### **MQCUOWC\_COMMIT**

작업 단위를 커밋합니다(DPL 전용).

#### **MQCUOWC\_BACKOUT**

작업 단위를 백아웃합니다(DPL 전용).

### **MQCIH에 대한 GetWait간격 (MQLONG)**

이 필드는 요청 필드입니다. 초기값은 MQCGWI\_DEFAULT입니다.

이 필드는 UOWControl이 MQCUOWC\_FIRST 값을 보유할 때만 적용됩니다. 이는 브릿지가 실행한 MQGET 호출이 이 메시지가 시작한 작업 단위에 대해 두 번째 및 후속 요청 메시지를 대기하는 개략적 시간(밀리초)을 송신 애플리케이션이 지정할 수 있도록 합니다. 이 기능은 브릿지가 사용하는 기본 대기 간격을 대체합니다. 다음 특수 값을 사용할 수 있습니다.

#### **MQCGWI\_DEFAULT**

기본 대기 간격.

이 값은 브릿지가 시작될 때 지정된 시간 동안 CICS bridge가 대기하도록 합니다.

#### **MQWI\_UNLIMITED**

무제한 대기 간격.

### **MQCIH용 LinkType (MQLONG)**

이 필드는 요청 필드입니다. 초기값은 MQCLT\_PROGRAM입니다.

이 값은 브릿지가 링크를 시도하는 오브젝트의 유형을 표시합니다. 다음 값 중 하나여야 합니다.

#### **MQCLT\_PROGRAM**

DPL 프로그램.

#### **MQCLT\_TRANSACTION**

3270 트랜잭션.

### **MQCIH의 OutputData길이 (MQLONG)**

이 필드는 DPL 프로그램에만 사용되는 요청 필드입니다. 초기값은 MQCODL\_AS\_INPUT입니다.

이 값은 응답 메시지에서 클라이언트로 리턴되는 사용자 데이터의 길이입니다. 이 길이에는 8바이트 프로그램 이름이 포함됩니다. 링크된 프로그램으로 전달된 COMMAREA의 길이는 이 필드의 최대 값 및 요청 메시지의 사용자 데이터 길이에서 8을 뺀 수입니다.

**참고:** 메시지에서 사용자 데이터의 길이는 MQCIH 구조를 제외한 메시지의 길이입니다.

요청 메시지에서 사용자 데이터의 길이가 *OutputDataLength* 미만이면 LINK 명령의 DATALENGTH 옵션이 사용되며, LINK가 다른 CICS 리전에 대해 효율적으로 기능 제공되도록 합니다.

다음 특수 값을 사용할 수 있습니다.

#### **MQCODL\_AS\_INPUT**

출력 길이가 입력 길이와 동일합니다.

이 값은 링크된 프로그램에 전달된 COMMAREA가 충분한 크기인지 확인하기 위해 응답이 요청되지 않아도 필요할 수 있습니다.

#### **FacilityKeepMQCIH의 시간 (MQLONG)**

FacilityKeepTime은 사용자 트랜잭션이 종료된 이후 브릿지 기능이 유지되는 시간의 길이(초)입니다.

의사 대화식 트랜잭션의 경우, 의사 대화의 예상 기간에 대응되는 값을 지정하십시오. 의사 대화의 마지막 트랜잭션에 대해 0을 지정하고, 기타 트랜잭션 유형에 대해 0을 지정하십시오.

이 필드는 3270 트랜잭션에만 사용되는 요청 필드입니다. 이 필드의 초기값은 0입니다.

#### **MQCIH용 ADSDescriptor (MQLONG)**

이 필드는 SEND 및 RECEIVE BMS 요청에서 ADS 디스크립터를 전송할지 지정하는 표시기입니다.

다음 값이 정의됩니다.

#### **MQCADSD\_NONE**

ADS 디스크립터를 송신 또는 수신하지 않습니다.

#### **MQCADSD\_SEND**

ADS 디스크립터를 송신합니다.

#### **MQCADSD\_RECV**

ADS 디스크립터를 수신합니다.

#### **MQCADSD\_MSGFORMAT**

ADS 디스크립터의 메시지 형식을 사용합니다.

이는 ADS 디스크립터의 긴 양식을 사용하여 ADS 디스크립터를 송신하거나 수신합니다. 긴 양식에는 4바이트 경계에서 맞추어진 필드가 있습니다.

다음과 같이 *ADSDescriptor* 필드를 설정하십시오.

- ADS 디스크립터를 사용 중이 아니면 필드를 MQCADSD\_NONE으로 설정하십시오.
- 각 환경에서 동일한 CCSID의 ADS 디스크립터를 사용 중이면 필드를 MQCADSD\_SEND 및 MQCADSD\_RECV의 합으로 설정하십시오.
- 각 환경에서 상이한 CCSID의 ADS 디스크립터를 사용 중이면 필드를 MQCADSD\_SEND, MQCADSD\_RECV 및 MQCADSD\_MSGFORMAT의 합으로 설정하십시오.

이 필드는 3270 트랜잭션에만 사용된 요청 필드입니다. 이 필드의 초기값은 MQCADSD\_NONE입니다.

#### **MQCIH용 ConversationalTask (MQLONG)**

이 필드는 태스크가 추가 정보에 대한 요청을 발행하도록 허용할지 또는 태스크를 중지하고 이상 종료 메시지를 발행할지를 지정하는 표시기입니다.

값은 다음 옵션 중 하나여야 합니다.

#### **MQCCT\_YES**

태스크가 대화식입니다.

#### **MQCCT\_NO**

태스크가 대화식이 아닙니다.

이 필드는 3270 트랜잭션에만 사용되는 요청 필드입니다. 이 필드의 초기값은 MQCCT\_NO입니다.

#### **MQCIH에 대한 TaskEnd상태 (MQLONG)**

이 필드는 태스크의 종료 시에 사용자 트랜잭션의 상태를 표시하는 응답 필드입니다. 이 필드는 3270 트랜잭션에만 사용되며 초기값은 MQCTES\_NOSYNC입니다.

다음 값 중 하나가 리턴됩니다.

#### **MQCTES\_NOSYNC**

동기화되지 않습니다.

사용자 트랜잭션은 아직 완료되지 않았으며 동기점이 없습니다. MQMD의 *MsgType* 필드는 이 경우에 MQMT\_REQUEST입니다.

#### **MQCTES\_COMMIT**

작업 단위를 커밋합니다.

사용자 트랜잭션은 아직 완료되지 않았지만 첫 번째 작업 단위의 동기점이 조정됩니다. MQMD의 *MsgType* 필드는 이 경우에 MQMT\_DATAGRAM입니다.

#### **MQCTES\_BACKOUT**

작업 단위를 백아웃합니다.

사용자 트랜잭션이 아직 완료되지 않았습니다. 현재 작업 단위가 백아웃됩니다. MQMD의 *MsgType* 필드는 이 경우에 MQMT\_DATAGRAM입니다.

#### **MQCTES\_ENDTASK**

태스크를 종료합니다.

사용자 트랜잭션이 종료되었습니다(또는 이상종료됨). MQMD의 *MsgType* 필드는 이 경우에 MQMT\_REPLY입니다.

### **MQCIH용 기능 (MQBYTE8)**

이 필드는 8바이트 브릿지 기능 토큰을 표시합니다.

브릿지 기능 토큰을 사용하여 의사 대화의 다수의 트랜잭션은 동일한 브릿지 기능(가상 3270 터미널)을 사용할 수 있습니다. 의사 대화의 최초의 또는 유일한 메시지에서 MQCFAC\_NONE의 값을 설정하십시오. 이 값은 이 메시지에 대해 새 브릿지 기능을 할당하도록 CICS에 알립니다. 브릿지 기능 토큰은 0이 아닌 *FacilityKeepTime*이 입력 메시지에 지정될 때 응답 메시지에서 리턴됩니다. 그리고 의사 대화 내의 후속 입력 메시지는 동일한 브릿지 기능 토큰을 사용해야 합니다.

다음 특수 값이 정의됩니다.

#### **MQCFAC\_NONE**

기능 토큰이 지정되지 않습니다.

C 프로그래밍 언어의 경우, MQCFAC\_NONE\_ARRAY 상수 역시 정의되며 MQCFAC\_NONE과 동일한 값을 갖지만 문자열이 아닌 문자의 배열입니다.

이 필드는 3270 트랜잭션에만 사용되는 요청 및 응답 필드입니다. 이 필드의 길이는 MQ\_FACILITY\_LENGTH에서 제공됩니다. 이 필드의 초기값은 MQCFAC\_NONE입니다.

### **MQCIH의 함수 (MQCHAR4)**

이 필드는 응답 필드입니다. 이 필드의 길이는 MQ\_FUNCTION\_LENGTH에서 제공됩니다. 이 필드의 초기값은 MQCFUNC\_NONE입니다.

이 필드에서 리턴된 값은 *ReturnCode*에 의존합니다. [296 페이지의 표 476](#)를 참조하십시오. *Function*에 IBM MQ 호출 이름이 포함되면 다음 값이 가능합니다.

#### **MQCFUNC\_MQCONN**

MQCONN 호출입니다.

#### **MQCFUNC\_MQGET**

MQGET 호출.

#### **MQCFUNC\_MQINQ**

MQINQ 호출입니다.

#### **MQCFUNC\_MQOPEN**

MQOPEN 호출.

#### **MQCFUNC\_MQPUT**

MQPUT 호출.

## **MQCFUNC\_MQPUT1**

MQPUT1 호출.

## **MQCFUNC\_NONE**

호출 없음.

모든 경우, C 프로그래밍 언어에 대해 MQCFUNC\_\*\_ARRAY 상수도 정의됩니다. 이러한 상수는 대응되는 MQCFUNC\_\* 상수와 동일한 값을 갖지만, 문자열 대신 문자의 배열입니다.

## **MQCIH의 AbendCode (MQCHAR4)**

AbendCode는 응답 필드입니다. 이 필드의 길이는 MQ\_ABEND\_CODE\_LENGTH에서 제공됩니다. 이 필드의 초기값은 4개의 공백 문자입니다.

이 필드에서 리턴된 값은 ReturnCode 필드의 값이 MQCRC\_APPLICATION\_ABEND 또는 MQCRC\_BRIDGE\_ABEND인 경우에만 중요합니다. 포함하는 경우 AbendCode에는 CICS ABCODE값이 포함됩니다.

## **MQCIH용 인증자 (MQCHAR8)**

이 필드의 값은 비밀번호 또는 패스 티켓입니다.

사용자 ID 인증이 CICS bridge에 대해 활성화인 경우에는 Authenticator를 MQMD ID 컨텍스트의 사용자 ID와 함께 사용하여 메시지의 송신자를 인증합니다.

이 필드는 요청 필드입니다. 이 필드의 길이는 MQ\_AUTHENTICATOR\_LENGTH로 지정됩니다. 이 필드의 초기값은 8개의 공백입니다.

## **MQCIH의 경우 Reserved1 (MQCHAR8)**

이 필드는 예약 필드입니다. 값은 8개의 공백이어야 합니다.

## **MQCIH의 ReplyTo형식 (MQCHAR8)**

이 필드의 값은 현재 메시지에 대한 응답으로 송신된 응답 메시지의 IBM MQ 형식 이름입니다.

이 필드를 코딩하는 규칙은 MQMD의 Format 필드를 코딩하는 규칙과 같습니다.

이 필드는 DPL 프로그램에만 사용되는 요청 필드입니다. 이 필드의 길이는 MQ\_FORMAT\_LENGTH에 지정되어 있습니다. 이 필드의 초기값은 MQFMT\_NONE입니다.

## **MQCIH의 RemoteSysID (MQCHAR4)**

이 필드는 요청을 처리 중인 CICS 시스템의 CICS 시스템 ID를 표시합니다.

이 필드가 공백인 경우, CICS 시스템 요청은 브릿지 모니터와 동일한 CICS 시스템에서 처리됩니다. 사용된 SYSID는 응답 메시지에서 리턴됩니다.

3270 의사 대화의 경우, 대화의 모든 후속 메시지는 초기 응답에서 리턴된 원격 SYSID를 지정해야 합니다. 지정된 경우, SYSID는 다음과 같아야 합니다.

- 활성화입니다.
- IBM MQ 요청 큐에 대한 액세스 권한을 갖습니다.
- 브릿지 모니터의 CICS 시스템에서 CICS ISC 링크에 의해 액세스 가능합니다.

## **MQCIH의 RemoteTransID (MQCHAR4)**

이 필드는 선택적 요청 필드입니다. 이 필드의 길이는 MQ\_TRANSACTION\_ID\_LENGTH에서 제공됩니다.

지정된 경우, 필드는 CICS START의 RTRANSID 값으로 사용됩니다.

## **MQCIH용 TransactionId (MQCHAR4)**

이 필드는 요청 필드입니다. 해당 길이는 MQ\_TRANSACTION\_ID\_LENGTH에서 제공됩니다. 이 필드의 초기값은 4개의 공백입니다.

LinkType의 값이 MQCLT\_TRANSACTION인 경우, TransactionId는 실행될 사용자 트랜잭션의 트랜잭션 ID입니다. 이 경우에 공백이 아닌 값을 지정하십시오.

*LinkType*의 값이 MQCLT\_PROGRAM인 경우, *TransactionId*는 작업 단위 내의 모든 프로그램이 실행되는 트랜잭션 코드입니다. 공백 값을 지정하는 경우에는 CICS DPL 브릿지 기본 트랜잭션 코드(CKBP)가 사용됩니다. 값이 공백이 아닌 경우, 사용자는 이를 CSQCBP00인 초기 프로그램에서 로컬 트랜잭션으로서 CICS로 정의해야 합니다. 이 필드는 *UOWControl*의 값이 MQCUOWC\_FIRST 또는 MQCUOWC\_ONLY인 경우에만 적용됩니다.

### **MQCIH의 FacilityLike (MQCHAR4)**

FacilityLike는 브릿지 기능의 모델로 사용되는 설치된 터미널의 이름입니다.

공백 값은 *FacilityLike*를 브릿지 트랜잭션 프로파일 정의에서 가져오거나 기본값이 사용됨을 의미합니다.

이 필드는 3270 트랜잭션에만 사용되는 요청 필드입니다. 이 필드의 길이는 MQ\_FACILITY\_LIKE\_LENGTH에서 제공됩니다. 이 필드의 초기값은 4개의 공백입니다.

### **MQCIH용 AttentionId (MQCHAR4)**

이 필드의 값은 트랜잭션이 시작될 때 AID 키의 초기값을 판별합니다. 1바이트 값이며 왼쪽에 맞춰 정렬합니다.

AttentionId는 3270 트랜잭션에만 사용되는 요청 필드입니다. 이 필드의 길이는 MQ\_ATTENTION\_ID\_LENGTH에서 제공됩니다. 이 필드의 초기값은 4개의 공백입니다.

### **MQCIH용 StartCode (MQCHAR4)**

이 필드의 값은 START으로 시작된 트랜잭션 또는 터미널 트랜잭션을 브릿지가 에뮬레이트하는지 여부를 지정하는 표시기입니다.

값은 다음 중 하나여야 합니다.

#### **MQCSC\_START**

시작.

#### **MQCSC\_STARTDATA**

데이터 시작.

#### **MQCSC\_TERMINPUT**

터미널 입력.

#### **MQCSC\_NONE**

없음

모든 경우, C 프로그래밍 언어에 대해 MQCSC\_\*\_ARRAY 상수도 정의됩니다. 이러한 상수는 대응되는 MQCSC\_\* 상수와 동일한 값을 갖지만, 문자열 대신 문자의 배열입니다.

브릿지의 응답에서, 이 필드는 *NextTransactionId* 필드에 포함된 다음 트랜잭션 ID에 적절한 시작 코드로 설정됩니다. 다음 시작 코드는 다음 응답에 가능합니다.

- MQCSC\_START
- MQCSC\_STARTDATA
- MQCSC\_TERMINPUT

CICS Transaction Server 1.2의 경우 이 필드는 요청 필드 전용입니다. 응답에서 해당 값은 정의되어 있지 않습니다.

CICS Transaction Server 1.3 및 후속 릴리스의 경우 이 필드는 요청 필드인 동시에 응답 필드입니다.

이 필드는 3270 트랜잭션에만 사용됩니다. 이 필드의 길이는 MQ\_START\_CODE\_LENGTH에서 제공됩니다. 이 필드의 초기값은 MQCSC\_NONE입니다.

### **MQCIH의 경우 CancelCode (MQCHAR4)**

이 필드의 값은 트랜잭션(일반적으로 추가 데이터를 요청 중인 대화식 트랜잭션)을 종료하는 데 사용되는 이상종료 코드입니다. 그렇지 않으면 이 필드는 공백으로 설정됩니다.

이 필드는 3270 트랜잭션에만 사용되는 요청 필드입니다. 이 필드의 길이는 MQ\_CANCEL\_CODE\_LENGTH에서 제공됩니다. 이 필드의 초기값은 4개의 공백입니다.



### **MQCIH의 NextTransactionID (MQCHAR4)**

이 값은 사용자 트랜잭션에 의해 리턴된 다음 트랜잭션의 이름입니다(일반적으로 EXEC CICS RETURN TRANSID에 의해). 다음 트랜잭션이 없는 경우, 이 필드는 공백으로 설정됩니다.

이 필드는 3270 트랜잭션에만 사용되는 응답 필드입니다. 이 필드의 길이는 MQ\_TRANSACTION\_ID\_LENGTH에서 제공됩니다. 이 필드의 초기값은 4개의 공백입니다.

### **MQCIH의 경우 Reserved2 (MQCHAR8)**

이 필드는 예약 필드입니다. 값은 8개의 공백이어야 합니다.

### **MQCIH용 Reserved3 (MQCHAR8)**

이 필드는 예약 필드입니다. 값은 8개의 공백이어야 합니다.

### **MQCIH용 CursorPosition (MQLONG)**

이 필드의 값은 트랜잭션이 시작될 때 초기 커서 위치를 표시합니다. 대화식 트랜잭션의 경우 커서 위치는 RECEIVE 백터에 있습니다.

이 필드는 3270 트랜잭션에만 사용되는 요청 필드입니다. 이 필드의 초기값은 0입니다. *Version*이 MQCIH\_VERSION\_2 미만이면 이 필드가 존재하지 않습니다.

### **MQCIH의 경우 ErrorOffset (MQLONG)**

ErrorOffset 필드는 브릿지 엑시트에 의해 감지된 올바르지 않은 데이터의 위치를 표시합니다. 이 필드는 메시지 시작부터 유효하지 않은 데이터 위치까지의 오프셋을 제공합니다.

ErrorOffset은 3270 트랜잭션에만 사용되는 응답 필드입니다. 이 필드의 초기값은 0입니다. *Version*이 MQCIH\_VERSION\_2 미만이면 이 필드가 존재하지 않습니다.

### **MQCIH용 InputItem (MQLONG)**

이 필드는 예약 필드입니다. 값은 0이어야 합니다.

*Version*이 MQCIH\_VERSION\_2 미만이면 이 필드가 존재하지 않습니다.

### **MQCIH의 경우 Reserved4 (MQLONG)**

이 필드는 예약 필드입니다. 값은 0이어야 합니다.

*Version*이 MQCIH\_VERSION\_2 미만이면 이 필드가 존재하지 않습니다.

## **MQCMHO - 메시지 핸들 작성 옵션**

MQCMHO 구조를 사용하여 애플리케이션이 메시지 핸들을 작성하는 방법을 제어할 수 있습니다. 이 구조는 MQCRTMH 호출의 입력 매개변수입니다.

### **가용성**

MQCMHO 구조는 다음 플랫폼에서 사용 가능합니다.

-  AIX
-  IBM i
-  Linux
-  Windows
-  z/OS

및 해당 시스템에 연결된 IBM MQ MQI clients

## 문자 세트 및 인코딩

**MQCMHO**의 데이터는 애플리케이션의 문자 세트 및 애플리케이션의 인코딩 (**MQENC\_NATIVE**)에 있어야 합니다.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 478. MQCMHO의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQCMHO_STRUC_ID	'CMHO'
버전 (구조 버전 번호)	MQCMHO_VERSION_1	1
옵션 (옵션)	MQCMHO_DEFAULT_VAL IDATION	0
<p><b>참고:</b></p> <p>1. C 프로그래밍 언어의 매크로 변수MQCMHO_DEFAULT에는 표에 나열된 값이 포함됩니다. 즉, 구조 내의 필드에 대한 초기값을 제공하기 위해 다음과 같은 방법으로 사용될 수 있습니다.</p> <pre>MQCMHO MyCMHO = {MQCMHO_DEFAULT};</pre>		

## 언어 선언

### MQCMHO의 C 선언

```
struct tagMQCMHO {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    Options;          /* Options that control the action of MQCRTMH */
};
```

### MQCMHO의 COBOL 선언

```
** MQCMHO structure
10 MQCMHO.
** Structure identifier
15 MQCMHO-STRUCID PIC X(4).
** Structure version number
15 MQCMHO-VERSION PIC S9(9) BINARY.
** Options that control the action of MQCRTMH
15 MQCMHO-OPTIONS PIC S9(9) BINARY.
```

### MQCMHO의 PL/I 선언

```
dcl
1 MQCMHO based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 Options fixed bin(31), /* Options that control the action of MQCRTMH */
```

### MQCMHO의 상위 레벨 어셈블러 선언

```
MQCMHO DSECT
MQCMHO_STRUCID DS CL4 Structure identifier
```

MQCMHO_VERSION	DS	F	Structure version number
MQCMHO_OPTIONS	DS	F	Options that control the action of
*			MQCRTMH
MQCMHO_LENGTH	EQU	*	MQCMHO
MQCMHO_AREA	DS		CL(MQCMHO_LENGTH)

### **MQCMHO의 경우 StrucId (MQCHAR4)**

메시지 핸들 작성 옵션 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQCMHO\_STRUC\_ID입니다. 값은 다음과 같아야 합니다.

#### **MQCMHO\_STRUC\_ID**

메시지 핸들 작성 옵션 구조의 ID.

C 프로그래밍 언어의 경우 상수 **MQCMHO\_STRUC\_ID\_ARRAY**도 정의됩니다. 이는 **MQCMHO\_STRUC\_ID**와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### **MQCMHO의 버전 (MQLONG)**

이 필드는 항상 입력 필드입니다. 초기값은 MQCMHO\_VERSION\_1입니다.

구조 버전 번호이며 값은 다음과 같아야 합니다.

#### **MQCMHO\_VERSION\_1**

버전-1 작성 메시지 핸들 옵션 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### **MQCMHO\_CURRENT\_VERSION**

작성 메시지 핸들 옵션 구조의 현재 버전.

### **MQCMHO의 옵션 (MQLONG)**

이 필드는 항상 입력 필드입니다. 초기값은 MQCMHO\_DEFAULT\_VALIDATION입니다.

다음 중 하나를 지정할 수 있습니다.

#### **MQCMHO\_VALIDATE**

**MQSETMP**가 이 메시지 핸들에서 특성을 설정하기 위해 호출될 때 특성 이름은 다음을 확인하기 위해 유효성 검증됩니다.

- 올바르게 않은 문자가 포함되지 않습니다.
- JMS 또는 usr로 시작하지 않습니다. 다음을 제외한 JMS :

- JMSCorrelationID
- JMSReplyTo
- JMSType
- JMSXGroupID
- JMSXGroupSeq

이러한 이름은 JMS 특성에 예약됩니다.

- 이는 다음 키워드 중 하나가 아닙니다(대소문자 혼합).
  - 및
  - BETWEEN
  - ESCAPE
  - FALSE
  - IN

- IS
- LIKE
- NOT
- NULL
- 또는
- TRUE

• 이는 Body. 또는 Root.로 시작하지 않습니다. ( Root.MQMD제외).

특성이 MQ-정의된 경우 (mq. \*) 이름이 인식되면 특성 디스크립터 필드가 특성의 올바른 값으로 설정됩니다. 특성이 인식되지 않으면, 특성 디스크립터의 *Support* 필드가 **MQPD\_OPTIONAL**로 설정됩니다.

### MQCMHO\_DEFAULT\_VALIDATION

이 값은 특성 이름의 기본 유효성 검증 레벨이 발생함을 지정합니다.

유효성 검증의 기본 레벨은 **MQCMHO\_VALIDATE**에서 지정하는 레벨과 동등합니다.

이 값이 기본값입니다.

### MQCMHO\_NO\_VALIDATION

특성 이름의 유효성 검증이 발생하지 않습니다. **MQCMHO\_VALIDATE**의 설명을 참조하십시오.

**기본 옵션:** 설명된 선행 옵션 중 필요한 옵션이 없으면 다음 옵션을 사용할 수 있습니다.

### MQCMHO\_NONE

모든 옵션은 자체 기본값을 가정합니다. 기타 옵션을 지정하지 않도록 표시하려면 이 값을 사용하십시오.

**MQCMHO\_NONE**은 프로그램 문서화를 지원합니다. 이 옵션은 다른 옵션과 함께 사용하기 위한 용도는 아니지만, 해당 값이 0이므로 해당 사용을 감지할 수 없습니다.

## MQCNO - 연결 옵션

MQCNO 구조를 사용하면 애플리케이션이 큐 관리자에 대한 연결과 관련된 옵션을 지정할 수 있습니다. 구조는 MQCONNX 호출의 입/출력 매개변수입니다.

공유 핸들 사용 및 MQCONNX 호출에 대한 자세한 정보는 [MQCONNX와의 공유\(스레드 독립\) 연결](#)을 참조하십시오.

## 가용성

MQCNO 구조는 다음 플랫폼에서 사용 가능합니다.

-  AIX
-  IBM i
-  Linux
-  Windows

및 해당 시스템에 연결된 IBM MQ MQI clients

## 버전

지원되는 프로그래밍 언어에 제공되는 헤더, COPY 및 INCLUDE 파일에는 MQCNO의 최신 버전이 포함되지만 *Version* 필드의 초기값은 MQCNO\_VERSION\_1로 설정됩니다. version-1 구조에 없는 필드를 사용하려면 애플리케이션이 *Version* 필드를 필요한 버전 번호로 설정해야 합니다.

## 문자 세트 및 인코딩

MQCNO의 데이터는 MQENC\_NATIVE로 지정된 로컬 큐 관리자의 인코딩 및 **CodedCharSetId** 큐 관리자 속성으로 지정된 문자 세트에 있어야 합니다. 그러나 애플리케이션이 IBM MQ MQI client로 실행 중인 경우 구조는 클라이언트의 문자 세트 및 인코딩에 있어야 합니다.

### 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 479. MQCNO에 대한 MQCNO의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
<u>StrucId</u> (구조 ID)	MQCNO_STRUC_ID	'CNO~'
<u>버전</u> (구조 버전 번호)	MQCNO_VERSION_1	1
<u>옵션</u> (MQCONN의 조치를 제어하는 옵션)	MQCNO_NONE	0
<b>참고:</b> 나머지 필드는 <i>Version</i> 이 MQCNO_VERSION_2 미만일 경우 무시됩니다.		
<u>ClientConnOffset</u> (클라이언트 연결에 대한 MQCD 구조의 오프셋)	없음	0
<u>ClientConnPtr</u> (클라이언트 연결을 위한 MQCD 구조의 주소)	없음	널 포인터 또는 널 바이트
<b>참고:</b> 나머지 필드는 <i>Version</i> 이 MQCNO_VERSION_3 미만인 경우 무시됩니다.		
<u>ConnTag</u> (큐 관리자 연결 태그)	MQCT_NONE	널
<b>참고:</b> 나머지 필드는 <i>Version</i> 이 MQCNO_VERSION_4 미만인 경우 무시됩니다.		
<u>SSLConfigPtr</u> (클라이언트 연결을 위한 MQSCO 구조의 주소)	없음	널 포인터 또는 널 바이트
<u>SSLConfigOffset</u> (클라이언트 연결에 대한 MQSCO 구조의 오프셋)	없음	0
<b>참고:</b> <i>Version</i> 이 MQCNO_VERSION_5 미만인 경우 나머지 필드는 무시됩니다.		
<u>ConnectionId</u> (고유 연결 ID)	없음	널 포인터 또는 널 바이트
<u>SecurityParmsOffset</u> (보안 매개변수에 대한 MQSCO 구조의 오프셋)	없음	널 포인터 또는 널 바이트
<u>SecurityParmsPtr</u> (보안 매개변수에 대한 MQSCO 구조의 주소)	없음	널 포인터 또는 널 바이트
<b>참고:</b> <i>Version</i> 이 MQCNO_VERSION_6 미만인 경우 나머지 필드는 무시됩니다.		
<u>예약됨</u> (예약된 필드)	없음	구조를 64비트 경계로 채우는 예약된 필드입니다.
<u>CCDTUrlLength</u> (CCDT URL 길이)	없음	<u>CCDTUrlPtr</u> 또는 <u>CCDTUrlOffset</u> 으로 식별되는 문자열의 길이

표 479. MQCNO에 대한 MQCNO의 필드 (계속)		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
<u>CCDTUrlPtr</u> (CCDT URL 포인터)	없음	연결에 사용할 클라이언트 연결 채널 테이블의 위치를 식별하기 위해 URL이 포함된 문자열에 대한 포인터입니다.
<u>CCDTUrlOffset</u> (CCDT URL 오프셋)	없음	연결에 사용할 클라이언트 연결 채널 테이블의 위치를 식별하는 URL이 포함된 문자열에서 오프셋(바이트)입니다.
<b>참고:</b> <i>Version</i> 이 MQCNO_VERSION_7 미만인 경우 나머지 필드는 무시됩니다.		
<u>ApplName</u> (애플리케이션에서 설정한 이름)	없음	큐 관리자에 대한 연결을 식별하기 위해 애플리케이션에서 설정한 이름
<u>Reserved2</u> (예약 필드)	없음	구조를 64비트 경계로 채우는 예약된 필드입니다.
<b>참고:</b> <i>Version</i> 이 MQCNO_VERSION_8 미만인 경우 나머지 필드는 무시됩니다.		
<u>BalanceParmsOffset</u>	없음	MQBNO 구조에 대한 오프셋(바이트)
<u>BalanceParmsPtr</u>	없음	MQBNO 구조의 위치에 대한 포인터
<b>참고:</b> 1. ~ 기호는 단일 공백 문자를 나타냅니다. 2. C 프로그래밍 언어의 매크로 변수MQCNO_DEFAULT에는 표에 나열된 값이 포함됩니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오. <pre style="background-color: #f0f0f0; padding: 5px; margin-top: 10px;">MQCNO MyCNO = {MQCNO_DEFAULT};</pre>		

## 언어 선언

### MQCNO의 C 선언

```
typedef struct tagMQCNO MQCNO;
struct tagMQCNO {
    MQCHAR4      StructId;          /* Structure identifier */
    MQLONG       Version;           /* Structure version number */
    MQLONG       Options;           /* Options that control the action of
    MQCONNEX */
    MQLONG       ClientConnOffset; /* Offset of MQCD structure for client
    connection */
    MQPTR        ClientConnPtr;     /* Address of MQCD structure for client
    connection */
    MQBYTE128    ConnTag;           /* Queue manager connection tag */
    PMQSCO       SSLConfigPtr;      /* Address of MQSCO structure for client
    connection */
    MQLONG       SSLConfigOffset;   /* Offset of MQSCO structure for client
    connection */
    MQBYTE24     ConnectionId;      /* Unique connection identifier */
    MQLONG       SecurityParmsOffset /* Security fields */
    PMQCSP       SecurityParmsPtr   /* Security parameters */
    MQLONG       CCDTurlLength      /* Length of string identified by Ptr or offset */
};
```

```

MQLONG    CCDTurlOffset    /* Offset in bytes to URL of client connection channel */
PMQURL    CCDTurlPtr      /* Address of string containing URL */
MQBYTE4   Reserved        /* Reserved field to pad out to 64 bit boundary */
MQCHAR28  ApplName       /* Name set by the application to identify the connection to
                           the queue manager */
MQBYTE4   Reserved2      /* Reserved field to pad out to 64 bit boundary */
MQLONG    BalanceParmsOffset /* Offset of the MQBMO structure */
PMQBMO    BalanceParmsPtr /* Address of the location of the MQBMO structure */
};

```

## MQCNO의 COBOL 선언

```

** MQCNO structure
10 MQCNO.
**   Structure identifier
15 MQCNO-STRUCID      PIC X(4).
**   Structure version number
15 MQCNO-VERSION     PIC S9(9) BINARY.
**   Options that control the action of MQCONN
15 MQCNO-OPTIONS     PIC S9(9) BINARY.
**   Offset of MQCD structure for client connection
15 MQCNO-CLIENTCONNOFFSET PIC S9(9) BINARY.
**   Address of MQCD structure for client connection
15 MQCNO-CLIENTCONNPTR  POINTER.
**   Queue manager connection tag
15 MQCNO-CONNTRAG    PIC X(128).
**   Address of MQSCO structure for client connection
15 MQCNO-SSLCONFIGPTR  POINTER.
**   Offset of MQSCO structure for client connection
15 MQCNO-SSLCONFIGOFFSET PIC S9(9) BINARY.
**   Unique connection identifier
15 MQCNO-CONNECTIONID  PIC X(24).
**   Offset of MQCSP structure for security parameters
15 MQCNO-SECURITYPARMSOFFSET PIC S9(9) BINARY.
**   Address of MQCSP structure for security parameters
15 MQCNO-SECURITYPARMSPTR  POINTER.
**   Length of string identified by CCDTURLLOFFSET or CCDTURLPTR
15 MQCNO-CCDTURLLENGTH
**   Pointer to a string which contains a URL, to identify the location of the client
connection channel
15 MQCNO-CCDTURLPTR
**   Address of string which contains a URL that identifies the location of the client
connection channel table
15 MQCNO-CCDTURLLOFFSET
**   Reserved field to pad to 64 bit boundary
15 MQCNO-RESERVED
**   Name set by the application to identify the connection to the queue manager
15 MQCNO-APPLNAME
**   Reserved field to pad to 64 bit boundary
15 MQCNO-RESERVED2
**   Address of the MQBMO structure
15 MQCNO-BALANCEPARMSOFFSET
**   Pointer to the MQBMO structure
15 MQCNO-BALANCEPARMSPTR

```

## MQCNO의 PL/I 선언

```

dcl
1 MQCNO based,
3 StrucId          char(4),          /* Structure identifier */
3 Version          fixed bin(31),    /* Structure version number */
3 Options          fixed bin(31),    /* Options that control the action
of MQCONN */
3 ClientConnOffset fixed bin(31),    /* Offset of MQCD structure for
client connection */
3 ClientConnPtr    pointer,          /* Address of MQCD structure for
client connection */
3 ConnTag          char(128),        /* Queue manager connection tag */
3 SSLConfigPtr     pointer,          /* Address of MQSCO structure for
client connection */
3 SSLConfigOffset  fixed bin(31),    /* Offset of MQSCO structure for
client connection */
3 ConnectionId     char(24),         /* Unique connection identifier
3 SecurityParmsOffset fixed bin(31) /* Offset of MQCSP structure for
security parameters */
3 SecurityParmsPtr pointer,          /* Address of MQCSP structure for
security parameters */

```

3	CCDURLLength	fixed bin(31)	/* Length of string identified by CCDURLPtr or CCDURLOffset */
3	CCDURLOffset	fixed bin(31)	/* Offset in bytes to URL of client connection channel */
3	CCDURLPtr	pointer	/* Pointer to string containing URL */
3	Reserved	char(4)	/* Reserved field to pad out to 64 bit boundary */
3	AppName	char(28)	/* Name set by the application to identify the connection to
			the queue manager */
3	Reserved2	char(4)	/* Reserved field to pad out to 64 bit boundary */
3	BalanceParmsOffset	fixed bin(31)	/* Offset of the MQBMO structure */
3	BalanceParmsPtr	pointer	/* Address of the MQBMO structure */

## MQCNO의 상위 레벨 어셈블러 선언

MQCNO	DSECT		
MQCNO_STRUCID	DS	CL4	Structure identifier
MQCNO_VERSION	DS	F	Structure version number
MQCNO_OPTIONS	DS	F	Options that control the action of MQCONN
* MQCNO_CLIENTCONNOFFSET	DS	F	Offset of MQCD structure for client connection
* MQCNO_CLIENTCONNPTR	DS	F	Address of MQCD structure for client connection
MQCNO_CONNTAG	DS	XL128	Queue manager connection tag
MQCNO_CONNECTIONID	DS	XL24	Unique connection identifier
MQCNO_SSLCONFIGOFFSET	DS	F	Offset of MQCSP structure for security parameters
* MQCNO_SSLCONFIGPTR	DS	F	Address of MQCSP structure for security parameters
* MQCNO_LENGTH	EQU	*-MQCNO	
	ORG	MQCNO	
MQCNO_AREA	DS	CL(MQCNO_LENGTH)	
MQCNO_CCDURLLENGTH	DS	F	Length of string identified by CCDURLPTR or CCDURLOFFSET
* MQCNO_CCDURLOFFSET	DS	F	Offset in bytes to URL of client connection channel
MQCNO_CCDURLPTR	DS	F	Pointer to string containing URL
RESERVED	DS	XL4	Reserved field to pad out to 64 bit boundary
APPLNAME	DS	CL28	Name set by the application to identify the connection to the queue manager
* RESERVED2	DS	XL4	Reserved field to pad out to 64 bit boundary
MQCNO_BALANCEPARMSOFFSET	DS	F	Offset of the MQBMO structure
MQCNO_BALANCEPARMSPTR	DS	F	Address of the MQBMO structure

## MQCNO의 Visual Basic 선언

Type MQCNO		
StrucId	As String*4	'Structure identifier'
Version	As Long	'Structure version number'
Options	As Long	'Options that control the action of MQCONN'
ClientConnOffset	As Long	'Offset of MQCD structure for client connection'
ClientConnPtr	As MQPTR	'Address of MQCD structure for client connection'
ConnTag	As MQBYTE128	'Queue manager connection tag'
SSLConfigPtr	As MQPTR	'Address of MQSCO structure for client connection'
SSLConfigOffset	As Long	'Offset of MQSCO structure for client connection'
ConnectionId	As MQBYTE24	'Unique connection identifier'
SecurityParmsOffset	As Long	'Offset of MQCSP structure for security parameters'
SecurityParmsPtr	As MQPTR	'Address of MQCSP structure for security parameters'
CCDURLLength	As Long	'Length of string identified by CCDURLPtr or CCDURLOffset'
CCDURLOffset	As Long	'Offset in bytes to URL of client connection channel'
CCDURLPtr	As MQPTR	'Pointer to string containing URL'
Reserved	As MQBYTE4	'Reserved field to pad out to 64 bit boundary'
AppName	As String*28	'Name set by the application to identify the connection to the queue manager'
Reserved2	As MQBYTE4	'Reserved field to pad out to 64 bit boundary'
BalanceParmsOffset	As Long	'Offset in bytes to MQBNO structure'
BalanceParmsPtr	As MQPTR	'Address of MQBNO structure'
End Type		



## 관련 태스크

[MQCONNX 사용](#)

### **MQCNO의 경우 StrucId (MQCHAR4)**

연결 옵션 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQCNO\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

#### **MQCNO\_STRUC\_ID**

연결 옵션 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQCNO\_STRUC\_ID\_ARRAY도 정의됩니다. 이 상수의 값은 MQCNO\_STRUC\_ID와 동일하지만 문자열이 아닌 문자의 배열입니다.

### **MQCNO의 버전 (MQLONG)**

버전은 항상 입력 필드입니다. 초기값은 MQCNO\_VERSION\_1입니다.

값은 다음 중 하나여야 합니다.

#### **MQCNO\_VERSION\_1**

버전-1 연결 옵션 구조.

#### **MQCNO\_VERSION\_2**

버전-2 연결 옵션 구조.

#### **MQCNO\_VERSION\_3**

버전-3 연결 옵션 구조.

#### **MQCNO\_VERSION\_4**

버전-4 연결 옵션 구조.

#### **MQCNO\_VERSION\_5**

버전-5 연결 옵션 구조.

#### **MQCNO\_VERSION\_6**

버전-6 연결 옵션 구조입니다.

#### **MQCNO\_VERSION\_7**

버전-7 연결 옵션 구조입니다.

#### **MQCNO\_VERSION\_8**

버전-8 연결 옵션 구조입니다.

구조의 최신 버전에만 있는 필드는 필드 설명에 그렇게 식별되어 있습니다. 다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### **MQCNO\_CURRENT\_VERSION**

연결 옵션 구조의 현재 버전입니다.

### **MQCNO에 대한 옵션 (MQLONG)**

MQCONNX의 조치를 제어하는 옵션입니다.

## 계정 옵션

다음 옵션은 **AccountingConnOverride** 큐 관리자 속성이 MQMON\_ENABLED로 설정된 경우 계정 유형을 제어합니다.

#### **MQCNO\_ACCOUNTING\_MQI\_ENABLED**

**MQIAccounting** 속성을 MQMON\_OFF로 설정하여 큐 관리자 정의에서 모니터링 데이터 콜렉션을 사용 불가능하게 한 경우, 이 플래그를 설정하면 MQI 계정 데이터 콜렉션을 사용할 수 있습니다.

#### **MQCNO\_ACCOUNTING\_MQI\_DISABLED**

**MQIAccounting** 속성을 MQMON\_OFF로 설정하여 큐 관리자 정의에서 모니터링 데이터 콜렉션을 사용 불가능하게 한 경우, 이 플래그를 설정하면 MQI 계정 데이터 콜렉션을 중지할 수 있습니다.

### **MQCNO\_ACCOUNTING\_Q\_ENABLED**

**MQIAccounting** 속성을 MQMON\_OFF로 설정하여 큐 관리자 정의에서 큐-계정 데이터 콜렉션을 사용 불가능하게 한 경우, 이 플래그를 설정하면 해당 큐 정의의 **MQIAccounting** 필드에 큐 관리자를 지정한 큐에 계정 데이터 콜렉션을 사용할 수 있습니다.

### **MQCNO\_ACCOUNTING\_Q\_DISABLED**

**MQIAccounting** 속성을 MQMON\_OFF로 설정하여 큐 관리자 정의에서 큐-계정 데이터 콜렉션을 사용 불가능하게 한 경우, 이 플래그를 설정하면 해당 큐 정의의 **MQIAccounting** 필드에 큐 관리자를 지정한 큐에 대해 계정 데이터 콜렉션이 꺼짐으로 전환됩니다.

플래그를 정의하지 않으면 연결 계정이 큐 관리자 속성에 정의된 대로 적용됩니다.

## **바인딩 옵션**

다음 옵션은 사용할 IBM MQ 바인딩의 유형을 제어합니다. 다음 옵션 중 하나만 지정하십시오.

### **MQCNO\_STANDARD\_BINDING**

애플리케이션 및 로컬 큐 관리자 에이전트(큐잉 조작을 관리하는 컴포넌트)는 별도의 실행 단위(대개, 분리된 프로세스)에서 실행합니다. 이러한 배열은 큐 관리자의 무결성을 유지합니다. 즉, 잘못된 프로그램으로부터 큐 관리자를 보호합니다.

큐 관리자가 다중 바인딩 유형을 지원하고 MQCNO\_STANDARD\_BINDING을 설정하는 경우, 큐 관리자는 qm.ini 파일의 Connection 스탠자에 있는 **DefaultBindType** 속성을 사용하여 실제 바인딩 유형을 선택합니다. 이 스탠자가 정의되지 않았거나 값을 사용할 수 없거나 애플리케이션에 적절하지 않은 경우, 큐 관리자는 적절한 바인딩 유형을 선택합니다. 큐 관리자는 연결 옵션에 사용되는 실제 바인딩 유형을 설정합니다.

애플리케이션이 완전히 테스트되지 않았거나 신뢰할 수 없는 경우 MQCNO\_STANDARD\_BINDING을 사용하십시오. MQCNO\_STANDARD\_BINDING이 기본값입니다.

모든 환경에서 이 옵션이 지원됩니다.

mqm 라이브러리에 링크 중인 경우, 기본 바인드 유형을 사용하는 표준 서버 연결이 먼저 시도됩니다. 기본 서버 라이브러리가 로드하는 데 실패한 경우, 클라이언트 연결이 대신 시도됩니다.

- MQCONN(또는 MQCNO\_STANDARD\_BINDING이 지정된 경우 MQCONNX)의 동작을 변경하려면 MQ\_CONNECT\_TYPE 환경 변수를 다음 옵션 중 하나로 설정하십시오. 이에 대한 예외가 있습니다. MQ\_CONNECT\_TYPE을 LOCAL 또는 STANDARD로 설정한 상태에서 MQCNO\_FASTPATH\_BINDING을 지정한 경우 애플리케이션에서 관련 항목을 변경하지 않고 관리자가 빠른 경로 연결을 강등할 수 있습니다.

표 480. MQCONN 또는 MQCONNX의 동작을 변경하는 MQ_CONNECT_TYPE의 값	
값	의미
클라이언트	클라이언트 연결만 시도합니다.
FASTPATH	이 값은 이전 릴리스에서 지원되었지만 지금은 지정된 경우에 무시됩니다.
로컬	서버 연결만 시도합니다. 빠른 경로 연결이 표준 서버 연결로 다운그레이드됩니다.
STANDARD	이전 릴리스와의 호환성을 위해 지원됩니다. 이 값은 이제 LOCAL로 처리됩니다.

- MQCONNX가 호출될 때 MQ\_CONNECT\_TYPE 환경 변수가 설정되어 있지 않은 경우 기본 바인드 유형을 사용하는 표준 서버 연결을 시도합니다. 서버 라이브러리를 로드하는 데 실패하는 경우 클라이언트 연결이 시도됩니다.




### **MQCNO\_FASTPATH\_BINDING**

애플리케이션 및 로컬 큐 관리자 에이전트는 동일한 실행 단위의 일부입니다. 이것은 애플리케이션 및 로컬 큐 관리자 에이전트가 별도의 실행 단위에서 실행되는 일반적인 바인딩 방법과는 반대입니다.


큐 관리자가 이러한 유형의 바인딩을 지원하지 않으면 MQCNO\_FASTPATH\_BINDING이 무시됩니다. 옵션이 지정되지 않았더라도 처리는 계속됩니다.

다중 프로세스가 애플리케이션에 사용된 전체 자원보다 더 많은 자원을 이용하는 상황에서는 MQCNO\_FASTPATH\_BINDING이 이점이 될 수 있습니다. 빠른 경로 바인딩을 사용하는 애플리케이션을 신뢰할 수 있는 애플리케이션이라고 합니다.

빠른 경로 바인딩을 사용할지 여부를 결정할 때 다음과 같은 중요 사항을 고려하십시오.


- MQCNO\_FASTPATH\_BINDING 옵션을 사용하면 애플리케이션이 큐 관리자에 속한 기타 데이터 영역과 메시지를 대체하거나 손상시키는 것을 방지할 수 없습니다. 이러한 문제점을 완전히 평가한 경우에만 이 옵션을 사용하십시오.
- 애플리케이션에서 비동기 신호 또는 타이머 인터럽트(예: sigkill)를 MQCNO\_FASTPATH\_BINDING과 함께 사용하지 않아야 합니다. 또한 공유 메모리 세그먼트 사용에 관한 제한사항도 있습니다.
- 애플리케이션에서 큐 관리자 연결을 끊으려면 MQDISC 호출을 사용해야 합니다.
- 애플리케이션은 endmqm 명령으로 큐 관리자를 종료하기 전에 완료되어야 합니다.
-  IBM i에서는 작업이 QMQADM 그룹에 속한 사용자 프로파일에 따라 실행되어야 합니다. 또한 프로그램이 비정상적으로 종료되어서는 안 됩니다. 그렇지 않으면 예측하지 못한 결과가 발생할 수 있습니다.
-   AIX and Linux에서 mqm 사용자 ID는 유효한 사용자 ID여야 하며, mqm 그룹 ID는 유효한 그룹 ID여야 합니다. 애플리케이션이 이런 식으로 실행되게 하려면 프로그램을 mqm 사용자 ID 및 mqm 그룹 ID가 소유하도록 구성한 다음, 이 프로그램에 setuid 및 setgid 권한 비트를 설정하십시오.

IBM MQ Object Authority Manager(OAM)는 여전히 권한 검사에 이 실제 사용자 ID를 사용합니다.

-  Windows에서 프로그램이 mqm 그룹의 멤버여야 합니다. 빠른 경로 바인딩은 64비트 애플리케이션에 지원되지 않습니다.

MQCNO\_FASTPATH\_BINDING 옵션은 다음 환경에서 지원됩니다.

-  AIX
-  IBM i
-  Linux
-  Windows

 z/OS에서 옵션은 허용되었지만 무시됩니다.

신뢰할 수 있는 애플리케이션을 사용하는 경우 미치는 영향에 대한 자세한 정보는 [신뢰할 수 있는 애플리케이션 제한의 내용](#)을 참조하십시오.

## MQCNO\_SHARED\_BINDING

MQCNO\_SHARED\_BINDING을 사용하는 경우, 애플리케이션과 로컬 큐 관리자 에이전트가 일부 자원을 공유합니다. 큐 관리자가 이 바인딩 유형을 지원하지 않는 경우 MQCNO\_SHARED\_BINDING은 무시됩니다. 이 옵션이 지정되지 않은 것처럼 처리가 계속 수행됩니다.

## MQCNO\_ISOLATED\_BINDING

이 경우, 애플리케이션 프로세스 및 로컬 큐 관리자 에이전트가 서로 분리되어 자원을 공유하지 않습니다. 큐 관리자가 이 바인딩 유형을 지원하지 않는 경우 MQCNO\_ISOLATED\_BINDING은 무시됩니다. 이 옵션이 지정되지 않은 것처럼 처리가 계속 수행됩니다.

## MQCNO\_CLIENT\_BINDING


애플리케이션이 클라이언트 연결만 시도하게 하려면 이 옵션을 지정하십시오. 이 옵션에는 다음 제한사항이 있습니다.

-  MQCNO\_CLIENT\_BINDING은 z/OS에서 무시됩니다.

- MQCNO\_CLIENT\_BINDING은 MQCNO\_STANDARD\_BINDING 이외의 MQCNO 바인딩 옵션과 함께 지정된 경우 MQRC\_OPTIONS\_ERROR로 거부됩니다.
- MQCNO\_CLIENT\_BINDING은 바인드 유형을 선택하는 고유의 메커니즘이 있는 Java 또는 .NET에는 사용할 수 없습니다.

## MQCNO\_LOCAL\_BINDING

애플리케이션이 서버 연결을 시도하게 하려면 이 옵션을 지정하십시오. MQCNO\_FASTPATH\_BINDING, MQCNO\_ISOLATED\_BINDING 또는 MQCNO\_SHARED\_BINDING도 지정된 경우, 해당 유형의 연결이 대신 작성되고 이 절에서 설명됩니다. 그렇지 않으면 표준 서버 연결은 기본 바인드 유형을 사용하여 시도됩니다. MQCNO\_LOCAL\_BINDING에는 다음 제한사항이 있습니다.

-  MQCNO\_LOCAL\_BINDING은 z/OS에서 무시됩니다.
- MQCNO\_LOCAL\_BINDING은 MQCNO\_RECONNECT\_AS\_DEF 이외의 MQCNO 다시 연결 옵션과 함께 지정된 경우 MQRC\_OPTIONS\_ERROR로 거부됩니다.
- MQCNO\_LOCAL\_BINDING은 바인드 유형을 선택하는 고유의 메커니즘이 있는 Java 또는 .NET에는 사용할 수 없습니다.

다음 플랫폼에서 Options 필드에 지정된 바인드 유형으로 환경 변수 MQ\_CONNECT\_TYPE을 사용하여 사용되는 바인딩 유형을 제어할 수 있습니다.

-  AIX
-  Linux
-  Windows

이 환경 변수를 지정하는 경우, 값이 FASTPATH 또는 STANDARD여야 합니다. 다른 값을 지정하면 무시됩니다. 환경 변수 값은 대소문자를 구분합니다. 자세한 정보는 [MQCONNХ 환경 변수](#)를 참조하십시오.

환경 변수와 Options 필드는 다음과 같이 상호작용합니다.

- 환경 변수를 생략하거나 지원되지 않는 값을 제공하는 경우 빠른 경로 바인딩의 사용은 Options 필드를 통해서만 결정됩니다.
- 환경 변수에 지원되는 값을 제공하는 경우, 빠른 경로 바인딩은 환경 변수 및 Options 필드가 모두 빠른 경로 바인딩을 지정할 때에만 사용됩니다.

## 연결 태그 옵션

다음 옵션은 연결 태그의 사용을 제어합니다.ConnTag . IBM MQ for z/OS 큐 관리자에 연결하는 경우에만 지원됩니다. 다음과 같은 경우에는 유효하지 않습니다.Version 보다 작다MQCNO\_VERSION\_3 . 다음 옵션 중 하나만 지정하십시오. 멀티플랫폼 큐 관리자에서 연결하는 경우 다음을 지정하십시오.

MQCNO\_GENERATE\_CONN\_TAG .

### MQCNO\_GENERATE\_CONN\_TAG

큐 관리자가 이 연결과 관련된 연결 태그를 출력 MQCNO 구조로 리턴합니다.

반환된 연결 태그는 큐 관리자가 단일 애플리케이션 인스턴스로 간주하는 모든 연결에 대해 동일합니다.

### MQCNO\_SERIALIZE\_CONN\_TAG\_Q\_MGR

이 옵션은 로컬 큐 관리자 내에서 연결 태그의 독점 사용을 요청합니다. 연결 태그가 로컬 큐 관리자에서 이미 사용되고 있으면 이유 코드 MQRC\_CONN\_TAG\_IN\_USE가 표시되면서 MQCONNХ 호출이 실패합니다. 호출의 결과는 로컬 큐 관리자가 속한 큐 공유 그룹의 다른 곳에서 사용되는 연결 태그의 영향을 받지 않습니다.

### MQCNO\_SERIALIZE\_CONN\_TAG\_QSG

이 옵션은 로컬 큐 관리자가 속한 큐 공유 그룹 내에서 연결 태그의 독점 사용을 요청합니다. 연결 태그가 큐 공유 그룹에서 이미 사용되고 있으면 이유 코드 MQRC\_CONN\_TAG\_IN\_USE가 표시되면서 MQCONNХ 호출이 실패합니다.

## z/OS MQCNO\_RESTRICT\_CONN\_TAG\_Q\_MGR

이 옵션은 로컬 큐 관리자 내에서 연결 태그의 공유 사용을 요청합니다. 연결 태그가 로컬 큐 관리자에서 이미 사용되고 있는 경우, 요청 중인 애플리케이션이 태그를 이미 사용하고 있는 사용자와 동일한 처리 영역에서 실행 중이면 MQCONNX 호출이 성공할 수 있습니다. 이 조건이 충족되지 않으면 이유 코드 MQRC\_CONN\_TAG\_IN\_USE가 표시되면서 MQCONNX 호출이 실패합니다. 호출의 결과는 로컬 큐 관리자가 속한 큐 공유 그룹의 다른 곳에서 연결 태그 사용의 영향을 받지 않습니다.

- 연결 태그를 공유하려면 애플리케이션이 동일한 MVS 주소 공간 내에서 실행되어야 합니다. 연결 태그를 사용 중인 애플리케이션이 클라이언트 애플리케이션인 경우 MQCNO\_RESTRICT\_CONN\_TAG\_Q\_MGR이 허용되지 않습니다.

## z/OS MQCNO\_RESTRICT\_CONN\_TAG\_QSG

이 옵션은 로컬 큐 관리자가 속한 큐 공유 그룹 내에서 연결 태그의 공유 사용을 요청합니다. 연결 태그가 큐 공유 그룹에서 이미 사용되고 있는 경우, 요청 중인 애플리케이션이 태그를 이미 사용하고 있는 사용자와 동일한 큐 관리자에 연결되고 동일한 처리 영역에서 실행 중이면 MQCONNX 호출이 성공할 수 있습니다.

이러한 조건이 충족되지 않으면 이유 코드 MQRC\_CONN\_TAG\_IN\_USE가 표시되면서 MQCONNX 호출이 실패합니다.

- 연결 태그를 공유하려면 애플리케이션이 동일한 MVS 주소 공간 내에서 실행되어야 합니다. 연결 태그를 사용 중인 애플리케이션이 클라이언트 애플리케이션인 경우 MQCNO\_RESTRICT\_CONN\_TAG\_QSG가 허용되지 않습니다.

옵션이 지정되지 않은 경우, ConnTag 사용되지 않습니다.

## 핸들 공유 옵션

### Multi

이러한 옵션은 다음 환경에서 지원됩니다.

- **AIX** AIX
- **IBM i** IBM i
- **Linux** Linux
- **Windows** Windows

해당 옵션은 동일한 프로세스 내에서 서로 다른 스레드(병렬 처리 단위) 간의 핸들 공유를 제어합니다. 다음 옵션 중 하나만 지정할 수 있습니다.

### MQCNO\_HANDLE\_SHARE\_NONE

이 옵션은 연결 및 오브젝트 핸들이 핸들 할당을 유발하는 스레드(즉, MQCONN, MQCONNX 또는 MQOPEN 호출을 발행한 스레드)에만 사용될 수 있음을 표시합니다. 핸들은 동일한 프로세스에 속한 다른 스레드에서 사용될 수 없습니다.

### MQCNO\_HANDLE\_SHARE\_BLOCK

이 옵션은 프로세스의 한 스레드에 의해 할당된 연결 및 오브젝트 핸들이 동일한 프로세스에 속한 다른 스레드에 사용될 수 있음을 표시합니다. 그러나 한 번에 한 스레드만 특정 핸들을 사용할 수 있습니다. 즉, 한 핸들의 순차적 사용만 허용됩니다. 스레드가 다른 스레드에서 이미 사용되고 있는 핸들을 사용하는 경우 호출은 핸들이 사용 가능하게 될 때까지 차단(대기)됩니다.

### MQCNO\_HANDLE\_SHARE\_NO\_BLOCK


핸들이 다른 스레드에 사용 중인 경우 핸들이 사용 가능하게 될 때까지 호출을 차단하는 대신 MQCC\_FAILED 및 MQRC\_CALL\_IN\_PROGRESS가 표시되면서 호출이 즉시 완료되는 점을 제외하고, 이 옵션은 MQCNO\_HANDLE\_SHARE\_BLOCK과 동일합니다.

스레드에는 비공유 핸들이 0개 또는 1개 있을 수 있습니다.

- MQCNO\_HANDLE\_SHARE\_NONE을 지정하는 각 MQCONN 또는 MQCONNX 호출은 첫 번째 호출에서 새 비공유 핸들을 리턴하고 두 번째와 그 이후 호출에서 동일한 비공유 핸들을 리턴합니다(끼어드는 MQDISC 호출이 없다고 가정함). 두 번째 호출과 그 이후의 호출에 대한 이유 코드는 MQRC\_ALREADY\_CONNECTED입니다.
- MQCNO\_HANDLE\_SHARE\_BLOCK 또는 MQCNO\_HANDLE\_SHARE\_NO\_BLOCK을 지정하는 각 MQCONNX 호출은 각 호출에서 새 공유 핸들을 리턴합니다.

오브젝트 핸들에는 이 오브젝트 핸들을 작성한 MQOPEN 호출에 지정된 연결 핸들과 동일한 공유 특성이 상속됩니다. 또한 작업 단위는 이 작업 단위를 시작하는 데 사용된 연결 핸들과 동일한 공유 특성을 상속합니다. 작업 단위가 공유 핸들을 사용하여 하나의 스레드에서 시작되는 경우 이 작업 단위는 동일한 핸들을 사용하는 다른 스레드에서 업데이트될 수 있습니다.

핸들 공유 옵션을 지정하지 않은 경우 기본값은 환경에 의해 판별됩니다.

-  Microsoft Transaction Server(MTS) 환경에서 기본값이 MQCNO\_HANDLE\_SHARE\_BLOCK과 동일합니다.
- 다른 환경에서는 기본값이 MQCNO\_HANDLE\_SHARE\_NONE과 동일합니다.

## 재연결 옵션

재연결 옵션은 연결이 재연결 가능한지 여부를 판별합니다. 클라이언트 연결만 재연결이 가능합니다.

### MQCNO\_RECONNECT\_AS\_DEF

재연결 옵션이 기본값으로 해석됩니다. 기본값이 설정되지 않은 경우 이 옵션의 값은 DISABLED로 해석됩니다. 옵션의 값은 서버로 전달되며 PCF 또는 MQSC로 조회될 수 있습니다.

### MQCNO\_RECONNECT

애플리케이션은 MQCONN의 **QmgrName** 매개변수 값과 일치하는 큐 관리자에 다시 연결될 수 있습니다. 처음 연결을 설정한 큐 관리자와 클라이언트 애플리케이션 사이에 연관관계가 없는 경우에만 MQCNO\_RECONNECT 옵션을 사용하십시오. 옵션의 값은 서버로 전달되며 PCF 또는 MQSC로 조회될 수 있습니다.

### MQCNO\_RECONNECT\_DISABLED

애플리케이션이 다시 연결될 수 없습니다. 옵션의 값이 서버로 전달되지 않습니다.

### MQCNO\_RECONNECT\_Q\_MGR

애플리케이션은 원래 연결했던 큐 관리자에만 다시 연결될 수 있습니다. 클라이언트가 다시 연결될 수 있지만 원래 연결을 설정한 큐 관리자와 클라이언트 애플리케이션 사이에 연관관계가 있는 경우에 이 값을 사용하십시오. 고가용성 큐 관리자의 대기 인스턴스에 클라이언트를 자동으로 다시 연결시키려면 이 값을 선택하십시오. 옵션의 값은 서버로 전달되며 PCF 또는 MQSC로 조회될 수 있습니다.

클라이언트 연결에 대해서만 MQCNO\_RECONNECT, MQCNO\_RECONNECT\_DISABLED 및 MQCNO\_RECONNECT\_Q\_MGR 옵션을 사용하십시오. 이 옵션을 바인딩 연결에 사용하면 완료 코드 MQCC\_FAILED 및 이유 코드 MQRC\_OPTIONS\_ERROR가 표시되면서 MQCONN이 실패합니다. IBM MQ classes for Java에서 자동 클라이언트 재연결을 지원하지 않습니다.

## 대화 공유 옵션

다음 옵션은 TCP/IP 클라이언트 연결에만 적용됩니다. SNA, SPX 및 NetBios 채널의 경우, 이 값이 무시되고 채널이 이전 버전의 제품에서처럼 실행됩니다.

### MQCNO\_NO\_CONV\_SHARING

이 옵션은 대화 공유를 허용하지 않습니다.

대화가 가득 로드되어 대화 공유가 존재하는 채널 인스턴스의 서버 연결 끝에 경합이 발생할 가능성이 있는 경우 MQCNO\_NO\_CONV\_SHARING을 사용할 수 있습니다. MQCNO\_NO\_CONV\_SHARING은 대화 공유를 지원하는 채널에 연결된 경우에는 sharecnv(1)와 유사하게 작동하고, 대화 공유를 지원하지 않는 채널에 연결된 경우에는 sharecnv(0)와 유사하게 작동합니다.



## **MQCNO\_ALL\_CONVS\_SHARE**

이 옵션은 대화 공유를 허용합니다. 애플리케이션이 채널 인스턴스에서 연결 수를 제한하지 않습니다. 이 옵션이 기본값입니다.

애플리케이션이 채널 인스턴스가 공유할 수 있지만 채널의 서버 연결 끝에 있는 *SharingConversations* (SHARECNV) 정의가 1로 설정되었음을 나타내는 경우, 공유가 발생하지 않고 애플리케이션에 경고가 표시되지 않습니다.

마찬가지로 애플리케이션에서 공유가 허용됨을 표시하지만 서버 연결 *SharingConversations* 정의가 0으로 설정된 경우, 경고는 표시되지 않고 애플리케이션이 IBM WebSphere MQ 7.0 이전 버전의 클라이언트와 동일한 작동을 보입니다. 대화 공유와 관련된 애플리케이션 설정이 무시됩니다.

MQCNO\_NO\_CONV\_SHARING 및 MQCNO\_ALL\_CONVS\_SHARE는 상호 배타적입니다. 특정 연결에 두 옵션을 모두 지정하면 MQRC\_OPTIONS\_ERROR 이유 코드가 표시되면서 연결이 거부됩니다.

## **채널 정의 옵션**

다음 옵션은 MQCNO에 전달된 채널 정의 구조의 사용을 제어합니다.

### **MQCNO\_CD\_FOR\_OUTPUT\_ONLY**

이 옵션은 MQCNO의 채널 정의 구조를 성공적인 MQCONNX 호출에 사용된 채널 이름을 리턴하는 데만 사용할 수 있도록 합니다.

올바른 채널 정의 구조를 제공하지 않으면 이유 코드 MQRC\_CD\_ERROR가 표시되면서 호출이 실패합니다.

애플리케이션이 클라이언트로 실행되지 않는 경우 이 옵션은 무시됩니다.

리턴된 채널 이름은 동일한 채널 정의를 사용하여 다시 연결하기 위해 MQCNO\_USE\_CD\_SELECTION 옵션을 사용하는 후속 MQCONN 호출에서 사용될 수 있습니다. 이는 클라이언트 채널 테이블에 적용 가능한 채널 정의가 여러 개 있을 때 유용할 수 있습니다.

### **MQCNO\_USE\_CD\_SELECTION**

이 옵션은 MQCONNX 호출이 MQCNO에 전달된 채널 정의 구조에 포함된 채널 이름을 사용하여 연결할 수 있도록 합니다.

MQSERVER 환경 변수가 설정되면 이 변수가 정의하는 채널 정의가 사용됩니다. MQSERVER가 설정되지 않은 경우에는 클라이언트 채널 테이블이 사용됩니다.

일치하는 채널 이름 및 큐 관리자 이름이 있는 채널 정의를 찾을 수 없는 경우, 이유 코드 MQRC\_Q\_MGR\_NAME\_ERROR가 표시되면서 호출이 실패합니다.

올바른 채널 정의 구조를 제공하지 않으면 이유 코드 MQRC\_CD\_ERROR가 표시되면서 호출이 실패합니다.

애플리케이션이 클라이언트로 실행되지 않는 경우 이 옵션은 무시됩니다.

## **기본 옵션**

위에 설명된 옵션 중 필요한 옵션이 없으면 다음 옵션을 사용할 수 있습니다.

### **MQCNO\_NONE**

지정된 옵션이 없습니다.

프로그램 문서를 보조하려면 MQCNO\_NONE을 사용하십시오. 이 옵션은 다른 MQCNO\_\* 옵션과 함께 사용할 수 없지만, 값이 0이므로 이러한 사용을 감지할 수 없습니다.

## **ClientConnMQCNO에 대한 오프셋 (MQLONG)**

ClientConnOffset는 MQCNO 구조의 시작에서 MQCD 채널 정의 구조의 오프셋(바이트)입니다. 오프셋은 양수 또는 음수일 수 있습니다. 이 필드는 초기값이 0인 입력 필드입니다.

MQCONNX 호출을 발행하는 애플리케이션이 IBM MQ MQI client로 실행 중인 경우에만 *ClientConnOffset* 를 사용하십시오. 이 필드를 사용하는 방법에 대한 정보는 *ClientConnPtr* 필드의 설명을 참조하십시오.

*Version*이 MQCNO\_VERSION\_2 미만이면 이 필드가 무시됩니다.

### **MQCNO의 경우 ClientConnPtr (MQPTR)**

ClientConnPtr은 입력 필드입니다. 초기값은 널 포인터(포인터를 지원하는 해당 프로그래밍 언어에서) 및 모두 널인 바이트 문자열(그 이외의 경우)입니다.

ClientConnOffset 및 ClientConnPtr은 MQCONNX 호출을 발행하는 애플리케이션이 IBM MQ MQI client로 실행되는 경우에만 사용하십시오. 이들 필드의 하나 또는 다른 하나를 지정하면 애플리케이션이 필수 값을 포함한 MQCD 채널 정의 구조를 제공하여 클라이언트 연결 채널의 정의를 제어할 수 있습니다.

애플리케이션이 IBM MQ MQI client로 실행 중이지만 MQCD 구조를 제공하지 않으면 MQSERVER 환경 변수를 사용하여 채널 정의를 선택합니다. MQSERVER를 설정하지 않으면 클라이언트 채널 테이블이 사용됩니다.

애플리케이션이 IBM MQ MQI client로 실행되지 않으면 ClientConnOffset 및 ClientConnPtr이 무시됩니다.

애플리케이션이 MQCD 구조를 제공하는 경우, 나열된 필드를 필수 값으로 설정하십시오. MQCD의 다른 필드는 무시됩니다. 필드의 길이에 맞게 문자열을 공백으로 채우거나, 널 문자로 종료할 수 있습니다. MQCD 구조의 필드에 대한 자세한 정보는 [1410 페이지의 『필드』](#)의 내용을 참조하십시오.

표 481. MQCD의 필드

MQCD의 필드	값
ChannelName	채널 이름.
Version	구조 버전 번호입니다. MQCD_VERSION_7 이상이어야 합니다.
TransportType	지원되는 전송 유형.
ModeName	LU 6.2 모드 이름
TpName	LU 6.2 트랜잭션 프로그램 이름
SecurityExit	채널 보안 엑시트의 이름.
SendExit	채널 전송 엑시트의 이름.
ReceiveExit	채널 수신 엑시트의 이름.
MaxMsgLength	클라이언트 연결 채널을 통해 전송할 수 있는 메시지의 최대 길이(바이트).
SecurityUserData	보안 엑시트에 대한 사용자 데이터.
SendUserData	전송 엑시트에 대한 사용자 데이터.
ReceiveUserData	수신 엑시트에 대한 사용자 데이터.
UserIdentifier	LU 6.2 세션을 설정하는 데 사용되는 사용자 ID.
Password	LU 6.2 세션을 설정하는 데 사용되는 비밀번호.
ConnectionName	연결 이름.
HeartbeatInterval	하트비트 플로우 사이의 시간(초).
StrucLength	MQCD 구조의 길이.
ExitNameLength	SendExitPtr 및 ReceiveExitPtr에 의해 처리되는 엑시트 이름의 길이. SendExitPtr 또는 ReceiveExitPtr이 널 포인터가 아닌 값으로 설정된 경우 0보다 커야 합니다.
ExitDataLength	SendUserDataPtr 및 ReceiveUserDataPtr에 의해 처리되는 엑시트 데이터의 길이. SendUserDataPtr 또는 ReceiveUserDataPtr이 널 포인터가 아닌 값으로 설정된 경우 0보다 커야 합니다.



MQCD의 필드	값
<i>SendExitsDefined</i>	<i>SendExitPtr</i> 에 의해 처리되는 송신 엑시트 수. 0이면 <i>SendExit</i> 및 <i>SendUserData</i> 가 엑시트 이름 및 데이터를 제공합니다. 0보다 크면 <i>SendExitPtr</i> 및 <i>SendUserDataPtr</i> 이 엑시트 이름과 데이터를 제공하고 <i>SendExit</i> 및 <i>SendUserData</i> 가 공백이어야 합니다.
<i>ReceiveExitsDefined</i>	<i>ReceiveExitPtr</i> 에 의해 처리되는 수신 엑시트 수. 0이면 <i>ReceiveExit</i> 및 <i>ReceiveUserData</i> 가 엑시트 이름 및 데이터를 제공합니다. 0보다 크면 <i>ReceiveExitPtr</i> 및 <i>ReceiveUserDataPtr</i> 이 엑시트 이름과 데이터를 제공하고 <i>ReceiveExit</i> 및 <i>ReceiveUserData</i> 가 공백이어야 합니다.
<i>SendExitPtr</i>	첫 번째 전송 엑시트 이름의 주소.
<i>SendUserDataPtr</i>	첫 번째 전송 엑시트에 대한 데이터의 주소.
<i>ReceiveExitPtr</i>	첫 번째 수신 엑시트 이름의 주소.
<i>ReceiveUserDataPtr</i>	첫 번째 수신 엑시트에 대한 데이터의 주소.
<i>LongRemoteUserIdLength</i>	긴 리모트 사용자 ID의 길이.
<i>LongRemoteUserIdPtr</i>	긴 리모트 사용자 ID의 주소.
<i>RemoteSecurityId</i>	리모트 보안 ID.
<i>SSLCipherSpec</i>	TLS CipherSpec.
<i>SSLPeerNamePtr</i>	TLS 피어 이름의 주소.
<i>SSLPeerNameLength</i>	TLS 피어 이름의 길이.
<i>KeepAliveInterval</i>	통신 스택에 전달되는 채널의 활성 유지(keepalive) 시간 값
<i>LocalAddress</i>	사용할 로컬 네트워크 어댑터의 IP 주소를 포함하는 로컬 통신 주소 및 발신 연결에 사용할 포트의 범위.

두 가지 방법 중 하나로 채널 정의 구조를 제공하십시오.

- 오프셋 필드 *ClientConnOffset* 사용

이 경우, 애플리케이션은 MQCNO와 채널 정의 구조 MQCD가 뒤에 따라오는 복합 구조를 선언하고 MQCNO 시작부터 *ClientConnOffset*을 채널 정의 구조의 오프셋으로 설정해야 합니다. 이 오프셋이 올바른지 확인하십시오. *ClientConnPtr*이 널 포인터 또는 널 바이트로 설정되어야 합니다.

포인터 데이터 유형을 지원하지 않거나 다른 환경으로 이동 불가능하게 포인터 데이터 유형을 구현하는 프로그래밍 언어(예: COBOL 프로그래밍 언어)에는 *ClientConnOffset*을 사용하십시오.

Visual Basic 프로그래밍 언어의 경우, MQCNOCD라는 복합 구조가 헤더 파일 CMQXB.BAS에 제공됩니다. 이 구조는 MQCD 구조가 뒤에 오는 MQCNO 구조를 포함합니다. MQCNOCD\_DEFAULTS 서브루틴을 호출하여 MQCNOCD를 초기화하십시오. MQCNOCD는 MQCONNX 호출의 MQCONNXAny 변형과 함께 사용됩니다. 자세한 정보는 MQCONNX 호출에 대한 설명을 참조하십시오.

- 포인터 필드 *ClientConnPtr* 사용

이 경우, 애플리케이션은 MQCNO 구조와 별도로 채널 정의 구조를 선언하고, *ClientConnPtr*을 채널 정의 구조의 주소로 설정할 수 있습니다. *ClientConnOffset*을 0으로 설정하십시오.

다른 환경으로 이동 가능하게 포인터 데이터 유형을 지원하는 프로그래밍 언어(예: C 프로그래밍 언어)에는 *ClientConnPtr*을 사용하십시오.

C 프로그래밍 언어에서는 매크로 변수 MQCD\_CLIENT\_CONN\_DEFAULT를 사용하여 MQCD\_DEFAULT에서 제공하는 초기값보다 MQCONNX 호출에 사용하기에 더 적합한 초기 값을 구조에 제공할 수 있습니다.

어떤 기술을 선택하든 *ClientConnOffset* 및 *ClientConnPtr* 중 하나만 사용할 수 있으며, 둘 다 0이 아니면 이유 코드 MQRC\_CLIENT\_CONN\_ERROR와 함께 호출이 실패합니다.

MQCONNX 호출이 완료되면 MQCD 구조가 다시 참조되지 않습니다.

*Version*이 MQCNO\_VERSION\_2 미만이면 이 필드가 무시됩니다.

**참고:** 프로그래밍 언어가 포인터 데이터 유형을 지원하지 않는 플랫폼에서 이 필드는 적절한 길이의 바이트 문자 열로 선언되며, 초기값은 모두 널 바이트 문자열입니다.

**Multi**

### 멀티플랫폼의 MQCNO에 대한 ConnTag (MQBYTE128)

연결 태그는 연결 ID와 개념적으로 유사하지만 단일 애플리케이션 인스턴스로 식별하는 여러 관련 연결에 걸쳐 있을 수 있습니다. Multiplatforms에서 연결 태그는 연결 시 큐 관리자에 의해 생성됩니다.

자세한 정보는 [연결 ID 및 애플리케이션 인스턴스를 참조하십시오](#).

생성된 연결 태그는 semi형의 사람이 읽을 수 있는 태그입니다. 즉, 로컬 문자 세트의 문자열처럼 MQSC에서 표시되고 필터링될 수 있다. IBM MQ에 의해 관련되는 것으로 인식되는 연결에는 자동으로 동일한 연결 태그가 지정됩니다. 이 지정은 [애플리케이션 밸런싱](#)에 특히 중요합니다.

생성된 연결 태그는 다음 세 가지 방법으로 표시됩니다.

- `MQCNO GENERATE CONN_TAG`를 지정할 때 MQCONNX 호출의 출력 MQCNO 구조.
- `DISPLAY CONN`의 출력(또는 프로그래밍 방식의 등가물).
- `DISPLAY APSTATUS`의 출력(또는 등가물).

태그는 애플리케이션이 MQDISC 호출을 종료하거나 실행할 때 무효화됩니다.

#### 관련 참조

326 페이지의 『[ConnTag \(MQBYTE128\) for MQCNO on IBM MQ for z/OS](#)』

A connection tag is conceptually similar to a connection identifier, but might span multiple related connections, identifying them as a single application instance. On IBM MQ for z/OS, the connection tag is an input field, provided by the application and used in conjunction with MQCNO\_\*\_CONN\_TAG options to serialize connections from that application instance

**z/OS**

### ConnTag (MQBYTE128) for MQCNO on IBM MQ for z/OS

A connection tag is conceptually similar to a connection identifier, but might span multiple related connections, identifying them as a single application instance. On IBM MQ for z/OS, the connection tag is an input field, provided by the application and used in conjunction with MQCNO\_\*\_CONN\_TAG options to serialize connections from that application instance

Where there are multiple instances of an application that are intended to be simultaneously connected, they must each supply a unique value for this field. See the descriptions of these [connection tag options](#) for further details.

#### Notes:

- On IBM MQ for z/OS, there is no way to administratively determine the connection tag associated with an application at runtime.
- Connection tag values beginning with MQ in upper, lower, or mixed case in either ASCII or EBCDIC are reserved for use by IBM products. Do not use connection tag values beginning with these letters.

Use the following special value if you require no tag:

#### MQCT\_NONE

The value is binary zero for the length of the field.

For the C programming language, the constant MQCT\_NONE\_ARRAY is also defined; this constant has the same value as MQCT\_NONE, but is an array of characters instead of a string.

The ConnTag field is used when connecting to a z/OS queue manager.

The length of this field is given by MQ\_CONN\_TAG\_LENGTH. This field is ignored if *Version* is less than MQCNO\_VERSION\_3.

**Multi**

See “[멀티플랫폼의 MQCNO에 대한 ConnTag \(MQBYTE128\)](#)” on page 326 for information on using the connection tag on IBM MQ for Multiplatforms.

## **MQCNO의 경우 SSLConfigPtr (PMQSCO)**

SSLConfigPtr은 입력 필드입니다. 초기값은 널 포인터(포인터를 지원하는 해당 프로그래밍 언어에서) 및 모두 널 인 바이트 문자열(그 이외의 경우)입니다.

MQCONNX 호출을 발행하는 애플리케이션이 IBM MQ MQI client 로 실행 중이고 채널 프로토콜이 TCP/IP인 경우에만 *SSLConfigPtr* 및 *SSLConfigOffset* 를 사용하십시오. 애플리케이션이 IBM MQ 클라이언트로서 실행 중이지 않거나 채널 프로토콜이 TCP/IP가 아닌 경우에는 *SSLConfigPtr* 및 *SSLConfigOffset*이 무시됩니다.

*SSLConfigPtr* 또는 *SSLConfigOffset*와 함께 *ClientConnPtr* 또는 *ClientConnOffset* 중 하나를 지정하여, 애플리케이션은 클라이언트 연결을 위한 TLS의 사용을 제어할 수 있습니다. TLS 정보가 이러한 방식으로 지정된 경우, 환경 변수 MQSSLKEYR 및 MQSSLCRYP는 무시됩니다. 클라이언트 채널 정의 테이블(CCDT)의 TLS 관련 정보도 무시됩니다.

TLS 정보는 다음에서만 지정될 수 있습니다.

- 클라이언트 프로세스의 첫 번째 MQCONNX 호출 또는
- 후속 MQCONNX 호출(큐 관리자에 대한 모든 이전 TLS 연결이 MQDISC를 사용하여 결정된 경우).

이는 프로세스 전체 TLS 환경이 초기화될 수 있는 유일한 상태입니다. TLS 환경이 이미 존재할 때 TLS 정보를 지정하여 MQCONNX 호출이 실행되는 경우, 호출의 TLS 정보는 무시되며 기존 TLS 환경을 사용하여 연결이 작성됩니다. 이 경우에 호출은 완료 코드 MQCC\_WARNING 및 이유 코드 MQRC\_SSL\_ALREADY\_INITIALIZED를 리턴합니다.

*SSLConfigPtr*에서 주소를 지정하거나 *SSLConfigOffset*에서 오프셋을 지정하여 MQCD 구조와 동일한 방식으로 MQSCO 구조를 제공할 수 있습니다. 이를 수행하는 방법의 세부사항은 *ClientConnPtr*의 설명을 참조하십시오. 그러나 *SSLConfigPtr* 및 *SSLConfigOffset* 중 하나만 사용할 수 있습니다. 호출은 이유 코드 MQRC\_SSL\_CONFIG\_ERROR로 실패합니다. (둘 모두 0이 아닌 경우).

일단 MQCONNX 호출이 완료되면 MQSCO 구조가 다시 참조되지 않습니다.

*Version*이 MQCNO\_VERSION\_4 미만이면 이 필드가 무시됩니다.

**참고:** 프로그래밍 언어가 포인터 데이터 유형을 지원하지 않는 플랫폼에서 이 필드는 적절한 길이의 바이트 문자열로 선언됩니다.

## **MQCNO의 경우 SSLConfigOffset (MQLONG)**

SSLConfigOffset은 MQCNO 구조의 시작에서 MQSCO 구조의 오프셋(바이트)입니다. 오프셋은 양수 또는 음수일 수 있습니다. 이 필드는 초기값이 0인 입력 필드입니다.

MQCONNX 호출을 발행하는 애플리케이션이 IBM MQ MQI client로 실행 중인 경우에만 *SSLConfigOffset* 를 사용하십시오. 이 필드를 사용하는 방법에 대한 정보는 *SSLConfigPtr* 필드의 설명을 참조하십시오.

*Version*이 MQCNO\_VERSION\_4 미만이면 이 필드가 무시됩니다.

## **MQCNO의 경우 ConnectionId (MQBYTE24)**

ConnectionId는 IBM MQ가 애플리케이션을 안정적으로 식별할 수 있게 허용하는 고유 24바이트 ID입니다. 애플리케이션은 PUT 및 GET 호출의 상관에 이 ID를 사용할 수 있습니다. 이 출력 매개변수의 초기값은 모든 프로그래밍 언어에서 24 널 바이트입니다.

어떤 설정이든 큐 관리자는 모든 연결에 대해 고유 ID를 지정합니다. MQCONNX가 버전 5 MQCNO와의 연결을 설정하는 경우, 애플리케이션은 리턴된 MQCNO에서 ConnectionId를 판별할 수 있습니다. 지정된 ID는 IBM MQ가 생성하는 기타 모든 ID 중에서 고유하도록 보장됩니다(예: CorrelId, MsgId 및 GroupId).

ConnectionId를 사용하면 PCF 명령 Inquire Connection 또는 MQSC 명령 DISPLAY CONN을 사용하여 장기 실행 작업 단위를 식별할 수 있습니다. MQSC 명령(CONN)에서 사용하는 ConnectionId는 여기서 리턴된 ConnectionId에서 도출됩니다. PCF Inquire 및 Stop Connection 명령은 수정 없이 여기서 리턴된 ConnectionId를 사용할 수 있습니다.

PCF 명령 Stop Connection 또는 MQSC 명령 STOP CONN을 사용하여 ConnectionId를 지정함으로써, ConnectionId를 사용하여 장기 실행 작업 단위의 종료를 강제 실행할 수 있습니다. 이러한 명령의 사용에 대한 자세한 정보는 [Stop Connection](#) 및 [STOP CONN](#)을 참조하십시오.

버전이 MQCNO\_VERSION\_5 미만이면 이 필드가 리턴되지 않습니다.

이 필드의 길이는 MQ\_CONNECTION\_ID\_LENGTH에서 제공됩니다.

### **SecurityParmsMQCNO에 대한 오프셋 (MQLONG)**

SecurityParmsOffset는 MQCNO 구조의 시작에서 MQCSP 구조의 오프셋(바이트)입니다. 오프셋은 양수 또는 음수일 수 있습니다. 이 필드는 초기값이 0인 입력 필드입니다.

Version이 MQCNO\_VERSION\_5 미만이면 이 필드는 무시됩니다.

MQCSP 구조는 [329 페이지의 『MQCSP - 보안 매개변수』](#)에 정의되어 있습니다.

### **MQCNO의 경우 SecurityParmsPtr (PMQCSP)**

SecurityParmsPtr는 권한 서비스에 의한 인증을 위한 사용자 ID 및 비밀번호를 지정하는 데 사용되는 MQCSP 구조의 주소입니다. 이 필드는 입력 필드이며 초기값은 널 포인터 또는 널 바이트입니다.

Version이 MQCNO\_VERSION\_5 미만이면 이 필드는 무시됩니다.

MQCSP 구조는 [329 페이지의 『MQCSP - 보안 매개변수』](#)에 정의되어 있습니다.

### **MQCNO에 대해 예약됨 (MQBYTE4)**

구조를 64비트 경계까지 채우기 위해 예약된 필드입니다. 필드의 초기값은 필드 길이만큼의 2진수 0입니다.

Version이 MQCNO\_VERSION\_6 미만이면 이 필드가 무시됩니다.

### **MQCNO의 경우 CCDUrlLength (MQLONG)**

CCDUrlLength는 연결에 사용할 클라이언트 연결 채널 테이블의 위치를 식별하는 URL이 포함된 CCDUrlPtr 또는 CCDUrlOffset으로 식별된 문자열의 길이입니다. 필드의 초기값은 0입니다.

MQCONNX 호출을 발행하는 애플리케이션이 IBM MQ MQI client로 실행 중인 경우에만 CCDUrlLength를 사용하십시오.

프로그래밍 방식으로 MQCHLLIB 및 MQCHLTAB 환경 변수를 설정하는 대체 방식입니다.

애플리케이션이 클라이언트로 실행 중이 아니면 CCDUrlLength가 무시됩니다.

Version이 MQCNO\_VERSION\_6 미만이면 이 필드가 무시됩니다.

### **MQCNO의 경우 CCDUrlPtr (PMQCHAR)**

CCDUrlPtr은 연결에 사용할 클라이언트 연결 채널 테이블의 위치를 식별하기 위한 URL이 포함된 문자열에 대한 선택적 포인터입니다. 이 필드는 입력 필드이며, 초기값은 널 포인터(포인터를 지원하는 프로그래밍 언어에서) 및 모두 널인 바이트 문자열(그 이외의 경우)입니다.

MQCONNX 호출을 발행하는 애플리케이션이 IBM MQ MQI client로 실행 중인 경우에만 CCDUrlPtr를 사용하십시오.

**중요사항:** CCDUrlPtr 및 CCDUrlOffset 중에서 하나만 사용할 수 있습니다. 두 필드가 모두 0이 아니면 호출이 이유 코드 MQRC\_CCDT\_URL\_ERROR로 실패합니다.

프로그래밍 방식으로 MQCHLLIB 및 MQCHLTAB 환경 변수를 설정하는 대체 방식입니다.

애플리케이션이 클라이언트로 실행 중이 아니면 CCDUrlPtr이 무시됩니다.

Version이 MQCNO\_VERSION\_6 미만이면 이 필드가 무시됩니다.

### **MQCNO의 경우 CCDUrlOffset (MQLONG)**

CCDUrlOffset은 연결에 사용할 클라이언트 연결 채널 테이블의 위치를 식별하는 URL이 포함된 문자열에 대해 MQCNO 구조의 시작으로부터의 오프셋(바이트)입니다. 오프셋은 양수 또는 음수일 수 있으며 필드의 초기값은 0입니다.

MQCONNX 호출을 발행하는 애플리케이션이 IBM MQ MQI client로 실행 중인 경우에만 CCDUr10ffset 를 사용하십시오.

**중요사항:** CCDUr1Ptr 및 CCDUr10ffset 중에서 하나만 사용할 수 있습니다. 두 필드가 모두 0이 아니면 호출이 이유 코드 MQRC\_CCDT\_URL\_ERROR로 실패합니다.

프로그래밍 방식으로 MQCHLLIB 및 MQCHLTAB 환경 변수를 설정하는 대체 방식입니다.


애플리케이션이 클라이언트로 실행 중이 아니면 CCDUr10ffset이 무시됩니다.

Version이 MQCNO\_VERSION\_6 미만이면 이 필드가 무시됩니다.

### **MQCNO의 경우 ApplName (MQCHAR28)**

큐 관리자에 대한 연결을 식별하기 위해 애플리케이션에 의해 설정된 이름입니다. 필드 초기값은 MQAN\_NONE\_ARRAY(공백 문자)입니다.

Version이 MQCNO\_VERSION\_7 미만이거나 값이 공백으로 설정된 경우 이 필드는 무시됩니다.

 z/OS에서는 이 필드를 설정할 수 없습니다. 이를 설정하려고 시도하면 MQRC\_CNO\_ERROR 이유 코드를 받게 됩니다.

### **MQCNO의 경우 Reserved2 (MQBYTE4)**

구조를 64비트 경계까지 채우기 위해 예약된 필드입니다. 필드의 초기값은 필드 길이 만큼의 2진수 0입니다.

Version이 MQCNO\_VERSION\_7 미만이면 이 필드가 무시됩니다.

### **BalanceParmsMQCNO에 대한 오프셋 (MQLONG)**

애플리케이션의 밸런싱 동작에 대한 정보를 포함하고 있는 MQBNO 유형의 구조에 맞는 메모리 위치입니다. 애플리케이션이 클라이언트 채널을 통해 연결하지 않으면 구조가 완전히 무시됩니다.

Version 이 MQCNO\_VERSION\_8미만인 경우 이 필드는 무시됩니다.

자세한 정보는 MQBNO를 참조하십시오.

이 필드를 제공하면 329 페이지의 『MQCNO의 경우 BalanceParmsPtr (MQPTR)』 필드를 제공할 수 없습니다. 두 필드를 모두 제공하려고 하면 MQRC\_CNO\_ERROR가 표시됩니다. 이 필드는 클라이언트 연결과만 관련이 있으므로 다른 연결 유형에도 이 필드를 제공하면 MQRC\_CNO\_ERROR가 표시됩니다.

### **MQCNO의 경우 BalanceParmsPtr (MQPTR)**

애플리케이션의 밸런싱 동작에 대한 정보를 포함하는 MQBNO 유형의 구조에 맞는 메모리 위치의 포인터입니다. 애플리케이션이 클라이언트 채널을 통해 연결하지 않으면 구조가 완전히 무시됩니다.

Version 이 MQCNO\_VERSION\_8미만인 경우 이 필드는 무시됩니다.

자세한 정보는 MQBNO를 참조하십시오.

이 필드를 제공하면 329 페이지의 『BalanceParmsMQCNO에 대한 오프셋 (MQLONG)』 필드를 제공할 수 없습니다. 두 필드를 모두 제공하려고 하면 MQRC\_CNO\_ERROR가 표시됩니다. 이 필드는 클라이언트 연결과만 관련이 있으므로 다른 연결 유형에도 이 필드를 제공하면 MQRC\_CNO\_ERROR가 표시됩니다.

## **MQCSP - 보안 매개변수**

IBM MQ 연결 보안 매개변수 구조는 애플리케이션이 큐 관리자에 대한 MQCONNX 호출에서 인증 정보를 플로우 하는 데 사용됩니다. 또한 이를 사용하여 민감한 데이터를 암호화하는 IBM MQ 비밀번호 보호 시스템과 함께 사용되는 초기 키를 제공할 수도 있습니다.

버전 1의 사용자 ID 및 비밀번호를 포함하려면 *AuthenticationType* 를 MQCSP\_AUTH\_USER\_ID\_AND\_PWD 로 설정하십시오.

버전 2에서 초기 키 정보를 제공할 때 *AuthenticationType* 의 기본값은 MQCSP\_AUTH\_NONE입니다.

 IBM MQ 9.3.4에서 *AuthenticationType* 를 사용하여 인증 토큰 정보를 포함하십시오.

**V 9.4.0** MQCSP\_AUTH\_USER\_ID\_AND\_PWD 또는 MQCSP\_AUTH\_ID\_TOKEN을 사용할 수 있지만 둘 다 사용할 수는 없습니다.

**경고:** 일부 경우에 클라이언트 애플리케이션에 대한 MQCSP 구조의 비밀번호 또는 인증 토큰은 일반 텍스트로 네트워크를 통해 전송됩니다. 클라이언트 애플리케이션 비밀번호 및 인증 토큰이 적절하게 보호되는지 확인하려면 [MQCSP 비밀번호 보호](#)를 참조하십시오.

## 가용성

MQCSP 구조는 지원되는 모든 IBM MQ 플랫폼에서 사용 가능합니다.

## 버전

지원되는 프로그래밍 언어에 제공되는 헤더, COPY 및 INCLUDE 파일은 MQCSP의 최신 버전을 포함하지만 *Version* 필드의 초기값이 MQCSP\_VERSION\_1로 설정되어 있습니다. version-1 구조에 없는 필드를 사용하려면 애플리케이션이 *Version* 필드를 필요한 버전 번호로 설정해야 합니다.

## 문자 세트 및 인코딩

MQCSP의 데이터는 로컬 큐 관리자의 문자 세트 및 인코딩에 있어야 하며, 이는 각각 **CodedCharSetId** 큐 관리자 속성 및 MQENC\_NATIVE에 의해 제공됩니다.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 482. MQCSP에 대한 MQCSP의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQCSP_STRUC_ID	'CSP-'
버전 (구조 버전 번호)	MQCSP_VERSION_1	1
AuthenticationType (인증 유형)	없음	MQCSP_AUTH_NONE
Reserved1 (IBM i에서 포인터 맞추기에 필요함)	없음	널 문자열 또는 공백
CSPUserIdPtr (사용자 ID의 주소)	없음	널 포인터 또는 널 바이트
CSPUserIdOffset (사용자 ID의 오프셋)	없음	0
CSPUserId길이 (사용자 ID의 길이)	없음	0
Reserved2 (IBM i에서 포인터 맞추기에 필요함)	없음	널 문자열 또는 공백
CSPPasswordPtr (비밀번호 주소)	없음	널 포인터 또는 널 바이트
CSPPasswordOffset (비밀번호 오프셋)	없음	0
CSPPasswordLength (비밀번호 길이)	없음	0
<b>참고:</b> <i>Version</i> 이 MQCSP_VERSION_2미만인 경우 나머지 필드는 무시됩니다.		
Reserved3 (IBM i에서 포인터 맞추기에 필요함)	없음	널 문자열 또는 공백
InitialKeyPtr	없음	널 포인터 또는 공백
InitialKeyOffset (암호 보호 시스템의 초기 키 오프셋)	없음	0
InitialKey길이 (암호 보호 시스템의 초기 키 길이)	없음	0
<b>참고:</b> <i>Version</i> 이 MQCSP_VERSION_3미만인 경우 나머지 필드는 무시됩니다.		



표 482. MQCSP에 대한 MQCSP의 필드 (계속)

필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
<b>V 9.4.0</b> Reserved4 (IBM i에서 포인터 맞추기에 필요함)	없음	널 문자열 또는 공백
<b>V 9.4.0</b> TokenPtr (인증 토큰 주소)	없음	널 포인터 또는 공백
<b>V 9.4.0</b> TokenKey오프셋 (인증 토큰 오프셋)	없음	0
<b>V 9.4.0</b> TokenLength (인증 토큰 길이)	없음	0

**참고:**

1. ~ 기호는 단일 공백 문자를 나타냅니다.
2. C 프로그래밍 언어의 매크로 변수MQCSP\_DEFAULT에는 표에 나열된 값이 포함됩니다. 즉, 구조 내의 필드에 대한 초기값을 제공하기 위해 다음과 같은 방법으로 사용될 수 있습니다.

```
MQCSP MyCSP = {MQCSP_DEFAULT};
```

## 언어 선언

### MQCSP의 C 선언

```
typedef struct tagMQCSP MQCSP;
struct tagMQCSP {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;          /* Structure version number */
    MQLONG     AuthenticationType; /* Type of authentication */
    MQBYTE4    Reserved1;       /* Required for IBM i pointer alignment */
    MQPTR      CSPUserIdPtr;     /* Address of user ID */
    MQLONG     CSPUserIdOffset;  /* Offset of user ID */
    MQLONG     CSPUserIdLength;  /* Length of user ID */
    MQBYTE8    Reserved2;       /* Required for IBM i pointer alignment */
    MQPTR      CSPPasswordPtr;   /* Address of password */
    MQLONG     CSPPasswordOffset; /* Offset of password */
    MQLONG     CSPPasswordLength; /* Length of password */
    /* Ver:1 */

    MQBYTE8    Reserved3;       /* Required for IBM i pointer alignment */
    MQPTR      InitialKeyPtr;    /* Address of initial key */
    MQLONG     InitialKeyOffset; /* Offset of initial key */
    MQLONG     InitialKeyLength; /* Length of initial key */
    /* Ver:2 */
};
```

**V 9.4.0**

```
typedef struct tagMQCSP MQCSP;
struct tagMQCSP {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;          /* Structure version number */
    MQLONG     AuthenticationType; /* Type of authentication */
    MQBYTE4    Reserved1;       /* Required for IBM i pointer alignment */
    MQPTR      CSPUserIdPtr;     /* Address of user ID */
    MQLONG     CSPUserIdOffset;  /* Offset of user ID */
    MQLONG     CSPUserIdLength;  /* Length of user ID */
    MQBYTE8    Reserved2;       /* Required for IBM i pointer alignment */
    MQPTR      CSPPasswordPtr;   /* Address of password */
    MQLONG     CSPPasswordOffset; /* Offset of password */
    MQLONG     CSPPasswordLength; /* Length of password */
    /* Ver:1 */

    MQBYTE8    Reserved3;       /* Required for IBM i pointer alignment */
    MQPTR      InitialKeyPtr;    /* Address of initial key */
    MQLONG     InitialKeyOffset; /* Offset of initial key */
    MQLONG     InitialKeyLength; /* Length of initial key */
};
```

```

/* Ver:2 */
MQBYTE8   Reserved4;           /* Required for IBM i pointer alignment */
MQPTR     TokenPtr;           /* Address of token */
MQLONG    TokenOffset;        /* Offset of token */
MQLONG    TokenLength;        /* Length of token */
/* Ver:3 */
};

```

## MQCSP의 COBOL 선언

```

** MQCSP structure
10 MQCSP.
** Structure identifier
15 MQCSP-STRUCID PIC X(4).
** Structure version number
15 MQCSP-VERSION PIC S9(9) BINARY.
** Type of authentication
15 MQCSP-AUTHENTICATIONTYPE PIC S9(9) BINARY.
** Required for IBM i pointer alignment
15 MQCSP-RESERVED1 PIC X(4).
** Address of user ID
15 MQCSP-CSPUSERIDPTR POINTER.
** Offset of user ID
15 MQCSP-CSPUSERIDOFFSET PIC S9(9) BINARY.
** Length of user ID
15 MQCSP-CSPUSERIDLENGTH PIC S9(9) BINARY.
** Required for IBM i pointer alignment
15 MQCSP-RESERVED2 PIC X(4).
** Address of password
15 MQCSP-CSPPASSWORDPTR POINTER.
** Offset of password
15 MQCSP-CSPPASSWORDOFFSET PIC S9(9) BINARY.
** Length of password
15 MQCSP-CSPPASSWORDLENGTH PIC S9(9) BINARY.
** Ver:1 **

** Reserved
15 MQCSP-RESERVED3 PIC X(8).
** Address of initial key
15 MQCSP-INITIALKEYPTR POINTER.
** Offset of initial key
15 MQCSP-INITIALKEYOFFSET PIC S9(9) BINARY.
** Length of initial key
15 MQCSP-INITIALKEYLENGTH PIC S9(9) BINARY.
** Ver:2 **

```

**V 9.4.0**

```

** MQCSP structure
10 MQCSP.
** Structure identifier
15 MQCSP-STRUCID PIC X(4).
** Structure version number
15 MQCSP-VERSION PIC S9(9) BINARY.
** Type of authentication
15 MQCSP-AUTHENTICATIONTYPE PIC S9(9) BINARY.
** Required for IBM i pointer alignment
15 MQCSP-RESERVED1 PIC X(4).
** Address of user ID
15 MQCSP-CSPUSERIDPTR POINTER.
** Offset of user ID
15 MQCSP-CSPUSERIDOFFSET PIC S9(9) BINARY.
** Length of user ID
15 MQCSP-CSPUSERIDLENGTH PIC S9(9) BINARY.
** Required for IBM i pointer alignment
15 MQCSP-RESERVED2 PIC X(4).
** Address of password
15 MQCSP-CSPPASSWORDPTR POINTER.
** Offset of password
15 MQCSP-CSPPASSWORDOFFSET PIC S9(9) BINARY.
** Length of password
15 MQCSP-CSPPASSWORDLENGTH PIC S9(9) BINARY.
** Ver:1 **

** Reserved
15 MQCSP-RESERVED3 PIC X(8).
** Address of initial key
15 MQCSP-INITIALKEYPTR POINTER.

```



```

**      Offset of initial key
15 MQCSP-INITIALKEYOFFSET PIC S9(9) BINARY.
**      Length of initial key
15 MQCSP-INITIALKEYLENGTH PIC S9(9) BINARY.
** Ver:2 **

**      Reserved
15 MQCSP-RESERVED4          PIC X(8).
**      Address of token
15 MQCSP-TOKENPTR          POINTER.
**      Offset of token
15 MQCSP-TOKENOFFSET      PIC S9(9) BINARY.
**      Length of token
15 MQCSP-TOKENLENGTH      PIC S9(9) BINARY.
** Ver:3 **

```

## MQCSP의 PL/I 선언

```

dcl
1 MQCSP based,
3 StrucId          char(4),          /* Structure identifier */
3 Version          fixed bin(31),    /* Structure version number */
3 AuthenticationType fixed bin(31), /* Type of authentication */
3 Reserved1        char(4),          /* Required for IBM i pointer
                                     alignment */
3 CSPUserIdPtr     pointer,          /* Address of user ID */
3 CSPUserIdOffset  fixed bin(31),    /* Offset of user ID */
3 CSPUserIdLength  fixed bin(31),    /* Length of user ID */
3 Reserved2        char(8),          /* Required for IBM i pointer
                                     alignment */
3 CSPPasswordPtr   pointer,          /* Address of password */
3 CSPPasswordOffset fixed bin(31), /* Offset of user ID */
3 CSPPasswordLength fixed bin(31), /* Length of user ID */
/* Version 1 */

3 Reserved3        char(8),          /* Reserved */
3 InitialKeyPtr    pointer,          /* Address of initial key */
3 InitialKeyOffset fixed bin(31),    /* Offset of initial key */
3 InitialKeyLength fixed bin(31); /* Length of initial key */
/* Version 2 */

```

### V 9.4.0

```

dcl
1 MQCSP based,
3 StrucId          char(4),          /* Structure identifier */
3 Version          fixed bin(31),    /* Structure version number */
3 AuthenticationType fixed bin(31), /* Type of authentication */
3 Reserved1        char(4),          /* Required for IBM i pointer
                                     alignment */
3 CSPUserIdPtr     pointer,          /* Address of user ID */
3 CSPUserIdOffset  fixed bin(31),    /* Offset of user ID */
3 CSPUserIdLength  fixed bin(31),    /* Length of user ID */
3 Reserved2        char(8),          /* Required for IBM i pointer
                                     alignment */
3 CSPPasswordPtr   pointer,          /* Address of password */
3 CSPPasswordOffset fixed bin(31), /* Offset of user ID */
3 CSPPasswordLength fixed bin(31), /* Length of user ID */
/* Version 1 */

3 Reserved3        char(8),          /* Reserved */
3 InitialKeyPtr    pointer,          /* Address of initial key */
3 InitialKeyOffset fixed bin(31),    /* Offset of initial key */
3 InitialKeyLength fixed bin(31); /* Length of initial key */
/* Version 2 */

3 Reserved4        char(8),          /* Reserved */
3 TokenPtr         pointer,          /* Address of Token */
3 TokenOffset      fixed bin(31),    /* Offset of Token */
3 TokenLength      fixed bin(31); /* Length of Token */
/* Version 3 */

```

## MQCSP의 Visual Basic 선언

```

Type MQCSP
StrucId As String*4 'Structure identifier'

```

Version	As Long	'Structure version number'
AuthenticationType	As Long	'Type of authentication'
Reserved1	As MQBYTE4	'Required for IBM i pointer alignment'
CSPUserIdPtr	As MQPTR	'Address of user ID'
CSPUserIdOffset	As Long	'Offset of user ID'
CSPUserIdLength	As Long	'Length of user ID'
Reserved2	As MQBYTE8	'Required for IBM i pointer alignment'
CSPPasswordPtr	As MQPTR	'Address of password'
CSPPasswordOffset	As Long	'Offset of password'
CSPPasswordLength	As Long	'Length of password'
End Type		

## 관련 개념

[인증 토큰에 대한 작업](#)

### **MQCSP의 경우 StrucId (MQCHAR4)**

보안 매개변수 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQCSP\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

#### **MQCSP\_STRUC\_ID**

보안 매개변수 구조 ID.

C 프로그래밍 언어의 경우 상수 MQCSP\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQCSP\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### **MQCSP의 버전 (MQLONG)**

MQCSP 구조 버전 번호입니다.

값은 다음과 같아야 합니다.

#### **MQCSP\_VERSION\_1**

버전-1 보안 매개변수 구조. 버전 1에서는 큐 관리자를 사용하여 인증하기 위해 MQCSP 구조에 사용자 ID 및 비밀번호를 포함할 수 있습니다.

#### **MQCSP\_VERSION\_2**

Version-2 보안 매개변수 구조. 버전 2에서는 큐 관리자를 사용하여 인증할 사용자 ID 및 비밀번호를 포함하고 비밀번호를 보호하는 데 사용되는 초기 키를 지정할 수 있습니다.

**V 9.4.0**

#### **MQCSP\_VERSION\_3**

Version-3 보안 매개변수 구조. 버전 3에서는 큐 관리자를 사용하여 인증하기 위해 MQCSP 구조에 사용자 ID 및 비밀번호 또는 인증 토큰을 포함할 수 있습니다. 비밀번호를 보호하는 데 사용되는 초기 키를 지정할 수도 있습니다.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### **MQCSP\_CURRENT\_VERSION**

보안 매개변수 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQCSP\_VERSION\_3입니다.

### **MQCSP의 경우 AuthenticationType (MQLONG)**

AuthenticationType은 입력 필드입니다. 초기값은 MQCSP\_AUTH\_NONE입니다.

이는 수행할 인증의 유형입니다. 올바른 값은 다음과 같습니다.

#### **MQCSP\_AUTH\_NONE**

사용자 ID 및 비밀번호 또는 인증 토큰 필드를 사용하지 마십시오.

#### **MQCSP\_AUTH\_USER\_ID\_AND\_PWD**

MQCSP 구조에서 사용자 ID 및 비밀번호를 사용하여 인증하십시오.

**V 9.4.0**

#### **MQCSP\_AUTH\_ID\_TOKEN**

MQCSP 구조에서 인증 토큰을 사용하여 인증하십시오.

기본값은 MQCSP\_AUTH\_NONE입니다. 기본 설정에서는 비밀번호 보호가 수행되지 않습니다.

인증이 필요한 경우 **MQCSP**를 설정해야 합니다.**AuthenticationType**  
-MQCSP\_AUTH\_USER\_ID\_AND\_PWD 또는 MQCSP\_AUTH\_ID\_TOKEN.

자세한 정보는 [MQCSP 비밀번호 보호](#)를 참조하십시오.

#### 관련 개념

[인증 토큰에 대한 작업](#)

#### **MQCSP의 경우 Reserved1 (MQBYTE4)**

IBM i의 포인터 정렬에 필요한 예약 필드입니다.

이 필드의 초기값은 모두 널입니다.

#### **MQCSP의 경우 CSPUserIdPtr (MQPTR)**

인증에 사용할 사용자 ID의 주소입니다.

입력 필드입니다. 이 필드의 초기값은 포인터를 지원하는 프로그래밍 언어로 된 널 포인터이며, 그렇지 않은 경우 모두 널 바이트 문자열입니다. *Version*이 MQCNO\_VERSION\_5 미만이면 이 필드는 무시됩니다.

이 필드는 *IDPWOS*의 **AUTHTYPE**이 큐 관리자의 *CONNAUTH* 필드에서 이름 지정된 경우 운영 체제 사용자 ID를 포함할 수 있습니다.

Windows에서 이는 완전한 도메인 사용자 ID일 수 있습니다.

이 필드는 *IDPWLDAP*의 **AUTHTYPE**이 큐 관리자의 *CONNAUTH* 필드에서 이름 지정된 경우 LDAP 사용자 ID를 포함할 수 있습니다.

#### **CSPUserIdMQCSP에 대한 오프셋 (MQLONG)**

인증에 사용할 사용자 ID의 오프셋 (바이트)입니다. 오프셋은 양수 또는 음수일 수 있습니다.

입력 필드입니다. 이 필드의 초기값은 0입니다.

#### **CSPUserIdMQCSP의 길이 (MQLONG)**

인증에 사용할 사용자 ID의 길이입니다.

사용자 ID의 최대 길이는 플랫폼에 따라 다릅니다. 사용자 ID를 참조하십시오. 사용자 ID의 길이가 허용되는 최대 길이보다 긴 경우 인증 요청이 MQRC\_CSP\_ERROR와 함께 실패합니다. IBM MQ의 이전 버전에서 리턴된 오류는 MQRC\_NOT\_AUTHORIZED입니다.

이 필드는 입력 필드입니다. 이 필드의 초기값은 0입니다.

#### **MQCSP의 경우 Reserved2 (MQBYTE8)**

IBM i의 포인터 정렬에 필요한 예약 필드입니다.

이 필드의 초기값은 모두 널입니다.

#### **MQCSP의 경우 CSPPasswordPtr (MQPTR)**

인증에 사용할 비밀번호의 주소입니다.

입력 필드입니다. 이 필드의 초기값은 포인터를 지원하는 프로그래밍 언어로 된 널 포인터이며, 그렇지 않은 경우 모두 널 바이트 문자열입니다. *Version*이 MQCNO\_VERSION\_5 미만이면 이 필드는 무시됩니다.

이 필드에는 설정에 따라 운영 체제 또는 LDAP 비밀번호 검사에서 거부되지만 인증 메소드에 전달되기 전에 IBM MQ에서 거부되지 않는 비어 있는 비밀번호가 포함될 수 있습니다.

#### **MQCSP의 경우 CSPPasswordOffset (MQLONG)**

이는 인증에 사용할 비밀번호의 오프셋 (바이트)입니다. 오프셋은 양수 또는 음수일 수 있습니다.

입력 필드입니다. 이 필드의 초기값은 0입니다.

## MQCSP의 경우 CSPPasswordLength (MQLONG)

인증에 사용할 비밀번호의 길이입니다.

비밀번호의 최대 길이는 MQ\_CSP\_PASSWORD\_LENGTH이고 256자입니다. 비밀번호의 길이가 허용되는 최대 길이보다 긴 경우 인증 요청이 MQRC\_CSP\_ERROR와 함께 실패합니다. IBM MQ의 이전 버전에서 리턴된 오류는 MQRC\_NOT\_AUTHORIZED입니다.

이 필드는 입력 필드입니다. 이 필드의 초기값은 0입니다.

## MQCSP의 경우 Reserved3 (MQBYTE8)

IBM i의 포인터 정렬에 필요한 예약 필드입니다.

이 필드의 초기값은 모두 널입니다.

## Multi MQCSP의 경우 InitialKeyPtr (MQPTR)

비밀번호 보호 시스템에 대한 초기 키의 주소입니다.

입력 필드입니다. 이 필드의 초기값은 포인터를 지원하는 프로그래밍 언어로 된 널 포인터이며, 그렇지 않은 경우 모두 널 바이트 문자열입니다. *Version* 이 MQCSP\_VERSION\_2미만인 경우 이 필드는 무시됩니다.

이 필드는 IBM i, AIX, Linux, and Windows 시스템에서 실행 중인 IBM MQ MQI clients에만 관련됩니다.

IBM MQ MQI clients는 IBM MQ 비밀번호 보호 시스템을 사용하여 일부 필드에 대해 암호화된 값을 제공할 수 있습니다. 초기 키를 사용하여 MQCSO 구조에 지정된 키 저장소의 비밀번호를 암호화한 경우, 동일한 클라이언트 애플리케이션의 MQCSP에 초기 키 필드를 포함하는지 확인하십시오.

IBM MQ MQI 클라이언트는 IBM MQ 비밀번호 보호 시스템을 사용하여 일부 필드에 대해 암호화된 값을 제공할 수 있습니다. 초기 키를 사용하여 MQCSO 구조에 지정된 키 저장소의 비밀번호를 암호화한 경우, 동일한 클라이언트 애플리케이션의 MQCSP 구조에 초기 키 필드를 포함하는지 확인하십시오.

초기 키는 암호화 알고리즘에서 이러한 값을 암호화하고 복호화하는 데 사용됩니다. 이러한 필드의 값이 **runmqicred** 유틸리티를 사용하여 암호화될 때 초기 키가 제공되는 경우, 큐 관리자에 연결할 때 클라이언트가 동일한 초기 키를 지정해야 합니다.

이 필드를 사용하여 지정된 초기 키는 *MQS MQI KEYFILE* 환경 변수 또는 클라이언트 구성 파일의 보안 스탠자에 있는 *MQIInitialKeyFile* 특성을 사용하여 지정된 초기 키를 대체합니다.

*InitialKeyOffset* 또는 *InitialKeyPtr* 를 사용하여 초기 키를 지정할 수 있지만 둘 다 지정할 수는 없습니다.

### 관련 태스크

[AIX, Linux 및 Windows에서 IBM MQ MQI 클라이언트에 대한 초기 키 제공](#)

[IBM MQ 컴포넌트 구성 파일에서 비밀번호 보호](#)

### 관련 참조

[runmqicred \(IBM MQ 클라이언트 비밀번호 보호\)](#)

548 페이지의 『MQSCO의 KeyRepoPasswordPtr (MQPTR)』

TLS키 저장소 비밀번호 문구의 주소 (바이트)입니다.

336 페이지의 『InitialKeyMQCSP의 오프셋 (MQLONG)』

MQCSP 구조의 시작부터 비밀번호 보호 시스템의 초기 키에 대한 오프셋 (바이트)입니다. 오프셋은 양수 또는 음수일 수 있습니다.

## Multi InitialKeyMQCSP의 오프셋 (MQLONG)

MQCSP 구조의 시작부터 비밀번호 보호 시스템의 초기 키에 대한 오프셋 (바이트)입니다. 오프셋은 양수 또는 음수일 수 있습니다.

*InitialKeyOffset* 또는 *InitialKeyPtr* 를 사용하여 초기 키를 지정할 수 있지만 둘 다 지정할 수는 없습니다. 자세한 정보는 *InitialKeyPtr* 필드의 설명을 참조하십시오.

입력 필드입니다. 이 필드의 초기값은 0입니다. *Version* 이 MQCSP\_VERSION\_2미만인 경우 이 필드는 무시됩니다.

## 관련 태스크

[IBM MQ 컴포넌트 구성 파일에서 비밀번호 보호](#)

[AIX, Linux 및 Windows에서 IBM MQ MQI 클라이언트에 대한 초기 키 제공](#)

## 관련 참조

336 페이지의 『MQCSP의 경우 InitialKeyPtr (MQPTR)』

[비밀번호 보호 시스템에 대한 초기 키의 주소입니다.](#)

### Multi InitialKeyMQCSP의 길이 (MQLONG)

비밀번호 보호 시스템의 초기 키 길이입니다.

입력 필드입니다. 이 필드의 초기값은 0입니다. *Version* 이 MQCSP\_VERSION\_2미만인 경우 이 필드는 무시됩니다.

## 관련 태스크

[IBM MQ 컴포넌트 구성 파일에서 비밀번호 보호](#)

[AIX, Linux 및 Windows에서 IBM MQ MQI 클라이언트에 대한 초기 키 제공](#)

### V 9.4.0 MQCSP의 경우 Reserved4 (MQBYTE8)

IBM i의 포인터 정렬에 필요한 예약 필드입니다.

이 필드의 초기값은 모두 널입니다.

### V 9.4.0 MQCSP의 경우 TokenPtr (MQPTR)

큐 관리자에 대한 인증에 사용되는 인증 토큰의 주소입니다.

입력 필드입니다. 이 필드의 초기값은 포인터를 지원하는 프로그래밍 언어로 된 널 포인터이며, 그렇지 않은 경우 모두 널 바이트 문자열입니다. *Version* 이 MQCSP\_VERSION\_3미만인 경우 이 필드는 무시됩니다.

이 필드는 AIX 또는 Linux 시스템에서 실행되는 IBM MQ 큐 관리자에 대한 IBM MQ MQI clients 연결과 관련됩니다.

*TokenOffset* 또는 *TokenPtr* 를 사용하여 인증 토큰을 지정할 수 있지만 둘 다 지정할 수는 없습니다.

자세한 정보는 [애플리케이션에서 인증 토큰 사용을 참조하십시오.](#)

## 관련 개념

[인증 토큰에 대한 작업](#)

## 관련 참조

337 페이지의 『MQCSP의 경우 TokenOffset (MQLONG)』

이는 MQCSP 구조의 시작부터 인증 토큰에 대한 오프셋 (바이트) 입니다. 오프셋은 양수 또는 음수일 수 있습니다.

### V 9.4.0 MQCSP의 경우 TokenOffset (MQLONG)

이는 MQCSP 구조의 시작부터 인증 토큰에 대한 오프셋 (바이트) 입니다. 오프셋은 양수 또는 음수일 수 있습니다.

입력 필드입니다. 이 필드의 초기값은 0입니다. *Version* 이 MQCSP\_VERSION\_3미만인 경우 이 필드는 무시됩니다.

*TokenOffset* 또는 *TokenPtr* 를 사용하여 토큰을 지정할 수 있지만 둘 다 지정할 수는 없습니다. 자세한 정보는 *TokenPtr* 필드의 설명을 참조하십시오.

## 관련 개념

[인증 토큰에 대한 작업](#)

## 관련 태스크

[애플리케이션에서 인증 토큰 사용](#)

## 관련 참조

337 페이지의 『MQCSP의 경우 TokenPtr (MQPTR)』

[큐 관리자에 대한 인증에 사용되는 인증 토큰의 주소입니다.](#)

## V 9.4.0 MQCSP의 경우 TokenLength (MQLONG)

큐 관리자에 대한 인증에 사용되는 인증 토큰의 길이입니다.

인증 토큰의 최대 길이는 8192바이트인 MQ\_CSP\_TOKEN\_LENGTH입니다. *TokenLength* 가 허용되는 최대 길이보다 큰 경우 인증 요청은 MQRC\_CSP\_ERROR로 실패합니다.

입력 필드입니다. 이 필드의 초기값은 0입니다. *Version* 이 MQCSP\_VERSION\_3미만인 경우 이 필드는 무시됩니다.

자세한 정보는 [애플리케이션에서 인증 토큰 사용을 참조하십시오](#).

### 관련 개념

[인증 토큰에 대한 작업](#)

## MQCTLO - 제어 콜백 옵션 구조

MQCTLO 구조는 제어 콜백 함수에 관하여 옵션을 지정하는 데 사용됩니다. 구조는 MQCTL 호출의 입력 및 출력 매개변수입니다.

### 가용성

MQCTLO 구조는 다음 플랫폼에서 사용 가능합니다.

-  AIX
-  IBM i
-  Linux
-  Windows
-  z/OS

및 해당 시스템에 연결된 IBM MQ MQI clients

### 버전

MQCTLO의 현재 버전은 MQCTLO\_VERSION\_1입니다.

### 문자 세트 및 인코딩

MQCTLO의 데이터는 **CodedCharSetId** 큐 관리자 속성에 의해 지정된 문자 세트 및 MQENC\_NATIVE에 의해 지정된 로컬 큐 관리자의 인코딩에 있어야 합니다. 그러나 애플리케이션이 MQ MQI 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트와 인코딩으로 작성되어야 합니다.

### 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 483. MQCTLO의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucID (구조 ID)	MQCTLO_STRUC_ID	'CTLO'
버전 (구조 버전 번호)	MQCTLO_VERSION_1	1
옵션 (옵션)	MQCTLO_NONE	널
옵션 (예약 필드)	Reserved 필드	
ConnectionArea (사용할 콜백 함수의 필드)	없음	널 포인터 또는 널 바이트

표 483. MQCTLO의 필드 (계속)

필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
<p><b>참고:</b></p> <p>1. C 프로그래밍 언어의 매크로 변수MQCTLO_DEFAULT는 테이블에 나열되는 값을 포함합니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오.</p> <pre>MQCTLO MyCTLO = {MQCTLO_DEFAULT};</pre>		

## 언어 선언

### MQCTLO의 C 선언

```
typedef struct tagMQCTLO MQCTLO;
struct tagMQCTLO {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    Options;          /* Options that control the action of MQCTL */
    MQLONG    Reserved;         /* Reserved field */

    MQPTR     ConnectionArea; /* Connection work area passed to the function */
};
```

### MQCTLO의 COBOL 선언

```
** MQCTLO structure
10  MQCTLO.
** Structure Identifier
15  MQCTLO-STRUCID                PIC X(4).
** Structure Version
15  MQCTLO-VERSION                PIC S9(9) BINARY.
** Options
15  MQCTLO-OPTIONS                PIC S9(9) BINARY.
** Reserved
15  MQCTLO-RESERVED                PIC S9(9) BINARY.
** ConnectionArea
15  MQCTLO-CONNECTIONAREA          POINTER
```

### MQCTLO의 PL/I 선언

```
dcl
1  MQCTLO based,
3  StrucId          char(4),          /* Structure identifier */
3  Version          fixed bin(31), /* Structure version */
3  Options          fixed bin(31), /* Options */
3  Reserved         fixed bin(31),
3  ConnectionArea  pointer;          /* Connection work area */
```

### MQCTLO의 경우 StrucId (MQCHAR4)

제어 옵션 구조의 구조 식별자입니다. 항상 입력 필드입니다. 해당 값은 MQCTLO\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

### MQCTLO\_STRUC\_ID

제어 옵션 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQCTLO\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQCTLO\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### MQCTLO의 버전 (MQLONG)

제어 옵션 구조 - Version 필드



구조 버전 번호이며 값은 다음과 같아야 합니다.

#### **MQCTLO\_VERSION\_1**

버전 1 제어 옵션 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### **MQCTLO\_CURRENT\_VERSION**

제어 옵션 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQCTLO\_VERSION\_1입니다.

#### **MQCTLO의 옵션 (MQLONG)**

제어 옵션 구조 - Options 필드

MQCTL의 조치를 제어하는 옵션.

#### **MQCTLO\_FAIL\_IF QUIESCING**

큐 관리자 또는 연결이 정지 상태인 경우, MQCTL 호출이 실패하도록 강제 실행합니다.

정지 상태인 경우에 메시지 이용자에 대한 알림이 발생하도록 하려면, MQCB 호출에서 전달되는 MQGMO 옵션에서 MQGMO\_FAIL\_IF QUIESCING을 지정하십시오.

#### **MQCTLO\_THREAD\_AFFINITY**

이 옵션은 애플리케이션이 동일 연결에 대해 모든 메시지 이용자가 동일한 스레드에서 호출되도록 요구함을 시스템에 알립니다. 이 스레드는 연결이 중지될 때까지 이용자의 모든 호출에 대해 사용됩니다.

**기본 옵션:** 설명한 옵션이 필요하지 않은 경우에는 다음 옵션을 사용하십시오.

#### **MQCTLO\_NONE**

기타 옵션이 지정되지 않았음을 표시하려면 이 값을 사용하십시오. 모든 옵션은 기본 값을 가정합니다.

MQCTLO\_NONE은 프로그램 문서를 지원하기 위해 정의되었습니다. 이는 이 옵션을 다른 옵션과 함께 사용하기 위한 용도가 아니지만, 해당 값이 0이므로 해당 사용을 감지할 수 없습니다.

입력 필드입니다. *Options* 필드의 초기값은 MQCTLO\_NONE입니다.

#### **MQCTLO에 대해 예약됨 (MQLONG)**

이 필드는 예약된 필드입니다. 값은 0이어야 합니다.

#### **MQCTLO의 경우 ConnectionArea (MQPTR)**

제어 옵션 구조 - ConnectionArea 필드

이 필드는 콜백 함수에서 사용할 수 있는 필드입니다.

큐 관리자는 이 필드의 콘텐츠를 기반으로 의사결정을 내리지 않으며, 이는 콜백에 대한 입력 매개변수인 MQCBC 구조의 ConnectionArea 필드에 변경 없이 전달됩니다.

MQOP\_START 및 MQOP\_START\_WAIT 이외의 모든 조작의 경우 이 필드는 무시됩니다.

이는 콜백 함수의 입력 및 출력 필드입니다. 이 필드의 초기값은 널 포인터이거나 널 바이트입니다.

#### **MQDH - 분배 헤더**

MQDH 구조는 해당 메시지가 전송 큐에 저장된 분배 목록 메시지일 때 메시지에 있는 추가 데이터를 설명합니다. 분배 목록 메시지는 다중 목적지 큐에 전송되는 메시지입니다. 추가 데이터는 MQOR 레코드의 배열 및 MQPMR 레코드의 배열 다음에 오는 MQDH 구조로 구성됩니다. 이 구조는 전송 큐에 메시지를 직접 넣거나, 전송 큐에서 메시지를 제거하는 특수화된 애플리케이션에서 사용됩니다(예: 메시지 채널 에이전트). 분배 목록에 메시지를 넣고자 하는 애플리케이션은 이 구조를 사용하지 않아야 합니다. 대신 MQOD 구조를 사용하여 분배 목록의 목적지를 정의해야 하고, MQPMO 구조를 사용하여 메시지 특성을 지정하거나 개별 목적지에 전송되는 메시지에 대한 정보를 수신해야 합니다.

#### **가용성**

MQDH 구조는 다음 플랫폼에서 사용 가능합니다.



-  AIX
-  IBM i
-  Linux
-  Windows

및 해당 시스템에 연결된 IBM MQ MQI clients

## 형식 이름

MQFMT\_DIST\_HEADER

## 문자 세트 및 인코딩

MQDH의 데이터는 MQENC\_NATIVE로 지정된 로컬 큐 관리자의 인코딩 및 **CodedCharSetId** 큐 관리자 속성으로 지정된 문자 세트에 있어야 합니다.

MQDH의 문자 세트 및 인코딩을 다음을 사용하여 **CodedCharSetId** 및 **Encoding** 필드로 설정하십시오.

- MQMD(MQDH 구조가 메시지 데이터의 시작에 있는 경우) 또는
- MQDH 구조에 선행하는 헤더 구조(다른 모든 경우).

## 사용법

애플리케이션이 메시지를 분배 목록에 넣고 일부 또는 모든 목적지가 원격인 경우, 큐 관리자는 애플리케이션 메시지 데이터에 MQXQH 및 MQDH 구조를 접두부로 지정하고 관련 전송 큐에 메시지를 배치합니다. 따라서 데이터는 메시지가 전송 큐에 있을 때 다음 순서로 발생합니다.

- MQXQH 구조
- MQDH 구조와 MQOR 및 MQPMR 레코드의 배열
- 애플리케이션 메시지 데이터

목적지에 따라 큐 관리자는 둘 이상의 그러한 메시지를 생성할 수 있고 이를 다른 전송 큐에 위치시킬 수 있습니다. 이 경우 해당 메시지에서 MQDH 구조는 애플리케이션에서 연 분배 목록에 의해 정의된 목적지의 다른 서브세트를 식별합니다.

분배 목록 메시지를 전송 큐에 직접 넣는 애플리케이션은 앞에서 설명한 순서를 따라야 하고 MQDH 구조가 올바른지 확인해야 합니다. MQDH 구조가 올바르지 않으면 큐 관리자는 이유 코드 MQRC\_DH\_ERROR로 MQPUT 또는 MQPUT1 호출에 실패할 수 있습니다.

분배 목록 메시지를 지원할 수 있도록 큐를 정의한 경우에만 분배 목록 형식으로 큐에 메시지를 저장할 수 있습니다. 804 페이지의 『큐의 속성』에 설명된 **DistLists** 큐 속성을 참조하십시오. 애플리케이션이 분배 목록을 지원하지 않는 큐에 직접 분배 목록 메시지를 넣는 경우 큐 관리자는 분배 목록 메시지를 개별 메시지로 분할하고 대신 큐에 배치합니다.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 484. MQDH에 대한 MQDH의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQDH_STRUC_ID	'DH---'
버전 (구조 버전 번호)	MQDH_VERSION_1	1
StrucLength (MQDH 구조의 길이+다음 레코드)	없음	0
인코딩 (MQPMR 레코드의 배열을 따르는 데이터의 숫자 인코딩)	없음	0

표 484. MQDH에 대한 MQDH의 필드 (계속)		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
CodedCharSetId (MQPMR 레코드의 배열을 따르는 데이터의 문자 세트 ID)	MQCCSI_UNDEFINED	0
형식 (MQPMR 레코드의 배열을 따르는 데이터의 형식 이름)	MQFMT_NONE	공백
플래그 (일반 플래그)	MQDHF_NONE	0
PutMsgRecFields (존재하는 MQPMR 필드를 표시하는 플래그)	MQPMRF_NONE	0
RecsPresent (존재하는 오브젝트 레코드 수)	없음	0
ObjectRec오프셋 (MQDH의 시작부터 첫 번째 오브젝트 레코드의 오프셋)	없음	0
PutMsgRecOffset (MQDH의 시작에서 첫 번째 넣기 메시지 레코드의 오프셋)	없음	0
<b>참고:</b> 1. ~ 기호는 단일 공백 문자를 나타냅니다. 2. C 프로그래밍 언어의 매크로 변수MQDH_DEFAULT는 테이블에 나열되는 값을 포함합니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오. <pre style="background-color: #f0f0f0; padding: 10px; margin: 10px 0;">MQDH MyDH = {MQDH_DEFAULT};</pre>		

## 언어 선언

### MQDH의 C 선언

```
typedef struct tagMQDH MQDH;
struct tagMQDH {
    MQCHAR4  StrucId;           /* Structure identifier */
    MQLONG   Version;          /* Structure version number */
    MQLONG   StrucLength;      /* Length of MQDH structure plus following
                               MQOR and MQPMR records */
    MQLONG   Encoding;        /* Numeric encoding of data that follows
                               the MQOR and MQPMR records */
    MQLONG   CodedCharSetId;  /* Character set identifier of data that
                               follows the MQOR and MQPMR records */
    MQCHAR8  Format;           /* Format name of data that follows the
                               MQOR and MQPMR records */
    MQLONG   Flags;           /* General flags */
    MQLONG   PutMsgRecFields; /* Flags indicating which MQPMR fields are
                               present */
    MQLONG   RecsPresent;     /* Number of MQOR records present */
    MQLONG   ObjectRecOffset; /* Offset of first MQOR record from start
                               of MQDH */
    MQLONG   PutMsgRecOffset; /* Offset of first MQPMR record from start
                               of MQDH */
};
```

### MQDH의 COBOL 선언

```
** MQDH structure
   10 MQDH.
**   Structure identifier
   15 MQDH-STRUCID      PIC X(4).
**   Structure version number
   15 MQDH-VERSION     PIC S9(9) BINARY.
**   Length of MQDH structure plus following MQOR and MQPMR records
   15 MQDH-STRUCLNGTH  PIC S9(9) BINARY.
```

```

** Numeric encoding of data that follows the MQOR and MQPMR records
15 MQDH-ENCODING PIC S9(9) BINARY.
** Character set identifier of data that follows the MQOR and MQPMR
** records
15 MQDH-CODEDCHARSETID PIC S9(9) BINARY.
** Format name of data that follows the MQOR and MQPMR records
15 MQDH-FORMAT PIC X(8).
** General flags
15 MQDH-FLAGS PIC S9(9) BINARY.
** Flags indicating which MQPMR fields are present
15 MQDH-PUTMSGRECFIELDS PIC S9(9) BINARY.
** Number of MQOR records present
15 MQDH-RECSPRESENT PIC S9(9) BINARY.
** Offset of first MQOR record from start of MQDH
15 MQDH-OBJECTRECOFFSET PIC S9(9) BINARY.
** Offset of first MQPMR record from start of MQDH
15 MQDH-PUTMSGRECOFFSET PIC S9(9) BINARY.

```

## MQDH의 PL/I 선언

```

dcl
1 MQDH based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 StrucLength fixed bin(31), /* Length of MQDH structure plus
/* following MQOR and MQPMR
/* records */
3 Encoding fixed bin(31), /* Numeric encoding of data that
/* follows the MQOR and MQPMR
/* records */
3 CodedCharSetId fixed bin(31), /* Character set identifier of data
/* that follows the MQOR and MQPMR
/* records */
3 Format char(8), /* Format name of data that follows
/* the MQOR and MQPMR records */
3 Flags fixed bin(31), /* General flags */
3 PutMsgRecFields fixed bin(31), /* Flags indicating which MQPMR
/* fields are present */
3 RecsPresent fixed bin(31), /* Number of MQOR records present */
3 ObjectRecOffset fixed bin(31), /* Offset of first MQOR record from
/* start of MQDH */
3 PutMsgRecOffset fixed bin(31); /* Offset of first MQPMR record from
/* start of MQDH */

```

## MQDH의 Visual Basic 선언

```

Type MQDH
StrucId As String*4 'Structure identifier'
Version As Long 'Structure version number'
StrucLength As Long 'Length of MQDH structure plus following'
'MQOR and MQPMR records'
Encoding As Long 'Numeric encoding of data that follows'
'the MQOR and MQPMR records'
CodedCharSetId As Long 'Character set identifier of data that'
'follows the MQOR and MQPMR records'
Format As String*8 'Format name of data that follows the'
'MQOR and MQPMR records'
Flags As Long 'General flags'
PutMsgRecFields As Long 'Flags indicating which MQPMR fields are'
'present'
RecsPresent As Long 'Number of MQOR records present'
ObjectRecOffset As Long 'Offset of first MQOR record from start'
'of MQDH'
PutMsgRecOffset As Long 'Offset of first MQPMR record from start'
'of MQDH'
End Type

```

### MQDH에 대한 StrucId (MQCHAR4)

분배 헤더 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQDH\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

### MQDH\_STRUC\_ID

분배 헤더 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQDH\_STRUC\_ID\_ARRAY도 정의됩니다. 이 값은 MQDH\_STRUC\_ID와 동일하지만 문자열이 아닌 문자의 배열입니다.

### **MQDH의 버전 (MQLONG)**

값은 다음과 같아야 합니다.

#### **MQDH\_VERSION\_1**

분배 헤더 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### **MQDH\_CURRENT\_VERSION**

분배 헤더 구조의 현재 버전.

이 필드의 초기값은 MQDH\_VERSION\_1입니다.

### **MQDH에 대한 StrucLength (MQLONG)**

이는 MQDH 구조의 시작에서 MQOR 및 MQPMR 레코드의 배열 다음에 오는 메시지 데이터의 시작까지의 바이트 수입니다. 데이터는 다음 순서로 발생합니다.

- MQDH 구조
- MQOR 레코드의 배열
- MQPMR 레코드의 배열
- 메시지 데이터

MQOR 및 MQPMR 레코드의 배열은 MQDH 구조 내에 포함된 오프셋에서 처리됩니다. 이러한 오프셋으로 인해 하나 이상의 MQDH 구조, 레코드 배열 및 메시지 데이터 사이의 사용되지 않은 바이트가 생성되는 경우, 사용되지 않은 바이트는 *StrucLength*의 값에 포함되어야 하지만 해당 바이트의 콘텐츠는 큐 관리자에 의해 보존되지 않습니다. MQOR 레코드의 배열이 MQPMR 레코드의 배열보다 선행하는 것이 올바릅니다.

이 필드의 초기값은 0입니다.

### **MQDH에 대한 인코딩 (MQLONG)**

이는 MQOR 및 MQPMR 레코드의 배열 다음에 오는 데이터의 숫자 인코딩을 지정합니다. MQDH 구조 자체의 숫자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다.

이 필드의 초기값은 0입니다.

### **MQDH의 CodedCharSetId (MQLONG)**

이는 MQOR 및 MQPMR 레코드의 배열 다음에 오는 데이터의 문자 세트 ID입니다. 이는 MQDH 구조 자체의 문자 데이터에 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 다음 특수 값을 사용할 수 있습니다.

#### **MQCCSI\_INHERIT**

이 구조의 문자 세트 ID를 상속합니다.

이 구조 다음에 오는 데이터의 문자 데이터는 이 구조와 동일한 문자 세트로 되어 있습니다.

큐 관리자가 구조의 실제 문자 세트 ID에 메시지로 송신한 구조에서 이 값을 변경합니다. 오류가 발생하지 않으면 MQGET 호출이 값 MQCCSI\_INHERIT를 리턴하지 않습니다.

MQMD의 *PutApplType* 필드 값이 MQAT\_BROKER인 경우 MQCCSI\_INHERIT를 사용할 수 없습니다.

이 값은 다음 환경에서 지원됩니다.

-  AIX
-  IBM i

-  Linux
-  Windows

및 해당 시스템에 연결된 IBM MQ 클라이언트  
이 필드의 초기값은 MQCCSI\_UNDEFINED입니다.

### **MQDHF에 대한 형식 (MQCHAR8)**

이는 MQOD 및 MQPMR 레코드의 배열 다음에 오는(마지막에 발생하는 사항) 데이터의 형식 이름입니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 이 필드의 코딩 규칙은 MQMD의 *Format* 필드에 대한 규칙과 동일합니다.

이 필드의 초기값은 MQFMT\_NONE입니다.

### **MQDHF의 플래그 (MQLONG)**

다음 플래그를 지정할 수 있습니다.

#### **MQDHF\_NEW\_MSG\_IDS**

분배 목록에서 각 목적지에 대한 새 메시지 ID를 생성하십시오. 넣기 메시지 레코드가 없거나 레코드가 있지만 *MsgId* 필드를 포함하지 않는 경우에만 설정하십시오.

이 플래그를 사용하면 분배 목록 메시지가 최종적으로 개별 메시지로 분할되는 순간까지 메시지 ID의 생성이 지연됩니다. 이는 분배 목록 메시지로 플로우해야 하는 제어 정보의 양을 최소화합니다.

애플리케이션이 메시지를 분배 목록에 넣으면, 큐 관리자는 다음 명령문이 모두 true인 경우 생성하는 MQDHF에서 MQDHF\_NEW\_MSG\_IDS를 설정합니다.

- 애플리케이션에서 제공하는 넣기 메시지 레코드가 없거나 제공된 레코드가 *MsgId* 필드를 포함하지 않습니다.
- MQMD의 *MsgId* 필드는 MQMI\_NONE이거나, MQPMO의 *Options* 필드가 MQPMO\_NEW\_MSG\_ID를 포함합니다.

플래그가 필요하지 않은 경우 다음을 지정하십시오.

#### **MQDHF\_NONE**

플래그가 지정되지 않았습니니다. MQDHF\_NONE은 프로그램 문서화를 지원하기 위해 정의됩니다. 이 상수는 다른 상수와 함께 사용하기 위한 용도는 아니지만, 값이 0이므로 그러한 사용을 감지할 수 없습니다.

이 필드의 초기값은 MQDHF\_NONE입니다.

### **MQDHF의 PutMsgRecFields (MQLONG)**

다음 플래그를 지정하지 않거나 하나 이상 지정할 수 있습니다.

#### **MQPMRF\_MSG\_ID**

메시지 ID 필드가 존재합니다.

#### **MQPMRF\_CORREL\_ID**

상관 ID 필드가 존재합니다.

#### **MQPMRF\_GROUP\_ID**

그룹 ID 필드가 존재합니다.

#### **MQPMRF\_FEEDBACK**

피드백 필드가 존재합니다.

#### **MQPMRF\_ACCOUNTING\_TOKEN**

Accounting-token 필드가 존재합니다.

MQPMR 필드가 존재하지 않는 경우 다음을 지정하십시오.

#### **MQPMRF\_NONE**

넣기 메시지 레코드 필드가 없습니다. MQPMRF\_NONE은 프로그램 문서화를 지원하기 위해 정의됩니다. 이 상수는 다른 상수와 함께 사용하기 위한 용도는 아니지만, 값이 0이므로 그러한 사용을 감지할 수 없습니다.

이 필드의 초기값은 MQPMRF\_NONE입니다.

### **MQDHE에 대한 RecsPresent (MQLONG)**

이는 목적지의 수입니다. 분배 목록은 항상 하나 이상의 목적지를 포함해야 하므로 *RecsPresent*은(는) 항상 0보다 커야 합니다.

이 필드의 초기값은 0입니다.

### **ObjectRecMQDHE에 대한 오프셋 (MQLONG)**

이는 목적지 큐의 이름을 포함하여, MQOR 오브젝트 레코드의 배열에서 첫 번째 레코드의 오프셋(바이트)을 제공합니다. 이 배열에 *RecsPresent*개의 레코드가 있습니다. 이러한 레코드(첫 번째 오브젝트 레코드와 이전 필드 사이에 건너뛴 바이트 포함)는 *StrucLength* 필드에서 제공한 길이에 포함됩니다.

분배 목록은 항상 하나 이상의 목적지를 포함해야 하므로 *ObjectRecOffset*은(는) 항상 0보다 커야 합니다.

이 필드의 초기값은 0입니다.

### **MQDHE에 대한 PutMsgRecOffset (MQLONG)**

이는 메시지 특성을 포함하여, MQPMR 넣기 메시지 레코드의 배열에서 첫 번째 레코드의 오프셋(바이트)을 제공합니다. 존재하는 경우 이 배열에 *RecsPresent*개의 레코드가 있습니다. 이러한 레코드(첫 번째 넣기 메시지 레코드와 이전 필드 사이에 건너뛴 바이트 포함)는 *StrucLength* 필드에서 제공한 길이에 포함됩니다.

넣기 메시지 레코드는 선택사항입니다. 레코드가 제공되지 않은 경우 *PutMsgRecOffset*은(는) 0이고, *PutMsgRecFields*에는 MQPMRF\_NONE 값이 있습니다.

이 필드의 초기값은 0입니다.

## **MQDLH - 데드 레터 헤더**

MQDLH 구조는 데드-레터 (미배달 메시지) 큐에 있는 메시지의 애플리케이션 메시지 데이터에 접두부를 붙이는 정보를 설명합니다. 큐 관리자 또는 메시지 채널 에이전트가 데드-레터 큐를 큐로 경로 재지정했거나 애플리케이션이 메시지를 큐에 직접 넣었기 때문에 메시지가 데드-레터 큐에 도착할 수 있습니다.

### **형식 이름**

MQFMT\_DEAD\_LETTER\_HEADER

### **문자 세트 및 인코딩**

MQDLH 구조의 필드는 *CodedCharSetId* 및 *Encoding* 필드에서 제공하는 문자 세트 및 인코딩으로 되어 있습니다. MQDLH가 애플리케이션 메시지 데이터의 시작에 있는 경우 이는 MQDLH 앞에 오는 헤더 구조 또는 MQMD 구조에 지정됩니다.

문자 세트는 큐 이름에 유효한 문자에 사용되는 단일 바이트 문자가 있어야 합니다.

Java/JMS용 IBM MQ 클래스를 사용 중이고 MQMD에 정의된 코드 페이지가 JVM (Java Virtual Machine) 에서 지원되지 않는 경우, MQDLH는 UTF-8 문자 세트로 작성됩니다.

### **사용법**

데드-레터 큐에 메시지를 직접 넣는 애플리케이션은 메시지 데이터에 MQDLH 구조로 접두부를 지정하고 적절한 값으로 필드를 초기화해야 합니다. 그러나 큐 관리자는 MQDLH 구조가 있는지 또는 올바른 값이 필드에 지정되었는지 확인하지 않습니다.

메시지가 너무 길어 데드-레터 큐에 넣을 수 없는 경우 애플리케이션은 다음 중 하나를 수행해야 합니다.

- 데드-레터 큐에 맞게 메시지 데이터를 자르십시오.
- 보조 저장영역에 메시지를 기록하고 이를 표시하는 데드-레터 큐에 예외 보고 메시지를 배치하십시오.

- 메시지를 제거하고 오류를 해당 발신자로 리턴하십시오. 메시지가 중요한 메시지인 경우 진원지에 메시지의 사본이 여전히 있는 경우에만 이를 수행하십시오(예를 들어, 통신 채널에서 메시지 채널 에이전트가 수신한 메시지).

앞선 조치 중 적절한 조치는(있는 경우) 애플리케이션의 디자인에 따라 다릅니다.

앞에서 MQDLH 구조와 함께 세그먼트인 메시지를 넣는 경우 큐 관리자가 특수 처리를 수행합니다. 추가 세부 사항은 MQMDE 구조에 대한 설명을 참조하십시오.

## 데드-레터 큐에 메시지 넣기

메시지를 데드-레터 큐에 넣을 때 MQPUT 또는 MQPUT1 호출에 사용되는 MQMD 구조는 다음을 제외하고 메시지와 연관된 MQMD (일반적으로 MQGET 호출에 의해 리턴되는 MQMD) 와 동일해야 합니다.

- *CodedCharSetId* 및 *Encoding* 필드를 MQDLH 구조의 필드에 사용되는 해당 문자 세트 및 인코딩으로 설정하십시오.
- *Format* 필드를 MQFMT\_DEAD\_LETTER\_HEADER로 설정하여 데이터가 MQDLH 구조로 시작함을 표시하십시오.
- 환경에 적합한 컨텍스트 옵션을 사용하여 컨텍스트 필드(*AccountingToken*, *ApplIdentityData*, *ApplOriginData*, *PutApplName*, *PutApplType*, *PutDate*, *PutTime*, *UserIdentifier*)를 설정하십시오.
  - 선행 메시지와 관련되지 않은 메시지를 데드-레터 큐에 넣는 애플리케이션은 MQPMO\_DEFAULT\_CONTEXT 옵션을 사용해야 합니다. 그러면 큐 관리자가 메시지 디스크립터의 모든 컨텍스트 필드를 기본값으로 설정합니다.
  - 방금 수신한 메시지를 데드-레터 큐에 넣는 서버 애플리케이션은 MQPMO\_PASS\_ALL\_CONTEXT 옵션을 사용하여 원래 컨텍스트 정보를 보존해야 합니다.
  - 방금 수신한 메시지에 대한 응답을 데드-레터 큐에 넣는 서버 애플리케이션은 MQPMO\_PASS\_IDENTITY\_CONTEXT 옵션을 사용해야 합니다. 이는 ID 정보를 보존하지만 원본 정보를 서버 애플리케이션의 정보로 설정합니다.
  - 통신 채널에서 수신한 메시지를 데드-레터 큐에 넣는 메시지 채널 에이전트는 MQPMO\_SET\_ALL\_CONTEXT 옵션을 사용하여 원래 컨텍스트 정보를 보존해야 합니다.

MQDLH 구조 자체에서 필드를 다음과 같이 설정하십시오.

- *CodedCharSetId*, *Encoding*, *Format* 필드를 MQDLH 구조 다음에 오는 데이터를 설명하는 값으로 설정하십시오(일반적으로, 원래 메시지 디스크립터의 값).
- *PutApplType*, *PutApplName*, *PutDate* 및 *PutTime* 컨텍스트 필드를 데드-레터 큐에 메시지를 넣는 애플리케이션에 적합한 값으로 설정하십시오. 이 값은 원래 메시지와 관련이 없습니다.
- 기타 필드는 적절하게 설정하십시오.

모든 필드에 올바른 값이 있고 문자 필드가 정의된 필드 길이까지 공백으로 채워졌는지 확인하십시오. 큐 관리자가 MQDLH 구조에서는 널 및 후속 문자를 공백으로 변환하지 않으므로 널 문자를 사용하여 문자 데이터를 이르게 종료하지 마십시오.

## 데드-레터 큐에서 메시지 가져오기

데드-레터 큐에서 메시지를 가져오는 애플리케이션은 메시지가 MQDLH 구조로 시작하는지 확인해야 합니다. 애플리케이션은 메시지 디스크립터 MQMD의 *Format* 필드를 조사하여 MQDLH 구조가 존재하는지 여부를 판별할 수 있습니다. 필드에 MQFMT\_DEAD\_LETTER\_HEADER 값이 있는 경우 메시지 데이터는 MQDLH 구조로 시작합니다. 또한 애플리케이션이 데드-레터 큐에서 가져오는 메시지가 원래 큐에 비해 너무 긴 경우에는 잘릴 수 있습니다.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 485. MQDLH에서 MQDLH의 필드

필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQDLH_STRUC_ID	'DLH~'
버전 (구조 버전 번호)	MQDLH_VERSION_1	1
이유 (데드-레터 큐에 도착한 이유 메시지)	MQRC_NONE	0
DestQName (원래 목적지 큐의 이름)	없음	널 문자열 또는 공백
DestQMgr이름 (원래 목적지 큐 관리자의 이름)	없음	널 문자열 또는 공백
인코딩 (MQDLH뒤에 오는 데이터의 숫자 인코딩)	없음	0
CodedCharSetId (MQDLH뒤에 오는 데이터의 문자 세트 ID)	MQCCSI_UNDEFINED	0
Format (MQDLH뒤에 오는 데이터의 형식 이름)	MQFMT_NONE	공백
PutAppl유형 (데드-레터 큐에 메시지를 넣는 애플리케이션의 유형)	없음	0
PutApplName (데드-레터 큐에 메시지를 넣는 애플리케이션의 이름)	없음	널 문자열 또는 공백
PutDate (메시지를 데드-레터 큐에 넣은 날짜)	없음	널 문자열 또는 공백
PutTime (메시지를 데드-레터 큐에 넣은 시간)	없음	널 문자열 또는 공백

**참고:**

- ~ 기호는 단일 공백 문자를 나타냅니다.
- 널 문자열 값 또는 공백은 C의 널 문자열과 기타 프로그래밍 언어의 공백 문자를 나타냅니다.
- C 프로그래밍 언어의 매크로 변수MQDLH\_DEFAULT는 테이블에 나열되는 값을 포함합니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오.

```
MQDLH MyDLH = {MQDLH_DEFAULT};
```

## 언어 선언

### MQDLH의 C 선언

```
typedef struct tagMQDLH MQDLH;
struct tagMQDLH {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    Reason;           /* Reason message arrived on dead-letter
    (undelivered-message) queue */

    MQCHAR48  DestQName;        /* Name of original destination queue */
    MQCHAR48  DestQMgrName;     /* Name of original destination queue
    manager */

    MQLONG    Encoding;         /* Numeric encoding of data that follows
    MQDLH */
    MQLONG    CodedCharSetId;  /* Character set identifier of data that
    follows MQDLH */
    MQCHAR8   Format;           /* Format name of data that follows
    MQDLH */
    MQLONG    PutApplType;      /* Type of application that put message on
    dead-letter (undelivered-message)
    queue */
    MQCHAR28  PutApplName;     /* Name of application that put message on
    dead-letter (undelivered-message)
    queue */
    MQCHAR8   PutDate;         /* Date when message was put on dead-letter
    (undelivered-message) queue */
    MQCHAR8   PutTime;         /* Time when message was put on the
```



```

                                dead-letter (undelivered-message)
                                queue */
};

```

## MQDLH의 COBOL 선언

```

**  MQDLH structure
   10 MQDLH.
**  Structure identifier
   15 MQDLH-STRUCID      PIC X(4).
**  Structure version number
   15 MQDLH-VERSION     PIC S9(9) BINARY.
**  Reason message arrived on dead-letter (undelivered-message) queue
   15 MQDLH-REASON     PIC S9(9) BINARY.
**  Name of original destination queue
   15 MQDLH-DESTQNAME   PIC X(48).
**  Name of original destination queue manager
   15 MQDLH-DESTQMGRNAME PIC X(48).
**  Numeric encoding of data that follows MQDLH
   15 MQDLH-ENCODING    PIC S9(9) BINARY.
**  Character set identifier of data that follows MQDLH
   15 MQDLH-CODEDCHARSETID PIC S9(9) BINARY.
**  Format name of data that follows MQDLH
   15 MQDLH-FORMAT     PIC X(8).
**  Type of application that put message on dead-letter
**  (undelivered-message) queue
   15 MQDLH-PUTAPPLTYPE PIC S9(9) BINARY.
**  Name of application that put message on dead-letter
**  (undelivered-message) queue
   15 MQDLH-PUTAPPLNAME PIC X(28).
**  Date when message was put on dead-letter (undelivered-message)
**  queue
   15 MQDLH-PUTDATE     PIC X(8).
**  Time when message was put on the dead-letter (undelivered-message)
**  queue
   15 MQDLH-PUTTIME    PIC X(8).

```

## MQDLH의 PL/I 선언

```

dcl
  1 MQDLH based,
    3 StrucId      char(4),          /* Structure identifier */
    3 Version      fixed bin(31), /* Structure version number */
    3 Reason       fixed bin(31), /* Reason message arrived on
                                dead-letter (undelivered-message)
                                queue */
    3 DestQName    char(48),        /* Name of original destination
                                queue */
    3 DestQMgrName char(48),        /* Name of original destination queue
                                manager */
    3 Encoding     fixed bin(31), /* Numeric encoding of data that
                                follows MQDLH */
    3 CodedCharSetId fixed bin(31), /* Character set identifier of data
                                that follows MQDLH */
    3 Format        char(8),         /* Format name of data that follows
                                MQDLH */
    3 PutApplType  fixed bin(31), /* Type of application that put
                                message on dead-letter
                                (undelivered-message) queue */
    3 PutApplName  char(28),        /* Name of application that put
                                message on dead-letter
                                (undelivered-message) queue */
    3 PutDate      char(8),         /* Date when message was put on
                                dead-letter (undelivered-message)
                                queue */
    3 PutTime      char(8);        /* Time when message was put on the
                                dead-letter (undelivered-message)
                                queue */

```

## MQDLH의 상위 레벨 어셈블러 선언

MQDLH	DSECT	
MQDLH_STRUCID	DS CL4	Structure identifier
MQDLH_VERSION	DS F	Structure version number
MQDLH_REASON	DS F	Reason message arrived on dead-letter

```

* (undelivered-message) queue
MQDLH_DESTQNAME DS CL48 Name of original destination queue
MQDLH_DESTQMGRNAME DS CL48 Name of original destination queue
* manager
MQDLH_ENCODING DS F Numeric encoding of data that follows
* MQDLH
MQDLH_CODEDCHARSETID DS F Character set identifier of data that
* follows MQDLH
MQDLH_FORMAT DS CL8 Format name of data that follows MQDLH
MQDLH_PUTAPPLTYPE DS F Type of application that put message on
* dead-letter (undelivered-message) queue
MQDLH_PUTAPPLNAME DS CL28 Name of application that put message on
* dead-letter (undelivered-message) queue
MQDLH_PUTDATE DS CL8 Date when message was put on
* dead-letter (undelivered-message) queue
MQDLH_PUTTIME DS CL8 Time when message was put on the
* dead-letter (undelivered-message) queue
*
MQDLH_LENGTH EQU *-MQDLH
ORG MQDLH
MQDLH_AREA DS CL(MQDLH_LENGTH)

```

## MQDLH의 Visual Basic 선언

```

Type MQDLH
  StrucId As String*4 'Structure identifier'
  Version As Long 'Structure version number'
  Reason As Long 'Reason message arrived on dead-letter'
  '(undelivered-message) queue'
  DestQName As String*48 'Name of original destination queue'
  DestQMgrName As String*48 'Name of original destination queue'
  'manager'
  Encoding As Long 'Numeric encoding of data that follows'
  'MQDLH'
  CodedCharSetId As Long 'Character set identifier of data that'
  'follows MQDLH'
  Format As String*8 'Format name of data that follows MQDLH'
  PutApplType As Long 'Type of application that put message on'
  'dead-letter (undelivered-message) queue'
  PutApplName As String*28 'Name of application that put message on'
  'dead-letter (undelivered-message) queue'
  PutDate As String*8 'Date when message was put on dead-letter'
  '(undelivered-message) queue'
  PutTime As String*8 'Time when message was put on the'
  'dead-letter (undelivered-message) queue'
End Type

```

### MQDLH용 StrucId (MQCHAR4)

데드-레터 헤더 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQDLH\_STRUC\_ID입니다. 값은 다음과 같아야 합니다.

### MQDLH\_STRUC\_ID

데드-레터 헤더 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQDLH\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQDLH\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### MQDLH용 버전 (MQLONG)

버전은 구조 버전 번호입니다.

값은 다음과 같아야 합니다.

### MQDLH\_VERSION\_1

데드 레터 헤더 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

### MQDLH\_CURRENT\_VERSION

데드 레터 헤더 구조의 현재 버전.

이 필드의 초기값은 MQDLH\_VERSION\_1입니다.

## MQDLH에 대한 이유 (MQLONG)

이유 필드는 메시지가 원래 목적지 큐 대신 데드-레터 큐에 배치된 이유를 식별합니다.

이는 메시지가 원래 목적지 큐 대신에 데드 레터에 큐에 배치된 이유를 식별합니다. 이는 MQFB\_\* 또는 MQRC\_\* 값 중 하나여야 합니다(예: MQRC\_Q\_FULL). 발생할 수 있는 공통 MQFB\_\* 값에 대한 세부사항은 412 페이지의 『MQMD - 메시지 디스크립터』에서 *Feedback* 필드에 대한 세부사항을 참조하십시오.

값의 범위가 MQFB\_IMS\_FIRST - MQFB\_IMS\_LAST인 경우 실제 IMS 오류 코드는 *Reason* 필드의 값에서 MQFB\_IMS\_ERROR를 빼서 판별할 수 있습니다.

일부 MQFB\_\* 값은 이 필드에서만 발생합니다. 이는 데드-레터 큐에 전송된 저장소 메시지, 트리거 메시지 또는 전송 큐 메시지와 관련됩니다. 이러한 항목은 다음과 같습니다.

### MQFB\_APPL\_CANNOT\_BE\_STARTED(X'00000109')

트리거 메시지를 처리하는 애플리케이션은 트리거 메시지의 *AppId* 필드에서 이름 지정된 애플리케이션을 시작할 수 없습니다(579 페이지의 『MQTM - 트리거 메시지』 참조).

z/OS에서 CKTI CICS 트랜잭션은 트리거 메시지를 처리하는 애플리케이션의 예제입니다.

### MQFB\_APPL\_TYPE\_ERROR(X'0000010B')

트리거 메시지를 처리하는 애플리케이션은 트리거 메시지의 *AppType* 필드가 올바르지 않아 애플리케이션을 시작할 수 없습니다(579 페이지의 『MQTM - 트리거 메시지』 참조).

z/OS에서 CKTI CICS 트랜잭션은 트리거 메시지를 처리하는 애플리케이션의 예제입니다.

### MQFB\_BIND\_OPEN\_CLUSRCVR\_DEL(X'00000119')

MQOO\_BIND\_ON\_OPEN 옵션으로 열린 클러스터 큐를 대상으로 하는 SYSTEM.CLUSTER.TRANSMIT.QUEUE에 메시지가 있지만 목적지 큐에 메시지를 전송하는 데 사용될 리모트 클러스터 수신자 채널이 메시지를 보낼 수 있기 전에 삭제되었습니다. MQOO\_BIND\_ON\_OPEN이 지정되었으므로 큐를 열었을 때 선택된 채널만 메시지를 전송하는 데 사용할 수 있습니다. 이 채널이 더 이상 사용 가능하지 않으므로 메시지가 데드-레터 큐에 배치됩니다.

### MQFB\_NOT\_A\_REPOSITORY\_MSG(X'00000118')

메시지가 저장소 메시지가 아닙니다.

### MQFB\_STOPPED\_BY\_CHAD\_EXIT(X'00000115')

채널 자동 정의 엑시트가 메시지를 중지했습니다.

### MQFB\_STOPPED\_BY\_MSG\_EXIT(X'0000010D')

채널 메시지 엑시트가 메시지를 중지했습니다.

### MQFB\_TM\_ERROR(X'0000010A')

MQMD의 *Format* 필드가 MQFMT\_TRIGGER를 지정하지만 메시지가 올바른 MQTM 구조로 시작되지 않습니다. 예를 들어, *StrucId* 니모닉 아이 캐처가 올바르지 않거나 *Version*이 인식되지 않거나 트리거 메시지의 길이가 MQTM 구조를 포함하기에 충분하지 않을 수 있습니다.

z/OS에서 CKTI CICS 트랜잭션은 트리거 메시지를 처리하는 애플리케이션의 예제이며 이 피드백 코드를 생성할 수 있습니다.

### MQFB\_XMIT\_Q\_MSG\_ERROR(X'0000010F')

메시지 채널 에이전트에서 전송 큐의 메시지가 올바른 형식이 아님을 발견했습니다. 메시지 채널 에이전트는 이 피드백 코드를 사용하여 데드-레터 큐에 메시지를 넣습니다.

일반적인 원인 하나는 메시지가 직접 전송 큐에 전달되어 메시지에 예상 XQH 헤더가 없다는 점입니다. 애플리케이션이 XQXQH 헤더를 빌드하지 않는 한 메시지는 리모트 큐를 통해 전송 큐로 전달되어야 합니다.

이 필드의 초기값은 MQRC\_NONE입니다.

## MQDLH의 경우 DestQName (MQCHAR48)

DestQName은 메시지의 원래 목적지인 메시지 큐의 이름입니다.

이 필드의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백입니다.

## DestQMGrMQDLH의 이름 (MQCHAR48)

DestQMGrName은 메시지의 원래 목적지인 큐 관리자의 이름입니다.

이 필드의 길이는 MQ\_Q\_MGR\_NAME\_LENGTH로 제공됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백입니다.

## MQDLH용 인코딩 (MQLONG)

인코딩은 MQDLH 구조 다음에 오는 데이터(일반적으로 원래 메시지의 데이터)의 숫자 인코딩입니다. 이는 MQDLH 구조 자체의 숫자 데이터에 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다.

이 필드의 초기값은 0입니다.

## MQDLH용 CodedCharSetId (MQLONG)

CodedCharSetId는 MQDLH 구조를 통해 플로우하는 데이터(일반적으로 원래 메시지의 데이터)의 문자 세트 ID입니다. 이는 MQDLH 구조 자체의 문자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 다음 특수 값을 사용할 수 있습니다.

### MQCCSI\_INHERIT

이 구조 다음에 오는 데이터의 문자 데이터는 이 구조와 동일한 문자 세트로 되어 있습니다.

큐 관리자가 구조의 실제 문자 세트 ID에 메시지로 송신한 구조에서 이 값을 변경합니다. 오류가 발생하지 않으면 MQGET 호출에서 MQCCSI\_INHERIT 값을 리턴하지 않습니다.

MQMD의 *PutApplType* 필드 값이 MQAT\_BROKER인 경우 MQCCSI\_INHERIT를 사용할 수 없습니다.

이 값은 다음 환경에서 지원됩니다.

-  AIX
-  IBM i
-  Linux
-  Windows

및 해당 시스템에 연결된 IBM MQ 클라이언트

이 필드의 초기값은 MQCCSI\_UNDEFINED입니다.

## MQDLH의 형식 (MQCHAR8)

형식은 MQDLH 구조 다음에 오는 데이터(일반적으로 원래 메시지의 데이터)의 형식 이름입니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 이 필드를 코딩하는 규칙은 MQMD의 *Format* 필드를 코딩하는 규칙과 같습니다.

이 필드의 길이는 MQ\_FORMAT\_LENGTH에 지정되어 있습니다. 이 필드의 초기값은 MQFMT\_NONE입니다.

## PutApplMQDLH의 유형 (MQLONG)

PutApplType은 데드-레터(전달되지 않은 메시지) 큐에 메시지를 넣는 애플리케이션의 유형입니다.

이 필드에는 메시지 디스크립터 MQMD의 *PutApplType* 필드와 동일한 의미가 있습니다(세부사항은 [412 페이지](#)의 『MQMD - 메시지 디스크립터』 참조).

큐 관리자가 메시지를 데드-레터 큐로 경로 재지정하는 경우 *PutApplType*은 값 MQAT\_QMGR을 가집니다.

이 필드의 초기값은 0입니다.

### **MQDLH의 PutAppl이름 (MQCHAR28)**

PutApplName은 데드-레터(전달되지 않은 메시지) 큐에 메시지를 넣는 애플리케이션의 이름입니다.

이름의 형식은 *PutApplType* 필드에 따라 다릅니다. 형식은 릴리스마다 다를 수 있습니다. [412 페이지의 『MQMD - 메시지 디스크립터』](#)에서 *PutApplName* 필드에 대한 설명을 참조하십시오.

큐 관리자가 메시지를 데드-레터 큐로 경로 재지정하는 경우 *PutApplName*은 필요한 경우 공백으로 채워진 큐 관리자 이름의 처음 28자를 포함합니다.

이 필드의 길이는 MQ\_PUT\_APPL\_NAME\_LENGTH로 지정됩니다. 이 필드의 초기값은 C에서는 널 문자열이고 기타 프로그래밍 언어에서는 28자의 공백 문자입니다.

### **MQDLH의 경우 PutDate (MQCHAR8)**

PutDate는 메시지를 데드-레터(전달되지 않은 메시지) 큐에 넣은 날짜입니다.

이 필드가 큐 관리자에 의해 생성될 때 날짜에 사용되는 형식입니다.

• YYYYMMDD

여기서 각 문자는 다음을 나타냅니다.

#### **YYYY**

연도(4자리 숫자)

#### **MM**

연 중 월(01에서 12)

#### **DD**

월 중 일(01에서 31)

그리니치 표준시(GMT)에 맞게 정확하게 설정된 시스템 시계에 따라 *PutDate* 및 *PutTime* 필드에 GMT가 사용 됩니다.

이 필드의 길이는 MQ\_PUT\_DATE\_LENGTH로 지정됩니다. 이 필드의 초기값은 C에서는 널 문자열이고 기타 프로그래밍 언어에서는 8자의 공백 문자입니다.

### **MQDLH의 경우 PutTime (MQCHAR8)**

PutTime은 메시지를 데드-레터(전달되지 않은 메시지) 큐에 넣은 시간입니다.

이 필드가 큐 관리자에 의해 생성될 때 시간에 사용되는 형식입니다.

• HHMMSSSTH

여기서 각 문자는 다음을 나타냅니다.

#### **HH**

시간(00에서 23까지)

#### **MM**

분(00에서 59까지)

#### **SS**

초(00 - 59, 참고 참조)

#### **T**

1/10초(0에서 9)

#### **H**

1/100초(0에서 9)

**참고:** 시스템 시계가 매우 정확한 시간 표준에 동기화되는 경우 매우 드물게 *PutTime*에서 초에 대해 60 또는 61이 리턴될 수 있습니다. 이는 윤초가 글로벌 시간 표준으로 삽입되는 경우 발생합니다.

그리니치 표준시(GMT)에 맞게 정확하게 설정된 시스템 시계에 따라 *PutDate* 및 *PutTime* 필드에 GMT가 사용 됩니다.

이 필드의 길이는 MQ\_PUT\_TIME\_LENGTH로 지정됩니다. 이 필드의 초기값은 C에서는 널 문자열이고 기타 프로그래밍 언어에서는 8자의 공백 문자입니다.

## MQDMHO - 메시지 핸들 삭제 옵션

MQDMHO 구조를 사용하여 애플리케이션이 메시지 핸들을 삭제하는 방법을 제어하는 옵션을 지정할 수 있습니다. 구조는 MQDLTMH 호출에서 입력 매개변수입니다.

## 문자 세트 및 인코딩

MQDMHO의 데이터는 애플리케이션의 문자 세트 및 애플리케이션의 인코딩 (MQENC\_NATIVE)에 있어야 합니다.

## 필드

참고: 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 486. MQDMHO의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQDMHO_STRUC_ID	'DMHO'
버전 (구조 버전 번호)	MQDMHO_VERSION_1	1
옵션 (옵션)	MQDMHO_NONE	0

**참고:**

- C 프로그래밍 언어의 매크로 변수MQDMHO\_DEFAULT는 테이블에 나열되는 값을 포함합니다. 즉, 구조 내의 필드에 대한 초기값을 제공하기 위해 다음과 같은 방법으로 사용될 수 있습니다.

```
MQDMHO MyDMHO = {MQDMHO_DEFAULT};
```

## 언어 선언

MQDMHO의 C 선언

```
typedef struct tagMQDMHO;
struct tagMQDMHO {
    MQCHAR4    StrucId;          /* Structure identifier */
    MQLONG    Version;         /* Structure version number */
    MQLONG    Options;         /* Options that control the action of MQDLTMH */
};
```

MQDMHO의 COBOL 선언

```
** MQDMHO structure
10 MQDMHO.
** Structure identifier
15 MQDMHO-STRUCID PIC X(4).
** Structure version number
15 MQDMHO-VERSION PIC S9(9) BINARY.
** Options that control the action of MQDLTMH
15 MQDMHO-OPTIONS PIC S9(9) BINARY.
```

MQDMHO의 PL/I 선언

```
dcl
1 MQDMHO based,
3 StrucId char(4), /* Structure identifier */
```

```

3 Version          fixed bin(31), /* Structure version number */
3 Options          fixed bin(31), /* Options that control the action of MQDLTMH */

```

MQDMHO의 상위 레벨 어셈블러 선언

```

MQDMHO           DSECT
MQDMHO_STRUCID   DS   CL4   Structure identifier
MQDMHO_VERSION   DS   F     Structure version number
MQDMHO_OPTIONS   DS   F     Options that control the action of
*                               MQDLTMH
MQDMHO_LENGTH    EQU   *-MQDMHO
MQDMHO_AREA      DS   CL(MQDMHO_LENGTH)

```

### MQDMHO의 경우 StrucId (MQCHAR4)

삭제 메시지 핸들 옵션 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQDMHO\_STRUC\_ID입니다. 값은 다음과 같아야 합니다.

#### MQDMHO\_STRUC\_ID

삭제 메시지 핸들 옵션 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQDMHO\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQDMHO\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### MQDMHO의 버전 (MQLONG)

구조 버전 번호이며 값은 다음과 같아야 합니다.

#### MQDMHO\_VERSION\_1

버전-1 삭제 메시지 핸들 옵션 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### MQDMHO\_CURRENT\_VERSION

삭제 메시지 핸들 옵션 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQDMHO\_VERSION\_1입니다.

### MQDMHO의 옵션 (MQLONG)

값은 다음과 같아야 합니다.

#### MQDMHO\_NONE

옵션이 지정되지 않았습니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQDMHO\_NONE입니다.

### MQDMPO - 메시지 특성 삭제 옵션

MQDMPO 구조를 사용하면 애플리케이션이 메시지의 특성이 삭제되는 방법을 제어하는 옵션을 지정할 수 있습니다. 구조는 MQDLTMP 호출의 입력 매개변수입니다.

### 문자 세트 및 인코딩

MQDMPO의 데이터는 애플리케이션의 문자 세트 및 애플리케이션의 인코딩 (MQENC\_NATIVE) 에 있어야 합니다.

### 필드

참고: 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 487. MQDPMO의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQDMPO_STRUC_ID	'DPMO'
버전 (구조 버전 번호)	MQDMPO_VERSION_1	1
옵션 (MQDPMO의 조치를 제어하는 옵션)	MQDLTMP 조치를 제어하는 옵션	MQDMPO_NONE
<p><b>참고:</b></p> <p>1. C 프로그래밍 언어의 매크로 변수MQDMPO_DEFAULT는 테이블에 나열되는 값을 포함합니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오.</p> <pre>MQDMPO MyDPMO = {MQDMPO_DEFAULT};</pre>		

## 언어 선언

### MQDPMO의 C 선언

```
typedef struct tagMQDMPO MQDMPO;
struct tagMQDMPO {
    MQCHAR4  StrucId;          /* Structure identifier */
    MQLONG   Version;         /* Structure version number */
    MQLONG   Options;        /* Options that control the action of
                             MQDLTMP */
};
```

### MQDPMO의 COBOL 선언

```
** MQDPMO structure
10 MQDPMO.
** Structure identifier
15 MQDPMO-STRUCID PIC X(4).
** Structure version number
15 MQDPMO-VERSION PIC S9(9) BINARY.
** Options that control the action of MQDLTMP
15 MQDPMO-OPTIONS PIC S9(9) BINARY.
```

### MQDPMO의 PL/I 선언

```
Dcl
1 MQDPMO based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 Options fixed bin(31), /* Options that control the action
of MQDLTMP */
```

### MQDPMO의 상위 레벨 어셈블러 선언

```
MQDMPO DSECT
MQDMPO_STRUCID DS CL4 Structure identifier
MQDMPO_VERSION DS F Structure version number
MQDMPO_OPTIONS DS F Options that control the
* action of MQDLTMP
MQDMPO_LENGTH EQU *-MQDMPO
MQDMPO_AREA DS CL(MQDMPO_LENGTH)
```

### MQDPMO의 경우 StrucId (MQCHAR4)

삭제 메시지 특성 옵션 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQDMPO\_STRUC\_ID입니다. 값은 다음과 같아야 합니다.



## **MQDMPO\_STRUC\_ID**

삭제 메시지 특성 옵션 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQDMPO\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQDMPO\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

## **MQDMPO의 버전 (MQLONG)**

메시지 특성 삭제 옵션 구조 - 버전 필드

구조 버전 번호입니다. 값은 다음과 같아야 합니다.

### **MQDMPO\_VERSION\_1**

삭제 메시지 특성 옵션 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

### **MQDMPO\_CURRENT\_VERSION**

삭제 메시지 특성 옵션 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQDMPO\_VERSION\_1입니다.

## **MQDMPO에 대한 옵션 (MQLONG)**

메시지 특성 삭제 옵션 구조 - 옵션 필드

**위치 옵션:** 다음 옵션은 특성 커서와 비교하여 특성의 상대 위치와 관련이 있습니다.

### **MQDMPO\_DEL\_FIRST**

지정된 이름과 일치하는 첫 번째 특성을 삭제합니다.

### **MQDMPO\_DEL\_PROP\_UNDER\_CURSOR**

특성 커서로 지정된 특성을 삭제합니다. 해당 특성은 MQIMPO\_INQ\_FIRST 또는 MQIMPO\_INQ\_NEXT 옵션을 사용하여 마지막으로 조회된 특성입니다.

메시지 핸들이 재사용될 때 특성 커서가 재설정됩니다. 이는 메시지 핸들이 MQPUT 호출의 MQPMO 구조 또는 MQGET 호출의 MQGMO 구조에 대한 *MsgHandle* 필드에서 지정될 때도 재설정됩니다.

특성 커서가 아직 설정되지 않았을 때 이 옵션이 사용되는 경우 완료 코드 MQCC\_FAILED 및 이유 MQRC\_PROPERTY\_NOT\_AVAILABLE과 함께 호출이 실패합니다. 특성 커서로 지정된 특성이 이미 삭제된 경우 완료 코드 MQCC\_FAILED 및 이유 MQRC\_PROPERTY\_NOT\_AVAILABLE과 함께 호출이 실패합니다.

해당 옵션이 필요하지 않은 경우 다음 옵션을 사용할 수 있습니다.

### **MQDMPO\_NONE**

옵션이 지정되지 않았습니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQDMPO\_DEL\_FIRST입니다.

## **MQEPH - 임베드된 PCF 헤더**

MQEPH 구조는 해당 메시지가 프로그래밍 가능 명령 형식(PCF) 메시지일 때 메시지에 있는 추가 데이터를 설명합니다. *PCFHeader* 필드는 이 구조를 따르는 PCF 매개변수를 정의하고 이는 다른 헤더가 있는 PCF 메시지를 따를 수 있도록 합니다.

## **형식 이름**

MQFMT\_EMBEDDED\_PCF

## **문자 세트 및 인코딩**

MQEPH의 데이터는 MQENC\_NATIVE로 지정된 로컬 큐 관리자의 인코딩 및 **CodedCharSetId** 큐 관리자 속성으로 지정된 문자 세트에 있어야 합니다.

MQEPH의 문자 세트 및 인코딩을 MQMD의 *CodedCharSetId* 및 *Encoding* 필드 (MQEPH 구조가 메시지 데이터의 시작에 있는 경우) 또는 MQEPH 구조 앞에 오는 헤더 구조 (다른 모든 경우) 로 설정하십시오.

## 사용법

명령을 명령 서버 또는 기타 큐 관리자 PCF 수용 서버에 보내기 위해 MQEPH 구조를 사용할 수 없습니다.

마찬가지로 명령 서버 또는 기타 큐 관리자 PCF-수용 서버는 MQEPH 구조를 포함하여 응답 또는 이벤트를 생성하지 않습니다.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 488. MQEPH에 대한 MQEPH의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQEPH_STRUC_ID	'EPH~'
버전 (구조 버전 번호)	MQEPH_VERSION_1	1
StrucLength (MQEPH 구조의 길이+MQCFH 및 뒤에 오는 매개변수 구조)	MQEPH_STRUC_LENGTH_FIXED	68
인코딩 (마지막 PCF 매개변수 구조를 따르는 데이터의 숫자 인코딩)	없음	0
CodedCharSetId (마지막 PCF 매개변수 구조를 따르는 데이터의 문자 세트 ID)	MQCCSI_UNDEFINED	0
형식 (마지막 PCF 매개변수 구조를 따르는 데이터의 형식 이름)	MQFMT_NONE	공백
플래그 (플래그)	MQEPH_NONE	0
PCFHeader (PCF (Programmable Command Format) 헤더)	361 페이지의 표 489에 정의된 이름 및 값	0

**참고:**

- ~ 기호는 단일 공백 문자를 나타냅니다.
- C 프로그래밍 언어의 매크로 변수 MQEPH\_DEFAULT는 테이블에 나열되는 값을 포함합니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오.

```
MQEPH MyEPH = {MQEPH_DEFAULT};
```

## 언어 선언

MQEPH의 C 선언

```
typedef struct tagMQEPH MQEPH;
struct tagMQDH {
    MQCHAR4  StrucId;           /* Structure identifier */
    MQLONG   Version;          /* Structure version number */
    MQLONG   StrucLength;      /* Total length of MQEPH including the MQCFH
                               and parameter structures that follow it */
    MQLONG   Encoding;         /* Numeric encoding of data that follows last
                               PCF parameter structure */
    MQLONG   CodedCharSetId;   /* Character set identifier of data that
                               follows last PCF parameter structure */
    MQCHAR8  Format;           /* Format name of data that follows last PCF
                               parameter structure */
    MQLONG   Flags;            /* Flags */
    MQCFH    PCFHeader;        /* Programmable command format header */
};
```

## MQEPH의 COBOL 선언

```
** MQEPH structure
10 MQEPH.
** Structure identifier
15 MQEPH-STRUCID PIC X(4).
** Structure version number
15 MQEPH-VERSION PIC S9(9) BINARY.
** Total length of MQEPH structure including the MQCFH
** and parameter structures that follow it
15 MQEPH-STRUCLength PIC S9(9) BINARY.
** Numeric encoding of data that follows last
** PCF structure
15 MQEPH-ENCODING PIC S9(9) BINARY.
** Character set identifier of data that
** follows last PCF parameter structure
15 MQEPH-CODEDCHARSETID PIC S9(9) BINARY.
** Format name of data that follows last PCF
** parameter structure
15 MQEPH-FORMAT PIC X(8).
** Flags
15 MQEPH-FLAGS PIC S9(9) BINARY.
** Programmable command format header
15 MQEPH-PCFHEADER.
** Structure type
20 MQEPH-PCFHEADER-TYPE PIC S9(9) BINARY.
** Structure length
20 MQEPH-PCFHEADER-STRUCLength PIC S9(9) BINARY.
** Structure version number
20 MQEPH-PCFHEADER-VERSION PIC S9(9) BINARY.
** Command identifier
20 MQEPH-PCFHEADER-COMMAND PIC S9(9) BINARY.
** Message sequence number
20 MQEPH-PCFHEADER-MSGSEQNUMBER PIC S9(9) BINARY.
** Control options
20 MQEPH-PCFHEADER-CONTROL PIC S9(9) BINARY.
** Completion code
20 MQEPH-PCFHEADER-COMPCODE PIC S9(9) BINARY.
** Reason code qualifying completion code
20 MQEPH-PCFHEADER-REASON PIC S9(9) BINARY.
** Count of parameter structures
20 MQEPH-PCFHEADER-PARAMETERCOUNT PIC S9(9) BINARY.
```

## MQEPH의 PL/I 선언

```
dcl
1 MQEPH based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 StrucLength fixed bin(31), /* Total Length of MQEPH including the
MQCFH and parameter structures that
follow it
3 Encoding fixed bin(31), /* Numeric encoding of data that follows
last PCF parameter structure
3 CodedCharSetId fixed bin(31), /* Character set identifier of data that
follows last PCF parameter structure
3 Format char(8), /* Format name of data that follows last
PCF parameter structure */
3 Flags fixed bin(31), /* Flags */
3 PCFHeader, /* Programmable command format header
5 Type fixed bin(31), /* Structure type */
5 StrucLength fixed bin(31), /* Structure length */
5 Version fixed bin(31), /* Structure version number */
5 Command fixed bin(31), /* Command identifier */
5 MsgseqNumber fixed bin(31), /* Message sequence number */
5 Control fixed bin(31), /* Control options */
5 CompCode fixed bin(31), /* Completion code */
5 Reason fixed bin(31), /* Reason code qualifying completion code */
5 ParameterCount fixed bin(31); /* Count of parameter structures */
```

## MQEPH의 상위 레벨 어셈블러 선언

```
MQEPH DSECT
MQEPH_STRUCID DS CL4 Structure identifier
MQEPH_VERSION DS F Structure version number
```

MQEPH_STRUCLength	DS	F	Total length of MQEPH including the MQCFH and parameter structures that follow it
MQEPH_ENCODING	DS	F	Numeric encoding of data that follows last PCF parameter structure
MQEPH_CODEDCHARSETID	DS	F	Character set identifier of data that follows last PCF parameter structure
MQEPH_FORMAT	DS	CL8	Format name of data that follows last PCF parameter structure
MQEPH_FLAGS	DS	F	Flags
MQEPH_PCFHEADER	DS	0F	Force fullword alignment
MQEPH_PCFHEADER_TYPE	DS	F	Structure type
MQEPH_PCFHEADER_STRUCLength	DS	F	Structure length
MQEPH_PCFHEADER_VERSION	DS	F	Structure version number
MQEPH_PCFHEADER_COMMAND	DS	F	Command identifier
MQEPH_PCFHEADER_MSGSEQNUMBER	DS	F	Structure length
MQEPH_PCFHEADER_CONTROL	DS	F	Control options
MQEPH_PCFHEADER_COMPCODE	DS	F	Completion code
MQEPH_PCFHEADER_REASON	DS	F	Reason code qualifying completion code
MQEPH_PCFHEADER_PARAMETER COUNT	DS	F	Count of parameter structures
MQEPH_PCFHEADER_LENGTH	EQU	*-MQEPH_PCFHEADER	
	ORG	MQEPH_PCFHEADER	
MQEPH_PCFHEADER_AREA	DS	CL(MQEPH_PCFHEADER_LENGTH)	
MQEPH_LENGTH	EQU	*-MQEPH	
	ORG	MQEPH	
MQEPH_AREA	DS	CL(MQEPH_LENGTH)	

## MQEPH의 Visual Basic 선언

```

Type MQEPH
  StrucId      As String*4 'Structure identifier'
  Version     As Long      'Structure version number'
  StrucLength As Long      'Total length of MQEPH structure including the MQCFH'
                'and parameter structures that follow it'
  Encoding    As Long      'Numeric encoding of data that follows last'
                'PCF parameter structure'
  CodedCharSetId As Long   'Character set identifier of data that'
                'follows last PCF parameter structure'
  Format      As String*8  'Format name of data that follows last PCF'
                'parameter structure'
  Flags      As Long      'Flags'
  PCFHeader  As MQCFH     'Programmable command format header'
End Type

Global MQEPH_DEFAULT As MQEPH

```

### MQEPH용 StrucId (MQCHAR4)

임베드된 PCF 헤더 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQEPH\_STRUC\_ID입니다. 값은 다음과 같아야 합니다.

#### MQEPH\_STRUC\_ID

임베드된 PCF 헤더 구조 ID.

C 프로그래밍 언어의 경우 상수 MQEPH\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQEPH\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### MQEPH용 버전 (MQLONG)

값은 다음과 같아야 합니다.

#### MQEPH\_VERSION\_1

임베드된 PCF 헤더 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### MQCFH\_VERSION\_3

임베드된 PCF 헤더 구조의 현재 버전.

이 필드의 초기값은 MQEPH\_VERSION\_1입니다.

### MQEPH용 StrucLength (MQLONG)

이는 다음 헤더 구조 앞에 오는 데이터의 양입니다. 여기에는 다음이 포함됩니다.

- MQEPH 헤더의 길이
- 헤더 다음에 오는 모든 PCF 매개변수의 길이
- 해당 매개변수 다음에 오는 공백 채우기

StrucLength는 4의 배수여야 합니다.

구조의 고정된 길이 부분은 MQEPH\_STRUC\_LENGTH\_FIXED에 의해 정의됩니다.

이 필드의 초기값은 68입니다.

### **MQEPH용 인코딩 (MQLONG)**

MQEPH 구조 및 연관된 PCF 매개변수 다음에 오는 데이터의 숫자 인코딩입니다. MQEPH 구조 자체에 있는 문자 데이터에는 적용되지 않습니다.

이 필드의 초기값은 0입니다.

### **MQEPH용 CodedCharSetId (MQLONG)**

이는 MQEPH 구조 뒤에 오는 데이터의 문자 세트 ID 및 연관된 PCF 매개변수입니다. MQEPH 구조 자체의 문자 데이터에는 적용되지 않습니다.

이 필드의 초기값은 MQCCSI\_UNDEFINED입니다.

### **MQEPH의 형식 (MQCHAR8)**

이는 MQEPH 구조 및 연관된 PCF 매개변수 다음에 오는 데이터의 형식 이름입니다.

이 필드의 초기값은 MQFMT\_NONE입니다.

### **MQEPH의 플래그 (MQLONG)**

다음 값을 사용할 수 있습니다.

#### **MQEPH\_NONE**

플래그가 지정되지 않았습니니다. MQEPH\_NONE은 프로그램 문서화를 지원하기 위해 정의됩니다. 이 상수는 다른 상수와 함께 사용하기 위한 용도는 아니지만, 값이 0이므로 그러한 사용을 감지할 수 없습니다.

#### **MQEPH\_CC SID\_EMBEDDED**

문자 데이터가 포함된 매개변수의 문자 세트는 각 구조의 CodedCharSetId 필드 내에서 개별적으로 지정됩니다. StrucId 및 Format 필드의 문자 세트는 MQEPH 구조 앞에 있는 헤더 구조의 CodedCharSetId 필드에 의해 정의되거나 MQEPH가 메시지의 시작 부분에 있는 경우 MQMD에 있는 CodedCharSetId 필드에 의해 정의됩니다.

이 필드의 초기값은 MQEPH\_NONE입니다.

### **MQEPH용 PCFHeader (MQCFH)**

이는 MQEPH 구조 다음에 오는 PCF 매개변수를 정의하는 PCF(Programmable Command Format) 헤더입니다. 이로 인해 다른 헤더가 있는 PCF 메시지 데이터를 따를 수 있습니다.

PCF 헤더는 처음에 다음 값으로 정의됩니다.

표 489. MQCFH의 필드		
필드 이름	상수의 이름	상수의 값
Type	MQCFT_NONE	0
StrucLength	MQCFH_STRUC_LENGTH	36
Version	MQCFH_VERSION_3	3
StrucLength	없음	0

표 489. MQCFH의 필드 (계속)		
필드 이름	상수의 이름	상수의 값
Command	MQCMD_NONE	0
MsgSeqNumber	없음	1
Control	MQCFC_LAST	1
CompCode	MQCC_OK	0
Reason	MQRC_NONE	0
ParameterCount	없음	0

애플리케이션은 Type을 MQCFT\_NONE에서 임베드된 PCF 헤더로 구성된 사용에 올바른 구조 유형으로 변경해야 합니다.

## MQGMO - 메시지 가져오기 옵션

MQGMO 구조를 사용하면 애플리케이션이 메시지가 큐에서 제거되는 방법을 제어할 수 있습니다. 구조는 MQGET 호출의 입출력(I/O) 매개변수입니다.

### 버전

MQGMO의 현재 버전은 MQGMO\_VERSION\_4입니다. 특정 필드는 MQGMO의 특정 버전에서만 사용 가능합니다. 여러 환경 간에 애플리케이션을 이식해야 하는 경우 MQGMO의 버전이 모든 환경에서 지속적인지 확인해야 합니다. 특정 버전의 구조에만 있는 필드는 362 페이지의 『MQGMO - 메시지 가져오기 옵션』 및 필드 설명서와 같이 식별됩니다.

지원되는 프로그래밍 언어에 제공되는 헤더, COPY 및 INCLUDE 파일에는 환경에서 지원되지만 *Version* 필드의 초기값이 MQGMO\_VERSION\_1로 설정된 MQGMO의 최신 버전이 포함되어 있습니다. 버전-1 구조에 있지 않은 필드를 사용하려면 *Version* 필드를 필요한 버전의 버전 번호로 설정하십시오.

### 문자 세트 및 인코딩

MQGMO의 데이터는 MQENC\_NATIVE로 지정된 로컬 큐 관리자의 인코딩 및 **CodedCharSetId** 큐 관리자 속성으로 지정된 문자 세트에 있어야 합니다. 그러나 애플리케이션이 MQ MQI 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트와 인코딩으로 작성되어야 합니다.

### 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 490. MQGMO에 대한 MQGMO의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQGMO_STRUC_ID	'GMO~'
버전 (구조 버전 번호)	MQGMO_VERSION_1	1
MQGMO-Options 필드 (MQGET의 조치를 제어하는 옵션)	MQGMO_NO_WAIT	0
WaitInterval (대기 간격)	없음	0
Signal1 (신호)	없음	z/OS의 널 포인터. 그 외의 경우 0
Signal2 (신호 ID)	없음	0
ResolvedQName (목적지 큐의 해석된 이름)	없음	널 문자열 또는 공백

표 490. MQGMO에 대한 MQGMO의 필드 (계속)		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
<b>참고:</b> <i>Version</i> 이 MQGMO_VERSION_2 미만인 경우 나머지 필드는 무시됩니다.		
MatchOptions (MQGET에 사용되는 선택 기준을 제어하는 옵션)	MQMO_MATCH_MSG_ID + MQMO_MATCH_CORREL_ID	3
GroupStatus (검색된 메시지가 그룹에 있는지 여부를 표시하는 플래그)	MQGS_NOT_IN_GROUP	'-'
SegmentStatus (검색된 메시지가 논리 메시지의 세그먼트인지 여부를 표시하는 플래그)	MQSS_NOT_A_SEGMENT	'-'
세그먼트화 (검색된 메시지에 추가 세그먼트화가 허용되는지 여부를 표시하는 플래그)	MQSEG_INHIBITED	'-'
Reserved1 (예약됨)	없음	'-'
<b>참고:</b> <i>Version</i> 이 MQGMO_VERSION_3 미만인 경우 나머지 필드는 무시됩니다.		
MsgToken (메시지 토큰)	MQMTOK_NONE	널
ReturnedLength (리턴된 메시지 데이터의 길이 (바이트))	MQRL_UNDEFINED	-1
<b>참고:</b> <i>Version</i> 이 MQGMO_VERSION_4 미만인 경우 나머지 필드는 무시됩니다.		
Reserved2 (예약됨)	없음	'-'
MsgHandle (큐에서 검색되는 메시지의 특성으로 채워질 메시지에 대한 핸들)	MQHM_NONE	0
<b>참고:</b> <ol style="list-style-type: none"> <li>- 기호는 단일 공백 문자를 나타냅니다.</li> <li>널 문자열 값 또는 공백은 C의 널 문자열과 기타 프로그래밍 언어의 공백 문자를 나타냅니다.</li> <li>C 프로그래밍 언어의 매크로 변수MQGMO_DEFAULT는 테이블에 나열되는 값을 포함합니다. 즉, 구조 내의 필드에 대한 초기값을 제공하기 위해 다음과 같은 방법으로 사용될 수 있습니다.</li> </ol> <pre>MQGMO MyGMO = {MQGMO_DEFAULT};</pre>		

## 언어 선언

### MQGMO의 C 선언

```
typedef struct tagMQGMO MQGMO;
struct tagMQGMO {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    Options;          /* Options that control the action of */
                                /* MQGET */
    MQLONG    WaitInterval;     /* Wait interval */
    MQLONG    Signal1;          /* Signal */
    MQLONG    Signal2;          /* Signal identifier */
    MQCHAR48  ResolvedQName;    /* Resolved name of destination queue */
    /* Ver:1 */
    MQLONG    MatchOptions;     /* Options controlling selection */
                                /* criteria used for MQGET */
    MQCHAR    GroupStatus;      /* Flag indicating whether message */
}
```

```

MQCHAR    SegmentStatus;    /* Flag indicating whether message */
/* Ver:2 */
MQCHAR    Segmentation;    /* Flag indicating whether further */
/* Ver:3 */
MQCHAR    Reserved1;        /* Reserved */
MQBYTE16  MsgToken;         /* Message token */
MQLONG    ReturnedLength;   /* Length of message data returned */
/* Ver:4 */
MQLONG    Reserved2;        /* Reserved */
MQHMSG    MsgHandle;        /* Message handle */
};

```

참고: z/OS에서 *Signal1* 필드는 *PMQLONG*로 선언됩니다.

### MQGMO의 COBOL 선언

```

** MQGMO structure
10 MQGMO.
** Structure identifier
15 MQGMO-STRUCID PIC X(4).
** Structure version number
15 MQGMO-VERSION PIC S9(9) BINARY.
** Options that control the action of MQGET
15 MQGMO-OPTIONS PIC S9(9) BINARY.
** Wait interval
15 MQGMO-WAITINTERVAL PIC S9(9) BINARY.
** Signal
15 MQGMO-SIGNAL1 PIC S9(9) BINARY.
** Signal identifier
15 MQGMO-SIGNAL2 PIC S9(9) BINARY.
** Resolved name of destination queue
15 MQGMO-RESOLVEDQNAME PIC X(48).
** Options controlling selection criteria used for MQGET
15 MQGMO-MATCHOPTIONS PIC S9(9) BINARY.
** Flag indicating whether message retrieved is in a group
15 MQGMO-GROUPSTATUS PIC X.
** Flag indicating whether message retrieved is a segment of a
** logical message
15 MQGMO-SEGMENTSTATUS PIC X.
** Flag indicating whether further segmentation is allowed for the
** message retrieved
15 MQGMO-SEGMENTATION PIC X.
** Reserved
15 MQGMO-RESERVED1 PIC X.
** Message token
15 MQGMO-MSGTOKEN PIC X(16).
** Length of message data returned (bytes)
15 MQGMO-RETURNEDLENGTH PIC S9(9) BINARY.
** Reserved
15 MQGMO-RESERVED2 PIC S9(9) BINARY.
** Message handle
15 MQGMO-MSGHANDLE PIC S9(18) BINARY.

```

참고: z/OS에서 *Signal1* 필드는 *POINTER*로 선언됩니다.

### MQGMO의 PL/I 선언

```

dcl
1 MQGMO based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 Options fixed bin(31), /* Options that control the action of
MQGET */
3 WaitInterval fixed bin(31), /* Wait interval */
3 Signal1 fixed bin(31), /* Signal */
3 Signal2 fixed bin(31), /* Signal identifier */
3 ResolvedQName char(48), /* Resolved name of destination
queue */
3 MatchOptions fixed bin(31), /* Options controlling selection
criteria used for MQGET */
3 GroupStatus char(1), /* Flag indicating whether message

```



```

3 SegmentStatus char(1), /* Flag indicating whether message
retrieved is a segment of a logical
message */
3 Segmentation char(1), /* Flag indicating whether further
segmentation is allowed for the
message retrieved */
3 Reserved1 char(1), /* Reserved */
3 MsgToken char(16), /* Message token */
3 ReturnedLength fixed bin(31); /* Length of message data returned
(bytes) */
3 Reserved2 fixed bin(31); /* Reserved */
3 MsgHandle fixed bin(63); /* Message handle */

```

**참고:** z/OS에서 *Signal1* 필드는 pointer로 선언됩니다.

MQGMO의 상위 레벨 어셈블러 선언

```

MQGMO          DSECT
MQGMO_STRUCID  DS CL4  Structure identifier
MQGMO_VERSION  DS F    Structure version number
MQGMO_OPTIONS  DS F    Options that control the action of
*              MQGET
MQGMO_WAITINTERVAL DS F    Wait interval
MQGMO_SIGNAL1  DS F    Signal
MQGMO_SIGNAL2  DS F    Signal identifier
MQGMO_RESOLVEDQNAME DS CL48 Resolved name of destination queue
MQGMO_MATCHOPTIONS DS F    Options controlling selection criteria
*              used for MQGET
MQGMO_GROUPSTATUS DS CL1  Flag indicating whether message
*              retrieved is in a group
MQGMO_SEGMENTSTATUS DS CL1  Flag indicating whether message
*              retrieved is a segment of a logical
*              message
MQGMO_SEGMENTATION DS CL1  Flag indicating whether further
*              segmentation is allowed for the message
*              retrieved
MQGMO_RESERVED1 DS CL1  Reserved
MQGMO_MSGTOKEN DS XL16  Message token
MQGMO_RETURNEDLENGTH DS F    Length of message data returned (bytes)
MQGMO_RESERVED2 DS F    Reserved
MQGMO_MSGHANDLE DS D    Message handle
MQGMO_LENGTH   EQU *-MQGMO
                ORG MQGMO
MQGMO_AREA     DS CL(MQGMO_LENGTH)

```

MQGMO의 상위 레벨 어셈블러 선언

```

Type MQGMO
  StrucId      As String*4  'Structure identifier'
  Version      As Long      'Structure version number'
  Options      As Long      'Options that control the action of MQGET'
  WaitInterval As Long      'Wait interval'
  Signal1      As Long      'Signal'
  Signal2      As Long      'Signal identifier'
  ResolvedQName As String*48 'Resolved name of destination queue'
  MatchOptions As Long      'Options controlling selection criteria'
  *            'used for MQGET'
  GroupStatus  As String*1  'Flag indicating whether message'
  *            'retrieved is in a group'
  SegmentStatus As String*1 'Flag indicating whether message'
  *            'retrieved is a segment of a logical'
  *            'message'
  Segmentation As String*1  'Flag indicating whether further'
  *            'segmentation is allowed for the message'
  *            'retrieved'
  Reserved1    As String*1  'Reserved'
  MsgToken     As MBYTE16   'Message token'
  ReturnedLength As Long    'Length of message data returned (bytes)'
End Type

```

## PROPCTL 채널 옵션 MQGMO

PROPCTL 채널 속성을 사용하여 IBM MQ 9.4 큐 관리자에서 IBM MQ의 이전 버전의 파트너 큐 관리자로 송신되는 메시지에 포함되는 메시지 특성을 제어할 수 있습니다.

표 491. 채널 메시지 특성 속성 설정	
PROPCTL	설명
all	<p>이전 버전에서 파트너 큐 관리자에 연결된 애플리케이션이 IBM MQ 9.4 애플리케이션에 의해 메시지에 놓인 특성을 처리할 수 있는 경우에는 이 옵션을 사용하십시오.</p> <p>MQRFH2에 놓인 임의의 이름-값 쌍과 함께, 모든 특성이 파트너 큐 관리자에 송신됩니다.</p> <p>두 가지 애플리케이션 디자인 문제를 고려해야 합니다.</p> <ol style="list-style-type: none"> <li>파트너 큐 관리자에 연결된 애플리케이션은 IBM MQ 9.4 큐 관리자에서 생성된 MQRFH2 헤더를 포함하는 메시지를 처리할 수 있어야 합니다.</li> <li>파트너 큐 관리자에 연결된 애플리케이션은 올바르게 MQPD_SUPPORT_REQUIRED로 플래그 지정된 새 메시지 특성을 처리해야 합니다.</li> </ol> <p>ALL 채널 옵션이 설정되면 JMS 애플리케이션은 채널을 사용하여 IBM MQ 9.4 및 이전 버전 사이에 상호 운용할 수 있습니다. 메시지 특성을 사용하는 새 IBM MQ 9.4 애플리케이션은 이전 버전의 애플리케이션이 MQRFH2 헤더를 핸들링하는 방법에 따라 이전 버전의 애플리케이션과 상호 운영될 수 있습니다.</p>
COMPAT	<p>이 옵션을 사용하여 이전 버전의 파트너 큐 관리자에 연결된 애플리케이션으로 메시지 특성을 송신할 수 있는 경우가 있지만 아닌 경우도 있습니다. 다음 두 조건을 충족하는 경우에만 메시지 특성이 송신됩니다.</p> <ol style="list-style-type: none"> <li>필수 메시지 특성 처리로 표시된 특성이 없어야 합니다.</li> <li>하나 이상의 메시지 특성이 "예약된" 폴더에 있어야 합니다. <a href="#">참고</a>를 참조하십시오.</li> </ol> <p>COMPAT 채널 옵션이 설정되면 JMS 애플리케이션은 채널을 사용하여 IBM MQ 9.4과 이전 버전 사이에 상호 운용할 수 있습니다.</p> <p>모든 애플리케이션이 메시지 특성을 사용하여 채널을 사용할 수 있는 것은 아닙니다. 예약된 폴더를 사용하는 애플리케이션만 가능합니다. 메시지 또는 특성이 송신되는지 여부와 관련된 규칙은 다음과 같습니다.</p> <ol style="list-style-type: none"> <li>메시지에 특성이 있지만 "예약된" 폴더와 연관된 특성이 없는 경우에는 메시지 특성이 송신되지 않습니다.</li> <li>메시지 특성이 "예약된" 특성 폴더에서 작성된 경우, 해당 메시지와 연관된 모든 메시지 특성이 송신됩니다. 그러나,             <ol style="list-style-type: none"> <li>지원이 필요한 것으로 표시되는 메시지 특성 MQPD_SUPPORT_REQUIRED 또는 MQPD_SUPPORT_REQUIRED_IF_LOCAL이 있는 경우 전체 메시지가 거부됩니다. 이는 보고서 옵션의 값에 따라 데드-레터 큐로 리턴되거나 제거되거나 송신됩니다.</li> <li>지원이 필요한 것으로 표시되는 메시지 특성이 없는 경우 개별 특성이 송신되지 않을 수 있습니다. 임의의 메시지 특성 디스크립터 필드가 기본값이 아닌 값으로 설정된 경우, 개별 특성이 송신되지 않습니다. 메시지는 여전히 송신됩니다. 기본값이 아닌 특성 디스크립터 필드 값의 예제는 MQPD_USER_CONTEXT입니다.</li> </ol> </li> </ol> <p><b>참고:</b> "예약된" 폴더 이름은 mcd., jms., usr. 또는 mqext.로 시작합니다. 이러한 폴더는 JMS 인터페이스를 사용하는 애플리케이션에 대해 작성됩니다. IBM MQ 9.4에서 이러한 폴더에 배치되는 모든 이름-값 쌍은 메시지 특성으로 처리됩니다.</p> <p>메시지 특성은 MQRFH2 헤더에 배치된 모든 이름-값 쌍과 함께 MQRFH2 헤더에서 송신됩니다. MQRFH2 헤더에 배치된 모든 이름-값 쌍은 메시지가 거부되지 않는 한 송신됩니다.</p>

표 491. 채널 메시지 특성 속성 설정 (계속)	
PROPCTL	설명
NONE	<p>이 옵션을 사용하면 이전 버전의 파트너 큐 관리자에 연결된 애플리케이션으로 메시지 특성이 송신되지 않도록 방지됩니다. 이름-값 쌍 및 메시지 특성을 포함하는 MQRFH2는 여전히 송신되지만 이름-값 쌍으로만 송신됩니다.</p> <p>NONE 채널 옵션을 설정하면 JMS 메시지가 JMS 메시지 특성 없이 JMSTextMessage 또는 JMSBytesMessage 로 송신됩니다. 이전 버전의 애플리케이션이 IBM MQ 9.4 애플리케이션의 특성 세트를 모두 무시할 수 있는 경우, 이 두 애플리케이션은 상호 운용할 수 있습니다.</p>

## MQGMO 에 대한 PROPCTL 큐 옵션

PROPCTL 큐 속성을 사용하여 MQGMO 메시지 특성 옵션을 설정하지 않고 MQGET를 호출하는 애플리케이션으로 메시지 특성을 리턴하는 방법을 제어합니다.

표 492. 큐 메시지 특성 속성 설정	
PROPCTL	설명
all	<p>동일한 큐에서 메시지를 읽는 다른 애플리케이션이 다른 방법으로 메시지를 처리할 수 있도록 ALL 옵션을 사용하십시오.</p> <ul style="list-style-type: none"> <li>이전 버전에서 변경되지 않고 마이그레이션되는 애플리케이션은 계속해서 MQRFH2를 직접 읽을 수 있습니다. 특성은 MQRFH2 헤더에서 직접 액세스할 수 있습니다.</li> </ul> <p>새 특성 및 새 특성 속성을 핸들링하려면 애플리케이션을 수정해야 합니다. MQRFH2 헤더의 레이아웃 및 수가 변경됨으로써 애플리케이션이 영향을 받을 수 있습니다. 일부 폴더 속성은 제거될 수 있습니다. 또는 해당 IBM MQ가 이전 버전에서 무시한 MQRFH2 헤더의 레이아웃에서 오류를 보고합니다.</p> <ul style="list-style-type: none"> <li>새로운 또는 변경된 애플리케이션은 메시지 특성 MQI를 사용하여 메시지 특성을 조회할 수 있으며 MQRFH2 헤더에서 직접 이름-값 쌍을 읽을 수 있습니다.</li> </ul> <p>메시지의 모든 특성은 애플리케이션에 리턴됩니다.</p> <ul style="list-style-type: none"> <li>애플리케이션이 MQCRTMH를 호출하여 메시지 핸들을 작성하는 경우에는 MQINQMP를 사용하여 메시지 특성을 조회해야 합니다. 메시지 특성이 아닌 이름-값 쌍은 메시지 특성이 제거된 MQRFH2에 남아 있습니다.</li> <li>애플리케이션이 메시지 핸들을 작성하지 않는 경우, 모든 메시지 특성 및 이름-값 쌍은 MQRFH2에 남아 있습니다.</li> </ul> <p>ALL 은 수신 애플리케이션이 MQGMO_PROPERTIES 옵션을 설정하지 않았거나 이를 MQGMO_PROPERTIES_AS_Q_DEF로 설정한 경우에만 적용됩니다.</p>

표 492. 큐 메시지 특성 속성 설정 (계속)

PROPCTL	설명
COMPAT (기본값)	<p>COMPAT는 기본 옵션입니다. GMO_PROPERTIES_*가 설정되지 않은 경우, 이전 버전에서 수정되지 않은 애플리케이션에서와 같이 COMPAT으로 간주됩니다. COMPAT 옵션을 기본값으로 설정하면 명시적으로 MQRFH2를 작성하지 않은 이전 버전 애플리케이션은 IBM MQ 9.4에서 변경 사항 없이 작동합니다.</p> <p>JMS 메시지를 읽도록 이전 버전 애플리케이션 MQI 애플리케이션을 작성하는 경우 이 옵션을 사용합니다.</p> <ul style="list-style-type: none"> <li>MQRFH2 헤더에 저장된 JMS 특성은 이름이 mcd., jms., usr. 또는 mqext로 시작하는 폴더의 MQRFH2 헤더에 있는 애플리케이션으로 리턴됩니다.</li> <li>메시지에 JMS 폴더가 있는 경우 및 IBM MQ 9.4 애플리케이션이 메시지에 새 특성 폴더를 추가하는 경우, MQRFH2에서 이 특성도 리턴됩니다. 따라서 새 특성 및 새 특성 속성을 핸들링하도록 애플리케이션을 수정해야 합니다. 수정되지 않은 애플리케이션은 MQRFH2 헤더의 레이아웃 또는 수가 변경될 때 영향을 받을 수 있습니다. 일부 폴더 속성이 제거되었음을 찾을 수 있습니다. 또는 해당 IBM MQ가 이전 버전에서 무시한 MQRFH2 헤더의 레이아웃에서 오류를 찾습니다.</li> </ul> <p><b>참고:</b> 이 시나리오에서 애플리케이션의 작동은 이전 버전 또는 IBM MQ 9.4 큐 관리자에 연결되었는지에 상관없이 동일합니다. 채널 PROPCTL 속성이 COMPAT 또는 ALL로 설정된 경우, 새 메시지 특성은 이전 버전 파트너 큐 관리자로 메시지에서 전송됩니다.</p> <ul style="list-style-type: none"> <li>메시지가 JMS 메시지가 아니지만 기타 특성을 포함하는 경우, 해당 특성은 MQRFH2 헤더에서 애플리케이션으로 리턴되지 않습니다.<sup>1</sup></li> <li>또한 이 옵션은 많은 경우에 명시적으로 MQRFH2를 작성하는 이전 버전 애플리케이션이 올바르게 작동하도록 합니다. 예를 들어, JMS 메시지 특성을 포함하는 MQRFH2 를 작성하는 MQI 프로그램은 계속해서 올바르게 작동합니다. 메시지가 JMS 메시지 특성 없이 작성되었지만 일부 기타 MQRFH2 폴더가 포함된 경우에는 해당 폴더가 애플리케이션으로 리턴됩니다. 폴더가 메시지 특성 폴더인 경우에만 해당 특정 폴더가 MQRFH2에서 제거됩니다. 메시지 특성 폴더는 새 폴더 속성 content='properties'를 사용하여 식별되거나 이름이 정의된 특성 폴더 이름 또는 그룹화되지 않은 특성 폴더 이름에 나열된 폴더입니다.</li> <li>애플리케이션이 MQCRTMH를 호출하여 메시지 핸들을 작성하는 경우에는 MQINQMP를 사용하여 메시지 특성을 조회해야 합니다. 메시지 특성이 MQRFH2 헤더에서 제거됩니다. 메시지 특성이 아닌 이름-값 쌍은 MQRFH2에 남아 있습니다.</li> <li>애플리케이션이 MQCRTMH를 호출하여 메시지 핸들을 작성하는 경우에는 메시지에 JMS 폴더가 있는지 여부에 관계없이 모든 메시지 특성을 조회할 수 있습니다.</li> <li>애플리케이션이 메시지 핸들을 작성하지 않는 경우, 모든 메시지 특성 및 이름-값 쌍은 MQRFH2에 남아 있습니다.</li> </ul> <p>메시지에 새 사용자 특성 폴더가 포함되는 경우에는 신규 또는 변경된 IBM MQ 9.4 애플리케이션에 의해 메시지가 작성되었음을 추론할 수 있습니다. 수신 애플리케이션이 MQRFH2에서 이러한 새 특성을 직접 처리하려는 경우, ALL 옵션을 사용하도록 애플리케이션을 수정해야 합니다. 기본 COMPAT 옵션을 설정하면 수정되지 않은 애플리케이션이 IBM MQ 9.4 특성 없이 MQRFH2의 나머지 부분을 계속 처리합니다.</p> <p>PROPCTL 인터페이스의 목적은 MQRFH2 폴더를 읽는 이전 애플리케이션과 메시지 특성 인터페이스를 사용하는 새 애플리케이션 및 변경된 애플리케이션을 지원하는 것입니다. 새 애플리케이션이 모든 사용자 메시지 특성에 대해 메시지 특성 인터페이스를 사용하고 MQRFH2 헤더를 직접 읽고 쓰지 않는 것이 목표입니다.</p> <p>COMPAT 는 수신 애플리케이션이 MQGMO_PROPERTIES 옵션을 설정하지 않았거나 이 옵션을 MQGMO_PROPERTIES_AS_Q_DEF로 설정한 경우에만 적용됩니다.</p>

표 492. 큐 메시지 특성 속성 설정 (계속)

PROPCTL	설명
FORCE	<p>FORCE 옵션은 모든 메시지 특성을 MQRFH2 헤더에 넣습니다. MQRFH2 헤더의 모든 메시지 특성과 이름-값 쌍은 메시지에 남아 있습니다. 메시지 특성은 MQRFH2에서 제거되지 않으며 메시지 핸들을 통해 사용 가능합니다. FORCE 옵션을 선택하면 새로 마이그레이션한 애플리케이션이 MQRFH2 헤더에서 메시지 특성을 읽을 수 있습니다.</p> <p>IBM MQ 9.4 메시지 특성을 처리하도록 애플리케이션을 수정했지만, 이전과 같이 MQRFH2 헤더를 직접 작업하는 기능도 보유한다고 가정합니다. 초기에 PROPCTL 큐 속성을 FORCE로 설정하여 메시지 특성을 사용하도록 애플리케이션을 전환할 시기를 결정할 수 있습니다. 메시지 특성 사용을 시작할 준비가 되면 <b>PROPCTL</b> 큐 속성을 다른 값으로 설정하십시오. 애플리케이션에서 새 함수가 예상대로 동작하지 않으면 <b>PROPCTL</b> 옵션을 다시 FORCE로 설정하십시오.</p> <p>FORCE 는 수신 애플리케이션이 MQGMO_PROPERTIES 옵션을 설정하지 않았거나 MQGMO_PROPERTIES_AS_Q_DEF로 설정한 경우에만 이 영향을 미칩니다.</p>
NONE	<p>기존 애플리케이션이 모든 메시지 특성을 무시하고 메시지를 처리할 수 있고 새 애플리케이션 또는 변경된 애플리케이션이 메시지 특성을 조회할 수 있도록 하려면 NONE 옵션을 사용하십시오.</p> <ul style="list-style-type: none"> <li>• 애플리케이션이 MQCRTMH를 호출하여 메시지 핸들을 작성하는 경우에는 MQINQMP를 사용하여 메시지 특성을 조회해야 합니다. 메시지 특성이 아닌 이름-값 쌍은 메시지 특성이 제거된 MQRFH2에 남아 있습니다.</li> <li>• 애플리케이션이 메시지 핸들을 작성하지 않는 경우, 모든 메시지 특성이 MQRFH2에서 제거됩니다. MQRFH2 헤더의 이름-값 쌍은 메시지에 남아 있습니다.</li> </ul> <p>NONE 은 수신 애플리케이션이 MQGMO_PROPERTIES 옵션을 설정하지 않았거나 MQGMO_PROPERTIES_AS_Q_DEF로 설정한 경우에만 이 영향을 미칩니다.</p>
V6COMPAT	<p>이 옵션을 사용하여 송신된 것과 동일한 형식으로 MQRFH2를 수신합니다. 송신 애플리케이션 또는 큐 관리자가 추가 메시지 특성을 작성하는 경우, 이는 메시지 핸들에서 리턴됩니다.</p> <p>이 옵션은 송신 및 수신 큐와 중간 전송 큐 모두에서 설정되어야 합니다. 이는 큐 이름 해석 경로의 큐 정의에서 설정된 다른 PROPCTL 옵션을 대체합니다.</p> <p>예외적인 상황에서만 V6COMPAT 옵션을 사용하십시오. 예를 들어, 이전 버전에서 IBM MQ 9.4로 애플리케이션을 마이그레이션하는 경우 이 옵션은 이전 버전의 동작을 유지하므로 유용합니다. 이 옵션은 메시지 처리량에 영향을 미칠 수 있습니다. 또한 관리하기가 더 어렵습니다. 이 옵션이 송신자, 수신자 및 중간 전송 큐에 설정되었는지 확인해야 합니다.</p> <p>V6COMPAT 는 수신 애플리케이션이 MQGMO_PROPERTIES 옵션을 설정하지 않았거나 MQGMO_PROPERTIES_AS_Q_DEF로 설정한 경우에만 적용됩니다.</p>

메시지 특성 및 이름-값 쌍에 대한 자세한 정보는 517 페이지의 『NameValue MQRFH2의 데이터 (MQCHARn)』의 내용을 참조하십시오.

### MQGMO의 메시지 특성 옵션

메시지 특성이 애플리케이션에 리턴되는 방법을 제어하려면 **MQGMO** 메시지 특성 옵션을 사용하십시오.

<sup>1</sup> IBM MQ classes for JMS에서 작성된 특정 특성 폴더가 있으면 JMS 메시지를 나타냅니다. 특성 폴더는 mcd., jms., usr. 또는 mqext입니다.

표 493. MQGMO 메시지 특성 옵션 설정

MQGMO 옵션	설명
MQGMO_PROPERTIES_AS_Q_DEF	<p>동일한 큐에서 읽고 GMO_PROPERTIES_*를 설정하지 않은 IBM MQ 애플리케이션은 메시지 특성을 다르게 수신합니다. 메시지 핸들을 작성하지 않는 IBM MQ 애플리케이션은 큐 <b>PROPCTL</b> 속성으로 제어됩니다. IBM MQ 애플리케이션은 MQRFH2에서 메시지 특성을 수신하거나 메시지 핸들을 작성하고 메시지 특성을 조회하도록 선택할 수 있습니다. 애플리케이션이 메시지 핸들을 작성하면 특성은 MQRFH2에서 제거됩니다.</p> <ul style="list-style-type: none"> <li>• GMO_PROPERTIES_* 를 설정하지 않거나 MQGMO_PROPERTIES_AS_Q_DEF 로 설정하는 신규 또는 변경된 IBM MQ 애플리케이션은 메시지 특성을 조회하도록 선택할 수 있습니다. MQINQMP MQI 호출을 사용하여 메시지 핸들을 작성하고 메시지 특성을 조회하려면 MQCRTMH를 설정해야 합니다.</li> <li>• 신규 또는 변경된 애플리케이션이 메시지 핸들을 작성하지 않는 경우 MQRFH2 헤더에서 직접 수신하는 모든 메시지 특성을 읽어야 합니다.</li> <li>• 큐 속성 <b>PROPCTL</b>이 FORCE로 설정되면 메시지 핸들에서 특성이 리턴되지 않습니다. 모든 특성은 MQRFH2 헤더에서 리턴됩니다.</li> <li>• 큐 속성 <b>PROPCTL</b>이 NONE 또는 COMPAT로 설정된 경우, 메시지 핸들을 작성하는 IBM MQ 애플리케이션은 모든 메시지 특성을 수신합니다.</li> </ul>
MQGMO_PROPERTIES_IN_HANDLE	<p>애플리케이션이 메시지 특성을 사용하도록 강제 실행합니다. 이 옵션을 사용하면 수정된 애플리케이션이 메시지 핸들을 작성하는데 실패하는지 여부를 감지할 수 있습니다. 애플리케이션이 MQINQMP를 호출하지 않고 MQRFH2에서 직접 메시지 특성을 읽으려고 시도 중일 수 있습니다.</p>
MQGMO_NO_PROPERTIES	<ul style="list-style-type: none"> <li>• 모든 특성이 제거됩니다. 큐 관리자가 생성한 특성(예: JMS 특성)은 제거됩니다.</li> <li>• 특성은 메시지 핸들이 작성된 경우에도 제거됩니다. 다른 MQRFH2 폴더에 있는 이름-값 쌍은 메시지 데이터에서 사용 가능합니다.</li> </ul>
MQGMO_PROPERTIES_FORCE_MQRFH2	<p>특성은 메시지 핸들이 작성된 경우에도 MQRFH2 헤더에서 리턴됩니다.</p> <ul style="list-style-type: none"> <li>• MQINQMP는 메시지 핸들이 작성된 경우에도 메시지 특성을 리턴하지 않습니다. 특성을 조회할 경우 MQRC_PROPERTY_NOT_AVAILABLE이 리턴됩니다.</li> </ul>

표 493. MQGMO 메시지 특성 옵션 설정 (계속)

MQGMO 옵션	설명
MQGMO_PROPERTIES_COMPATIBILITY	<p>JMS 클라이언트의 메시지인 경우 JMS 특성은 MQRFH2 헤더에서 리턴됩니다. 메시지 핸들을 작성하는 신규 또는 수정된 IBM MQ 애플리케이션은 다르게 작동합니다.</p> <ul style="list-style-type: none"> <li>• 메시지 특성 폴더의 모든 특성은 메시지에 mcd., jms., usr. 또는 mqext 폴더가 있는 경우 리턴됩니다.</li> <li>• 메시지에 특성 폴더가 있지만 mcd., jms., usr. 또는 mqext 폴더는 없는 경우에는 MQRFH2에서 메시지 특성이 리턴되지 않습니다.</li> <li>• 신규 또는 수정된 IBM MQ 애플리케이션이 메시지 핸들을 작성하는 경우 MQINQMP MQI 호출을 사용하여 메시지 특성을 조회합니다. 모든 메시지 특성은 MQRFH2에서 제거됩니다.</li> <li>• 신규 또는 수정된 IBM MQ 애플리케이션이 메시지 핸들을 작성하는 경우 메시지의 모든 특성을 조회할 수 있습니다. 메시지에 mcd., jms., usr. 또는 mqext 폴더가 없는 경우에도 모든 메시지 특성을 조회할 수 있습니다.</li> </ul>

**관련 참조**

PROPCTL

2471 (09A7) (RC2471): MQRC\_PROPERTY\_NOT\_AVAILABLE

**MQGMO의 경우 StrucId (MQCHAR4)**

메시지 가져오기 옵션 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQGMO\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

**MQGMO\_STRUC\_ID**

메시지 가져오기 옵션 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQGMO\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQGMO\_STRUC\_ID와 동일한 값을 갖지만 문자열이 아닌 문자의 배열입니다.

**MQGMO의 버전 (MQLONG)**

버전은 구조 버전 번호입니다.

값은 다음 중 하나여야 합니다.

**MQGMO\_VERSION\_1**

버전-1 메시지 가져오기 옵션 구조입니다.

이 버전은 모든 환경에서 지원됩니다.

**MQGMO\_VERSION\_2**

버전-2 메시지 가져오기 옵션 구조입니다.

이 버전은 모든 환경에서 지원됩니다.

**MQGMO\_VERSION\_3**

버전-3 메시지 가져오기 옵션 구조입니다.

이 버전은 모든 환경에서 지원됩니다.

**MQGMO\_VERSION\_4**

버전-4 메시지 가져오기 옵션 구조입니다.

이 버전은 모든 환경에서 지원됩니다.

구조의 최신 버전에만 있는 필드는 필드 설명에 그렇게 식별되어 있습니다. 다음 상수는 현재 버전의 버전 번호를 지정합니다.

**MQGMO\_CURRENT\_VERSION**

현재 버전의 가져오기 메시지 옵션 구조.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQGMO\_VERSION\_1입니다.

**MQGMO의 옵션(MQLONG)**

MQGMO 옵션은 MQGET의 조치를 제어합니다. 0개 이상의 옵션을 지정할 수 있습니다. 둘 이상의 선택적 값이 필요한 경우 다음을 수행하십시오.

- 값을 추가하십시오(동일한 상수를 두 번 이상 추가하지 않음). 또는
- 비트 단위의 OR 연산을 사용하여 값을 결합하십시오(프로그래밍 언어가 비트 조작을 지원하는 경우).

올바르지 않은 옵션의 결합이 설명되어 있습니다. 다른 모든 결합은 유효합니다.

**대기 옵션**

다음 옵션은 메시지가 큐에 도착하기까지 대기하는 것과 관련됩니다.

**MQGMO\_WAIT**

적당한 메시지가 도착할 때까지 애플리케이션이 대기합니다. 애플리케이션이 대기하는 최대 시간은 *WaitInterval*에 지정됩니다.

**중요사항:** 적당한 메시지를 즉시 사용할 수 있는 경우 대기 또는 지연이 없습니다.

대기 중에 MQGET 요청이 금지되거나 MQGET 요청이 금지되는 경우 대기가 취소됩니다. 호출은 큐에 적합한 메시지가 있는지 여부에 관계없이 MQCC\_FAILED 및 이유 코드 MQRC\_GET\_INHIBITED과(와) 함께 완료됩니다.

MQGMO\_WAIT 를 MQGMO\_BROWSE\_FIRST 또는 MQGMO\_BROWSE\_NEXT 옵션과 함께 사용할 수 있습니다.

여러 애플리케이션이 동일한 공유 큐에서 대기 중인 경우 다음 규칙은 적당한 메시지가 도착할 때 활성화되는 애플리케이션을 선택합니다.

표 494. 공유 큐에서 MQGET 호출을 활성화하기 위한 규칙		
활성화 대기 중인 MQGET 호출 수		결과
BROWSE 옵션 사용	BROWSE 옵션을 사용하지 않음 <sup>2</sup>	
없음	하나 이상	BROWSE 옵션을 사용하지 않는 하나의 MQGET 호출이 활성화됩니다.
하나 이상	없음	BROWSE 옵션을 사용하는 모든 MQGET 호출이 활성화됩니다.
하나 이상	하나 이상	BROWSE 옵션을 사용하지 않는 하나의 MQGET 호출이 활성화됩니다. BROWSE 옵션을 사용하는 MQGET 호출이 활성화되는 수를 예측할 수 없습니다.

BROWSE 옵션을 사용하지 않는 MQGET 호출이 동일한 큐에서 둘 이상 대기 중인 경우 한 개만 활성화됩니다. 큐 관리자는 대기 중인 호출에 대한 우선순위를 다음 순서로 지정하려고 시도합니다.

1. 특정 메시지(예: 특정 MsgId 또는 CorrelId(또는 둘 다)가 있는 메시지)에 의해서만 충족될 수 있는 특정 가져오기-대기 요청.
2. 어떤 메시지에 의해서도 충족될 수 있는 일반 가져오기-대기 요청

**참고:**

<sup>2</sup> MQGMO\_LOCK 옵션을 지정하는 MQGET 호출은 찾아보기가 아닌 호출로 처리됩니다.



- 첫 번째 카테고리에서는 더 특정한 가져오기-대기 요청에 추가 우선순위가 지정되지 않습니다. 예를 들어, `MsgId` 및 `CorrelId` 을(를) 둘 다 지정하는 요청입니다.
- 각 범주 내에서 어떤 애플리케이션이 선택되었는지 예측할 수 없습니다. 특히 가장 오래 대기 중인 애플리케이션이 반드시 선택되지는 않습니다.
- 경로 길이 및 운영 체제의 우선 순위 스케줄링 고려사항은 예상보다 하위 운영 체제 우선순위의 대기 중인 애플리케이션이 메시지를 검색함을 의미할 수 있습니다.
- 대기 중이지 않은 애플리케이션이 대기 중인 애플리케이션보다 우선하여 메시지를 검색하는 상황도 발생할 수 있습니다.

**z/OS** z/OS에서 다음 사항이 적용됩니다.

- 메시지가 도착하기를 기다리는 동안 애플리케이션이 다른 작업을 진행하도록 하려면 대신 신호 옵션 (`MQGMO_SET_SIGNAL`)을 사용하십시오. 그러나 신호 옵션은 환경에 특정됩니다. 서로 다른 환경 간에 이식하려는 애플리케이션은 이 옵션을 사용할 수 없습니다.
- 대기 및 신호 옵션을 혼합 사용하여 동일한 메시지에 대해 대기하는 `MQGET` 호출이 두 개 이상 있는 경우 각 대기 호출은 동일하게 간주됩니다. `MQGMO_SET_SIGNAL`을(를) `MQGMO_WAIT`(으)로 지정하는 것은 오류입니다. 또한 이 옵션을 신호가 미해결 상태인 큐 핸들과 함께 지정해서도 안 됩니다.
- `MQIT_MSG_TOKEN`의 `IndexType`이(가) 있는 큐에 대해 `MQGMO_WAIT` 또는 `MQGMO_SET_SIGNAL`을(를) 지정하는 경우, 선택 기준이 허용되지 않습니다. 이는 다음을 의미합니다.
  - 버전-1 `MQGMO`를 사용 중인 경우 `MQGET` 호출에 지정된 `MQMD`의 `MsgId` 및 `CorrelId` 필드를 `MQMI_NONE` 및 `MQCI_NONE`으로 설정하십시오.
  - version-2 이상의 `MQGMO`을(를) 사용하는 경우 `MatchOptions` 필드를 `MQMO_NONE`(으)로 설정하십시오.
- 공유 큐에 대한 `MQGET` 호출이며 호출이 찾아보기 요청이거나 그룹 메시지의 파괴적인 `Get`을 포함하고 `MsgId` 또는 `CorrelId` 중 아무 것도 일치하지 않는 경우, 사용자의 신호 `ECB`는 200밀리초 후 `MQEC_MSG_ARRIVED`로 게시됩니다.

이 상황은 `MQEC_WAIT_INTERVAL_EXPIRED`로 큐가 게시될 때 대기 간격이 만료될 때까지 적당한 메시지가 큐에 도착하지 않을 수 있는 경우에도 발생합니다. `MQEC_MSG_ARRIVED`가 게시될 때 하나가 사용 가능한 경우 두 번째 `MQGET` 호출을 다시 발행하여 메시지를 검색해야 합니다.

이 기술을 사용하면 적시에 메시지 도착 사실을 알 수 있지만, 비공유 큐에 대한 유사한 호출 순서와 비교할 때 예상치 못한 처리 오버헤드로 나타날 수 있습니다.

`MQGMO_WAIT`은(는) `MQGMO_BROWSE_MSG_UNDER_CURSOR` 또는 `MQGMO_MSG_UNDER_CURSOR`(으)로 지정된 경우 무시되며 오류가 발생하지 않습니다.

### **MQGMO\_NO\_WAIT**

적당한 메시지가 사용 가능하지 않은 경우 애플리케이션은 대기하지 않습니다. `MQGMO_NO_WAIT`은(는) `MQGMO_WAIT`과(와) 반대입니다. `MQGMO_NO_WAIT`은(는) 프로그램 문서화를 지원하기 위해 정의됩니다. 지정된 것이 없으면 이 값이 기본값입니다.

### **MQGMO\_SET\_SIGNAL**

`Signal1` 및 `Signal2` 필드와 함께 이 옵션을 사용하십시오. 애플리케이션이 메시지가 도착할 때까지 대기하는 동안 다른 작업을 진행할 수 있도록 합니다. 또한 적당한 운영 체제 기능이 사용 가능한 경우 애플리케이션이 두 개 이상의 큐에 메시지가 도착할 때까지 대기할 수 있도록 합니다.

**참고:** `MQGMO_SET_SIGNAL` 옵션은 환경에 따라 다르므로 이식하려는 애플리케이션에는 사용하지 마십시오.

두 가지 상황에서 호출은 이 옵션이 지정되지 않았을 때와 동일한 방법으로 완료됩니다.

1. 현재 사용 가능한 메시지가 메시지 디스크립터에서 지정된 기준을 충족하는 경우.
2. 매개변수 오류 또는 기타 동기 오류가 감지된 경우.

메시지 디스크립터에서 지정된 기준을 충족하는 메시지가 없는 경우 메시지가 도착할 때까지 기다리지 않고 애플리케이션으로 제어를 리턴합니다. **CompCode** 및 **Reason** 매개변수는 `MQCC_WARNING` 및 `MQRC_SIGNAL_REQUEST_ACCEPTED`(으)로 설정됩니다. `MQGET` 호출의 출력 매개변수 및 메시지 디스크

립터에서 기타 출력 필드가 설정되지 않습니다. 나중에 적당한 메시지가 도착할 때 ECB를 게시하여 신호가 전달됩니다.

그런 다음 호출자는 MQGET 호출을 재발행하여 메시지를 검색해야 합니다. 애플리케이션은 운영 체제가 제공하는 기능을 사용하여 이 신호에 대해 대기할 수 있습니다.

운영 체제가 다중 대기 메커니즘을 제공하는 경우 이를 사용하여 여러 큐 중 하나에서 메시지가 도착할 때까지 대기할 수 있습니다.

0이 아닌 WaitInterval이(가) 지정되면 대기 간격이 만료된 후 신호가 전달됩니다. 큐 관리자는 신호가 전달된 경우 대기를 취소할 수도 있습니다.

두 개 이상의 MQGET 호출이 동일한 메시지에 대한 신호를 설정할 수 있습니다. 애플리케이션이 활성화되는 순서는 MQGMO\_WAIT에 대해 설명한 것과 동일합니다.

동일한 메시지에 대해 두 개 이상의 MQGET 호출이 대기 중인 경우 대기 중인 각 호출은 동일하게 간주됩니다. 호출은 대기 및 신호 옵션의 혼합을 포함할 수 있습니다.

MQGET 호출은 특정 조건 아래에서 메시지를 검색할 수 있으며 동일한 메시지가 도착할 때 생성되는 신호가 전달될 수 있습니다. 신호가 전달되면 애플리케이션은 사용 가능한 메시지가 없음에 대비해야 합니다.

큐 핸들에서 신호 요청 미해결은 하나만 가능합니다.

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

- MQGMO\_UNLOCK
- MQGMO\_WAIT

공유 큐에 대한 MQGET 호출이며 호출이 찾아보기 요청이거나 그룹 메시지의 파괴적인 Get을 포함하고 MsgId 또는 CorrelId 중 아무 것도 일치하지 않는 경우, 사용자의 신호 ECB는 200밀리초 후 MQEC\_MSG\_ARRIVED로 게시됩니다.

이는 큐가 MQEC\_WAIT\_INTERVAL\_EXPIRED(으)로 게시될 때 대기 간격이 만료될 때까지 적합한 메시지가 큐에 도착하지 않은 경우에도 발생합니다. MQEC\_MSG\_ARRIVED가 게시될 때 하나가 사용 가능한 경우 두 번째 MQGET 호출을 다시 발행하여 메시지를 검색해야 합니다.

이 기술을 사용하면 적시에 메시지 도착 사실을 알 수 있지만, 비공유 큐에 대한 유사한 호출 순서와 비교할 때 예상치 못한 처리 오버헤드로 나타날 수 있습니다.

이는 메시지가 자주 추가되지 않을 때는 효율적인 메시지 검색 방법이 아닙니다. 찾아보기 작업 시 이런 오버헤드를 피하려면 MQGET 호출 시 일치하는 MsgId (MsgId에 의해 인덱스되지 않거나 인덱스되는 경우) 또는 CorrelId (CorrelId에 의해 인덱스되는 경우)를 지정하십시오.

**z/OS** 이 옵션은 z/OS에서만 지원됩니다.

### MQGMO\_FAIL\_IF QUIESCING

큐 관리자가 정지 상태에 있는 경우 MQGET 호출을 강제로 실패하게 합니다.

**z/OS** z/OS에서 이 옵션은 연결 (CICS 또는 IMS 애플리케이션의 경우) 이 정지 상태인 경우에도 MQGET 호출이 실패하도록 강제 실행합니다.

이 옵션이 MQGMO\_WAIT 또는 MQGMO\_SET\_SIGNAL(으)로 지정되고 큐 관리자가 정지 상태가 되는 시간에 대기 또는 신호가 미해결 상태인 경우:

- 대기가 취소되고 호출은 이유 코드 MQRC\_Q\_MGR QUIESCING 또는 MQRC\_CONNECTION QUIESCING 과(와) 함께 완료 코드 MQCC\_FAILED을(를) 리턴합니다.
- 신호가 환경에 특정한 신호 완료 코드로 취소됩니다.

**z/OS** z/OS에서 신호는 이벤트 완료 코드 MQEC\_Q\_MGR QUIESCING 또는 MQEC\_CONNECTION QUIESCING과(와) 함께 완료됩니다.

MQGMO\_FAIL\_IF QUIESCING이(가) 지정되지 않고 큐 관리자 또는 연결이 정지 상태가 되면 대기 또는 신호가 취소되지 않습니다.


## 동기점 옵션

다음 옵션은 작업 단위 내에서 MQGET 호출의 참여와 관련됩니다.

### MQGMO\_SYNCPOINT

요청이 일반 작업 단위 프로토콜 내에서 조작됩니다. 이 메시지는 다른 애플리케이션에 사용 불가능한 것으로 표시되지만, 작업 단위가 커밋되는 경우에만 큐에서 삭제됩니다. 작업 단위가 백아웃되는 경우 메시지가 다시 사용 가능하게 됩니다.

MQGMO\_SYNCPOINT 및 MQGMO\_NO\_SYNCPOINT을(를) 설정하지 않은 상태로 둘 수 있습니다. 이 경우 작업 단위 프로토콜에 가져오기 요청을 포함시키는 것은 큐 관리자를 실행하는 환경에 따라 판별됩니다. 이는 애플리케이션을 실행하는 환경에 따라 판별되지 않습니다.

-  z/OS에서 가져오기 요청은 작업 단위 내에 있습니다.
- z/OS 이외의 모든 환경에서는 가져오기 요청이 작업 단위 내에 있지 않습니다.

이러한 차이로 인해 인식하려는 애플리케이션은 이 옵션을 기본값으로 허용해서는 안 됩니다. MQGMO\_SYNCPOINT 또는 MQGMO\_NO\_SYNCPOINT을(를) 명시적으로 지정하십시오.

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

- MQGMO\_BROWSE\_FIRST
- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
- MQGMO\_BROWSE\_NEXT
- MQGMO\_LOCK
- MQGMO\_NO\_SYNCPOINT
- MQGMO\_SYNCPOINT\_IF\_PERSISTENT
- MQGMO\_UNLOCK

### MQGMO\_SYNCPOINT\_IF\_PERSISTENT

이 요청은 정상 작업 단위 프로토콜 내에서 조작하기 위한 요청입니다. 단, 검색된 메시지가 지속적인 경우에 만 해당됩니다. 지속 메시지에는 MQMD의 Persistence 필드에 MQPER\_PERSISTENT 값이 있습니다.

- 메시지가 지속적인 경우 큐 관리자는 애플리케이션이 MQGMO\_SYNCPOINT을(를) 지정한 것처럼 호출을 처리합니다.
- 메시지가 지속적이지 않은 경우 큐 관리자는 애플리케이션이 MQGMO\_NO\_SYNCPOINT을(를) 지정한 것처럼 호출을 처리합니다.

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

- MQGMO\_BROWSE\_FIRST
- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
- MQGMO\_BROWSE\_NEXT
- MQGMO\_COMPLETE\_MSG
- MQGMO\_MARK\_SKIP\_BACKOUT
- MQGMO\_NO\_SYNCPOINT
- MQGMO\_SYNCPOINT
- MQGMO\_UNLOCK

이 옵션은 다음 환경에서 지원됩니다.

-  AIX
-  IBM i
-  Linux

- **z/OS** z/OS

및 해당 시스템에 연결된 IBM MQ MQI clients

### **MQGMO\_NO\_SYNCPOINT**

요청은 일반 작업 단위 프로토콜의 외부에서 조작하는 것입니다. 찾아보기 옵션을 사용하지 않고 메시지를 가져오는 경우 메시지는 큐에서 즉시 삭제됩니다. 작업 단위를 백아웃하여 메시지를 다시 사용 가능하게 만들 수 없습니다.

이 옵션은 MQGMO\_BROWSE\_FIRST 또는 MQGMO\_BROWSE\_NEXT을(를) 지정한 경우 가정됩니다.

MQGMO\_SYNCPOINT 및 MQGMO\_NO\_SYNCPOINT을(를) 설정하지 않은 상태로 둘 수 있습니다. 이 경우 작업 단위 프로토콜에 가져오기 요청을 포함시키는 것은 큐 관리자를 실행하는 환경에 따라 판별됩니다. 이는 애플리케이션을 실행하는 환경에 따라 판별되지 않습니다.

- **z/OS** z/OS에서 가져오기 요청은 작업 단위 내에 있습니다.
- z/OS 이외의 모든 환경에서는 가져오기 요청이 작업 단위 내에 있지 않습니다.

이러한 차이로 인해 이식하려는 애플리케이션은 이 옵션을 기본값으로 허용해서는 안 됩니다. MQGMO\_SYNCPOINT 또는 MQGMO\_NO\_SYNCPOINT을(를) 명시적으로 지정하십시오.

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

- MQGMO\_MARK\_SKIP\_BACKOUT
- MQGMO\_SYNCPOINT
- MQGMO\_SYNCPOINT\_IF\_PERSISTENT

### **z/OS** **MQGMO\_MARK\_SKIP\_BACKOUT**

이 옵션으로 표시된 메시지를 큐에 재인스턴스화하지 않고 작업 단위를 백아웃하십시오.

이 옵션은 z/OS에서만 지원됩니다.

이 옵션을 지정하면 MQGMO\_SYNCPOINT도 지정해야 합니다. MQGMO\_MARK\_SKIP\_BACKOUT은(는) 다음 옵션 중 하나와 함께 사용할 수 없습니다.

- MQGMO\_BROWSE\_FIRST
- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
- MQGMO\_BROWSE\_NEXT
- MQGMO\_LOCK
- MQGMO\_NO\_SYNCPOINT
- MQGMO\_SYNCPOINT\_IF\_PERSISTENT
- MQGMO\_UNLOCK

**참고:** IMS 및 CICS에서 MQGMO\_MARK\_SKIP\_BACKOUT(으)로 표시된 메시지가 포함된 작업 단위를 백아웃 한 후에는 IBM MQ 호출을 실행해야 할 수도 있습니다. 표시된 메시지가 포함된 새 작업 단위를 커밋하기 전에 IBM MQ 호출을 발행해야 합니다. 호출은 사용자가 선호하는 IBM MQ 호출일 수 있습니다.

1. IMS에서 IMS APAR PN60855를 적용하지 않고 IMS MPP 또는 BMP 애플리케이션을 실행 중인 경우
2. CICS에서 임의 애플리케이션을 실행 중인 경우

두 경우에서 백아웃된 메시지가 포함된 새 작업 단위를 커밋하기 전에 IBM MQ 호출을 발행하십시오.

**참고:** 작업 단위 내에는 백아웃 건너뛰기로 표시된 가져오기 요청을 한 개만 포함할 수 있으며 표시되지 않은 여러 가져오기 요청도 포함하거나 포함하지 않을 수 있습니다.

애플리케이션이 작업 단위를 백아웃하면 MQGMO\_MARK\_SKIP\_BACKOUT을(를) 사용하여 검색된 메시지가 이전 상태로 복원되지 않습니다. 다른 자원 업데이트는 백아웃됩니다. 메시지는 백아웃 요청으로 시작된 새 작업 단위에서 검색된 것처럼 처리됩니다. 메시지는 MQGMO\_MARK\_SKIP\_BACKOUT 옵션 없이 검색됩니다.

MQGMO\_MARK\_SKIP\_BACKOUT은(는) 일부 자원이 변경된 후 작업 단위를 성공적으로 완료할 수 없는 경우에 유용합니다. 이 옵션을 생략하면 작업 단위를 백아웃할 때 큐에 있는 메시지가 재인스턴스화됩니다. 다음에 메시지를 검색할 때 동일한 순서의 이벤트가 다시 발생합니다.

그러나 원래 MQGET 호출에서 MQGMO\_MARK\_SKIP\_BACKOUT을 지정하면 작업 단위를 백아웃할 때 다른 자원에 대한 업데이트가 백아웃됩니다. 메시지는 새 작업 단위 아래에서 검색된 것처럼 처리됩니다. 애플리케이션은 적절한 오류 핸들링을 수행할 수 있습니다. 보고 메시지를 원래 메시지의 송신자로 전송하거나 원래 메시지를 데드-레터 큐에 배치할 수 있습니다. 그런 다음 새 작업 단위를 커밋할 수 있습니다. 새 작업 단위를 커밋하면 메시지가 원래 큐에서 영구적으로 제거됩니다.

MQGMO\_MARK\_SKIP\_BACKOUT은(는) 단일 실제 메시지를 표시합니다. 메시지가 메시지 그룹에 속한 경우 그룹에 있는 다른 메시지는 표시되지 않습니다. 마찬가지로 표시된 메시지가 논리 메시지의 세그먼트인 경우 이 논리 메시지의 다른 세그먼트는 표시되지 않습니다.

그룹에 있는 모든 메시지를 표시할 수 있지만 MQGMO\_LOGICAL\_ORDER을(를) 사용하여 메시지를 검색하는 경우 그룹의 첫 번째 메시지를 표시하는 것이 좋습니다. 작업 단위가 백아웃되는 경우 첫 번째 (표시된) 메시지가 새 작업 단위로 이동합니다. 그룹에서 두 번째 및 이후의 메시지는 큐에서 재인스턴스화됩니다.

MQGMO\_LOGICAL\_ORDER을(를) 사용하는 다른 애플리케이션에서 큐에 남아 있는 메시지를 검색할 수 없습니다. 그룹의 첫 번째 메시지는 더 이상 큐에 존재하지 않습니다. 그러나 작업 단위를 지원하는 애플리케이션은 MQGMO\_LOGICAL\_ORDER 옵션을 사용하여 두 번째 및 이후의 메시지를 새 작업 단위로 검색할 수 있습니다. 첫 번째 메시지는 이미 검색되었습니다.

가끔 새 작업 단위를 백아웃해야 하는 경우가 있습니다. 예를 들어, 데드-레터 큐가 가득 찼고 메시지를 제거해서는 안되기 때문입니다. 새 작업 단위를 백아웃하면 원래 큐에 있는 메시지가 재인스턴스화됩니다. 이렇게 하면 메시지가 손실되지 않습니다. 그러나 이 상황에서는 처리를 계속할 수 없습니다. 새 작업 단위를 백아웃한 후 애플리케이션은 운영자나 관리자에게 복구 불가능한 오류가 있음을 알린 다음 완료해야 합니다.

MQGMO\_MARK\_SKIP\_BACKOUT은(는) 가져오기 요청이 들어 있는 작업 단위가 애플리케이션이 이를 백킹하여 인터럽트되는 경우에만 작동합니다. 트랜잭션 또는 시스템이 실패하여 가져오기 요청을 포함하는 작업 단위가 백아웃되면 MQGMO\_MARK\_SKIP\_BACKOUT이(가) 무시됩니다. 이 옵션을 사용하여 검색된 메시지는 이 옵션을 사용하지 않고 검색된 메시지와 동일한 방법으로 큐에서 재인스턴스화됩니다.

## 찾아보기 옵션

다음 옵션은 큐에서 메시지를 찾아보는 것과 관련됩니다.

### MQGMO\_BROWSE\_FIRST

MQGMO\_BROWSE\_FIRST 옵션을 사용하여 큐를 여는 경우 찾아보기 커서가 설정되어 큐의 첫 번째 메시지 앞에 논리적으로 위치 지정됩니다. 그런 다음 MQGMO\_BROWSE\_FIRST, MQGMO\_BROWSE\_NEXT 또는

MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR 옵션을 지정하여 MQGET 호출을 사용하고 안전하게 큐에서 메시지를 검색할 수 있습니다. 찾아보기 커서는 큐에 있는 메시지 내에 해당 위치를 표시합니다.

MQGMO\_BROWSE\_NEXT를 사용하는 다음 MQGET 호출은 그 위치부터 적당한 메시지를 검색합니다.

MQGMO\_BROWSE\_FIRST은(는) 다음 옵션 중 하나와 함께 사용할 수 없습니다.

- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
- MQGMO\_BROWSE\_NEXT
- MQGMO\_MARK\_SKIP\_BACKOUT
- MQGMO\_MSG\_UNDER\_CURSOR
- MQGMO\_SYNCPOINT
- MQGMO\_SYNCPOINT\_IF\_PERSISTENT
- MQGMO\_UNLOCK

큐가 찾아보기에 대해 열려지지 않은 경우에도 오류가 발생합니다.

MQGMO\_BROWSE\_FIRST를 사용하는 MQGET 호출은 찾아보기 커서의 이전 위치를 무시합니다. 메시지 디스크립터에 지정된 조건을 충족하는 큐의 첫 번째 메시지가 검색됩니다. 메시지는 큐에 남아 있고 찾아보기 커서가 이 메시지에 위치 지정됩니다.

이 호출 후에 찾아보기 커서는 리턴된 메시지에 위치 지정됩니다. 다음 MQGET 호출이 MQGMO\_BROWSE\_NEXT를 사용하여 발행되기 전에 메시지가 큐에서 제거될 수 있습니다. 이 경우 찾아보기 커서는 큐에서 메시지가 차지했던 위치가 현재 비어 있더라도 그 위치에 남아 있습니다.

큐에서 메시지를 제거하려면 찾아보기하지 않는 MQGET 호출과 함께 MQGMO\_MSG\_UNDER\_CURSOR 옵션을 사용하십시오.

동일한 *Hobj* 핸들을 사용하는 경우에도 찾아보기하지 않는 MQGET 호출은 찾아보기 커서를 이동시킬 수 없습니다. 또는 완료 코드 MQCC\_FAILED 또는 이유 코드 MQRC\_TRUNCATED\_MSG\_FAILED를 리턴하는 찾아보기 MQGET 호출로도 찾아보기 커서를 이동시킬 수 없습니다.

탐색되는 메시지를 잠그려면 이 옵션과 함께 MQGMO\_LOCK 옵션을 지정하십시오.

논리 메시지의 그룹 및 세그먼트에서 메시지 처리를 제어하는 MQGMO\_\* 및 MQMO\_\* 옵션의 유효한 조합으로 MQGMO\_BROWSE\_FIRST을(를) 지정할 수 있습니다.

MQGMO\_LOGICAL\_ORDER을(를) 지정하면 메시지가 논리적 순서로 탐색됩니다. 해당 옵션을 생략하면 메시지를 실제 순서로 찾아봅니다. MQGMO\_BROWSE\_FIRST을(를) 지정하면 논리적 순서와 실제 순서 사이를 전환할 수 있습니다. MQGMO\_BROWSE\_NEXT를 사용하는 후속 MQGET 호출은 큐 핸들에 MQGMO\_BROWSE\_FIRST를 지정한 최신 호출과 동일한 순서로 큐를 찾아봅니다.

큐 관리자는 MQGET 호출에 대해 두 세트의 그룹 및 세그먼트 정보를 보유합니다. 찾아보기 호출에 대한 그룹 및 세그먼트 정보는 큐에서 메시지를 제거하는 호출에 대한 정보와 분리하여 보유됩니다.

MQGMO\_BROWSE\_FIRST을(를) 지정하면 큐 관리자가 찾아보기를 위한 그룹 및 세그먼트 정보를 무시합니다. 큐 관리자는 현재 그룹 및 현재 논리 메시지가 없는 경우에도 큐를 스캔합니다. MQGET 호출이 성공적이고 완료 코드가 MQCC\_OK 또는 MQCC\_WARNING이면 찾아보기에 대한 그룹 및 세그먼트 정보가 리턴된 메시지의 해당 정보로 설정됩니다. 호출을 실패하면 그룹 및 세그먼트 정보는 호출 이전의 상태와 동일하게 남아 있습니다.

### MQGMO\_BROWSE\_NEXT

찾아보기 커서를 MQGET 호출에서 지정된 선택 기준을 충족하는 큐의 다음 메시지로 이동합니다. 메시지는 애플리케이션으로 리턴되지만 큐에 남아 있습니다.

MQGMO\_BROWSE\_NEXT은(는) 다음 옵션 중 하나와 함께 사용할 수 없습니다.

- MQGMO\_BROWSE\_FIRST
- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
- MQGMO\_MARK\_SKIP\_BACKOUT
- MQGMO\_MSG\_UNDER\_CURSOR
- MQGMO\_SYNCPOINT
- MQGMO\_SYNCPOINT\_IF\_PERSISTENT
- MQGMO\_UNLOCK

큐가 찾아보기에 대해 열려지지 않은 경우에도 오류가 발생합니다.

MQGMO\_BROWSE\_NEXT은(는) 찾아보기를 위해 큐를 열고 나면 큐를 찾아보기 위한 첫 번째 호출인 경우 MQGMO\_BROWSE\_FIRST과(와) 동일한 방식으로 작동합니다.

커서 아래에 있는 메시지는 다음 MQGET 호출이 MQGMO\_BROWSE\_NEXT를 사용하여 발행되기 전에 큐에서 제거될 수 있습니다. 찾아보기 커서는 큐에서 메시지가 차지했던 위치가 현재 비어 있더라도 논리적으로 그 위치에 남아 있습니다.

메시지는 다음 두 가지 방법 중 하나로 큐에 저장됩니다.

- 우선순위 내에서 FIFO(MQMDS\_PRIORITY), 또는
- 우선순위에 관계없이 FIFO(MQMDS\_FIFO)

**MsgDeliverySequence** 큐 속성은 적용되는 메소드를 표시합니다(자세한 내용은 804 페이지의 『큐의 속성』 참조).



큐에 MQMDS\_PRIORITY의 MsgDeliverySequence이(가) 있을 수 있습니다. 메시지는 찾아보기 커서로 현재 지정된 것보다 더 높은 우선순위의 큐에 도착합니다. 이 경우, MQGMO\_BROWSE\_NEXT을(를) 사용하여 큐의 현재 스왑 동안 더 높은 우선순위 메시지를 찾을 수 없습니다. 찾아보기 커서가 MQGMO\_BROWSE\_FIRST(으)로 재설정된 후 또는 큐를 다시 열어서 찾을 수 있습니다.

MQGMO\_MSG\_UNDER\_CURSOR 옵션은 필요한 경우 찾아보기하지 않는 MQGET 호출과 함께 사용하여 큐에서 메시지를 제거하는 데 사용할 수 있습니다.

동일한 Hobj 핸들을 사용하여 찾아보기하지 않는 MQGET 호출은 찾아보기 커서를 이동시킬 수 없습니다.

찾아본 메시지를 잠그려면 이 옵션과 함께 MQGMO\_LOCK 옵션을 지정하십시오.

논리 메시지의 그룹 및 세그먼트에서 메시지 처리를 제어하는 MQGMO\_\* 및 MQMO\_\* 옵션의 유효한 조합으로 MQGMO\_BROWSE\_NEXT을(를) 지정할 수 있습니다.

MQGMO\_LOGICAL\_ORDER을(를) 지정하면 메시지가 논리적 순서로 탐색됩니다. 해당 옵션을 생략하면 메시지를 실제 순서로 찾아봅니다. MQGMO\_BROWSE\_FIRST을(를) 지정하면 논리적 순서와 실제 순서 사이를 전환할 수 있습니다. MQGMO\_BROWSE\_NEXT를 사용하는 후속 MQGET 호출은 큐 핸들에 MQGMO\_BROWSE\_FIRST를 지정한 최신 호출과 동일한 순서로 큐를 찾아봅니다. 이 조건이 충족되지 않으면 이유 코드 MQRC\_INCONSISTENT\_BROWSE과(와) 함께 호출이 실패합니다.

**참고:** MQGMO\_LOGICAL\_ORDER가 지정되지 않은 경우 MQGET 호출을 사용하여 메시지 그룹의 맨 끝을 넘어서 찾아보려면 특별히 주의하십시오. 예를 들어, 그룹의 마지막 메시지가 큐에서 그룹의 첫 번째 메시지에 선행한다고 가정하십시오. MQGMO\_BROWSE\_NEXT을(를) 사용하여 그룹 끝 너머를 찾아보고 MQMO\_MATCH\_MSG\_SEQ\_NUMBER을(를) MsgSeqNumber과(와) 함께 1(으)로 지정하면 이미 찾아본 그룹의 첫 번째 메시지가 리턴됩니다. 이 결과는 즉시 발생할 수 있거나 중간 그룹이 있는 경우 여러 번의 MQGET 호출 후에 발생할 수 있습니다. 동일한 고려사항이 그룹에 없는 논리 메시지에 적용됩니다.

찾아보기 호출에 대한 그룹 및 세그먼트 정보는 큐에서 메시지를 제거하는 호출에 대한 정보와 분리하여 보유됩니다.

### **MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR**

MQGMO의 MatchOptions 필드에 지정된 MQMO\_\* 옵션에 관계없이 찾아보기 커서가 가리키는 메시지를 비파괴적으로 검색합니다.

MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR은(는) 다음 옵션 중 하나와 함께 사용할 수 없습니다.

- MQGMO\_BROWSE\_FIRST
- MQGMO\_BROWSE\_NEXT
- MQGMO\_MARK\_SKIP\_BACKOUT
- MQGMO\_MSG\_UNDER\_CURSOR
- MQGMO\_SYNCPOINT
- MQGMO\_SYNCPOINT\_IF\_PERSISTENT
- MQGMO\_UNLOCK

큐가 찾아보기에 대해 열려지지 않은 경우에도 오류가 발생합니다.

찾아보기 커서가 가리키는 메시지는 MQGMO\_BROWSE\_FIRST 또는 MQGMO\_BROWSE\_NEXT 옵션을 사용하여 마지막으로 검색된 메시지입니다. 큐가 열려진 후에 이 큐에 대해 호출이 발행되지 않은 경우 호출이 실패합니다. 찾아보기 커서 아래에 있던 메시지가 파괴적으로 검색된 경우에도 호출이 실패합니다.

찾아보기 커서의 위치는 이 호출로 변경되지 않습니다.

MQGMO\_MSG\_UNDER\_CURSOR 옵션은 찾아보기하지 않는 MQGET 호출과 함께 사용하여 큐에서 메시지를 제거하는 데 사용할 수 있습니다.

동일한 Hobj 핸들을 사용하는 경우에도 찾아보기하지 않는 MQGET 호출은 찾아보기 커서를 이동시킬 수 없습니다. 또는 완료 코드 MQCC\_FAILED 또는 이유 코드 MQRC\_TRUNCATED\_MSG\_FAILED를 리턴하는 찾아보기 MQGET 호출로도 찾아보기 커서를 이동시킬 수 없습니다.

MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR이(가) MQGMO\_LOCK(으)로 지정된 경우:

- 이미 잠겨진 메시지가 있는 경우 커서 아래에 있는 메시지이므로 잠금 해제한 후 다시 잠글 필요없이 리턴됩니다. 메시지가 잠긴 상태로 있습니다.
- 잠겨진 메시지가 없고 찾아보기 커서 아래에 메시지가 있는 경우 잠금 후에 애플리케이션으로 리턴됩니다. 찾아보기 커서 아래에 메시지가 없는 경우 호출이 실패합니다.

MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR이(가) MQGMO\_LOCK 없이 지정된 경우:

- 이미 잠겨진 메시지가 있는 경우 메시지는 커서 아래에 있어야 합니다. 메시지는 애플리케이션으로 리턴된 다음 잠금 해제됩니다. 메시지가 이제 잠금 해제된 상태이므로 다시 찾아볼 수 있거나 동일한 애플리케이션에 의해 파괴적으로 검색될 수 있다고 확신할 수 없습니다. 큐에서 메시지를 가져오는 다른 애플리케이션에 의해 파괴적으로 검색되었을 수 있습니다.
- 잠겨진 메시지가 없고 찾아보기 커서 아래에 메시지가 있는 경우 애플리케이션으로 리턴됩니다. 찾아보기 커서 아래에 메시지가 없는 경우 호출이 실패합니다.

MQGMO\_COMPLETE\_MSG이(가) MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR(으)로 지정된 경우, 찾아보기 커서는 MQMD의 Offset 필드가 0인 메시지를 식별해야 합니다. 이 조건이 충족되지 않으면 호출이 실패하며 이유 코드는 MQRC\_INVALID\_MSG\_UNDER\_CURSOR입니다.

찾아보기 호출에 대한 그룹 및 세그먼트 정보는 큐에서 메시지를 제거하는 호출에 대한 정보와 분리하여 보유됩니다.

### **MQGMO\_MSG\_UNDER\_CURSOR**

MQGMO의 MatchOptions 필드에 지정된 MQMO\_\* 옵션에 관계없이 찾아보기 커서가 가리키는 메시지를 검색합니다. 메시지가 큐에서 제거됩니다.

찾아보기 커서가 가리키는 메시지는 MQGMO\_BROWSE\_FIRST 또는 MQGMO\_BROWSE\_NEXT 옵션을 사용하여 마지막으로 검색된 메시지입니다.

MQGMO\_COMPLETE\_MSG이(가) MQGMO\_MSG\_UNDER\_CURSOR(으)로 지정된 경우, 찾아보기 커서는 MQMD의 Offset 필드가 0인 메시지를 식별해야 합니다. 이 조건이 충족되지 않으면 호출이 실패하며 이유 코드는 MQRC\_INVALID\_MSG\_UNDER\_CURSOR입니다.

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

- MQGMO\_BROWSE\_FIRST
- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
- MQGMO\_BROWSE\_NEXT
- MQGMO\_UNLOCK

큐가 열림 및 입력 모두에 대해 열려지지 않은 경우에도 오류가 발생합니다. 찾아보기 커서가 현재 검색 가능한 메시지를 지정하고 있지 않은 경우 MQGET 호출이 오류를 리턴합니다.

### **MQGMO\_MARK\_BROWSE\_HANDLE**

성공적인 MQGET에 의해 리턴된 메시지 또는 리턴된 MsgToken으로 식별되는 메시지가 표시됩니다. 표시는 호출에 사용된 오브젝트 핸들에 특정합니다.

메시지가 큐에서 제거되지 않습니다.

MQGMO\_MARK\_BROWSE\_HANDLE은(는) 다음 옵션 중 하나도 지정된 경우에만 유효합니다.

- MQGMO\_BROWSE\_FIRST
- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
- MQGMO\_BROWSE\_NEXT

MQGMO\_MARK\_BROWSE\_HANDLE은(는) 다음 옵션 중 하나와 함께 사용할 수 없습니다.

- MQGMO\_ALL\_MSGS\_AVAILABLE
- MQGMO\_ALL\_SEGMENTS\_AVAILABLE
- MQGMO\_COMPLETE\_MSG
- MQGMO\_LOCK



- MQGMO\_LOGICAL\_ORDER
- MQGMO\_UNLOCK

메시지는 다음 이벤트 중 하나가 발생할 때까지 이 상태로 남아 있습니다.

- 관계된 오브젝트 핸들이 정상적으로 또는 다른 상태로 닫혀집니다.
- 메시지는 MQGMO\_UNMARK\_BROWSE\_HANDLE 옵션과 함께 MQGET에 대한 호출에 의해 이 핸들에 대한 표시가 해제됩니다.
- 메시지는 파괴적인 MQGET에 대한 호출에서 리턴되며 이 호출은 MQCC\_OK 또는 MQCC\_WARNING과 함께 완료됩니다. MQGET가 나중에 롤백되더라도 메시지는 변경된 상태로 남습니다.
- 메시지가 만기됩니다.

### **MQGMO\_MARK\_BROWSE\_CO\_OP**

성공적인 MQGET에 의해 리턴되거나 리턴된 *MsgToken*에 의해 식별된 메시지는 통합 세트에 있는 모든 핸들에 대해 표시됩니다.

모든 핸들 레벨 표시 외에도 통합 레벨 표시가 설정되었을 수 있습니다.

메시지가 큐에서 제거되지 않습니다.

MQGMO\_MARK\_BROWSE\_CO\_OP는 사용된 오브젝트 핸들이 MQOO\_CO\_OP를 지정한 MQOPEN 호출에 의해 리턴된 경우에만 유효합니다. 다음 MQGMO 옵션 중 하나를 지정해야 합니다.

- MQGMO\_BROWSE\_FIRST
- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
- MQGMO\_BROWSE\_NEXT

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

- MQGMO\_ALL\_MSGS\_AVAILABLE
- MQGMO\_ALL\_SEGMENTS\_AVAILABLE
- MQGMO\_COMPLETE\_MSG
- MQGMO\_LOCK
- MQGMO\_LOGICAL\_ORDER
- MQGMO\_UNLOCK

메시지가 이미 표시되어 있고 MQGMO\_UNMARKED\_BROWSE\_MSG 옵션이 지정되지 않은 경우 MQCC\_FAILED 및 이유 코드 MQRC\_MSG\_MARKED\_BROWSE\_CO\_OP과(와) 함께 호출이 실패합니다.

메시지는 다음 이벤트 중 하나가 발생할 때까지 이 상태로 남아 있습니다.

- 통합 세트에 있는 모든 오브젝트 핸들이 닫힙니다.
- 메시지는 MQGMO\_UNMARK\_BROWSE\_CO\_OP 옵션을 사용하는 MQGET 호출에 의해 통합 브라우저에 대한 표시가 해제됩니다.
- 메시지는 큐 관리자에 의해 자동으로 표시가 해제됩니다.
- 메시지가 찾아보기하지 않는 MQGET에 대한 호출에서 리턴됩니다. MQGET가 나중에 롤백되더라도 메시지는 변경된 상태로 남습니다.
- 메시지가 만기됩니다.

### **MQGMO\_UNMARKED\_BROWSE\_MSG**

MQGMO\_UNMARKED\_BROWSE\_MSG를 지정하는 MQGET 호출은 해당 핸들에 대한 표시가 해제된 것으로 간주되는 메시지를 리턴합니다. 메시지가 해당 핸들에 대해 표시되는 메시지는 리턴하지 않습니다. 큐가 MQOO\_CO\_OP 옵션을 지정하는 MQOPEN 호출에 의해 열렸고 메시지가 통합 세트의 멤버에 의해 표시된 경우에도 메시지를 리턴하지 않습니다.

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

- MQGMO\_ALL\_MSGS\_AVAILABLE

- MQGMO\_ALL\_SEGMENTS\_AVAILABLE
- MQGMO\_COMPLETE\_MSG
- MQGMO\_LOCK
- MQGMO\_LOGICAL\_ORDER
- MQGMO\_UNLOCK

#### **MQGMO\_UNMARK\_BROWSE\_CO\_OP**

이 옵션을 지정하는 MQGET을 호출한 후 통합 핸들 세트의 열린 핸들은 메시지를 더 이상 통합 세트에 대해 표시될 대상으로 고려하지 않습니다. 이 호출 이전에 메시지가 핸들 레벨에서 표시된 경우 메시지는 여전히 핸들 레벨에서 표시될 것으로 고려됩니다.

MQGMO\_UNMARK\_BROWSE\_CO\_OP는 MQOO\_CO\_OP 옵션을 사용한 성공적인 MQOPEN 호출로 리턴된 핸들 로만 사용될 수 있습니다. 메시지가 핸들 통합 세트에 의해 표시될 것으로 고려되지 않는 경우에도 MQGET은 성공합니다.

MQGMO\_UNMARK\_BROWSE\_CO\_OP는 찾아보기 하지 않는 MQGET 호출이나 다음 옵션과 함께 사용할 수 없습니다.

- MQGMO\_ALL\_MSGS\_AVAILABLE
- MQGMO\_ALL\_SEGMENTS\_AVAILABLE
- MQGMO\_COMPLETE\_MSG
- MQGMO\_LOCK
- MQGMO\_LOGICAL\_ORDER
- MQGMO\_MARK\_BROWSE\_CO\_OP
- MQGMO\_UNLOCK
- MQGMO\_UNMARKED\_BROWSE\_MSG

#### **MQGMO\_UNMARK\_BROWSE\_HANDLE**

이 옵션을 지정하는 MQGET을 호출한 후 찾은 메시지는 더 이상 이 핸들에 의해 표시될 대상으로 고려되지 않습니다.

메시지가 이 핸들에 대해 표시되지 않은 경우에도 호출은 성공합니다.

이 옵션은 찾아보기 하지 않는 MQGET 호출이나 다음 옵션과 함께 사용할 수 없습니다.

- MQGMO\_ALL\_MSGS\_AVAILABLE
- MQGMO\_ALL\_SEGMENTS\_AVAILABLE
- MQGMO\_COMPLETE\_MSG
- MQGMO\_LOCK
- MQGMO\_LOGICAL\_ORDER
- MQGMO\_MARK\_BROWSE\_CO\_OP
- MQGMO\_UNLOCK
- MQGMO\_UNMARKED\_BROWSE\_MSG

### **잠금 옵션**

다음 옵션은 큐에서 메시지를 잠그는 것과 관련됩니다.

#### **MQGMO\_LOCK**

메시지가 큐에 대해 열린 다른 핸들에 표시되지 않도록 찾아본 메시지를 잠그십시오. 이 옵션은 다음 옵션 중 하나가 지정된 경우에만 지정될 수 있습니다.

- MQGMO\_BROWSE\_FIRST
- MQGMO\_BROWSE\_NEXT

- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR

각 큐 핸들에 대해 하나의 메시지만 잠글 수 있습니다. 메시지는 논리 메시지 또는 물리적 메시지일 수 있습니다.

- MQGMO\_COMPLETE\_MSG을(를) 지정하면 논리 메시지를 구성하는 모든 메시지 세그먼트가 큐 핸들에 대해 잠깁니다. 메시지는 큐에 항상 존재하고 검색할 수 있어야 합니다.
- MQGMO\_COMPLETE\_MSG을(를) 생략하면 큐 핸들에 대해 하나의 실제 메시지만 잠깁니다. 이 메시지가 논리 메시지의 세그먼트가 되는 경우, 잠긴 세그먼트는 MQGMO\_COMPLETE\_MSG을(를) 사용하는 다른 애플리케이션이 논리 메시지를 검색하거나 찾아볼 수 없게 합니다.

잠긴 메시지는 항상 찾아보기 커서 아래에 있습니다. MQGMO\_MSG\_UNDER\_CURSOR 옵션을 지정하는 후속 MQGET 호출을 사용하여 메시지를 큐에서 제거할 수 있습니다. 큐 핸들을 사용하는 다른 MQGET 호출도 메시지를 제거할 수 있습니다(예를 들어, 잠긴 메시지의 메시지 ID를 지정하는 호출).

호출이 완료 코드 MQCC\_FAILED 또는 MQCC\_WARNING을(를) 이유 코드 MQRC\_TRUNCATED\_MSG\_FAILED과(와) 함께 리턴하는 경우 메시지가 잠기지 않습니다.

애플리케이션이 큐에서 메시지를 제거하지 않는 경우 잠금은 다음 조치 중 하나를 수행하여 해제될 수 있습니다.

- MQGMO\_BROWSE\_FIRST 또는 MQGMO\_BROWSE\_NEXT를 지정하여 이 핸들에 대해 다른 MQGET 호출을 발행합니다. 호출이 MQCC\_OK 또는 MQCC\_WARNING과(와) 함께 완료되면 잠금이 해제됩니다. 호출이 MQCC\_FAILED과(와) 함께 완료되면 메시지는 잠긴 상태로 유지됩니다. 그러나 다음 예외가 적용됩니다.
  - MQCC\_WARNING이(가) MQRC\_TRUNCATED\_MSG\_FAILED과(와) 함께 리턴되면 메시지가 잠금 해제되지 않습니다.
  - MQCC\_FAILED이(가) MQRC\_NO\_MSG\_AVAILABLE과(와) 함께 리턴되면 메시지가 잠금 해제됩니다.

MQGMO\_LOCK도 지정하면 리턴된 메시지가 잠깁니다. MQGMO\_LOCK을(를) 생략하면 호출 후에 잠긴 메시지가 없습니다.

MQGMO\_WAIT을(를) 지정하고 메시지를 즉시 사용할 수 없는 경우, 원래 메시지는 대기 시작 전에 잠금 해제됩니다.

- 이 핸들에 대해 MQGMO\_LOCK은 지정하지 않고 MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR를 지정하여 다른 MQGET 호출을 발행합니다. 호출이 MQCC\_OK 또는 MQCC\_WARNING과(와) 함께 완료되면 잠금이 해제됩니다. 호출이 MQCC\_FAILED과(와) 함께 완료되면 메시지는 잠긴 상태로 유지됩니다. 그러나 다음 예외가 적용됩니다.
  - MQCC\_WARNING이(가) MQRC\_TRUNCATED\_MSG\_FAILED과(와) 함께 리턴되면 메시지가 잠금 해제되지 않습니다.
- 이 핸들에 대해 MQGMO\_UNLOCK을 지정하여 다른 MQGET 호출을 발행합니다.

- 핸들을 사용하여 MQCLOSE 호출을 발행합니다. MQCLOSE는 애플리케이션이 종료될 때 포함될 수 있습니다.

부수적인 찾아보기 옵션을 지정하는 데 필요한 MQOO\_BROWSE 이외에는 MQGMO\_LOCK을 지정하는 데 특별한 MQOPEN 옵션이 필요하지 않습니다.

MQGMO\_LOCK은(는) 다음 옵션 중 하나와 함께 사용할 수 없습니다.

- MQGMO\_MARK\_SKIP\_BACKOUT
- MQGMO\_SYNCPOINT
- MQGMO\_SYNCPOINT\_IF\_PERSISTENT
- MQGMO\_UNLOCK

## MQGMO\_UNLOCK

잠금 해제될 메시지는 MQGMO\_LOCK 옵션을 사용하는 MQGET 호출에 의해 이미 잠겨져 있어야 합니다. 이 핸들에 대해 잠긴 메시지가 없는 경우, 호출은 MQCC\_WARNING 및 MQRC\_NO\_MSG\_LOCKED과(와) 함께 완료됩니다.

MQGMO\_UNLOCK을(를) 지정하는 경우 **MsgDesc**, **BufferLength**, **Buffer** 및 **DataLength** 매개변수는 검사되지 않거나 변경되지 않습니다. **Buffer**에 메시지가 리턴되지 않습니다.

MQGMO\_UNLOCK을(를) 지정하는 데 특별한 열기 옵션이 필요하지 않습니다(첫 번째 위치에서 잠금 요청을 발행하는 데는 MQOO\_BROWSE이(가) 필요함).

이 옵션은 다음을 제외한 옵션과 함께 사용할 수 없습니다.

- MQGMO\_NO\_WAIT
- MQGMO\_NO\_SYNCPOINT

해당 두 옵션이 지정되었는지 여부와 관계없이 가정됩니다.

## 메시지-데이터 옵션

다음 옵션은 큐에서 메시지를 읽을 때 메시지 데이터의 처리와 관련됩니다.

### MQGMO\_ACCEPT\_TRUNCATED\_MSG

메시지 버퍼가 너무 작아 전체 메시지를 보유할 수 없는 경우 MQGET 호출이 버퍼를 채우도록 허용합니다. MQGET은 버퍼에 가능한 많은 메시지를 채웁니다. 경고 완료 코드를 발행하고 처리를 완료합니다. 이는 다음을 의미합니다.

- 메시지를 찾아볼 때 찾아보기 커서가 리턴된 메시지로 이동합니다.
- 메시지를 제거할 때 리턴된 메시지가 큐에서 제거됩니다.
- 다른 오류가 발생하지 않으면 이유 코드 MQRC\_TRUNCATED\_MSG\_ACCEPTED가 리턴됩니다.

이 옵션을 사용하지 않으면 버퍼는 보유할 수 있는 만큼 많은 메시지로 계속 채워집니다. 경고 완료 코드가 발행되지만 처리가 완료되지 않습니다. 이는 다음을 의미합니다.

- 메시지를 찾아볼 때 찾아보기 커서가 이동되지 않습니다.
- 메시지를 제거할 때 메시지가 큐에서 제거되지 않습니다.
- 다른 오류가 발생하지 않으면 이유 코드 MQRC\_TRUNCATED\_MSG\_FAILED이(가) 리턴됩니다.

### MQGMO\_CONVERT

이 옵션은 MQGET 호출의 **MsgDesc** 매개변수에 지정된 CodedCharSetId 및 Encoding 값을 준수하도록 메시지의 애플리케이션 데이터를 변환합니다. 데이터는 **Buffer** 매개변수에 복사되기 전에 변환됩니다.

메시지를 넣을 때 지정된 Format 필드는 메시지 데이터의 특성을 식별하기 위해 변환 프로세스에 의해 가정됩니다. 메시지 데이터는 내장 형식의 경우 큐 관리자의 의해 변환되고 기타 형식의 경우 사용자 작성 엑시트에 의해 변환됩니다. 데이터 변환 엑시트에 대한 세부사항은 [869 페이지의 『데이터 변환 엑시트』](#)의 내용을 참조하십시오.

- 변환이 성공하면 MQGET 호출에서 리턴될 때 **MsgDesc** 매개변수에 지정된 CodedCharSetId 및 Encoding 필드가 변경되지 않습니다.
- 변환에 실패하면 메시지 데이터가 변환되지 않은 상태로 리턴됩니다. **MsgDesc**의 CodedCharSetId 및 Encoding 필드는 변환되지 않은 메시지의 값으로 설정됩니다. 이 경우 완료 코드는 MQCC\_WARNING입니다.

어느 경우에나 이 필드는 **Buffer** 매개변수에 리턴되는 메시지 데이터의 문자 세트 ID 및 인코딩을 설명합니다.

큐 관리자가 변환을 수행하는 형식 이름의 목록은 [412 페이지의 『MQMD - 메시지 디스크립터』](#)에 설명된 **Format** 필드를 참조하십시오.

## 그룹 및 세그먼트 옵션

다음 옵션은 논리 메시지의 그룹 및 세그먼트에서 메시지 처리와 관련됩니다. 옵션 설명 앞에는 중요한 용어에 대한 몇 가지 정의가 있습니다.

## 실제 메시지

물리적 메시지는 큐에 놓거나 제거할 수 있는 최소 단위의 정보입니다. 보통 단일 MQPUT, MQPUT1 또는 MQGET 호출에서 지정되거나 검색되는 정보에 해당합니다. 모든 물리적 메시지는 고유한 메시지 디스크립터 MQMD를 가집니다. 일반적으로 실제 메시지는 MQMD의 MsgId 필드인 메시지 ID에 대한 값을 다르게 하여 구별됩니다. 큐 관리자는 강제로 다른 값을 지정하지 않습니다.

## 논리 메시지

논리 메시지는 단일 단위의 애플리케이션 정보입니다. 시스템 제한 조건이 없는 경우 논리 메시지는 실제 메시지와 동일합니다. 논리 메시지가 큰 경우 시스템 제한조건이 논리 메시지를 두 개 이상의 물리적 메시지(세그먼트라고 함)로 나누도록 권장하거나 요청할 수 있습니다.

세그먼트화된 논리 메시지는 널이 아닌 그룹 ID(MQMD의 GroupId 필드)가 동일한 두 개 이상의 실제 메시지로 구성됩니다. 동일한 메시지 순서 번호(MQMD의 MsgSeqNumber 필드)가 있습니다. 세그먼트는 세그먼트 오프셋(MQMD의 Offset 필드)에 대한 값을 다르게 하여 구별됩니다. 세그먼트 오프셋은 논리 메시지에 있는 데이터의 시작에서 물리적 메시지에 있는 데이터의 오프셋입니다. 각 세그먼트는 물리적 메시지이므로 논리 메시지에 있는 세그먼트는 일반적으로 서로 다른 메시지 ID를 가집니다.

세그먼트화되지 않았지만 전송 애플리케이션에 의해 세그먼트화가 허용된 논리 메시지는 널이 아닌 그룹 ID를 가집니다. 이 경우 논리 메시지가 메시지 그룹에 속하지 않은 경우 해당 그룹 ID와 함께 하나의 물리적 메시지만 있게 됩니다. 전송 애플리케이션에 의해 세그먼트화가 금지된 논리 메시지는 논리 메시지가 메시지 그룹에 속하지 않는 한 널 그룹 ID(MQGI\_NONE)를 갖습니다.

## 메시지 그룹

메시지 그룹은 널이 아닌 동일한 그룹 ID가 있는 하나 이상의 논리 메시지 세트입니다. 그룹에 있는 논리 메시지는 메시지 순서 번호의 서로 다른 값으로 구분됩니다. 순서 번호는 1 - n 범위의 정수이며 여기서 n은 그룹에 있는 논리 메시지의 수입니다. 하나 이상의 논리 메시지가 세그먼트화되면 그룹에는 n보다 많은 실제 메시지가 있습니다.

## MQGMO\_LOGICAL\_ORDER

MQGMO\_LOGICAL\_ORDER는 큐 핸들에 대한 연속적인 MQGET 호출에 의해 메시지가 리턴되는 순서를 제어합니다. 옵션은 각 호출에서 지정되어야 합니다.

MQGMO\_LOGICAL\_ORDER가 동일한 큐 핸들에 대해 연속적인 MQGET 호출에 지정된 경우 그룹에 있는 메시지는 해당 메시지 순서 번호의 순서로 리턴됩니다. 논리 메시지의 세그먼트는 해당 세그먼트 오프셋이 제공하는 순서로 리턴됩니다. 이 순서는 해당 메시지 및 세그먼트가 큐에서 발생하는 순서와 차이가 있을 수 있습니다.

**참고:** MQGMO\_LOGICAL\_ORDER을(를) 지정하면 그룹에 속하지 않고 세그먼트가 아닌 메시지에 부정적인 영향을 주지 않습니다. 사실상 이러한 메시지는 각기 하나의 메시지로만 구성된 메시지 그룹에 속하는 것처럼 처리됩니다. 그룹에 있는 메시지의 혼합, 메시지 세그먼트 및 세그먼트화되지 않은 메시지가 포함된 큐에서 메시지를 검색할 때 MQGMO\_LOGICAL\_ORDER을(를) 지정하는 것이 안전합니다.

필요한 순서로 메시지를 리턴하기 위해 큐 관리자는 연속적인 MQGET 호출 사이에 그룹 및 세그먼트 정보를 보유하고 있습니다. 그룹 및 세그먼트 정보는 큐 핸들에 대한 현재 메시지 그룹 및 현재 논리 그룹을 식별합니다. 또한 그룹 및 논리 메시지 내의 현재 위치 및 메시지가 작업 단위 내에서 검색되고 있는지 여부도 식별합니다. 큐 관리자가 이 정보를 보유하고 있으므로 애플리케이션은 각 MQGET 호출 전에 그룹 및 세그먼트 정보를 설정할 필요가 없습니다. 특히, 이는 애플리케이션이 MQMD의 GroupId, MsgSeqNumber 및 Offset 필드를 설정할 필요가 없음을 의미합니다. 그러나 애플리케이션은 각 호출에서 MQGMO\_SYNCPOINT 또는 MQGMO\_NO\_SYNCPOINT 옵션을 올바르게 설정해야 합니다.

큐가 열려졌을 때는 현재 메시지 그룹 및 현재 논리 메시지가 없습니다. 메시지 그룹은 MQGET 호출에 의해 MQMF\_MSG\_IN\_GROUP 플래그를 가진 메시지가 리턴될 때 현재 메시지 그룹이 됩니다. 연속 호출에서 MQGMO\_LOGICAL\_ORDER이(가) 지정되면 해당 그룹은 다음이 있는 메시지가 리턴될 때까지 현재 그룹으로 유지됩니다.

- MQMF\_SEGMENT 없는 MQMF\_LAST\_MSG\_IN\_GROUP(즉, 그룹의 마지막 논리 메시지가 분할되지 않음), 또는
- MQMF\_LAST\_SEGMENT 있는 MQMF\_LAST\_MSG\_IN\_GROUP(즉, 리턴된 메시지가 그룹의 마지막 논리 메시지의 마지막 세그먼트임)

이러한 메시지가 리턴되면 메시지 그룹이 종료되고 MQGET 호출이 성공적으로 완료될 때 현재 그룹은 더 이상 존재하지 않습니다. 마찬가지로 논리 메시지는 MQGET 호출에 의해 MQMF\_SEGMENT 플래그를 가지는 메시지가 리턴될 때 현재 논리 메시지가 됩니다. MQMF\_LAST\_SEGMENT 플래그가 있는 메시지가 리턴되면 논리 메시지가 종료됩니다.

선택 기준이 지정되지 않은 경우 연속 MQGET 호출은 큐의 첫 번째 메시지 그룹에 대한 메시지를 올바른 순서로 리턴합니다. 그런 다음 더 이상 사용 가능한 메시지가 없을 때까지 두 번째 및 그 다음 메시지 그룹에 대한 메시지를 리턴합니다. MatchOptions 필드에 다음 옵션 중 하나 이상을 지정하여 리턴되는 특정 메시지 그룹을 선택할 수 있습니다.

- MQMO\_MATCH\_MSG\_ID
- MQMO\_MATCH\_CORREL\_ID
- MQMO\_MATCH\_GROUP\_ID

그러나 이러한 옵션은 현재 메시지 그룹 또는 논리 메시지가 없을 때만 유효합니다. 자세한 내용은 362 페이지의 『MQGMO - 메시지가져오기 옵션』에 설명된 MatchOptions 필드를 참조하십시오.

386 페이지의 표 495에서는 MQGET 호출 시 리턴할 메시지를 찾으려고 시도할 때 큐 관리자가 찾아보는 MsgId, CorrelId, GroupId, MsgSeqNumber, Offset 필드의 값을 표시합니다. 규칙은 큐에서 메시지를 제거하고 큐에 있는 메시지를 찾아볼 때 모두 적용됩니다. 표에서 둘 중 하나는 예 또는 아니오를 의미합니다.

**LOG ORD**

호출에 MQGMO\_LOGICAL\_ORDER 옵션이 지정되었는지 여부를 표시합니다.

**Cur grp**

현재 메시지 그룹이 호출 이전에 존재하는지 여부를 표시합니다.

**Cur log msg**

현재 논리 메시지가 호출 이전에 존재하는지 여부를 표시합니다.

**기타 열**

큐 관리자가 찾는 값을 표시합니다. 이전은 큐 핸들에 대한 이전 메시지에 있는 필드에 대해 리턴된 값을 표시합니다.

표 495. 논리 메시지의 세그먼트 및 그룹의 메시지와 관련된 MQGET 옵션							
지정하는 옵션	호출 이전의 그룹 및 로그 메시지 상태		큐 관리자가 찾는 값				
LOG ORD	Cur grp	Cur log msg	MsgId	CorrelId	GroupId	MsgSeqNumber	Offset
예	아니오	아니오	제어 주체: MatchOptions	제어 주체: MatchOptions	제어 주체: MatchOptions	1	0
예	아니오	예	모든 메시지 ID	모든 상관 ID	이전 그룹 ID	1	이전 오프셋 + 이전 세그먼트 길이
예	예	아니오	모든 메시지 ID	모든 상관 ID	이전 그룹 ID	이전 순서 번호 + 1	0
예	예	예	모든 메시지 ID	모든 상관 ID	이전 그룹 ID	이전 순서 번호	이전 오프셋 + 이전 세그먼트 길이
아니오	둘 중 하나	둘 중 하나	제어 주체: MatchOptions	제어 주체: MatchOptions	제어 주체: MatchOptions	제어 주체: MatchOptions	제어 주체: MatchOptions

큐에 다중 메시지 그룹이 존재하며 리턴이 가능한 경우 그룹은 각 그룹에 있는 첫 번째 논리 메시지의 첫 번째 세그먼트의 큐에서의 위치에 의해 판별된 순서로 리턴됩니다. 즉, 메시지 순서 번호가 1이고 오프셋이 0인 물리적 메시지가 해당 그룹이 리턴되는 순서를 판별합니다.

MQGMO\_LOGICAL\_ORDER 옵션은 다음과 같이 작업 단위에 영향을 줍니다.

- 그룹의 첫 번째 논리 메시지 또는 세그먼트가 작업 단위 내에서 검색되는 경우 동일한 큐 핸들이 사용되면 그룹의 다른 모든 논리 메시지 및 세그먼트가 작업 단위 내에서 검색되어야 합니다. 그러나 이를 동일한 작업 단위 내에서 검색할 필요는 없습니다. 이 방법은 여러 실제 메시지로 구성된 메시지 그룹이 큐 핸들에 대해 두 개 이상의 연속 작업 단위에서 나뉘질 수 있게 합니다.
- 그룹의 첫 번째 논리 메시지 또는 세그먼트가 작업 단위 내에서 검색되지 않고 동일한 큐 핸들이 사용되는 경우, 그룹의 다른 논리 메시지 및 세그먼트를 작업 단위 내에서 검색할 수 없습니다.

이러한 조건이 충족되지 않으면 MQGET 호출이 이유 코드 MQRC\_INCONSISTENT\_UOW와 함께 실패합니다.

MQGMO\_LOGICAL\_ORDER가 지정되면 MQGET 호출에서 제공된 MQGMO는 MQGMO\_VERSION\_2 미만이어야 하고 MQMD는 MQMD\_VERSION\_2 미만이어야 합니다. 이 조건이 충족되지 않으면 적절한 이유 코드 MQRC\_WRONG\_GMO\_VERSION 또는 MQRC\_WRONG\_MD\_VERSION과(와) 함께 호출이 실패합니다.

큐 핸들에 대한 연속 MQGET 호출에 MQGMO\_LOGICAL\_ORDER가 지정되지 않은 경우, 메시지는 메시지 그룹에 속하는지 또는 논리 메시지의 세그먼트인지 여부와 관계없이 리턴됩니다. 이는 특정 그룹 또는 논리 메시지의 메시지 또는 세그먼트가 순서없이 리턴되거나 다른 그룹 또는 논리 메시지의 메시지 또는 세그먼트와 섞이거나 세그먼트가 아니거나 그룹에 속하지 않은 메시지와 섞일 수 있음을 의미합니다. 이 상황에서 연속 MQGET 호출에 의해 리턴되는 특정 메시지는 해당 호출에 지정된 MQMO\_\* 옵션에 의해 제어됩니다 (이러한 옵션에 대한 세부사항은 362 페이지의 『MQGMO - 메시지 가져오기 옵션』에 설명된 MatchOptions 필드 참조).

이 방법은 시스템 실패가 발생한 후 중간에 있는 메시지 그룹 또는 논리 메시지를 재시작하는 데 사용될 수 있습니다. 시스템이 다시 시작하면 애플리케이션은 GroupId, MsgSeqNumber, Offset 및 MatchOptions 필드를 적절한 값으로 설정한 다음 MQGMO\_SYNCPOINT 또는 MQGMO\_NO\_SYNCPOINT를 설정하지만 MQGMO\_LOGICAL\_ORDER를 지정하지 않고 MQGET 호출을 실행할 수 있습니다. 이 호출에 성공하면 큐 관리자가 그룹 및 세그먼트 정보를 보유할 수 있으며 해당 큐 핸들을 사용하는 후속 MQGET 호출이 정상적으로 MQGMO\_LOGICAL\_ORDER를 지정할 수 있습니다.

큐 관리자가 MQGET 호출에 대해 보유하는 그룹 및 세그먼트 정보는 MQPUT 호출에 대해 보유하는 그룹 및 세그먼트 정보와 분리됩니다. 또한 큐 관리자는 다음에 대해 별도 정보를 보유합니다.

- 큐에서 메시지를 제거하는 MQGET 호출.
- 큐에 있는 메시지를 찾아보기하는 MQGET 호출.

제공된 큐 핸들에 대해 애플리케이션은 MQGMO\_LOGICAL\_ORDER를 지정하는 MQGET 호출과 이를 지정하지 않는 MQGET 호출을 혼합하여 사용할 수 있습니다. 그러나 다음 사항에 주의하십시오.

- MQGMO\_LOGICAL\_ORDER를 생략하는 경우 MQGET 호출을 성공할 때마다 큐 관리자는 저장된 그룹 및 세그먼트 정보를 리턴된 메시지에 해당하는 값으로 설정하게 됩니다. 이는 큐 관리자가 큐 핸들에 대해 보유하는 기존 그룹 및 세그먼트 정보를 대체합니다. 호출의 조치에 적절한 정보(찾아보기 또는 제거)만 수정됩니다.
- MQGMO\_LOGICAL\_ORDER을(를) 생략하면 현재 메시지 그룹 또는 논리 메시지가 있는 경우 호출이 실패하지 않습니다. 호출은 MQCC\_WARNING 완료 코드로 성공할 수 있습니다. 388 페이지의 표 496에서는 발생할 수 있는 다양한 사례를 보여줍니다. 이 경우 완료 코드가 MQCC\_OK이(가) 아닌 경우 이유 코드는 다음 중 하나입니다(해당되는 경우).
  - MQRC\_INCOMPLETE\_GROUP
  - MQRC\_INCOMPLETE\_MSG
  - MQRC\_INCONSISTENT\_UOW

**참고:** 큐 관리자는 큐를 찾아볼 때 또는 찾아보기를 위해 열린 큐를 닫을 때 그룹 및 세그먼트 정보를 확인하지 않습니다. 이러한 경우 완료 코드는 항상 MQCC\_OK입니다(다른 오류가 없는 경우).



현재 호출	이전 호출이 MQGET임 (MQGMO_LOGICAL_ORDER 사용)	이전 호출이 MQGET임 (MQGMO_LOGICAL_ORDER 사용 안 함)
MQGMO_LOGICAL_ORDER이(가) 있는 MQGET	MQCC_FAILED	MQCC_FAILED
MQGMO_LOGICAL_ORDER 없는 MQGET	MQCC_WARNING	MQCC_OK
종료되지 않은 그룹 또는 논리 메시지의 MQCLOSE	MQCC_WARNING	MQCC_OK

논리적 순서로 메시지 및 세그먼트를 검색하려는 애플리케이션은 사용할 수 있는 가장 간단한 옵션인 MQGMO\_LOGICAL\_ORDER을(를) 지정하는 것이 좋습니다. 이 옵션은 큐 관리자가 이들 정보를 관리할 때 때문에 애플리케이션이 그룹 및 세그먼트 정보를 관리할 필요가 없습니다. 그러나 특수 애플리케이션은 MQGMO\_LOGICAL\_ORDER 옵션에서 제공하는 것보다 더 많은 제어가 필요할 수 있으며 해당 옵션을 지정하지 않음으로써 이를 달성할 수 있습니다. 따라서 애플리케이션은 각 MQGET 호출 전에 MQMD의 MsgId, CorrelId, GroupId, MsgSeqNumber, Offset 필드와 MQGMO의 MatchOptions에 있는 MQMO\_\* 옵션이 올바르게 설정되었는지 확인해야 합니다.

예를 들어, 메시지가 논리 메시지의 그룹 또는 세그먼트에 있는지 여부에 관계없이 수신하는 실제 메시지를 전달하려는 애플리케이션은 MQGMO\_LOGICAL\_ORDER을(를) 지정해서는 안 됩니다. 송신 및 수신 큐 관리자 사이에 다중 경로가 포함된 복잡한 네트워크에서는 실제 메시지가 순서없이 도착합니다. MQPUT 호출에서 MQGMO\_LOGICAL\_ORDER 또는 해당 MQPMO\_LOGICAL\_ORDER을(를) 지정하지 않으면 전달 애플리케이션이 도착하는 즉시 각 실제 메시지를 검색 및 전달하여 논리적 순서로 다음 메시지가 도착할 때까지 기다리지 않아도 됩니다.

다른 MQGMO\_\* 옵션과 함께 MQGMO\_LOGICAL\_ORDER을(를) 지정할 수 있으며, 적절한 경우에는 다양한 MQMO\_\* 옵션을 사용하여 지정할 수 있습니다(이전 섹션 참조).

- ▶ **z/OS** z/OS에서 이 옵션은 개인 및 공유 큐에 대해 지원되지만 큐에는 MQIT\_GROUP\_ID의 색인 유형이 있어야 합니다. 공유 큐의 경우 큐가 맵핑하는 CFSTRUCT 오브젝트는 CFLEVEL(3) 이상에 있어야 합니다.
- 이 옵션은 다음 플랫폼의 모든 로컬 큐에 지원됩니다.
  - ▶ **AIX** AIX
  - ▶ **Linux** Linux
  - ▶ **IBM i** IBM i
  - ▶ **Windows** Windows

및 해당 시스템에 연결된 IBM MQ MQI clients.

### MQGMO\_COMPLETE\_MSG

완전한 논리 메시지만 MQGET 호출에 의해 리턴될 수 있습니다. 논리 메시지가 세그먼트화된 경우 큐 관리자는 이 세그먼트를 리어셈블링하여 완전한 논리 메시지를 애플리케이션으로 리턴합니다. 논리 메시지가 세그먼트화되었다는 사실은 이 메시지를 검색하는 애플리케이션에게 명시되지 않습니다.

**참고:** 이 옵션을 통해서만 큐 관리자가 메시지 세그먼트를 리어셈블링할 수 있습니다. 이 옵션을 지정하지 않으면 큐에 세그먼트가 있는 경우(및 MQGET 호출에서 지정된 기타 선택 기준을 충족하는 경우) 세그먼트가 개별적으로 애플리케이션으로 리턴됩니다. 개별 세그먼트를 수신하지 않으려는 애플리케이션은 항상 MQGMO\_COMPLETE\_MSG을(를) 지정해야 합니다.

이 옵션을 사용하려면 애플리케이션이 전체 메시지를 수용하기에 충분히 큰 버퍼를 제공하거나 MQGMO\_ACCEPT\_TRUNCATED\_MSG 옵션을 지정해야 합니다.



큐에 일부 세그먼트가 누락되어 세그먼트화된 메시지가 포함된 경우(네트워크에서 지연되어 아직 도달하지 않았기 때문일 수 있음), MQGMO\_COMPLETE\_MSG을(를) 지정하면 불완전한 논리 메시지에 속하는 세그먼트를 검색할 수 없습니다. 그러나 이러한 메시지 세그먼트는 여전히 **CurrentQDepth** 큐 속성의 값에 기여합니다. 이는 **CurrentQDepth**이(가) 0보다 큰 경우에도 검색 가능한 논리 메시지가 없을 수 있음을 의미합니다.

지속 메시지의 경우 큐 관리자는 작업 단위 내에서만 세그먼트를 리어셈블링할 수 있습니다.

- MQGET 호출이 사용자 정의 작업 단위 내에서 작동하는 경우 해당 작업 단위가 사용됩니다. 리어셈블링 프로세스 중에 호출이 실패하는 경우 큐 관리자는 리어셈블링 동안 제거된 모든 세그먼트를 큐에서 재인스턴스화합니다. 그러나 이 실패와 상관없이 작업 단위는 성공적으로 커밋됩니다.
- 호출이 사용자 정의 작업 단위 외부에서 작동 중이고 사용자 정의 작업 단위가 없는 경우 큐 관리자는 호출이 지속되는 동안 작업 단위를 작성합니다. 호출이 성공하면 큐 관리자가 자동으로 작업 단위를 커밋합니다(애플리케이션이 이를 수행할 필요가 없습니다). 호출이 실패하면 큐 관리자가 작업 단위를 백아웃합니다.
- 호출이 사용자 정의 작업 단위 외부에서 작동 중이지만 사용자 정의 작업 단위가 있는 경우 큐 관리자는 리어셈블링을 수행할 수 없습니다. 메시지를 리어셈블링할 필요가 없는 경우 호출은 계속 성공할 수 있습니다. 그러나 메시지를 리어셈블링해야 하는 경우 호출이 실패하며 이유 코드는 MQRC\_UOW\_NOT\_AVAILABLE입니다.

비지속 메시지의 경우 리어셈블링을 수행하기 위해 큐 관리자가 작업 단위를 사용할 필요가 없습니다.

세그먼트인 각 실제 메시지는 자체 메시지 디스크립터가 있습니다. 단일 논리 메시지를 구성하는 세그먼트의 경우, 메시지 디스크립터에 있는 대부분의 필드는 논리 메시지의 모든 세그먼트에 대해 동일합니다. 일반적으로 논리 메시지의 세그먼트 간에는 MsgId, Offset 및 MsgFlags 필드만 다릅니다. 그러나 중간 큐 관리자의 데드-레터 큐에 세그먼트가 배치되면 DLQ 핸들러는 MQGMO\_CONVERT 옵션을 지정하는 메시지를 검색하며 이로 인해 문자 세트 또는 세그먼트의 인코딩이 변경될 수 있습니다. DLQ 핸들러가 세그먼트를 성공적으로 전송한 경우 세그먼트는 세그먼트가 목적지 큐 관리자에 도착할 때 논리 메시지에 있는 다른 세그먼트와 다른 문자 세트 또는 인코딩을 가질 수 있습니다.

CodedCharSetId 및 Encoding 필드가 다른 세그먼트로 구성된 논리 메시지는 큐 관리자가 단일 논리 메시지로 리어셈블링할 수 없습니다. 대신, 큐 관리자는 동일한 문자 세트 ID 및 인코딩을 갖는 논리 메시지의 시작 부분에 있는 처음 몇 개의 연속 세그먼트를 리어셈블링하여 리턴하고 MQGET 호출은 완료 코드 MQCC\_WARNING 및 해당되는 이유 코드 MQRC\_INCONSISTENT\_CCSDS 또는 MQRC\_INCONSISTENT\_ENCODINGS와 함께 완료됩니다. 이는 MQGMO\_CONVERT이(가) 지정되었는지 여부에 관계없이 발생합니다. 나머지 세그먼트를 검색하려면 애플리케이션이 MQGMO\_COMPLETE\_MSG 옵션을 지정하지 않고 MQGET 호출을 다시 발행하여 세그먼트를 하나씩 검색해야 합니다. MQGMO\_LOGICAL\_ORDER을(를) 사용하여 순서대로 남아 있는 세그먼트를 검색할 수 있습니다.






세그먼트를 넣는 애플리케이션도 메시지 디스크립터에 있는 기타 필드를 세그먼트 간에 서로 다른 값으로 설정할 수 있습니다. 그러나 수신 애플리케이션이 MQGMO\_COMPLETE\_MSG을(를) 사용하여 논리 메시지를 검색하는 경우에는 이렇게 하는 것이 이점이 없습니다. 큐 관리자는 논리 메시지를 리어셈블링할 때 메시지 디스크립터에서 첫 번째 세그먼트에 대한 메시지 디스크립터의 값을 리턴합니다. 유일한 예외는 MsgFlags 필드이며, 이는 큐 관리자가 리어셈블링된 메시지가 유일한 세그먼트임을 표시하기 위해 설정합니다.

MQGMO\_COMPLETE\_MSG이(가) 보고 메시지에 지정된 경우 큐 관리자는 특수 처리를 수행합니다. 큐 관리자는 큐를 검사하여 논리 메시지에 있는 서로 다른 세그먼트와 관련된 해당 보고 유형의 모든 보고 메시지가 큐에 있는지 확인합니다. 있는 경우 MQGMO\_COMPLETE\_MSG을(를) 지정하여 단일 메시지로 검색할 수 있습니다. 이를 가능하게 하려면 세그먼트화를 지원하는 큐 관리자 또는 MCA가 보고 메시지를 생성해야 합니다. 그렇지 않으면 원래 애플리케이션이 최소한 100바이트의 메시지 데이터를 요청해야 합니다(즉, 적절한 MQRO\_\*\_WITH\_DATA 또는 MQRO\_\*\_WITH\_FULL\_DATA 옵션을 지정해야 함). 세그먼트에 대해 전체 애플리케이션 데이터보다 적은 양이 존재하는 경우 누락된 바이트는 리턴된 보고 메시지에서 널로 대체됩니다.

MQGMO\_COMPLETE\_MSG이(가) MQGMO\_MSG\_UNDER\_CURSOR 또는 MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR(으)로 지정된 경우, 찾아보기 커서는 MQMD의 Offset 필드에 0값이 있는 메시지에 위치해야 합니다. 이 조건이 충족되지 않으면 호출이 실패하며 이유 코드는 MQRC\_INVALID\_MSG\_UNDER\_CURSOR입니다.

MQGMO\_COMPLETE\_MSG은(는) 지정할 필요가 없는 MQGMO\_ALL\_SEGMENTS\_AVAILABLE을(를) 의미합니다.

MQGMO\_COMPLETE\_MSG은(는) MQGMO\_SYNCPOINT\_IF\_PERSISTENT 이외의 다른 MQGMO\_\* 옵션과 MQMO\_MATCH\_OFFSET 이외의 MQMO\_\* 옵션과 함께 지정될 수 있습니다.

-  z/OS에서 이 옵션은 개인 큐 및 공유 큐에 대해 지원되지만 큐에는 MQIT\_GROUP\_ID의 색인 유형이 있어야 합니다. 공유 큐의 경우 큐가 맵핑하는 CFSTRUCT 오브젝트는 CFLEVEL(3) 이상에 있어야 합니다.
- 다음 플랫폼에서:
  -  AIX
  -  IBM i
  -  Linux
  -  Windows

및 당 시스템에 연결된 IBM MQ MQI clients. 이 옵션은 모든 로컬 큐에 대해 지원됩니다.

### MQGMO\_ALL\_MSGS\_AVAILABLE

그룹의 메시지는 그룹에 있는 모든 메시지가 사용 가능할 때만 검색이 가능합니다. 큐에 일부 메시지가 누락된 메시지 그룹이 포함된 경우(네트워크에서 지연되어 아직 도달하지 않았기 때문일 수 있음), MQGMO\_ALL\_MSGS\_AVAILABLE을(를) 지정하면 완료되지 않은 그룹에 속하는 메시지의 검색을 방지할 수 있습니다. 그러나 이러한 메시지는 여전히 **CurrentQDepth** 큐 속성의 값에 기여합니다. 이는 CurrentQDepth이(가) 0보다 큰 경우에도 검색 가능한 메시지 그룹이 없을 수 있음을 의미합니다. 검색할 수 있는 다른 메시지가 없는 경우, 지정된 대기 간격(있는 경우)이 만료된 후 이유 코드 MQRC\_NO\_MSG\_AVAILABLE이(가) 리턴됩니다.

MQGMO\_ALL\_MSGS\_AVAILABLE의 처리는 MQGMO\_LOGICAL\_ORDER도 지정되었는지 여부에 따라 다릅니다.




- 두 옵션이 모두 지정된 경우, MQGMO\_ALL\_MSGS\_AVAILABLE은(는) 현재 그룹 또는 논리 메시지가 없는 경우에만 적용됩니다. 현재 그룹 또는 논리 메시지가 있는 경우 MQGMO\_ALL\_MSGS\_AVAILABLE은(는) 무시됩니다. 이는 메시지를 논리적 순서로 처리할 때 MQGMO\_ALL\_MSGS\_AVAILABLE이(가) 계속 유지될 수 있음을 의미합니다.
- MQGMO\_ALL\_MSGS\_AVAILABLE이(가) MQGMO\_LOGICAL\_ORDER 없이 지정된 경우 MQGMO\_ALL\_MSGS\_AVAILABLE은(는) 항상 영향을 받습니다. 이는 그룹의 나머지 메시지를 제거할 수 있도록 하려면 그룹의 첫 번째 메시지를 큐에서 제거한 후에 이 옵션을 꺼야 하는 것을 의미합니다.

MQGMO\_ALL\_MSGS\_AVAILABLE을 지정하여 MQGET 호출을 성공적으로 완료하는 것은 MQGET 호출이 발행되었을 때 그룹의 모든 메시지가 큐에 있었음을 의미합니다. 그러나 다른 애플리케이션은 여전히 그룹에서 메시지를 제거할 수 있다는 점을 유념하십시오(그룹은 해당 그룹의 첫 번째 메시지를 검색하는 애플리케이션에 대해 잠기지 않음).

이 옵션을 생략하면 그룹이 불완전한 때에도 그룹에 속한 메시지를 검색할 수 있습니다.

MQGMO\_ALL\_MSGS\_AVAILABLE은(는) 지정할 필요가 없는 MQGMO\_ALL\_SEGMENTS\_AVAILABLE을(를) 의미합니다.

MQGMO\_ALL\_MSGS\_AVAILABLE은(는) 다른 MQGMO\_\* 옵션과 MQMO\_\* 옵션 중 하나로 지정할 수 있습니다.

-  z/OS에서 이 옵션은 개인 큐 및 공유 큐에 대해 지원되지만 큐에는 MQIT\_GROUP\_ID의 색인 유형이 있어야 합니다. 공유 큐의 경우 큐가 맵핑하는 CFSTRUCT 오브젝트는 CFLEVEL(3) 이상에 있어야 합니다.
- 다음 플랫폼에서:
  -  AIX
  -  IBM i

-  Linux
-  Windows

및 당 시스템에 연결된 IBM MQ MQI clients. 이 옵션은 모든 로컬 큐에 대해 지원됩니다.

### MQGMO\_ALL\_SEGMENTS\_AVAILABLE

논리 메시지의 세그먼트는 논리 메시지에 있는 모든 세그먼트가 사용 가능할 때만 검색이 가능합니다. 큐에 일부 세그먼트가 누락되어 세그먼트화된 메시지가 포함된 경우(네트워크에서 지연되어 아직 도달하지 않았기 때문일 수 있음), MQGMO\_ALL\_SEGMENTS\_AVAILABLE을(를) 지정하면 불완전한 논리 메시지에 속하는 세그먼트의 검색을 방지할 수 있습니다. 그러나 해당 세그먼트는 여전히 **CurrentQDepth** 큐 속성의 값에 기여합니다. 이는 CurrentQDepth이(가) 0보다 큰 경우에도 검색 가능한 논리 메시지가 없을 수 있음을 의미합니다. 검색할 수 있는 다른 메시지가 없는 경우, 지정된 대기 간격(있는 경우)이 만료된 후 이유 코드 MQRC\_NO\_MSG\_AVAILABLE이(가) 리턴됩니다.

MQGMO\_ALL\_SEGMENTS\_AVAILABLE의 처리는 MQGMO\_LOGICAL\_ORDER도 지정되었는지 여부에 따라 다릅니다.

- 두 옵션을 모두 지정하면 현재 논리 메시지가 없는 경우에만 MQGMO\_ALL\_SEGMENTS\_AVAILABLE이(가) 적용됩니다. 현재 논리 메시지가 있는 경우 MQGMO\_ALL\_SEGMENTS\_AVAILABLE이(가) 무시됩니다. 이는 메시지를 논리적 순서로 처리할 때 MQGMO\_ALL\_SEGMENTS\_AVAILABLE이(가) 계속 유지될 수 있음을 의미합니다.
- MQGMO\_ALL\_SEGMENTS\_AVAILABLE이(가) MQGMO\_LOGICAL\_ORDER 없이 지정된 경우 MQGMO\_ALL\_SEGMENTS\_AVAILABLE은(는) 항상 영향을 받습니다. 이는 논리 메시지의 나머지 세그먼트를 제거할 수 있도록 하려면 논리 메시지의 첫 번째 세그먼트를 큐에서 제거한 후에 이 옵션을 꺼야 하는 것을 의미합니다.

이 옵션이 지정되지 않은 경우 논리 메시지가 불완전한 때에도 메시지 세그먼트가 검색될 수 있습니다.

MQGMO\_COMPLETE\_MSG 및 MQGMO\_ALL\_SEGMENTS\_AVAILABLE 모두를 검색하기 전에 모든 세그먼트를 사용할 수 있어야 하는 반면, 전자는 전체 메시지를 리턴하는 반면, 후자는 세그먼트를 하나씩 검색할 수 있습니다.

MQGMO\_ALL\_SEGMENTS\_AVAILABLE이(가) 보고서 메시지에 지정된 경우 큐 관리자는 큐를 검사하여 전체 논리 메시지를 구성하는 각 세그먼트에 대해 하나 이상의 보고 메시지가 있는지 확인합니다. 있는 경우 MQGMO\_ALL\_SEGMENTS\_AVAILABLE 조건이 충족됩니다. 그러나 큐 관리자는 존재하는 보고 메시지의 유형을 검사하지 않으므로 논리 메시지의 세그먼트와 관련된 보고 메시지에서 보고 유형이 혼합될 수 있습니다. 결과적으로 MQGMO\_ALL\_SEGMENTS\_AVAILABLE의 성공은 MQGMO\_COMPLETE\_MSG이(가) 성공할 것임을 의미하지는 않습니다. 특정 논리 메시지의 세그먼트에 대해 혼합된 보고 유형이 있는 경우 이러한 보고 메시지는 하나씩 검색되어야 합니다.

다른 MQGMO\_\* 옵션과 MQMO\_\* 옵션을 사용하여 MQGMO\_ALL\_SEGMENTS\_AVAILABLE을(를) 지정할 수 있습니다.

- z/OS에서 이 옵션은 개인 큐 및 공유 큐에 대해 지원되지만 큐에는 MQIT\_GROUP\_ID의 색인 유형이 있어야 합니다. 공유 큐의 경우 큐가 맵핑하는 CFSTRUCT 오브젝트는 CFLEVEL(3) 이상에 있어야 합니다.
- 다음 플랫폼에서:

-  AIX
-  IBM i
-  Linux
-  Windows

및 당 시스템에 연결된 IBM MQ MQI clients. 이 옵션은 모든 로컬 큐에 대해 지원됩니다.

### 특성 옵션

다음 옵션은 메시지의 특성과 관련되어 있습니다.

## MQGMO\_PROPERTIES\_AS\_Q\_DEF

메시지 디스크립터(또는 확장)에 포함된 메시지를 제외한 메시지의 특성은 **PropertyControl** 큐 속성에 의해 정의된 대로 표시되어야 합니다. **MsgHandle**이(가) 제공되면 **PropertyControl** 큐 속성의 값이 **MQPROP\_FORCE\_MQRFH2**이(가) 아닌 한 **MsgHandle**을(를) 통해 메시지의 특성을 사용할 수 있습니다.

이는 특성 옵션이 지정되지 않은 경우 기본 조치입니다.

## MQGMO\_PROPERTIES\_IN\_HANDLE

메시지 특성은 **MsgHandle**을(를) 통해 사용 가능해야 합니다. 메시지 핸들이 제공되지 않으면 **MQRC\_HMSG\_ERROR** 이유로 호출이 실패합니다.

**참고:** 메시지 핸들을 작성하지 않는 애플리케이션이 나중에 메시지를 읽는 경우, 큐 관리자는 메시지 특성을 **MQRFH2** 구조에 배치합니다. 예기치 않은 **MQRFH2** 헤더가 존재하면 기존 애플리케이션의 동작을 방해할 수 있습니다.

## MQGMO\_NO\_PROPERTIES

메시지 디스크립터(또는 확장자)에 포함된 특성을 제외한 메시지의 특성이 검색되지 않습니다. **MsgHandle**이(가) 제공되면 무시됩니다.

## MQGMO\_PROPERTIES\_FORCE\_MQRFH2

메시지 디스크립터(또는 확장)에 포함된 특성을 제외한 메시지 특성은 **MQRFH2** 헤더를 사용하여 표시해야 합니다. 이는 특성을 검색하려고 하지만 메시지 핸들을 사용하도록 변경될 수 없는 애플리케이션에 이전 버전과의 호환성을 제공합니다. **MsgHandle**이(가) 제공되면 무시됩니다.

## MQGMO\_PROPERTIES\_COMPATIBILITY

메시지에 접두부가 "**mcd.**", "**jms.**", "**usr.**" 또는 "**mqext.**"인 특성이 포함된 경우 모든 메시지 특성이 **MQRFH2** 헤더의 애플리케이션으로 전달됩니다. 그렇지 않은 경우에는 메시지 디스크립터(또는 확장자)의 특성을 제외한 메시지의 모든 특성이 제거되며 애플리케이션에 더 이상 액세스할 수 없습니다.

## 기본 옵션

설명된 옵션 중 필요한 사항이 없는 경우 다음 옵션을 사용할 수 있습니다.

### MQGMO\_NONE

기타 옵션이 지정되지 않았음을 표시하려면 이 값을 사용하십시오. 모든 옵션은 기본 값을 가정합니다.

**MQGMO\_NONE**은(는) 프로그램 문서화를 지원합니다. 이 옵션은 다른 옵션과 함께 사용할 수 없지만, 값이 0이므로 이러한 사용을 감지할 수 없습니다.

Options 필드의 초기값은 **MQGMO\_NO\_WAIT** 더하기 **MQGMO\_PROPERTIES\_AS\_Q\_DEF**입니다.

## MQGMO의 경우 WaitInterval (MQLONG)

**MQGET** 호출이 적합한 메시지가 도착하기를 기다리는 대략적인 시간(밀리초)입니다(즉 **MQGET** 호출의 **MsgDesc** 매개변수에 지정된 선택 기준을 충족하는 메시지).

**중요사항:** 적당한 메시지를 즉시 사용할 수 있는 경우 대기 또는 지연이 없습니다.

자세한 정보는 412 페이지의 『MQMD - 메시지 디스크립터』에 설명된 **MsgId** 필드를 참조하십시오. 이 시간이 경과된 후 적당한 메시지가 도달하지 않은 경우 **MQCC\_FAILED** 및 이유 코드 **MQRC\_NO\_MSG\_AVAILABLE** 과 함께 호출이 완료됩니다.

**z/OS**에서 **MQGET** 호출이 실제로 대기하는 시간은 시스템 로딩 및 시스템 스케줄링 고려사항에 영향을 받으며 **WaitInterval**에 대해 지정된 값 및 **WaitInterval**보다 큰 약 100밀리초 간에 다를 수 있습니다.

**WaitInterval**은 **MQGMO\_WAIT** 또는 **MQGMO\_SET\_SIGNAL** 옵션과 함께 사용됩니다. 이는 지정된 사항이 없는 경우 무시됩니다. 하나가 지정되면 **WaitInterval**은 0 이상이거나 다음 특수 값이어야 합니다.

### MQWI\_UNLIMITED

무제한 대기 간격.

이 필드의 초기값은 0입니다.

## MQGMO의 경우 *Signal1 (MQLONG)*

이는 오직 MQGMO\_SET\_SIGNAL 옵션과 함께 사용되는 입력 필드입니다. 메시지가 사용 가능할 때 전달되는 신호를 식별합니다.

**참고:** 이 필드의 데이터 유형과 사용법은 환경에 의해 판별됩니다. 이러한 이유로, 다른 환경 사이에 이식하려는 애플리케이션은 신호를 사용하지 않아야 합니다.

- z/OS에서 이 필드는 이벤트 제어 블록(ECB)의 주소를 포함해야 합니다. MQGET 호출이 발생되기 전에 애플리케이션이 ECB를 지워야 합니다. 큐가 처리완료될 때까지 ECB를 포함하는 스토리지는 사용 가능하게 되지 않아야 합니다. ECB은 설명된 신호 완료 코드 중 하나와 함께 큐 관리자에 의해 게시됩니다. 이러한 완료 코드는 사용자 완료 코드를 위한 것으로서 z/OS 맵핑 매크로 IHAECB에서 정의된 영역인 ECB의 비트 2 ~ 31에 설정됩니다.
- 다른 모든 환경에서 이는 예약 필드입니다. 해당 값은 중요하지 않습니다.

신호 완료 코드는 다음과 같습니다.

### MQEC\_MSG\_ARRIVED

적당한 메시지가 큐에 도착했습니다. 이 메시지는 호출자에 대해 예약되지 않았습니다. 두 번째 MQGET 요청이 발생되어야 하지만 두 번째 요청이 발생되기 전에 다른 애플리케이션이 해당 메시지를 검색했을 수 있습니다.

### MQEC\_WAIT\_INTERVAL\_EXPIRED

적당한 메시지가 도착하지 않고 지정된 *WaitInterval*이 만료되었습니다.

### MQEC\_WAIT\_CANCELED

불확실한 이유로 대기가 취소되었습니다(예: 큐 관리자가 종료했거나 큐가 사용 안함으로 설정됨). 추가 분석이 필요한 경우 요청을 다시 실행하십시오.

### MQEC\_Q\_MGR QUIESCING

큐 관리자가 정지 상태에 진입했기 때문에 대기가 취소되었습니다(MQGMO\_FAIL\_IF\_QUIESCING이 MQGET 호출에 지정됨).

### MQEC\_CONNECTION\_QUIESCING

연결이 정지 상태에 진입했기 때문에 대기가 취소되었습니다(MQGMO\_FAIL\_IF\_QUIESCING이 MQGET 호출에 지정됨).

이 필드의 초기값은 환경에 의해 판별됩니다.

- z/OS에서 초기값은 널 포인터입니다.
- 다른 모든 환경에서 초기값은 0입니다.

## MQGMO의 경우 *Signal2 (MQLONG)*

이는 MQGMO\_SET\_SIGNAL 옵션과 함께 사용되는 입력 필드입니다. 이는 예약된 필드입니다. 해당 값은 중요하지 않습니다.

이 필드의 초기값은 0입니다.

## MQGMO의 경우 *ResolvedQName (MQCHAR48)*

이는 큐 관리자가 로컬 큐 관리자에 정의된 대로 메시지가 검색된 큐의 로컬 이름으로 설정하는 출력 필드입니다. 이는 큐를 여는 데 사용된 이름과 다릅니다.

- 알리어스 큐가 열렸음(이 경우 알리어스가 해석된 로컬 큐의 이름이 리턴됨), 또는
- 모델 큐가 열렸음(이 경우 동적 로컬 큐의 이름이 리턴됨)

이 필드의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백입니다.

## MQGMO의 경우 *MatchOptions (MQLONG)*

이러한 옵션은 애플리케이션이 MQGET 호출로 리턴된 메시지를 선택하는 데 사용하기 위해 **MsgDesc** 매개변수에서 필드를 선택하도록 허용합니다. 애플리케이션은 이 필드에 필수 옵션을 설정한 후 **MsgDesc** 매개변수의 해당 필드를 해당 필드에 필요한 값으로 설정합니다. 메시지의 MQMD에 해당 값이 있는 메시지만 MQGET 호출에

서 해당 **MsgDesc** 매개변수를 사용하는 검색 후보입니다. 리턴할 메시지를 선택할 때 해당하는 일치 옵션이 지정되지 않는 필드는 무시됩니다. MQGET 호출에서 선택 기준을 지정하지 않는 경우(즉, 임의 메시지가 허용 가능함) *MatchOptions*를 MQMO\_NONE으로 설정하십시오.

- z/OS에서 사용할 수 있는 선택 기준은 큐에 사용된 색인 유형으로 제한될 수 있습니다. 추가 세부사항은 **IndexType** 큐 속성을 참조하십시오.

MQGMO\_LOGICAL\_ORDER를 지정하는 경우 특정 메시지만 다음 MQGET 호출로 리턴할 수 있습니다.

- 현재 그룹 또는 논리 메시지가 없는 경우 *MsgSeqNumber*이(가) 1이고 *Offset*이(가) 0인 메시지만 리턴할 수 있습니다. 이 상황에서 다음 일치 옵션 중 하나 이상을 사용하여 리턴되는 적합한 메시지를 선택할 수 있습니다.

- MQMO\_MATCH\_MSG\_ID
- MQMO\_MATCH\_CORREL\_ID
- MQMO\_MATCH\_GROUP\_ID

- 현재 그룹 또는 논리 메시지가 있는 경우 그룹의 다음 메시지 또는 논리 메시지의 다음 세그먼트만 리턴할 수 있으며, 이는 MQMO\_\* 옵션을 지정하여 대체할 수 없습니다.

앞의 두 경우에서 적용되지 않는 일치 옵션을 지정할 수 있지만 **MsgDesc** 매개변수의 관련 필드 값이 리턴할 메시지의 해당 필드 값과 일치해야 합니다. 이 조건이 충족되지 않으면 이유 코드 MQRC\_MATCH\_OPTIONS\_ERROR와 함께 호출이 실패합니다.

MQGMO\_MSG\_UNDER\_CURSOR 또는 MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR를 지정하는 경우 *MatchOptions*가 무시됩니다.

메시지 특성을 기반으로 한 메시지 가져오기는 일치 옵션을 사용하여 수행되지 않습니다. 자세한 정보는 478 페이지의 『MQOD의 경우 SelectionString (MQCHARV)』의 내용을 참조하십시오.

다음 일치 옵션 중 하나 이상을 지정할 수 있습니다.

#### **MQMO\_MATCH\_MSG\_ID**

검색할 메시지에는 MQGET 호출의 **MsgDesc** 매개변수에 있는 *MsgId* 필드의 값과 일치하는 메시지 ID가 있어야 합니다. 이 일치는 적용할 수 있는 기타 일치에 추가됩니다(예: 상관 ID).

이 옵션을 생략하면 **MsgDesc** 매개변수의 *MsgId* 필드가 무시되고 모든 메시지 ID가 일치합니다.

**참고:** 메시지 ID MQMI\_NONE은 메시지에 대한 MQMD의 메시지 ID와도 일치하는 특수 값입니다. 따라서 MQMI\_NONE과 함께 MQMO\_MATCH\_MSG\_ID를 지정하는 것은 MQMO\_MATCH\_MSG\_ID를 지정하지 않는 것과 동일합니다.

#### **MQMO\_MATCH\_CORREL\_ID**

검색할 메시지에는 MQGET 호출의 **MsgDesc** 매개변수에 있는 *CorrelId* 필드의 값과 일치하는 상관 ID가 있어야 합니다. 이 일치는 적용할 수 있는 기타 일치에 추가됩니다(예: 메시지 ID).

이 옵션을 생략하면 **MsgDesc** 매개변수의 *CorrelId* 필드가 무시되고 상관 ID가 일치합니다.

**참고:** 상관 ID MQCI\_NONE은 메시지의 MQMD에서 임의 상관 ID와 일치하는 특수 값입니다. 따라서 MQCI\_NONE과 함께 MQMO\_MATCH\_CORREL\_ID를 지정하는 것은 MQMO\_MATCH\_CORREL\_ID를 지정하지 않는 것과 동일합니다.

#### **MQMO\_MATCH\_GROUP\_ID**

검색할 메시지에는 MQGET 호출의 **MsgDesc** 매개변수에 있는 *GroupId* 필드의 값과 일치하는 그룹 ID가 있어야 합니다. 이 일치는 적용할 수 있는 기타 일치에 추가됩니다(예: 상관 ID).

이 옵션을 생략하면 **MsgDesc** 매개변수의 *GroupId* 필드가 무시되고 모든 그룹 ID가 일치합니다.

**참고:** 그룹 ID MQGI\_NONE은 메시지의 MQMD에서 임의 그룹 ID와 일치하는 특수 값입니다. 따라서 MQGI\_NONE과 함께 MQMO\_MATCH\_GROUP\_ID를 지정하는 것은 MQMO\_MATCH\_GROUP\_ID를 지정하지 않는 것과 동일합니다.

#### **MQMO\_MATCH\_MSG\_SEQ\_NUMBER**

검색할 메시지에는 MQGET 호출의 **MsgDesc** 매개변수에 있는 *MsgSeqNumber* 필드의 값과 일치하는 메시지 순서 번호가 있어야 합니다. 이 일치는 적용할 수 있는 기타 일치에 추가됩니다(예: 그룹 ID).



이 옵션을 생략하면 **MsgDesc** 매개변수의 *MsgSeqNumber* 필드가 무시되고 모든 메시지 순서 번호가 일치합니다.

### **MQMO\_MATCH\_OFFSET**

검색할 메시지에는 MQGET 호출의 **MsgDesc** 매개변수에 있는 *Offset* 필드의 값과 일치하는 오프셋이 있어야 합니다. 이 일치는 적용할 수 있는 기타 일치에 추가됩니다(예: 메시지 순서 번호).

이 옵션을 생략하면 **MsgDesc** 매개변수의 *Offset* 필드가 무시되고 모든 오프셋이 일치합니다.

- 이 옵션은 z/OS에서 지원되지 않습니다.

### **MQMO\_MATCH\_MSG\_TOKEN**

검색할 메시지에는 MQGET 호출에서 지정된 MQGMO 구조에서 *MsgToken* 필드의 값과 일치하는 메시지 토큰이 있어야 합니다.

모든 로컬 큐에 대해 이 옵션을 지정할 수 있습니다. MQIT\_MSG\_TOKEN(WLM 관리되는 큐)의 *IndexType* 이 있는 큐에 대해 이를 지정하는 경우, MQMO\_MATCH\_MSG\_TOKEN과 함께 다른 일치 옵션을 지정할 수 없습니다.

MQGMO\_WAIT 또는 MQGMO\_SET\_SIGNAL과 함께 MQMO\_MATCH\_MSG\_TOKEN을 지정할 수 없습니다. 애플리케이션이 MQIT\_MSG\_TOKEN의 *IndexType*이 있는 큐에 메시지가 도착할 때까지 대기하려는 경우 MQMO\_NONE을 지정하십시오.

이 옵션을 생략하면 MQGMO의 *MsgToken* 필드가 무시되며 메시지 토큰이 일치합니다.

설명된 옵션을 지정하지 않는 경우 다음 옵션을 사용할 수 있습니다.

### **MQMO\_NONE**

리턴될 메시지를 선택할 때 일치 안함을 사용합니다. 큐의 모든 메시지는 검색 가능합니다(그러나 MQGMO\_ALL\_MSGS\_AVAILABLE, MQGMO\_ALL\_SEGMENTS\_AVAILABLE 및 MQGMO\_COMPLETE\_MSG 옵션의 제어에 따라 달라짐).

MQMO\_NONE은 프로그램 문서화를 지원합니다. 이 옵션은 다른 MQMO\_\* 옵션과 함께 사용되기 위한 용도는 아니지만 값이 0이므로 그러한 사용을 감지할 수 없습니다.

입력 필드입니다. 이 필드의 초기값은 MQMO\_MATCH\_CORREL\_ID가 있는 MQMO\_MATCH\_MSG\_ID입니다. *Version*이(가) MQGMO\_VERSION\_2보다 작으면 이 필드는 무시됩니다.

**참고:** *MatchOptions* 필드의 초기값이 이전 MQSeries® 큐 관리자와의 호환성을 위해 정의됩니다. 그러나 선택 기준을 사용하지 않고 큐에서 일련의 메시지를 읽을 때 이 초기값을 위해 애플리케이션이 각 MQGET 호출 전에 *MsgId* 및 *CorrelId* 필드를 MQMI\_NONE 및 MQCI\_NONE으로 재설정해야 합니다. *Version*을 MQGMO\_VERSION\_2로 설정하고 *MatchOptions*를 MQMO\_NONE으로 설정하여 *MsgId* 및 *CorrelId*를 재설정하지 않을 수 있습니다.

### **관련 개념**

[JMS의 메시지 선택자](#)

### **MQGMO의 경우 GroupStatus (MQCHAR)**

이 플래그는 검색된 메시지가 그룹에 있는지 표시합니다.

값은 다음 중 하나입니다.

#### **MQGS\_NOT\_IN\_GROUP**

메시지는 그룹에 없습니다.

#### **MQGS\_MSG\_IN\_GROUP**

메시지가 그룹에 있지만 그룹의 마지막이 아닙니다.

#### **MQGS\_LAST\_MSG\_IN\_GROUP**

메시지가 그룹의 마지막에 있습니다.

또한 이는 그룹이 하나의 메시지로만 구성된 경우 리턴된 값입니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 MQGS\_NOT\_IN\_GROUP입니다. *Version*이(가) MQGMO\_VERSION\_2보다 작으면 이 필드는 무시됩니다.

## **MQGMO의 경우 SegmentStatus (MQCHAR)**

이는 검색된 메시지가 논리 메시지의 세그먼트인지 여부를 표시하는 플래그입니다. 값은 다음 중 하나입니다.

### **MQSS\_NOT\_A\_SEGMENT**

메시지가 세그먼트가 아닙니다.

### **MQSS\_SEGMENT**

메시지가 세그먼트이지만 논리 메시지의 마지막 세그먼트가 아닙니다.

### **MQSS\_LAST\_SEGMENT**

메시지는 논리적 메시지의 마지막 세그먼트입니다.

또한 이는 논리적 메시지가 하나의 세그먼트로만 구성된 경우에 리턴되는 값입니다.

z/OS에서 큐 관리자는 항상 이 필드를 MQSS\_NOT\_A\_SEGMENT로 설정합니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 MQSS\_NOT\_A\_SEGMENT입니다. *Version*이(가) MQGMO\_VERSION\_2보다 작으면 이 필드는 무시됩니다.

## **MQGMO의 세그먼트화 (MQCHAR)**

이는 검색된 메시지에 대해 추가 세그먼트화가 허용되는지 여부를 표시하는 플래그입니다. 값은 다음 중 하나입니다.

### **MQSEG\_INHIBITED**

허용되지 않은 세그먼트화.

### **MQSEG\_ALLOWED**

세그먼트화가 허용됩니다.

z/OS에서 큐 관리자는 항상 이 필드를 MQSEG\_INHIBITED로 설정합니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 MQSEG\_INHIBITED입니다. *Version*이(가) MQGMO\_VERSION\_2보다 작으면 이 필드는 무시됩니다.

## **MQGMO의 경우 Reserved1 (MQCHAR)**

이 필드는 예약된 필드입니다. 이 필드의 초기값은 공백 문자입니다. *Version*이(가) MQGMO\_VERSION\_2보다 작으면 이 필드는 무시됩니다.

## **MQGMO의 경우 MsgToken (MQBYTE16)**

MsgToken 필드 - MQGMO 구조입니다. 이 필드는 메시지를 고유하게 식별하기 위해 큐 관리자가 사용합니다.

이는 큐에서 메시지를 고유하게 식별하기 위해 큐 관리자가 생성하는 바이트 문자열입니다. 메시지 토큰은 메시지가 큐 관리자에 처음 배치될 때 생성되며, 큐 관리자가 재시작하는 경우를 제외하고 메시지가 영구적으로 큐 관리자에서 제거될 때까지 메시지와 함께 지속됩니다.

메시지가 큐에서 제거되면 메시지의 해당 인스턴스를 식별하는 *MsgToken*이 더 이상 올바르지 않아 재사용되지 않습니다. 큐 관리자가 재시작되면 재시작 전에 큐에서 메시지를 식별한 *MsgToken*이 재시작 후에 올바르지 않을 수 있습니다. 그러나 다른 메시지 인스턴스를 식별하기 위해 *MsgToken*을 재사용하지 않습니다. *MsgToken*은 큐 관리자가 생성하며 외부 애플리케이션에 표시되지 않습니다.

메시지가 버전 3 이상의 MQGMO가 제공되는 MQGET으로의 호출로 리턴될 때 큐에서 메시지를 식별하는 *MsgToken*은 큐 관리자에 의해 MQGMO에서 리턴됩니다. 이에 대한 하나의 예외가 있습니다. 메시지가 동기점 외부의 큐에서 제거되면 후속 MQGET 호출에서 리턴된 메시지를 식별하는 데 유용하지 않으므로 큐 관리자가 *MsgToken*을 리턴하지 않을 수 있습니다. 애플리케이션은 *MsgToken*만 사용하여 후속 MQGET 호출의 메시지를 참조해야 합니다.

*MsgToken*이 제공되고 *MatchOption* MQMO\_MATCH\_MSG\_TOKEN이 지정되며 MQGMO\_MSG\_UNDER\_CURSOR 및 MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR가 지정되지 않은 경우 해당 *MsgToken*이 식별한 메시지만 리턴할 수 있습니다. 옵션은 INDXTYPE에 상관없이 모든 로컬 큐에 올바르며 z/OS에서는 워크로드 관리자(WLM) 큐에서만 INDXTYPE(MSGTOKEN)을 사용해야 합니다.



지정된 기타 *MatchOptions*를 확인하고 일치하지 않는 경우 MQRC\_NO\_MSG\_AVAILABLE이 리턴됩니다. MQGMO\_BROWSE\_NEXT가 MQMO\_MATCH\_MSG\_TOKEN과 함께 코딩되면 호출 핸들의 찾아보기 커서 너머에 있는 경우에만 *MsgToken*이 식별한 메시지가 리턴됩니다.

MQGMO\_MSG\_UNDER\_CURSOR 또는 MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR가 지정되면 MQMO\_MATCH\_MSG\_TOKEN은 무시됩니다.

MQMO\_MATCH\_MSG\_TOKEN은 다음 가져오기 메시지 옵션과 함께 사용하면 올바르지 않습니다.

- MQGMO\_WAIT
- MQGMO\_SET\_SIGNAL

MQMO\_MATCH\_MSG\_TOKEN을 지정하는 MQGET 호출의 경우 버전 3 이상의 MQGMO가 호출에 제공되어야 하며 그렇지 않으면 MQRC\_WRONG\_GMO\_VERSION이 리턴됩니다.

*MsgToken*이 현재 올바르지 않은 경우 다른 오류가 없으면 MQRC\_NO\_MSG\_AVAILABLE과 함께 MQCC\_FAILED가 리턴됩니다.

### **MQGMO의 경우 ReturnedLength (MQLONG)**

이는 큐 관리자가 **Buffer** 매개변수에서 MQGET 호출로 리턴된 메시지 데이터의 길이(바이트)로 설정하는 출력 필드입니다. 큐 관리자가 이 기능을 지원하지 않는 경우 *ReturnedLength*는 값 MQRL\_UNDEFINED로 설정됩니다.

메시지가 인코딩 또는 문자 세트 간에 변환할 때 메시지 데이터는 종종 크기를 변경할 수 있습니다. MQGET 호출에서 리턴하는 경우:

- *ReturnedLength*가 MQRL\_UNDEFINED가 아닌 경우, 리턴된 메시지 데이터의 바이트 수는 *ReturnedLength*로 지정됩니다.
- *ReturnedLength*에 값 MQRL\_UNDEFINED가 있는 경우 리턴된 메시지 데이터의 바이트 수는 일반적으로 더 작은 *BufferLength* 및 *DataLength*로 지정되지만 이유 코드 MQRC\_TRUNCATED\_MSG\_ACCEPTED와 함께 MQGET 호출이 완료되는 경우 해당 값 미만일 수 있습니다. 이 경우, **Buffer** 매개변수의 중요하지 않은 바이트는 널로 설정됩니다.

다음 특수 값이 정의됩니다.

#### **MQRL\_UNDEFINED**

리턴된 데이터의 길이가 정의되지 않습니다.

z/OS에서 *ReturnedLength* 필드에 대해 리턴된 값은 항상 MQRL\_UNDEFINED입니다.

이 필드의 초기값은 MQRL\_UNDEFINED입니다. 이 필드는 *Version*이 MQGMO\_VERSION\_3 미만인 경우 무시됩니다.

### **MQGMO의 경우 Reserved2 (MQLONG)**

이 필드는 예약된 필드입니다. 이 필드의 초기값은 공백 문자입니다. 이 필드는 *Version*이 MQGMO\_VERSION\_4 미만인 경우 무시됩니다.

### **MQGMO의 경우 MsgHandle (MQHMSG)**

MQGMO\_PROPERTIES\_AS\_Q\_DEF 옵션이 지정되고 **PropertyControl** 큐 속성이 MQPROP\_FORCE\_MQRFH2 로 설정되지 않은 경우 큐에서 검색되는 메시지의 특성으로 채워질 메시지에 대한 핸들입니다. 핸들은 MQCRTMH 호출에 의해 작성됩니다. 핸들과 이미 연관된 특성은 메시지를 검색하기 전에 지워집니다.

다음 값을 지정할 수도 있습니다.

MQHM\_NONE

제공된 메시지 핸들이 없습니다.

유효한 메시지 핸들이 제공되고 메시지 특성, 입력 필드에 사용된 메시지 핸들과 연관된 메시지 디스크립터를 포함하기 위해 출력에 사용된 경우 메시지 디스크립터는 MQGET 호출에 필요합니다.

메시지 디스크립터가 MQGET 호출에 지정되면 항상 메시지 핸들과 연관된 메시지 디스크립터에 우선합니다.

MQGMO\_PROPERTIES\_FORCE\_MQRFH2 가 지정되거나 MQGMO\_PROPERTIES\_AS\_Q\_DEF가 지정되고 **PropertyControl** 큐 속성이 MQPROP\_FORCE\_MQRFH2 인 경우 메시지 디스크립터 매개변수가 지정되지 않으면 이유 코드 MQRC\_MD\_ERROR와 함께 호출이 실패합니다.

MQGET 호출에서 리턴 시 이 메시지 핸들과 연관된 특성 및 메시지 디스크립터는 검색된(하나의 MQGET 호출에 제공되는 경우 메시지 디스크립터 또한) 메시지의 상태를 반영하기 위해 업데이트됩니다. 그런 다음 메시지의 특성은 MQINQMP 호출을 사용하여 조회할 수 있습니다.

메시지 디스크립터 확장자를 제외하고(존재할 경우) MQINQMP 호출로 조회할 수 있는 특성은 메시지 데이터에 포함되지 않습니다. 메시지 데이터의 특성에 포함된 큐의 메시지는 데이터가 애플리케이션으로 리턴되기 전에 메시지 데이터에서 제거됩니다.

메시지 핸들이 제공되거나 버전이 MQGMO\_VERSION\_4 미만인 경우 MQGET 호출에서 올바른 메시지 디스크립터를 제공해야 합니다. 모든 메시지 특성 (메시지 디스크립터에 포함된 특성 제외) 은 MQGMO 구조 및 **PropertyControl** 큐 속성의 특성 옵션 값에 따라 메시지 데이터에 리턴됩니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQHM\_NONE입니다. **Version** 이 MQGMO\_VERSION\_4 미만인 경우 이 필드는 무시됩니다.

## MQIIH - IMS 정보 헤더

MQIIH 구조는 IMS 브릿지에서 IMS에 전송된 메시지에 대한 헤더 정보를 설명합니다. IBM MQ 지원 플랫폼의 경우 MQIIH 구조를 포함하는 메시지를 작성하고 전송할 수 있지만 IBM MQ for z/OS 큐 관리자만 IMS 브릿지를 사용할 수 있습니다. 따라서 메시지가 비z/OS 큐 관리자에서 IMS에 도달하려면 큐 관리자 네트워크에 메시지를 라우팅할 수 있는 하나 이상의 z/OS 큐 관리자가 포함되어야 합니다.

## 가용성

모든 IBM MQ 시스템 및 IBM MQ 클라이언트.

## 형식 이름

MQFMT\_IMS

## 문자 세트 및 인코딩

특수 조건은 MQIIH 구조 및 애플리케이션 메시지 데이터에 사용되는 문자 세트 및 인코딩에 적용됩니다.

- IMS 브릿지 큐를 소유하는 큐 관리자에 연결되는 애플리케이션은 큐 관리자의 문자 세트 및 인코딩에 있는 MQIIH 구조를 제공해야 합니다. 이는 MQIIH 구조의 데이터 변환이 이 경우에 수행되지 않기 때문입니다.
- 기타 큐 관리자에 연결하는 애플리케이션은 지원되는 문자 세트 및 인코딩에 있는 MQIIH 구조를 제공할 수 있습니다. IMS 브릿지 큐를 소유하는 큐 관리자에 연결된 수신 메시지 채널 에이전트는 MQIIH를 변환합니다.
- MQIIH 구조 뒤에 오는 애플리케이션 메시지 데이터는 MQIIH 구조와 동일한 문자 세트 및 인코딩에 있어야 합니다. 애플리케이션 메시지 데이터의 문자 세트 및 인코딩을 지정하기 위해 MQIIH 구조의 *CodedCharSetId* 및 *Encoding* 필드를 사용하지 마십시오.

데이터가 큐 관리자가 지원하는 내장 형식 중 하나가 아닌 경우 애플리케이션 메시지 데이터를 변환하기 위해 데이터 변환 엑시트를 제공해야 합니다.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 497. MQIIH에서 MQIIH의 필드

필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQIIH_STRUC_ID	'IIH~'
버전 (구조 버전 번호)	MQIIH_VERSION_1	1
StrucLength (MQIIH 구조의 길이)	MQIIH_LENGTH_1	84
인코딩 (예약됨 - 398 페이지의 『문자 세트 및 인코딩』 참조)	없음	0
CodedCharSetId (예약됨 - 398 페이지의 『문자 세트 및 인코딩』 참조)	없음	0
형식 (MQIIH 다음에 오는 데이터의 MQ 형식 이름)	MQFMT_NONE	공백
플래그 (플래그)	MQIIH_NONE	0
LTermOverride (논리 터미널 대체)	없음	공백
MFSMapName (메시지 형식 서비스 맵 이름)	없음	공백
ReplyTo 형식 (응답 메시지의 MQ 형식 이름)	MQFMT_NONE	공백
인증자 (RACF® 비밀번호 또는 패스티켓)	MQIAUT_NONE	공백
TranInstanceId (트랜잭션 인스턴스 ID)	MQITII_NONE	널
TranState (트랜잭션 상태)	MQITS_NOT_IN_CONVE RSATION	'~'
CommitMode (커미트 모드)	MQICM_COMMIT_THEN _SEND	'0'
SecurityScope (보안 범위)	MQISS_CHECK	'C'
예약됨 (예약됨)	없음	'~'

**참고:**

1. ~ 기호는 단일 공백 문자를 나타냅니다.
2. C 프로그래밍 언어의 매크로 변수 MQIIH\_DEFAULT는 테이블에 나열되는 값을 포함합니다. 즉, 구조 내의 필드에 대한 초기값을 제공하기 위해 다음과 같은 방법으로 사용될 수 있습니다.

```
MQIIH MyIIH = {MQIIH_DEFAULT};
```

**언어 선언**

MQIIH의 C 선언

```
typedef struct tagMQIIH MQIIH;
struct tagMQIIH {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;          /* Structure version number */
    MQLONG     StrucLength;      /* Length of MQIIH structure */
    MQLONG     Encoding;        /* Reserved */
    MQLONG     CodedCharSetId;  /* Reserved */
    MQCHAR8    Format;          /* MQ format name of data that follows
                               MQIIH */
    MQLONG     Flags;           /* Flags */
    MQCHAR8    LTermOverride;   /* Logical terminal override */
    MQCHAR8    MFSMapName;     /* Message format services map name */
    MQCHAR8    ReplyToFormat;  /* MQ format name of reply message */
    MQCHAR8    Authenticator;  /* RACF password or passticket */
    MQBYTE16   TranInstanceId; /* Transaction instance identifier */
};
```

```

MQCHAR    TranState;      /* Transaction state */
MQCHAR    CommitMode;     /* Commit mode */
MQCHAR    SecurityScope;  /* Security scope */
MQCHAR    Reserved;       /* Reserved */
};

```

## MQIIH의 COBOL 선언

```

** MQIIH structure
10 MQIIH.
** Structure identifier
15 MQIIH-STRUCID PIC X(4).
** Structure version number
15 MQIIH-VERSION PIC S9(9) BINARY.
** Length of MQIIH structure
15 MQIIH-STRUCLength PIC S9(9) BINARY.
** Reserved
15 MQIIH-ENCODING PIC S9(9) BINARY.
** Reserved
15 MQIIH-CODEDCHARSETID PIC S9(9) BINARY.
** MQ format name of data that follows MQIIH
15 MQIIH-FORMAT PIC X(8).
** Flags
15 MQIIH-FLAGS PIC S9(9) BINARY.
** Logical terminal override
15 MQIIH-LTERMOVERRIDE PIC X(8).
** Message format services map name
15 MQIIH-MFSMAPNAME PIC X(8).
** MQ format name of reply message
15 MQIIH-REPLYTOFORMAT PIC X(8).
** RACF password or passticket
15 MQIIH-AUTHENTICATOR PIC X(8).
** Transaction instance identifier
15 MQIIH-TRANINSTANCEID PIC X(16).
** Transaction state
15 MQIIH-TRANSTATE PIC X.
** Commit mode
15 MQIIH-COMMITMODE PIC X.
** Security scope
15 MQIIH-SECURITYSCOPE PIC X.
** Reserved
15 MQIIH-RESERVED PIC X.

```

## MQIIH의 PL/I 선언

```

dcl
1 MQIIH based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 StrucLength fixed bin(31), /* Length of MQIIH structure */
3 Encoding fixed bin(31), /* Reserved */
3 CodedCharSetId fixed bin(31), /* Reserved */
3 Format char(8), /* MQ format name of data that follows
MQIIH */
3 Flags fixed bin(31), /* Flags */
3 LTermOverride char(8), /* Logical terminal override */
3 MFSMapName char(8), /* Message format services map name */
3 ReplyToFormat char(8), /* MQ format name of reply message */
3 Authenticator char(8), /* RACF password or passticket */
3 TranInstanceId char(16), /* Transaction instance identifier */
3 TranState char(1), /* Transaction state */
3 CommitMode char(1), /* Commit mode */
3 SecurityScope char(1), /* Security scope */
3 Reserved char(1); /* Reserved */

```

## MQIIH의 상위 레벨 어셈블러 선언

```

MQIIH          DSECT
MQIIH_STRUCID  DS CL4  Structure identifier
MQIIH_VERSION  DS F    Structure version number
MQIIH_STRUCLength DS F    Length of MQIIH structure
MQIIH_ENCODING DS F    Reserved
MQIIH_CODEDCHARSETID DS F    Reserved
MQIIH_FORMAT   DS CL8  MQ format name of data that follows
*              MQIIH

```

MQIIH_FLAGS	DS	F	Flags
MQIIH_LTERM_OVERRIDE	DS	CL8	Logical terminal override
MQIIH_MFSMAPNAME	DS	CL8	Message format services map name
MQIIH_REPLYTOFORMAT	DS	CL8	MQ format name of reply message
MQIIH_AUTHENTICATOR	DS	CL8	RACF password or passticket
MQIIH_TRANINSTANCEID	DS	XL16	Transaction instance identifier
MQIIH_TRANSTATE	DS	CL1	Transaction state
MQIIH_COMMITMODE	DS	CL1	Commit mode
MQIIH_SECURITYSCOPE	DS	CL1	Security scope
MQIIH_RESERVED	DS	CL1	Reserved
*			
MQIIH_LENGTH	EQU	*-MQIIH	
	ORG	MQIIH	
MQIIH_AREA	DS	CL(MQIIH_LENGTH)	

## MQIIH의 Visual Basic 선언

```

Type MQIIH
  StrucId      As String*4 'Structure identifier'
  Version     As Long      'Structure version number'
  StrucLength  As Long      'Length of MQIIH structure'
  Encoding     As Long      'Reserved'
  CodedCharSetId As Long    'Reserved'
  Format       As String*8  'MQ format name of data that follows MQIIH'
  Flags       As Long      'Flags'
  LTermOverride As String*8 'Logical terminal override'
  MFSMapName  As String*8  'Message format services map name'
  ReplyToFormat As String*8 'MQ format name of reply message'
  Authenticator As String*8 'RACF password or passticket'
  TranInstanceID As MQBYTE16 'Transaction instance identifier'
  TranState    As String*1  'Transaction state'
  CommitMode   As String*1  'Commit mode'
  SecurityScope As String*1 'Security scope'
  Reserved     As String*1  'Reserved'
End Type

```

### MQIIH의 StrucId (MQCHAR4)

IMS 정보 헤더 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQIIH\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

#### MQIIH\_STRUC\_ID

IMS 정보 헤더 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQIIH\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQIIH\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### MQIIH의 버전 (MQLONG)

구조 버전 번호입니다. 값은 다음과 같아야 합니다.

#### MQIIH\_VERSION\_1

IMS 정보 헤더 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### MQIIH\_CURRENT\_VERSION

IMS 정보 헤더 구조의 현재 버전.

이 필드의 초기값은 MQIIH\_VERSION\_1입니다.

### MQIIH의 StrucLength (MQLONG)

이는 MQIIH 구조의 길이입니다. 값은 다음과 같아야 합니다.

#### MQIIH\_LENGTH\_1

IMS 정보 헤더 구조의 길이입니다.

이 필드의 초기값은 MQIIH\_LENGTH\_1입니다.

### MQIIH의 인코딩 (MQLONG)

이 필드는 예약 필드이며 값은 중요하지 않습니다. 이 필드의 초기값은 0입니다.

MQIIH 구조를 따르는 지원되는 구조에 대한 Encoding은 MQIIH 구조 자체의 Encoding과 동일하며 선행 MQ 헤더에서 가져옵니다.

### **MQIIH의 CodedCharSetId (MQLONG)**

이 필드는 예약 필드이며 값은 중요하지 않습니다. 이 필드의 초기값은 0입니다.

MQIIH 구조를 따르는 지원되는 구조에 대한 문자 세트 ID는 MQIIH 구조 자체의 문자 세트 ID와 동일하며 선행 MQ 헤더에서 가져옵니다.

### **MQIIH의 형식 (MQCHAR8)**

이는 MQIIH 구조 다음에 오는 데이터의 MQ 형식 이름을 지정합니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다.

이 필드의 길이는 MQ\_FORMAT\_LENGTH에 지정되어 있습니다. 이 필드의 초기값은 MQFMT\_NONE입니다.

### **MQIIH의 플래그 (MQLONG)**

플래그 값은 다음과 같아야 합니다.

#### **MQIIH\_NONE**

플래그가 없습니다.

#### **MQIIH\_PASS\_EXPIRATION**

응답 메시지는 다음을 포함합니다.

- 요청 메시지와 동일한 만기 보고서 옵션
- 브릿지의 처리 시간 동안 조정되지 않은 요청 메시지에서 남은 만기 시간

이 값을 설정하지 않으면 만기 시간이 무제한으로 설정됩니다.

#### **MQIIH\_REPLY\_FORMAT\_NONE**

응답의 MQIIH.Format 필드를 MQFMT\_NONE으로 설정합니다.

#### **MQIIH\_IGNORE\_PURG**

OTMA 접두부에서 TMAMIPRG 표시기를 설정합니다. 이는 CMO 트랜잭션에 대한 TP PCB에서 OTMA가 PURG 호출을 무시하도록 요청합니다.

#### **MQIIH\_CMO\_REQUEST\_RESPONSE**

커미트 모드 0(CMO) 트랜잭션의 경우 이 플래그는 OTMA 접두부에서 TMAMHRSP 지표를 설정합니다. 이 표시기를 설정하면 원래 IMS 애플리케이션 프로그램이 다른 트랜잭션에 대한 메시지 스위치 또는 IOPCB에 응답하지 않을 때 OTMA/IMS가 DFS2082 RESPONSE MODE TRANSACTION TERMINATED WITHOUT REPLY 메시지를 생성하도록 요청합니다.

이 필드의 초기값은 MQIIH\_NONE입니다.

### **MQIIH의 LTermOverride (MQCHAR8)**

IO PCB 필드에 위치하는 논리 터미널 오버라이드입니다. 이는 선택사항입니다. 지정되지 않는 경우 TPIPE 이름이 사용됩니다. 이는 첫 번째 바이트가 공백 또는 널이면 무시됩니다.

이 필드의 길이는 MQ\_LTERM\_OVERRIDE\_LENGTH로 지정됩니다. 이 필드의 초기값은 8개의 빈 문자입니다.

### **MQIIH용 MFSMapName (MQCHAR8)**

IO PCB 필드에 위치하는 메시지 형식 서비스 맵 이름입니다. 이는 선택사항입니다. 입력 시에는 MID를 나타내며 출력 시에는 MOD를 나타냅니다. 이는 첫 번째 바이트가 공백 또는 널이면 무시됩니다.

이 필드의 길이는 MQ\_MFS\_MAP\_NAME\_LENGTH로 지정됩니다. 이 필드의 초기값은 8개의 빈 문자입니다.

### **MQIIH의 ReplyTo형식 (MQCHAR8)**

이는 현재 메시지에 대한 응답으로 전송되는 응답 메시지의 MQ 형식 이름입니다. 이 필드의 길이는 MQ\_FORMAT\_LENGTH에 지정되어 있습니다. 이 필드의 초기값은 MQFMT\_NONE입니다.

MQGMO\_CONVERT를 사용하여 응답 메시지의 데이터를 변환하려면 MQIIH.replyToFormat=MQFMT\_STRING 또는 MQIIH.replyToFormat=MQFMT\_IMS\_VAR\_STRING을 지정하십시오. 이러한 필드의 사용에 대한 설명은 435 페이지의 『MQMD의 형식 (MQCHAR8)』의 내용을 참조하십시오.

기본값(MQIIH.replyToFormat=MQFMT\_NONE)이 요청 메시지에서 사용되고 응답 메시지가 MQGMO\_CONVERT를 사용하여 검색되는 경우 데이터 변환이 수행되지 않습니다.

### **MQIIH용 인증자 (MQCHAR8)**

이는 RACF 비밀번호 또는 PassTicket입니다. 이는 선택사항입니다. 지정된 경우 보안 컨텍스트 제공을 위해 IMS에 전송된 UTOKEN을 빌드하도록 MQMD 보안 컨텍스트에서 사용자 ID와 함께 사용됩니다. 지정되지 않으면, 사용자 ID가 검증 없이 사용됩니다. 이는 인증자가 존재해야 할 수 있는 RACF 스위치의 설정에 따라 다릅니다.

이는 첫 번째 바이트가 공백 또는 널이면 무시됩니다. 다음 특수 값을 사용할 수 있습니다.

#### **MQIAUT\_NONE**

인증이 없습니다.

C 프로그래밍 언어의 경우 상수 MQIAUT\_NONE\_ARRAY도 정의됩니다. 이는 MQIAUT\_NONE과 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

이 필드의 길이는 MQ\_AUTHENTICATOR\_LENGTH로 지정됩니다. 이 필드의 초기값은 MQIAUT\_NONE입니다.

### **MQIIH의 TranInstanceID (MQBYTE16)**

이는 트랜잭션 인스턴스 ID입니다. 이 필드는 IMS에서 출력 메시지에 의해 사용되므로 첫 번째 입력에서 무시됩니다. TranState를 MQITS\_IN\_CONVERSATION으로 설정하는 경우, 다음 입력 및 모든 후속 입력에 이를 제공하여 IMS에서 메시지를 올바른 대화로 상관할 수 있도록 해야 합니다. 다음 특수 값을 사용할 수 있습니다.

#### **MQITII\_NONE**

트랜잭션 인스턴스 ID가 없습니다.

C 프로그래밍 언어의 경우 상수 MQITII\_NONE\_ARRAY도 정의됩니다. 이는 MQITII\_NONE과 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

이 필드의 길이는 MQ\_TRAN\_INSTANCE\_ID\_LENGTH로 지정됩니다. 이 필드의 초기값은 MQITII\_NONE입니다.

### **MQIIH의 경우 TranState (MQCHAR)**

이는 IMS 대화상태를 표시합니다. 대화가 존재하지 않으므로 이는 첫 번째 입력에서 무시됩니다. 후속 입력 시 대화가 활성인지 여부를 표시합니다. 출력 시 IMS에 의해 설정됩니다. 값은 다음 중 하나여야 합니다.

#### **MQITS\_IN\_CONVERSATION**


대화 중입니다.

#### **MQITS\_NOT\_IN\_CONVERSATION**

대화 중이 아닙니다.

#### **MQITS\_ARCHITECTED**

아키텍처 양식으로 트랜잭션 상태 데이터를 리턴하십시오.

이 값은 IMS /DISPLAY TRAN 명령으로만 사용됩니다. 이는 문자 양식 대신 IMS 아키텍처 양식으로 트랜잭션 상태 데이터를 리턴합니다.  자세한 정보는 [IBM MQ를 통해 IMS 트랜잭션 프로그램 작성의 내용을 참조하십시오.](#)

이 필드의 초기값은 MQITS\_NOT\_IN\_CONVERSATION입니다.

### **MQIIH의 CommitMode (MQCHAR)**

이는 IMS 커밋 모드입니다. IMS 커밋 모드에 대한 자세한 정보는 OTMA 참조를 참조하십시오. 값은 다음 중 하나여야 합니다.

#### **MQICM\_COMMIT\_THEN\_SEND**

커밋 후 전송합니다.

이 모드는 출력의 이중 큐잉이지만 부족한 영역 점유 시간을 의미합니다. 빠른 경로 및 대화 트랜잭션은 이 모드로 실행할 수 없습니다.

### **MQICM\_SEND\_THEN\_COMMIT**

전송 후 커밋합니다.

MQICM\_SEND\_THEN\_COMMIT의 커밋 모드의 결과로 시작된 IMS 트랜잭션은 트랜잭션이 IMS 시스템 정의에서 정의된 방법에 상관 없이 RESPONSE 모드에서 실행됩니다(TRANSACT 매크로의 MSGTYPE 매개변수). 이는 또한 트랜잭션 스위치를 사용하여 시작된 트랜잭션에 적용됩니다.

이 필드의 초기값은 MQICM\_COMMIT\_THEN\_SEND입니다.

### **MQIIH용 SecurityScope (MQCHAR)**

이는 필요한 IMS 보안 처리를 표시합니다. 다음 값이 정의됩니다.

#### **MQISS\_CHECK**

보안 검사 범위: ACEE는 종속 영역이 아니라 제어 영역에서 빌드됩니다.

#### **MQISS\_FULL**

전체 보안 범위: 캐시된 ACEE는 제어 영역에서 빌드되며 캐시되지 않은 ACEE는 종속 영역에서 빌드됩니다. MQISS\_FULL을 사용하는 경우 ACEE가 빌드되는 사용자 ID에 종속 영역에서 사용된 자원에 대한 액세스가 있는지 확인하십시오.

이 필드에 대해 MQISS\_CHECK 및 MQISS\_FULL 모두 지정되지 않으면 MQISS\_CHECK가 가정됩니다.

이 필드의 초기값은 MQISS\_CHECK입니다.

### **MQIIH용으로 예약됨 (MQCHAR)**

이는 예약된 필드이며 비어 있어야 합니다.

### **MQIMPO - 메시지 특성 조회 옵션**

MQIMPO 구조를 사용하여 애플리케이션이 메시지 특성이 조회되는 방법을 제어하는 옵션을 지정할 수 있습니다. 구조는 MQINQMP 호출의 입력 매개변수입니다.

### **가용성**

모든 IBM MQ 시스템 및 IBM MQ 클라이언트.

### **문자 세트 및 인코딩**

MQIMPO의 데이터는 애플리케이션의 문자 세트 및 애플리케이션의 인코딩 (MQENC\_NATIVE)에 있어야 합니다.

### **필드**

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 498. MQIPMO의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQIMPO_STRUC_ID	'IMPO'
버전 (구조 버전 번호)	MQIMPO_VERSION_1	1
옵션 (MQINQMP의 조치를 제어하는 옵션)	MQIMPO_INQ_FIRST	
RequestedEncoding (조회된 특성이 변환되는 인코딩)	MQENC_NATIVE	
RequestedCCSID (조회된 특성의 문자 세트)	MQCCSI_APPL	
ReturnedEncoding (리턴된 값의 인코딩)	MQENC_NATIVE	



표 498. MQIPMO의 필드 (계속)

필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
ReturnedCCSID	0	
Reserved1 (예약 필드)	공백 문자 (4바이트필드)	
ReturnedName (조회된 특성의 이름)	MQCHARV_DEFAULT	
TypeString (특성 데이터 유형의 문자열 표시)	널 문자열 또는 공백	

**참고:**

1. 널 문자열 값 또는 공백은 C의 널 문자열과 기타 프로그래밍 언어의 공백 문자를 나타냅니다.
2. C 프로그래밍 언어의 매크로 변수MQIPMO\_DEFAULT는 테이블에 나열되는 값을 포함합니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오.

```
MQIPMO MyIMPO = {MQIPMO_DEFAULT};
```

## 언어 선언

### MQIPMO의 C 선언

```
typedef struct tagMQIPMO MQIPMO;
struct tagMQIPMO {
    MQCHAR4   StructId;           /* Structure identifier */
    MQLONG    Version;           /* Structure version number */
    MQLONG    Options;           /* Options that control the action of
    MQINQMP */
    MQLONG    RequestedEncoding; /* Requested encoding of Value */
    MQLONG    RequestedCCSID;    /* Requested character set identifier
    of Value */
    MQLONG    ReturnedEncoding;  /* Returned encoding of Value */
    MQLONG    ReturnedCCSID;     /* Returned character set identifier
    of Value */
    MQCHAR    Reserved1;        /* Reserved field */
    MQCHARV   ReturnedName;     /* Returned property name */
    MQCHAR8   TypeString;       /* Property data type as a string */
};
```

### MQIPMO의 COBOL 선언

```
** MQIPMO structure
10 MQIPMO.
** Structure identifier
15 MQIPMO-STRUCID PIC X(4).
** Structure version number
15 MQIPMO-VERSION PIC S9(9) BINARY.
** Options that control the action of MQINQMP
15 MQIPMO-OPTIONS PIC S9(9) BINARY.
** Requested encoding of VALUE
15 MQIPMO-REQUESTEDENCODING PIC S9(9) BINARY.
** Requested character set identifier of VALUE
15 MQIPMO-REQUESTEDCCSID PIC S9(9) BINARY.
** Returned encoding of VALUE
15 MQIPMO-RETURNEDENCODING PIC S9(9) BINARY.
** Returned character set identifier of VALUE
15 MQIPMO-RETURNEDCCSID PIC S9(9) BINARY.
** Reserved field
15 MQIPMO-RESERVED1
** Returned property name
15 MQIPMO-RETURNEDNAME.
** Address of variable length string
20 MQIPMO-RETURNEDNAME-VSPTR POINTER.
** Offset of variable length string
20 MQIPMO-RETURNEDNAME-VSOFFSET PIC S9(9) BINARY.
** CCSID of variable length string
20 MQIPMO-RETURNEDNAME-VSCCSID PIC S9(9) BINARY.
```

```
** Property data type as string
15 MQIMPO-TYPESTRING PIC S9(9) BINARY.
```

## MQIMPO의 PL/I 선언

```
dcl
1 MQIMPO based,
3 StrucId          char(4),          /* Structure identifier */
3 Version          fixed bin(31), /* Structure version number */
3 Options          fixed bin(31), /* Options that control the
                                action of MQINQMP */
3 RequestedEncoding fixed bin(31), /* Requested encoding of
                                Value */
3 RequestedCCSID   fixed bin(31), /* Requested character set
                                identifier of Value */
3 ReturnedEncoding fixed bin(31), /* Returned encoding of
                                Value */
3 ReturnedCCSID    fixed bin(31), /* Returned character set
                                identifier of Value */
3 Reserved1        fixed bin(31), /* Reserved field */
3 ReturnedName,    /* Returned property name */
5 ReturnedName_VSPtr pointer,      /* Address of returned
                                name */
5 5 ReturnedName_VSOFFSET fixed bin(31), /* Offset of returned
                                name */
5 5 ReturnedName_VSCCSID fixed bin(31), /* CCSID of returned
                                name */
3 TypeString       char(8);        /* Property data type as
                                string */
```

## MQIMPO의 상위 레벨 어셈블러 선언

MQIMPO	DSECT		
MQIMPO_STRUCID	DS	CL4	Structure identifier
MQIMPO_VERSION	DS	F	Structure version number
MQIMPO_OPTIONS	DS	F	Options that control the action of MQINQMP
*			
MQIMPO_REQUESTEDENCODING	DS	F	Requested encoding of VALUE
MQIMPO_REQUESTEDCCSID	DS	F	Requested character set identifier of VALUE
*			
MQIMPO_RETURNEDENCODING	DS	F	Returned encoding of VALUE
MQIMPO_RETURNEDCCSID	DS	F	Returned character set identifier of VALUE
*			
MQIMPO_RESERVED1	DS	F	Reserved field
MQIMPO_RETURNEDNAME	DS	0F	Force fullword alignment
MQIMPO_RETURNEDNAME_VSPTR	DS	F	Address of returned name
MQIMPO_RETURNEDNAME_VSOFFSET	DS	F	Offset of returned name
MQIMPO_RETURNEDNAME_VSLENGTH	DS	F	Length of returned name
MQIMPO_RETURNEDNAME_VSCCSID	DS	F	CCSID of returned name
MQIMPO_RETURNEDNAME_LENGTH	EQU	**MQIMPO_RETURNEDNAME	
	ORG	MQIMPO_RETURNEDNAME	
MQIMPO_RETURNEDNAME_AREA	DS	CL(MQIMPO_RETURNEDNAME_LENGTH)	
*			
MQIMPO_TYPESTRING	DS	CL8	Property data type as string
MQIMPO_LENGTH	EQU	**MQIMPO	
MQIMPO_AREA	DS	CL(MQIMPO_LENGTH)	

### MQIMPO의 경우 StrucId (MQCHAR4)

이는 조회 메시지 특성 옵션 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQIMPO\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

#### MQIMPO\_STRUC\_ID

조회 메시지 특성 옵션 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQIMPO\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQIMPO\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### MQIMPO의 버전 (MQLONG)

메시지 특성 조회 옵션 구조 - 버전 필드

구조 버전 번호입니다. 값은 다음과 같아야 합니다.

### **MQIMPO\_VERSION\_1**

조회 메시지 특성 옵션 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

### **MQIMPO\_CURRENT\_VERSION**

조회 메시지 특성 옵션 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQIMPO\_VERSION\_1입니다.

## **MQIMPO의 옵션 (MQLONG)**

메시지 특성 조회 옵션 구조 - 옵션 필드

다음 옵션은 MQINQMP의 조치를 제어합니다. 이 옵션 중 하나 이상을 지정할 수 있습니다. 둘 이상의 옵션을 지정하려면 값을 함께 추가하거나(동일한 상수를 두 번 이상 추가하지 않음), 비트 단위 OR 연산을 사용하여 값을 결합하십시오(프로그래밍 언어가 비트 연산을 지원하는 경우).

올바르지 않은 옵션의 결합이 설명되어 있습니다. 다른 모든 결합은 유효합니다.

**값 데이터 옵션:** 다음 옵션은 특성이 메시지에서 검색될 때 값 데이터의 처리와 관련됩니다.

### **MQIMPO\_CONVERT\_VALUE**

이 옵션은 MQINQMP 호출이 *Value* 영역에서 특성 값을 리턴하기 전에 특성의 값이 지정된 *RequestedCCSID* 및 *RequestedEncoding* 값을 따르게 변환되도록 요청합니다.

- 변환에 성공하면 *ReturnedCCSID* 및 *ReturnedEncoding* 필드가 MQINQMP 호출에서 리턴 시 *RequestedCCSID* 및 *RequestedEncoding*과 동일하게 설정됩니다.

- 변환에 실패해도 MQINQMP 호출이 오류 없이 완료되면 특성 값은 변환되지 않은 상태로 리턴됩니다.

특성이 문자열인 경우 *ReturnedCCSID* 및 *ReturnedEncoding* 필드는 변환되지 않은 문자열의 문자 세트 및 인코딩으로 설정됩니다.

이 경우 완료 코드는 MQCC\_WARNING이고 이유 코드는 MQRC\_PROP\_VALUE\_NOT\_CONVERTED입니다. 특성 커서는 리턴된 특성으로 이동됩니다.

특성 값이 변환 중에 확장되고 **Value** 매개변수의 크기를 초과하는 경우 값은 이유 코드 MQCC\_FAILED와 함께 변환되지 않은 상태로 리턴됩니다. 이유 코드는 MQRC\_PROPERTY\_VALUE\_TOO\_BIG으로 설정됩니다.

MQINQMP 호출의 **DataLength** 매개변수는 애플리케이션이 변환된 특성 값을 수용하는 데 필요한 버퍼 크기를 판별할 수 있도록 특성 값이 변환된 길이를 리턴합니다. 특성 커서가 변경되지 않습니다.

이 옵션은 또한 다음을 요청합니다.

- 특성 이름에 와일드카드가 포함된 경우 및
- *ReturnedName* 필드는 리턴된 이름의 주소 또는 오프셋으로 초기화됩니다.

그러면 리턴된 이름은 *RequestedCCSID* 및 *RequestedEncoding* 값을 따르도록 변환됩니다.

- 변환에 성공하면 *ReturnedName*의 *VSCSID* 필드 및 리턴된 이름의 인코딩이 *RequestedCCSID* 및 *RequestedEncoding*의 입력 값으로 설정됩니다.

- 변환에 실패해도 MQINQMP 호출이 오류 또는 경고 없이 완료되면 리턴된 이름은 변환되지 않습니다. 이 경우 완료 코드는 MQCC\_WARNING이고 이유 코드는 MQRC\_PROP\_NAME\_NOT\_CONVERTED입니다.

특성 커서는 리턴된 특성으로 이동됩니다. 값과 이름 모두 변환되지 않으면 MQRC\_PROP\_VALUE\_NOT\_CONVERTED가 리턴됩니다.

리턴된 이름이 변환 중에 확장되고 *ReturnedName*의 *VSBuFSIZE* 필드의 크기를 초과하는 경우 리턴된 문자열은 완료 코드 MQCC\_FAILED와 함께 변환되지 않은 상태로 남고 이유 코드는 MQRC\_PROPERTY\_NAME\_TOO\_BIG으로 설정됩니다.

MQCHARV 구조의 *VSLength* 필드는 애플리케이션이 변환된 특성 값을 수용하는 데 필요한 버퍼 크기를 판별할 수 있도록 특성 값이 변환된 길이를 리턴합니다. 특성 커서가 변경되지 않습니다.

### MQIMPO\_CONVERT\_TYPE

이 옵션은 특성 값을 현재 데이터 유형에서 MQINQMP 호출의 **Type** 매개변수에 지정된 데이터 유형으로 변환하도록 요청합니다.

- 변환에 성공한 경우 **Type** 매개변수는 MQINQMP 호출의 리턴 시 변경되지 않습니다.
- 변환에 실패하지만 MQINQMP 호출이 오류와 함께 완료되는 경우 이유 MQRC\_PROP\_CONV\_NOT\_SUPPORTED와 함께 호출이 실패합니다. 특성 커서가 변경되지 않습니다.

데이터 유형의 변환으로 인해 변환 중에 값이 확장되고 변환된 값이 **Value** 매개변수의 크기를 초과하는 경우 값은 완료 코드 MQCC\_FAILED와 함께 변환되지 않은 상태로 리턴되고 이유 코드는 MQRC\_PROPERTY\_VALUE\_TOO\_BIG으로 설정됩니다.

MQINQMP 호출의 **DataLength** 매개변수는 애플리케이션이 변환된 특성 값을 수용하는 데 필요한 버퍼 크기를 판별할 수 있도록 특성 값이 변환된 길이를 리턴합니다. 특성 커서가 변경되지 않습니다.

MQINQMP 호출의 **Type** 매개변수 값이 올바르지 않은 경우 이유 MQRC\_PROPERTY\_TYPE\_ERROR와 함께 호출이 실패합니다.

요청된 데이터 유형 변환이 지원되지 않는 경우 이유 MQRC\_PROP\_CONV\_NOT\_SUPPORTED와 함께 호출이 실패합니다. 다음의 데이터 유형 변환이 지원됩니다.

특성 데이터 유형	지원되는 대상 데이터 유형
MQTYPE_BOOLEAN	MQTYPE_STRING, MQTYPE_INT8, MQTYPE_INT16, MQTYPE_INT32, MQTYPE_INT64
MQTYPE_BYTE_STRING	MQTYPE_STRING
MQTYPE_INT8	MQTYPE_STRING, MQTYPE_INT16, MQTYPE_INT32, MQTYPE_INT64
MQTYPE_INT16	MQTYPE_STRING, MQTYPE_INT32, MQTYPE_INT64
MQTYPE_INT32	MQTYPE_STRING, MQTYPE_INT64
MQTYPE_INT64	MQTYPE_STRING
MQTYPE_FLOAT32	MQTYPE_STRING, MQTYPE_FLOAT64
MQTYPE_FLOAT64	MQTYPE_STRING
MQTYPE_STRING	MQTYPE_BOOLEAN, MQTYPE_INT8, MQTYPE_INT16, MQTYPE_INT32, MQTYPE_INT64, MQTYPE_FLOAT32, MQTYPE_FLOAT64
MQTYPE_NULL	없음

지원되는 변환에 적용되는 일반 규칙은 다음과 같습니다.

- 변환 중에 데이터가 손실되지 않는 경우 숫자 특성 값을 하나의 데이터 유형에서 다른 데이터 유형으로 변환할 수 없습니다.  
예를 들어, 데이터 유형이 MQTYPE\_INT32인 특성 값을 데이터 유형이 MQTYPE\_INT64인 값으로 변환할 수 있지만 데이터 유형이 MQTYPE\_INT16인 값으로 변환할 수는 없습니다.
- 데이터 유형의 특성 값은 문자열로 변환될 수 있습니다.
- 문자열이 변환에 맞게 형식화된 경우 문자열 특성 값은 다른 데이터 유형으로 변환될 수 있습니다. 애플리케이션이 올바르게 형식화되지 않은 문자열 특성 값을 변환하려고 시도하는 경우 IBM MQ가 이유 코드 MQRC\_PROP\_NUMBER\_FORMAT\_ERROR를 리턴합니다.
- 애플리케이션이 지원되지 않는 변환을 시도하는 경우 IBM MQ는 이유 코드 MQRC\_PROP\_CONV\_NOT\_SUPPORTED를 리턴합니다.

하나의 데이터 유형에서 다른 데이터 유형으로의 특성 값 변환에 대한 특정 규칙은 다음과 같습니다.

- MQTYPE\_BOOLEAN 특성 값을 문자열로 변환할 때 값 TRUE는 문자열 "TRUE"로 변환되고 값 false는 문자열 "FALSE"로 변환됩니다.
- MQTYPE\_BOOLEAN 특성 값을 숫자 데이터 유형으로 변환할 때 값 TRUE는 1로 변환되고 값 FALSE는 0으로 변환됩니다.
- 문자열 특성 값을 MQTYPE\_BOOLEAN 값으로 변환할 때 문자열 "TRUE" 또는 "1"은 TRUE로 변환되고 문자열 "FALSE" 또는 "0"은 FALSE로 변환됩니다.

용어 "TRUE" 및 "FALSE"는 대소문자가 구분되지 않습니다.

기타 문자열은 변환할 수 없습니다. IBM MQ가 이유 코드 MQRC\_PROP\_NUMBER\_FORMAT\_ERROR를 리턴합니다.

- 문자열 특성 값을 데이터 유형이 MQTYPE\_INT8, MQTYPE\_INT16, MQTYPE\_INT32 또는 MQTYPE\_INT64인 값으로 변환할 때 문자열의 형식은 다음과 같아야 합니다.

```
[blanks][sign]digits
```

문자열의 컴포넌트의 의미는 다음과 같습니다.

**blanks**

선택적 선두 공백 문자

**sign**

선택적 더하기 부호(+) 또는 빼기 부호(-) 문자.

**digits**

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

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

문자열이 올바르게 형식화되지 않는 경우 IBM MQ가 이유 코드 MQRC\_PROP\_NUMBER\_FORMAT\_ERROR를 리턴합니다.

- 문자열 특성 값을 데이터 유형이 MQTYPE\_FLOAT32 또는 MQTYPE\_FLOAT64인 값으로 변환할 때 문자열의 형식은 다음과 같아야 합니다.

```
[blanks][sign]digits[.digits][e_char[e_sign]e_digits]
```

문자열의 컴포넌트의 의미는 다음과 같습니다.

**blanks**

선택적 선두 공백 문자

**sign**

선택적 더하기 부호(+) 또는 빼기 부호(-) 문자.

**digits**

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

**e\_char**

지수 문자("E" 또는 "e" 중 하나).

**e\_sign**

지수에 대한 선택적 더하기 부호(+) 또는 빼기 부호(-) 문자.

**e\_digits**

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

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

문자열이 올바르게 형식화되지 않는 경우 IBM MQ가 이유 코드 MQRC\_PROP\_NUMBER\_FORMAT\_ERROR를 리턴합니다.

- 숫자 특성 값을 문자열로 변환할 때 이 값은 해당 값의 ASCII 문자를 포함하는 문자열이 아니라 10진수로서의 값에 대한 문자열 표현으로 변환됩니다. 예를 들어, 정수 65는 "65" 문자열로 변환되며 "A" 문자열로 변환되지 않습니다.
- 바이트 문자열 특성 값을 문자열로 변환할 때 각 바이트는 바이트를 표시하는 두 개의 16진 문자로 변환됩니다. 예를 들어, 바이트 배열 {0xF1, 0x12, 0x00, 0xFF}는 "F11200FF" 문자열로 변환됩니다.

### MQIMPO\_QUERY\_LENGTH

특성 값의 유형 및 길이를 조회합니다. 길이는 MQINQMP 호출의 **DataLength** 매개변수에서 리턴됩니다. 특성 값은 리턴되지 않습니다.

**ReturnedName** 버퍼가 지정되는 경우 MQCHARV 구조의 *VSLength* 필드는 특성 이름의 길이로 채워집니다. 특성 이름은 리턴되지 않습니다.

**반복 옵션:** 다음 옵션은 와일드 카드 문자가 포함된 이름을 사용한 특성 반복과 관련이 있습니다.

### MQIMPO\_INQ\_FIRST

지정된 이름과 일치하는 첫 번째 특성을 조회하십시오. 이 호출 이후 커서는 리턴된 특성에서 설정됩니다.

이는 기본값입니다.

MQIMPO\_INQ\_PROP\_UNDER\_CURSOR 옵션은 필요에 따라 동일한 특성을 다시 조회하기 위해 MQINQMP 호출과 함께 나중에 사용할 수 있습니다.

하나의 특성 커서만 있으므로 특성 이름이 MQINQMP 호출에 지정된 경우 특성 이름이 변경하면 커서가 재설정됩니다.

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

MQIMPO\_INQ\_NEXT  
MQIMPO\_INQ\_PROP\_UNDER\_CURSOR

### MQIMPO\_INQ\_NEXT

특성 커서로부터 검색을 계속하면서 지정된 이름과 일치하는 다음 특성을 조회합니다. 커서는 리턴된 특성으로 이동됩니다.

호출이 지정된 이름의 첫 번째 MQINQMP 호출인 경우, 지정된 이름과 일치하는 첫 번째 특성이 리턴됩니다.

MQIMPO\_INQ\_PROP\_UNDER\_CURSOR 옵션은 필요에 따라 동일한 특성을 다시 조회하기 위해 MQINQMP 호출과 함께 나중에 사용할 수 있습니다.

커서 아래에 있는 특성이 삭제되는 경우 MQINQMP는 삭제된 특성 다음에 오는 다음 일치 특성을 리턴합니다.

반복이 진행 중인 동안 와일드 카드와 일치하는 특성이 추가되는 경우 특성은 반복이 완료되는 동안 리턴되거나 리턴되지 않을 수 있습니다. MQIMPO\_INQ\_FIRST를 사용하여 반복이 재시작되면 특성이 리턴됩니다.

반복이 진행 중인 동안 삭제된 와일드 카드와 일치하는 특성은 해당 삭제 후에 리턴되지 않습니다.

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

MQIMPO\_INQ\_FIRST  
MQIMPO\_INQ\_PROP\_UNDER\_CURSOR

### MQIMPO\_INQ\_PROP\_UNDER\_CURSOR

특성 커서가 지시하는 특성의 값을 검색합니다. 특성 커서로 지정된 특성은 MQIMPO\_INQ\_FIRST 또는 MQIMPO\_INQ\_NEXT 옵션을 사용하여 조회된 특성입니다.

메시지 핸들이 재사용될 때 메시지 핸들이 MQGET 호출에서 MQGMO의 *MsgHandle* 필드에 지정될 때 또는 메시지 핸들이 MQPUT 호출에서 MQPMO 구조의 *OriginalMsgHandle* 또는 *NewMsgHandle* 필드에 지정될 때 특성 커서가 재설정됩니다.

이 옵션이 특성 커서가 아직 설정되지 않았을 때 사용되거나 특성 커서로 지정된 특성이 삭제된 경우 완료 코드 MQCC\_FAILED 및 이유 MQRC\_PROPERTY\_NOT\_AVAILABLE과 함께 호출이 실패합니다.

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

MQIMPO\_INQ\_FIRST  
MQIMPO\_INQ\_NEXT

이전에 설명된 옵션 중 필요한 사항이 없는 경우 다음 옵션을 사용할 수 있습니다.

#### **MQIMPO\_NONE**

기타 옵션이 지정되지 않았음을 표시하려면 이 값을 사용하십시오. 모든 옵션은 기본 값을 가정합니다.

MQIMPO\_NONE은 프로그램 문서화를 지원합니다. 이 옵션은 다른 옵션과 함께 사용하도록 의도되지 않았지만 값이 0인 경우에는 그러한 사용을 발견할 수 없습니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQIMPO\_INQ\_FIRST입니다.

#### **MQIMPO의 경우 RequestedEncoding (MQLONG)**

메시지 특성 조회 옵션 구조 - RequestedEncoding 필드

이는 MQIMPO\_CONVERT\_VALUE 또는 MQIMPO\_CONVERT\_TYPE이 지정될 때 조회된 특성 값이 변환되는 인코딩입니다.

이 필드의 초기값은 MQENC\_NATIVE입니다.

#### **MQIMPO의 경우 RequestedCCSID (MQLONG)**

메시지 특성 조회 옵션 구조 - RequestedCCSID 필드

값이 문자열인 경우 조회한 특성 값을 변환할 문자 세트입니다. 또한 MQIMPO\_CONVERT\_VALUE 또는 MQIMPO\_CONVERT\_TYPE이 지정된 경우 *ReturnedName*이 변환되는 문자 세트입니다.

이 필드의 초기값은 MQCCSI\_APPL입니다.

#### **MQIMPO의 경우 ReturnedEncoding (MQLONG)**

메시지 특성 조회 옵션 구조 - ReturnedEncoding 필드

출력에서 이는 리턴된 값의 인코딩입니다.

MQIMPO\_CONVERT\_VALUE 옵션이 지정되고 변환이 완료되면 리턴 시 *ReturnedEncoding* 필드가 전달된 값과 동일한 값입니다.

이 필드의 초기값은 MQENC\_NATIVE입니다.

#### **MQIMPO의 경우 ReturnedCCSID (MQLONG)**

메시지 특성 조회 옵션 구조 - ReturnedCCSID 필드

출력 시 이는 MQINQMP 호출의 **Type** 매개변수가 MQTYPE\_STRING인 경우 리턴된 값의 문자 세트입니다.

MQIMPO\_CONVERT\_VALUE 옵션이 지정되고 변환이 완료되면 리턴 시 *ReturnedCCSID* 필드가 전달된 값과 동일한 값입니다.

이 필드의 초기값은 0입니다.

#### **MQIMPO의 경우 Reserved1 (MQCHAR)**

이 필드는 예약된 필드입니다. 이 필드의 초기값은 공백 문자(4바이트 필드)입니다.

#### **MQIMPO의 경우 ReturnedName (MQCHARV)**

메시지 특성 조회 옵션 구조 - ReturnedName 필드

조회된 특성의 실제 이름.

입력 시 문자열 버퍼는 MQCHARV 구조의 *VSPtr* 또는 *VSOffset* 필드를 사용하여 전달할 수 있습니다. 문자열 버퍼의 길이는 MQCHARV 구조의 *VSBuFSIZE* 필드를 사용하여 지정됩니다.

MQINQMP 호출에서 리턴 시 이름을 완전히 포함할 수 있을 만큼 긴 문자열 버퍼는 조회한 특성의 이름으로 완료됩니다. MQCHARV 구조의 *VSLength* 필드는 특성 이름의 길이로 채워집니다. MQCHARV 구조의 *VSCCSID* 필드는 이름 변환의 실패 여부와 관계없이 리턴된 이름의 문자 세트를 표시하기 위해 채워집니다.

이 필드는 입출력(I/O) 필드입니다. 이 필드의 초기값은 MQCHARV\_DEFAULT입니다.

## MQIMPO의 경우 TypeString (MQCHAR8)

메시지 특성 조회 옵션 구조 - TypeString 필드

특성의 데이터 유형의 문자열 표현.

특성이 MQRFH2 헤더에 지정되었고 MQRFH2 dt 속성이 인식되지 않는 경우 이 필드는 특성의 데이터 유형을 판별하는 데 사용할 수 있습니다. *TypeString*이 코드화 문자 세트 1208(UTF-8)로 리턴되고 인식되지 않는 실패한 특성의 dt 속성 값의 첫 8바이트인임

이 필드는 항상 출력 필드입니다. 이 필드의 초기값은 C 프로그래밍 언어에서는 널 문자열이며 기타 프로그래밍 언어에서는 8자의 공백입니다.

## MQMD - 메시지 디스크립터

MQMD 구조에는 메시지가 송신 및 수신 애플리케이션 간에 이동할 때 애플리케이션 데이터를 수반하는 제어 정보가 포함되어 있습니다. 이 구조는 MQGET, MQPUT 및 MQPUT1 호출의 입력/출력 매개변수입니다.

### 가용성

모든 IBM MQ 시스템 및 해당 시스템에 연결된 IBM MQ MQI clients

### 버전

MQMD의 현재 버전은 MQMD\_VERSION\_2입니다. 여러 환경 간에 이식하려는 애플리케이션은 MQMD의 필수 버전이 관련된 모든 환경에서 지원되는지 확인해야 합니다. 최신 버전의 구조에만 있는 필드는 다음에 오는 설명에서 최신 필드로 식별됩니다.

지원되는 프로그래밍 언어에 제공되는 헤더, COPY, INCLUDE 파일은 MQPMO\_VERSION\_1로 설정된, *Version* 필드의 초기값과 함께 환경이 지원하는 MQMD의 최신 버전을 포함합니다. version-1 구조에 없는 필드를 사용하려면 애플리케이션이 *Version* 필드를 필요한 버전의 버전 번호로 설정해야 합니다.

버전-1 구조의 선언은 MQMD1 이름으로 사용 가능합니다.

### 문자 세트 및 인코딩

MQMD의 데이터는 로컬 큐 관리자의 문자 세트 및 인코딩에 있어야 합니다. 이는 **CodedCharSetId** 큐 관리자 속성 및 MQENC\_NATIVE에서 제공됩니다. 그러나 애플리케이션이 IBM MQ MQI client로 실행 중인 경우 구조는 클라이언트의 문자 세트 및 인코딩에 있어야 합니다.

송신 및 수신 큐 관리자가 서로 다른 문자 세트 또는 인코딩을 사용하는 경우, MQMD의 데이터가 자동으로 변환됩니다. 애플리케이션이 MQMD를 변환할 필요가 없습니다.

### MQMD의 다른 버전 사용

version-2 MQMD는 version-1 MQMD를 사용하고 메시지 데이터에 MQMDE 구조를 접두부로 붙이는 것과 같습니다. 그러나 MQMDE 구조의 모든 필드가 해당 기본값을 가진 경우 MQMDE는 생략될 수 있습니다. 버전-1 MQMD 및 MQMDE는 다음 설명대로 사용됩니다.

- MQPUT 및 MQPUT1 호출에서 MQPUT 및 MQPUT1 호출에서 애플리케이션이 버전-1 MQMD를 제공하는 경우 MQMDE가 있음을 표시하기 위해 MQMD의 *Format* 필드를 MQFMT\_MD\_EXTENSION으로 설정하여 애플



리케이션이 선택적으로 메시지 데이터에 MQMDE를 접두부로 설정할 수 있습니다. 애플리케이션이 MQMDE를 제공하지 않는 경우 큐 관리자는 MQMDE의 필드에 대해 기본값 가정합니다.

**참고:** 버전-1 MQMD가 아니라 버전-2 MQMD에 있는 여러 필드는 MQPUT 및 MQPUT1의 입출력(I/O) 필드입니다. 그러나 큐 관리자는 MQPUT 및 MQPUT1 호출의 출력에서 MQMDE의 동일 필드에는 아무 값도 리턴하지 않습니다. 애플리케이션은 이들 출력 값이 필요한 경우 버전-2 MQMD를 사용해야 합니다.

- MQGET 호출에서 애플리케이션이 버전-1 MQMD를 제공하는 경우 큐 관리자는 리턴된 메시지에 MQMDE로 접두부를 붙입니다. 단, MQMDE의 필드 중 하나 이상이 기본값이 아닌 값을 가진 경우에만 해당됩니다. MQMD의 *Format* 필드에는 MQMDE가 있음을 나타내는 값 MQFMT\_MD\_EXTENSION이 있습니다.

큐 관리자가 MQMDE의 필드에 사용하는 기본값은 [460 페이지의 표 503](#)에 표시된, 해당 필드의 초기값과 동일합니다.

메시지가 전송 큐에 있으면 MQMD의 일부 필드가 특정한 값으로 설정됩니다. 세부사항은 [597 페이지의 『MQXQH - 전송 큐 헤더』](#)의 내용을 참조하십시오.

## 메시지 컨텍스트

MQMD의 특정 필드에 메시지 컨텍스트가 포함됩니다. 메시지 컨텍스트에는 ID 컨텍스트 및 원본 컨텍스트의 두 가지 유형이 있습니다. 일반적으로, 다음과 같습니다.

- ID 컨텍스트는 원래 메시지를 넣은 애플리케이션에 관련됩니다.
- 원본 컨텍스트는 가장 최근에 메시지를 넣은 애플리케이션에 관련됩니다.

이러한 두 애플리케이션은 동일한 애플리케이션일 수 있지만, 서로 다른 애플리케이션일 수도 있습니다(예를 들어, 메시지가 한 애플리케이션에서 다른 애플리케이션으로 전달되는 경우).

ID 및 원본 컨텍스트가 일반적으로 의미가 설명되어 있기는 하지만 MDMD의 컨텍스트 필드의 콘텐츠는 두 유형 모두 메시지를 넣을 때 지정된 MQPMO\_\*\_CONTEXT 옵션에 따라 다릅니다. 따라서 ID 컨텍스트는 반드시 원래 메시지를 넣은 애플리케이션에 관련될 필요는 없으며 원본 컨텍스트도 가장 최근에 메시지를 넣은 애플리케이션에 관련될 필요가 없습니다. 이는 애플리케이션 스위트의 디자인에 따라 다릅니다.

메시지 채널 에이전트(MCA)는 메시지 컨텍스트를 대체하지 않습니다. 리모트 큐 관리자에서 메시지를 수신하는 MCA는 MQPUT 또는 MQPUT1 호출에서 컨텍스트 옵션 MQPMO\_SET\_ALL\_CONTEXT를 사용합니다. 이는 수신 MCA가 송신 MCA로부터 메시지와 함께 전달된 메시지 컨텍스트를 정확히 보존할 수 있도록 합니다. 그러나 결과적으로 원본 컨텍스트는 메시지를 전송하고 수신한 MCA에 관련되지 않습니다. 원본 컨텍스트는 메시지를 넣은 이전 애플리케이션을 참조합니다. 모든 중간 애플리케이션이 메시지 컨텍스트를 전달한 경우 원본 컨텍스트는 원래 애플리케이션 자체를 참조합니다.

설명에서 컨텍스트 필드는 이전에 설명된 대로 사용되는 것처럼 설명됩니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트](#)를 참조하십시오.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 500. MQMD에 대한 MQMD의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQMD_STRUC_ID	'MD'
버전 (구조 버전 번호)	MQMD_VERSION_1	1
<a href="#">보고서</a> (보고 메시지 옵션)	MQRO_NONE	0
MsgType (메시지 유형)	MQMT_DATAGRAM	8
<a href="#">MQMD-만기 필드</a> (메시지 수명)	MQEI_UNLIMITED	-1
<a href="#">MQMD-피드백 필드</a> (피드백 또는 이유 코드)	MQFB_NONE	0
인코딩 (메시지 데이터의 숫자 인코딩)	MQENC_NATIVE	환경에 따라 다름
<a href="#">CodedCharSetId</a> (메시지 데이터의 문자 세트 ID)	MQCCSI_Q_MGR	0

표 500. MQMD에 대한 MQMD의 필드 (계속)		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
<u>형식</u> (메시지 데이터의 형식 이름)	MQFMT_NONE	공백
<u>우선순위</u> (메시지 우선순위)	MQPRI_PRIORITY_AS_Q_DEF	-1
<u>Persistence</u> (메시지 지속성)	MQPER_PERSISTENCE_AS_Q_DEF	2
MQMD- <u>MsgId 필드</u> (메시지 ID)	MQMI_NONE	널
<u>CorrelId</u> (상관 ID)	MQCI_NONE	널
<u>BackoutCount</u> (백아웃 카운터)	없음	0
<u>ReplyToQ</u> (응답 큐의 이름)	없음	널 문자열 또는 공백
<u>ReplyToQMGr</u> (응답 큐 관리자의 이름)	없음	널 문자열 또는 공백
<u>UserIdentifier</u> (사용자 ID)	없음	널 문자열 또는 공백
<u>AccountingToken</u> (계정 토큰)	MQACT_NONE	널
<u>ApplIdentity</u> 데이터 (ID와 관련된 애플리케이션 데이터)	없음	널 문자열 또는 공백
<u>PutAppl</u> 유형 (메시지를 넣는 애플리케이션의 유형)	MQAT_NO_CONTEXT	0
<u>PutApplName</u> (메시지를 넣는 애플리케이션의 이름)	없음	널 문자열 또는 공백
<u>PutDate</u> (메시지를 넣은 날짜)	없음	널 문자열 또는 공백
<u>PutTime</u> (메시지를 넣은 시간)	없음	널 문자열 또는 공백
<u>ApplOriginData</u> (원본과 관련된 애플리케이션 데이터)	없음	널 문자열 또는 공백
<b>참고:</b> <i>Version</i> 이 MQMD_VERSION_2 미만인 경우 나머지 필드는 무시됩니다.		
<u>GroupId</u> (그룹 ID)	MQGI_NONE	널
<u>MsgSeqNumber</u> (그룹 내 논리 메시지의 순서 번호)	없음	1
<u>오프셋</u> (논리 메시지의 시작부터 실제 메시지의 데이터 오프셋)	없음	0
MQMD- <u>MsgFlags 필드</u> (메시지 플래그)	MQMF_NONE	0
<u>OriginalLength</u> (원래 메시지의 길이)	MQOL_UNDEFINED	-1
<b>참고:</b> 1. 널 문자열 값 또는 공백은 C의 널 문자열과 기타 프로그래밍 언어의 공백 문자를 나타냅니다. 2. C 프로그래밍 언어의 매크로 변수 MQMD_DEFAULT는 테이블에 나열되는 값을 포함합니다. 즉, 구조 내의 필드에 대한 초기값을 제공하기 위해 다음과 같은 방법으로 사용될 수 있습니다.		
<pre>MQMD MyMD = {MQMD_DEFAULT};</pre>		

## 언어 선언

### MQMD의 C 선언

```
typedef struct tagMQMD MQMD;
```

```

struct tagMQMD {
    MQCHAR4  StrucId;          /* Structure identifier */
    MQLONG   Version;         /* Structure version number */
    MQLONG   Report;          /* Options for report messages */
    MQLONG   MsgType;         /* Message type */
    MQLONG   Expiry;          /* Message lifetime */
    MQLONG   Feedback;        /* Feedback or reason code */
    MQLONG   Encoding;        /* Numeric encoding of message data */
    MQLONG   CodedCharSetId;  /* Character set identifier of message
                               data */

    MQCHAR8  Format;           /* Format name of message data */
    MQLONG   Priority;         /* Message priority */
    MQLONG   Persistence;     /* Message persistence */
    MQBYTE24 MsgId;           /* Message identifier */
    MQBYTE24 CorrelId;        /* Correlation identifier */
    MQLONG   BackoutCount;    /* Backout counter */
    MQCHAR48 ReplyToQ;        /* Name of reply queue */
    MQCHAR48 ReplyToQMgr;     /* Name of reply queue manager */
    MQCHAR12 UserIdentifier;   /* User identifier */
    MQBYTE32 AccountingToken; /* Accounting token */
    MQCHAR32 ApplIdentityData; /* Application data relating to
                               identity */

    MQLONG   PutApplType;     /* Type of application that put the
                               message */
    MQCHAR28 PutApplName;     /* Name of application that put the
                               message */

    MQCHAR8  PutDate;         /* Date when message was put */
    MQCHAR8  PutTime;         /* Time when message was put */
    MQCHAR4  ApplOriginData;  /* Application data relating to origin */
    MQBYTE24 GroupId;         /* Group identifier */
    MQLONG   MsgSeqNumber;    /* Sequence number of logical message
                               within group */

    MQLONG   Offset;          /* Offset of data in physical message
                               from start of logical message */

    MQLONG   MsgFlags;        /* Message flags */
    MQLONG   OriginalLength;  /* Length of original message */
};

```

## MQMD의 COBOL 선언

```

** MQMD structure
10 MQMD.
** Structure identifier
15 MQMD-STRUCID PIC X(4).
** Structure version number
15 MQMD-VERSION PIC S9(9) BINARY.
** Options for report messages
15 MQMD-REPORT PIC S9(9) BINARY.
** Message type
15 MQMD-MSGTYPE PIC S9(9) BINARY.
** Message lifetime
15 MQMD-EXPIRY PIC S9(9) BINARY.
** Feedback or reason code
15 MQMD-FEEDBACK PIC S9(9) BINARY.
** Numeric encoding of message data
15 MQMD-ENCODING PIC S9(9) BINARY.
** Character set identifier of message data
15 MQMD-CODEDCHARSETID PIC S9(9) BINARY.
** Format name of message data
15 MQMD-FORMAT PIC X(8).
** Message priority
15 MQMD-PRIORITY PIC S9(9) BINARY.
** Message persistence
15 MQMD-PERSISTENCE PIC S9(9) BINARY.
** Message identifier
15 MQMD-MSGID PIC X(24).
** Correlation identifier
15 MQMD-CORRELID PIC X(24).
** Backout counter
15 MQMD-BACKOUTCOUNT PIC S9(9) BINARY.
** Name of reply queue
15 MQMD-REPLYTOQ PIC X(48).
** Name of reply queue manager
15 MQMD-REPLYTOQMGR PIC X(48).
** User identifier
15 MQMD-USERIDENTIFIER PIC X(12).
** Accounting token
15 MQMD-ACCOUNTINGTOKEN PIC X(32).
** Application data relating to identity

```

```

15 MQMD-APPLIDENTITYDATA PIC X(32).
** Type of application that put the message
15 MQMD-PUTAPPLTYPE     PIC S9(9) BINARY.
** Name of application that put the message
15 MQMD-PUTAPPLNAME     PIC X(28).
** Date when message was put
15 MQMD-PUTDATE         PIC X(8).
** Time when message was put
15 MQMD-PUTTIME         PIC X(8).
** Application data relating to origin
15 MQMD-APPLORIGINDATA PIC X(4).
** Group identifier
15 MQMD-GROUPID         PIC X(24).
** Sequence number of logical message within group
15 MQMD-MSGSEQNUMBER    PIC S9(9) BINARY.
** Offset of data in physical message from start of logical message
15 MQMD-OFFSET          PIC S9(9) BINARY.
** Message flags
15 MQMD-MSGFLAGS        PIC S9(9) BINARY.
** Length of original message
15 MQMD-ORIGINALLENGTH PIC S9(9) BINARY.

```

## MQMD의 PL/I 선언

```

dcl
  1 MQMD based,
  3 StrucId          char(4),          /* Structure identifier */
  3 Version          fixed bin(31),    /* Structure version number */
  3 Report           fixed bin(31),    /* Options for report messages */
  3 MsgType          fixed bin(31),    /* Message type */
  3 Expiry           fixed bin(31),    /* Message lifetime */
  3 Feedback         fixed bin(31),    /* Feedback or reason code */
  3 Encoding         fixed bin(31),    /* Numeric encoding of message
                                     data */
  3 CodedCharSetId  fixed bin(31),    /* Character set identifier of
                                     message data */
  3 Format            char(8),          /* Format name of message data */
  3 Priority          fixed bin(31),    /* Message priority */
  3 Persistence      fixed bin(31),    /* Message persistence */
  3 MsgId            char(24),         /* Message identifier */
  3 CorrelId         char(24),         /* Correlation identifier */
  3 BackoutCount     fixed bin(31),    /* Backout counter */
  3 ReplyToQ         char(48),         /* Name of reply queue */
  3 ReplyToMgr       char(48),         /* Name of reply queue manager */
  3 UserIdentifier   char(12),         /* User identifier */
  3 AccountingToken  char(32),         /* Accounting token */
  3 ApplIdentityData char(32),         /* Application data relating to
                                     identity */
  3 PutApplType      fixed bin(31),    /* Type of application that put the
                                     message */
  3 PutApplName      char(28),         /* Name of application that put the
                                     message */
  3 PutDate          char(8),          /* Date when message was put */
  3 PutTime          char(8),          /* Time when message was put */
  3 ApplOriginData  char(4),          /* Application data relating to
                                     origin */
  3 GroupId          char(24),         /* Group identifier */
  3 MsgSeqNumber     fixed bin(31),    /* Sequence number of logical
                                     message within group */
  3 Offset           fixed bin(31),    /* Offset of data in physical
                                     message from start of logical
                                     message */
  3 MsgFlags         fixed bin(31),    /* Message flags */
  3 OriginalLength   fixed bin(31);   /* Length of original message */

```

## MQMD의 상위 레벨 어셈블러 선언

MQMD	DSECT	
MQMD_STRUCID	DS CL4	Structure identifier
MQMD_VERSION	DS F	Structure version number
MQMD_REPORT	DS F	Options for report messages
MQMD_MSGTYPE	DS F	Message type
MQMD_EXPIRY	DS F	Message lifetime
MQMD_FEEDBACK	DS F	Feedback or reason code
MQMD_ENCODING	DS F	Numeric encoding of message data
MQMD_CODEDCHARSETID	DS F	Character set identifier of message data
*		

MQMD_FORMAT	DS	CL8	Format name of message data
MQMD_PRIORITY	DS	F	Message priority
MQMD_PERSISTENCE	DS	F	Message persistence
MQMD_MSGID	DS	XL24	Message identifier
MQMD_CORRELID	DS	XL24	Correlation identifier
MQMD_BACKOUTCOUNT	DS	F	Backout counter
MQMD_REPLYTOQ	DS	CL48	Name of reply queue
MQMD_REPLYTOQMGR	DS	CL48	Name of reply queue manager
MQMD_USERIDENTIFIER	DS	CL12	User identifier
MQMD_ACCOUNTINGTOKEN	DS	XL32	Accounting token
MQMD_APPLIDENTITYDATA	DS	CL32	Application data relating to identity
MQMD_PUTAPPLTYPE	DS	F	Type of application that put the message
* MQMD_PUTAPPLNAME	DS	CL28	Name of application that put the message
* MQMD_PUTDATE	DS	CL8	Date when message was put
MQMD_PUTTIME	DS	CL8	Time when message was put
MQMD_APPLORIGINDATA	DS	CL4	Application data relating to origin
MQMD_GROUPID	DS	XL24	Group identifier
MQMD_MSGSEQNUMBER	DS	F	Sequence number of logical message within group
* MQMD_OFFSET	DS	F	Offset of data in physical message from start of logical message
* MQMD_MSGFLAGS	DS	F	Message flags
MQMD_ORIGINALLENGTH	DS	F	Length of original message
* MQMD_LENGTH	EQU	*-MQMD	
	ORG	MQMD	
MQMD_AREA	DS	CL(MQMD_LENGTH)	

## MQMD의 Visual Basic 선언

```

Type MQMD
  StrucId      As String*4  'Structure identifier'
  Version     As Long      'Structure version number'
  Report      As Long      'Options for report messages'
  MsgType     As Long      'Message type'
  Expiry      As Long      'Message lifetime'
  Feedback    As Long      'Feedback or reason code'
  Encoding    As Long      'Numeric encoding of message data'
  CodedCharSetId As Long      'Character set identifier of message'
  data

  Format      As String*8  'Format name of message data'
  Priority    As Long      'Message priority'
  Persistence As Long      'Message persistence'
  MsgId      As MQBYTE24  'Message identifier'
  CorrelId   As MQBYTE24  'Correlation identifier'
  BackoutCount As Long      'Backout counter'
  ReplyToQ   As String*48  'Name of reply queue'
  ReplyToQMgr As String*48  'Name of reply queue manager'
  UserIdentifier As String*12 'User identifier'
  AccountingToken As MQBYTE32 'Accounting token'
  ApplIdentityData As String*32 'Application data relating to identity'
  PutApplType As Long      'Type of application that put the'
  message

  PutApplName As String*28 'Name of application that put the'
  message

  PutDate    As String*8  'Date when message was put'
  PutTime    As String*8  'Time when message was put'
  ApplOriginData As String*4  'Application data relating to origin'
  GroupId    As MQBYTE24  'Group identifier'
  MsgSeqNumber As Long      'Sequence number of logical message'
  within group

  Offset     As Long      'Offset of data in physical message'
  from start of logical message

  MsgFlags   As Long      'Message flags'
  OriginalLength As Long      'Length of original message'
End Type

```

### MQMD의 경우 StrucId (MQCHAR4)

메시지 디스크립터 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQMD\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

### MQMD\_STRUC\_ID

메시지 디스크립터 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQMD\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQMD\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

## MQMD의 버전 (MQLONG)

이는 구조 버전 번호이며 다음 중 하나여야 합니다.

### MQMD\_VERSION\_1

버전-1 메시지 디스크립터 구조입니다.

이 버전은 모든 환경에서 지원됩니다.

### MQMD\_VERSION\_2

버전-2 메시지 디스크립터 구조입니다.

이 버전은 모든 IBM MQ V6.0 이상의 환경 및 해당 시스템에 연결된 IBM MQ MQI clients에서 지원됩니다.

**참고:** 버전-2 MQMD가 사용되는 경우 큐 관리자가 애플리케이션 메시지 데이터의 시작 부분에 있을 수 있는 MQ 헤더 구조에 대한 추가 검사를 수행합니다. 추가 세부사항은 MQPUT 호출에 대한 사용법 참고를 참조하십시오.

최신 버전의 구조에만 있는 필드는 필드의 설명에서 최신 필드로 식별됩니다. 다음 상수는 현재 버전의 버전 번호를 지정합니다.

### MQMD\_CURRENT\_VERSION

메시지 디스크립터 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQMD\_VERSION\_1입니다.

## MQMD에 대한 보고서 (MQLONG)

보고 메시지는 다른 메시지에 대한 메시지이며 원래 메시지와 관련된 예상 이벤트 또는 예상치 못한 이벤트에 대해 애플리케이션에 알리는 데 사용됩니다. *Report* 필드를 사용하면 원래 메시지를 전송하는 애플리케이션에 필요한 보고 메시지, 애플리케이션 메시지 데이터가 포함되는지 여부, 보고 및 응답 둘 다의 경우에 보고 또는 응답 메시지에서 메시지 및 상관 ID가 설정되는 방법을 지정할 수 있습니다. 다음 유형의 보고 메시지 중 일부 또는 모두(또는 없음) 요청될 수 있습니다.

- 예외
- 만기
- 도착 시 확인(COA)
- 전달 시 확인(COD)
- 긍정적 조치 알림(PAN)
- 부정적 조치 알림(NAN)

이 옵션 중 하나 이상을 지정할 수 있습니다. 둘 이상의 옵션을 지정하려면 값을 함께 추가하거나(동일한 상수를 두 번 이상 추가하지 않음), 비트 단위 OR 연산을 사용하여 값을 결합하십시오(프로그래밍 언어가 비트 연산을 지원하는 경우).

보고 메시지를 수신하는 애플리케이션은 MQMD에서 *Feedback* 필드를 검사하여 보고서가 생성된 이유를 판별할 수 있습니다. 자세한 정보는 *Feedback* 필드를 참조하십시오.

메시지를 토픽에 넣을 때 보고서 옵션을 사용하면 보고 메시지가 생성되지 않거나 한 개 또는 여러 개 생성되어 애플리케이션에 전송될 수 있습니다. 그 이유는 발행 메시지가 0, 1 또는 다수의 구독 애플리케이션에 전송될 수 있기 때문입니다.

**예외 옵션:** 예외 보고 메시지를 요청하려면 나열된 옵션 중 하나를 지정하십시오.

### MQRO\_EXCEPTION

메시지가 다른 큐 관리자로 전송되어 지정된 목적지 큐로 메시지를 배달할 수 없는 경우 메시지 채널 에이전트가 이 유형의 보고서를 생성합니다. 예를 들어, 목적지 큐 또는 중간 전송 큐가 가득 찼거나 메시지가 큐에 비해 너무 큰 경우입니다.

예외 보고 메시지의 생성은 기존 메시지의 지속성 및 기존 메시지가 통과하는 메시지 채널의 속도(보통 또는 빠름)에 따라 다릅니다.

- 모든 지속 메시지의 경우 및 보통 메시지 채널을 통과하는 비지속 메시지의 경우, 오류 조건에 대해 송신 애플리케이션에서 지정된 조치가 성공적으로 완료될 수 있는 경우에 한해서만 예외 보고서가 생성됩니다. 송신 애플리케이션은 다음 조치 중 하나를 지정하여 오류 조건 발생 시 기존 메시지의 배치를 제어할 수 있습니다.

- MQRO\_DEAD\_LETTER\_Q(원래 메시지를 데드-레터 큐에 배치합니다.)

- MQRO\_DISCARD\_MSG(원래 메시지를 제거합니다.)

송신 애플리케이션에서 지정된 조치가 성공적으로 완료될 수 없는 경우, 기존 메시지가 전송 큐에 남아 있으며 예외 보고 메시지가 생성되지 않습니다.

- 빠른 메시지 채널을 통해 이동하는 비지속 메시지의 경우 원래 메시지가 전송 큐에서 제거되고 오류 조건에 대해 지정된 조치가 성공적으로 완료되지 못한 경우에도 예외 보고서가 생성됩니다. 예를 들어, MQRO\_DEAD\_LETTER\_Q가 지정되었으나 해당 큐가 가득 차서 원래 메시지를 데드-레터 큐에 배치하지 못하면 예외 보고 메시지가 생성되고 원래 메시지가 제거됩니다.

보통 및 빠른 메시지 채널에 대한 자세한 정보는 [비지속 메시지 속도\(NPMSPEED\)](#)를 참조하십시오.

기존 메시지를 넣은 애플리케이션이 MQPUT 또는 MQPUT1 호출에서 리턴된 이유 코드로 문제점에 대해 동기적으로 알림을 받을 수 있는 경우 예외 보고서는 생성되지 않습니다.

메시지를 처리할 수 없음을 표시하기 위해 애플리케이션이 예외 보고서를 전송할 수도 있습니다. 예를 들어, 계정이 대변 한계를 초과하도록 만드는 차변 거래 등이 있습니다.

원래 메시지의 메시지 데이터는 보고 메시지에 포함되지 않습니다.

MQRO\_EXCEPTION, MQRO\_EXCEPTION\_WITH\_DATA, MQRO\_EXCEPTION\_WITH\_FULL\_DATA 중 둘 이상을 지정하지 마십시오.

#### **MQRO\_EXCEPTION\_WITH\_DATA**

이는 원래 메시지의 애플리케이션 메시지 데이터 중 첫 100바이트가 보고 메시지에 포함된다는 점을 제외하고 MQRO\_EXCEPTION과 동일합니다. 원래 메시지에 하나 이상의 MQ 헤더 구조가 포함되면 100바이트의 애플리케이션 바이트 외에 해당 구조도 보고 메시지에 포함됩니다.

MQRO\_EXCEPTION, MQRO\_EXCEPTION\_WITH\_DATA, MQRO\_EXCEPTION\_WITH\_FULL\_DATA 중 둘 이상을 지정하지 마십시오.

#### **MQRO\_EXCEPTION\_WITH\_FULL\_DATA**

전체 데이터가 포함된 필수 예외 보고서입니다.

이는 원래 메시지의 모든 애플리케이션 메시지 데이터가 보고 메시지에 포함된다는 점을 제외하고 MQRO\_EXCEPTION과 동일합니다.

MQRO\_EXCEPTION, MQRO\_EXCEPTION\_WITH\_DATA, MQRO\_EXCEPTION\_WITH\_FULL\_DATA 중 둘 이상을 지정하지 마십시오.

**만기 옵션:** 만기 보고 메시지를 요청하려면 나열된 옵션 중 하나를 지정하십시오.

#### **MQRO\_EXPIRATION**

만기 시간이 경과하여 메시지가 애플리케이션에 전달되기 전에 제거되는 경우 큐 관리자가 이 유형의 보고서를 생성합니다(*Expiry* 필드 참조). 이 옵션을 설정하지 않으면 MQRO\_EXCEPTION\_\* 옵션 중 하나를 지정하더라도 이 이유로 메시지가 제거되는 경우 보고 메시지가 생성되지 않습니다.

원래 메시지의 메시지 데이터는 보고 메시지에 포함되지 않습니다.

MQRO\_EXPIRATION, MQRO\_EXPIRATION\_WITH\_DATA, MQRO\_EXPIRATION\_WITH\_FULL\_DATA 중 둘 이상을 지정하지 마십시오.

#### **MQRO\_EXPIRATION\_WITH\_DATA**

이는 원래 메시지의 애플리케이션 메시지 데이터 중 첫 100바이트가 보고 메시지에 포함된다는 점을 제외하고 MQRO\_EXPIRATION과 동일합니다. 원래 메시지에 하나 이상의 MQ 헤더 구조가 포함되면 100바이트의 애플리케이션 바이트 외에 해당 구조도 보고 메시지에 포함됩니다.

MQRO\_EXPIRATION, MQRO\_EXPIRATION\_WITH\_DATA, MQRO\_EXPIRATION\_WITH\_FULL\_DATA 중 둘 이상을 지정하지 마십시오.

## **MQRO\_EXPIRATION\_WITH\_FULL\_DATA**

이는 원래 메시지의 모든 애플리케이션 메시지 데이터가 보고 메시지에 포함된다는 점을 제외하고 MQRO\_EXPIRATION과 동일합니다.

MQRO\_EXPIRATION, MQRO\_EXPIRATION\_WITH\_DATA, MQRO\_EXPIRATION\_WITH\_FULL\_DATA 중 둘 이상을 지정하지 마십시오.

**도착 확인 옵션:** 도착 확인 보고 메시지를 요청하려면 나열된 옵션 중 하나를 지정하십시오.

## **MQRO\_COA**

메시지가 목적지 큐에 배치될 때 목적지 큐를 소유한 큐 관리자가 이 유형의 보고서를 생성합니다. 원래 메시지의 메시지 데이터는 보고 메시지에 포함되지 않습니다.

메시지를 작업 단위의 일부로 넣고 목적지 큐가 로컬 큐인 경우 작업 단위가 커밋되는 경우에만 큐 관리자가 생성한 COA 보고 메시지가 검색됩니다.

메시지 디스크립터의 *Format* 필드가 MQFMT\_XMIT\_Q\_HEADER 또는 MQFMT\_DEAD\_LETTER\_HEADER 인 경우 COA 보고서가 생성되지 않습니다. 이로 인해 메시지를 전송 큐에 넣거나 메시지를 배달할 수 없어 데드-레터 큐에 넣는 경우 COA 보고서가 생성되지 않습니다.

IMS 브릿지 큐의 경우 COA 보고서는 메시지를 MQ 브릿지 큐에 넣을 때가 아니라 메시지가 IMS 큐에 도달할 때(IMS에서 수신된 수신확인) 생성됩니다. 이는 IMS가 활성화되지 않은 경우 IMS가 시작되고 메시지가 IMS 큐에 들어갈 때까지 COA 보고서가 생성되지 않음을 의미합니다.

메시지에 MQMD.Report=MQRO\_COA를 넣는 프로그램을 실행하는 사용자는 응답 큐에 대해 +passid 권한을 가지고 있어야 합니다. 사용자에게 +passid 권한이 없으면, COA 보고서 메시지는 응답 큐에 도달하지 못합니다. 데드-레터 큐에 보고 메시지를 넣으려고 합니다.

MQRO\_COA, MQRO\_COA\_WITH\_DATA, MQRO\_COA\_WITH\_FULL\_DATA 중 둘 이상의 지정하지 마십시오.

## **MQRO\_COA\_WITH\_DATA**

이는 원래 메시지의 애플리케이션 메시지 데이터 중 첫 100바이트가 보고 메시지에 포함된다는 점을 제외하고 MQRO\_COA와 동일합니다. 원래 메시지에 하나 이상의 MQ 헤더 구조가 포함되면 100바이트의 애플리케이션 바이트 외에 해당 구조도 보고 메시지에 포함됩니다.

MQRO\_COA, MQRO\_COA\_WITH\_DATA, MQRO\_COA\_WITH\_FULL\_DATA 중 둘 이상의 지정하지 마십시오.

## **MQRO\_COA\_WITH\_FULL\_DATA**

이는 원래 메시지의 모든 애플리케이션 메시지 데이터가 보고 메시지에 포함된다는 점을 제외하고 MQRO\_COA와 동일합니다.

MQRO\_COA, MQRO\_COA\_WITH\_DATA, MQRO\_COA\_WITH\_FULL\_DATA 중 둘 이상의 지정하지 마십시오.

**전달 확인 옵션:** 전달 확인 보고 메시지를 요청하려면 나열된 옵션 중 하나를 지정하십시오.

## **MQRO\_COD**

애플리케이션이 큐에서 메시지를 삭제하는 방법으로 목적지 큐에서 메시지를 검색할 때 큐 관리자가 이 유형의 보고서를 생성합니다. 원래 메시지의 메시지 데이터는 보고 메시지에 포함되지 않습니다.

메시지를 작업 단위의 일부로 검색하는 경우 작업 단위가 커밋될 때까지 보고서가 사용 불가능하도록 동일한 작업 단위 내에서 보고 메시지가 생성됩니다. 작업 단위가 백아웃되면 보고서가 송신되지 않습니다.

MQGMO\_MARK\_SKIP\_BACKOUT 옵션을 사용하여 메시지를 검색하는 경우 COD 메시지가 항상 생성되는 않습니다. 1차 작업 단위는 백아웃되지만 2차 작업 단위는 커밋되는 경우 메시지는 큐에서 제거되지만 COD 보고서는 생성되지 않습니다.

메시지 디스크립터의 *Format* 필드가 MQFMT\_DEAD\_LETTER\_HEADER인 경우 COD 보고서가 생성되지 않습니다. 이로 인해 메시지를 배달할 수 없어 데드-레터 큐에 넣는 경우 COD 보고서가 생성되지 않습니다.

목적지 큐가 XCF 큐인 경우에는 MQRO\_COD가 올바르지 않습니다.

MQRO\_COD, MQRO\_COD\_WITH\_DATA, MQRO\_COD\_WITH\_FULL\_DATA 중 둘 이상을 지정하지 마십시오.

## **MQRO\_COD\_WITH\_DATA**

이는 원래 메시지의 애플리케이션 메시지 데이터 중 첫 100바이트가 보고 메시지에 포함된다는 점을 제외하고 MQRO\_COD와 동일합니다. 원래 메시지에 하나 이상의 MQ 헤더 구조가 포함되면 100바이트의 애플리케이션 바이트 외에 해당 구조도 보고 메시지에 포함됩니다.



원래 메시지에 대해 MQGET 호출에서 MQGMO\_ACCEPT\_TRUNCATED\_MSG가 지정되고 검색된 메시지가 잘린 경우 보고 메시지에 배치되는 애플리케이션 메시지 데이터의 양은 환경에 따라 다릅니다.

- z/OS에서 이는 다음의 최소값입니다.
  - 기존 메시지의 길이
  - 메시지를 검색하는 데 사용되는 버퍼의 길이
  - 100바이트
- 다른 환경에서 이는 다음의 최소값입니다.
  - 기존 메시지의 길이
  - 100바이트

목적지 큐가 XCF 큐인 경우 MQRO\_COD\_WITH\_DATA가 올바르지 않습니다.

MQRO\_COD, MQRO\_COD\_WITH\_DATA, MQRO\_COD\_WITH\_FULL\_DATA 중 둘 이상을 지정하지 마십시오.

#### **MQRO\_COD\_WITH\_FULL\_DATA**

이는 원래 메시지의 모든 애플리케이션 메시지 데이터가 보고 메시지에 포함된다는 점을 제외하고 MQRO\_COD와 동일합니다.

목적지 큐가 XCF 큐인 경우 MQRO\_COD\_WITH\_FULL\_DATA가 올바르지 않습니다.

MQRO\_COD, MQRO\_COD\_WITH\_DATA, MQRO\_COD\_WITH\_FULL\_DATA 중 둘 이상을 지정하지 마십시오.

**조치 알림 옵션:** 수신 애플리케이션이 긍정적인 조치 또는 부정적인 조치 보고 메시지를 전송하도록 요청하려면 나열된 옵션 중 하나 또는 둘 다를 지정하십시오.

#### **MQRO\_PAN**

이 유형의 보고서는 메시지를 검색하고 이에 대해 조치를 취하는 애플리케이션에 의해 생성됩니다. 이는 메시지에 요청된 조치가 성공적으로 수행되었음을 나타냅니다. 보고서를 생성하는 애플리케이션은 데이터가 보고서에 포함되는지 여부를 판별합니다.

메시지를 검색하는 애플리케이션에 이 요청을 전달하는 것 외에는 큐 관리자가 이 옵션을 기반으로 하여 어떠한 조치도 수행하지 않습니다. 검색 애플리케이션은 적절한 경우 보고서를 생성해야 합니다.

#### **MQRO\_NAN**

이 유형의 보고서는 메시지를 검색하고 이에 대해 조치를 취하는 애플리케이션에 의해 생성됩니다. 이는 메시지에 요청된 조치가 성공적으로 수행되지 않았음을 나타냅니다. 보고서를 생성하는 애플리케이션은 데이터가 보고서에 포함되는지 여부를 판별합니다. 예를 들어, 요청을 수행할 수 없는 이유를 표시하는 일부 데이터를 포함하기를 원할 수도 있습니다.

메시지를 검색하는 애플리케이션에 이 요청을 전달하는 것 외에는 큐 관리자가 이 옵션을 기반으로 하여 어떠한 조치도 수행하지 않습니다. 검색 애플리케이션은 적절한 경우 보고서를 생성해야 합니다.

애플리케이션은 긍정적인 조치에 해당하는 조건 및 부정적인 조치에 해당하는 조건을 판별해야 합니다. 그러나 요청이 부분적으로만 수행된 경우 요청에 따라 PAN 보고서가 아니라 NAN 보고서가 생성됩니다. 가능한 모든 조건이 긍정적인 조치 또는 부정적인 조치에 해당되어야 하지만 둘 다는 아닙니다.

**메시지 ID 옵션:** 보고 메시지 또는 응답 메시지의 *MsgId*를 설정하는 방법을 제어하려면 나열된 옵션 중 하나를 지정하십시오.

#### **MQRO\_NEW\_MSG\_ID**

이는 기본 조치입니다. 보고서 또는 응답이 이 메시지의 결과로 생성되는지, 새 *MsgId*가 보고 메시지 또는 응답 메시지에 대해 생성되는지 여부를 표시합니다.

#### **MQRO\_PASS\_MSG\_ID**

보고서 또는 응답이 이 메시지의 결과로 생성되는 경우 이 메시지의 *MsgId*가 보고 메시지 또는 응답 메시지의 *MsgId*에 복사됩니다.

발행 메시지의 *MsgId*는 발행물의 사본을 수신하는 각 구독자마다 다르므로, 보고 또는 응답 메시지에 복사된 *MsgId*가 각각 다릅니다.

이 옵션을 지정하지 않으면 MQRO\_NEW\_MSG\_ID인 것으로 가정됩니다.

**상관 ID 옵션:** 보고 메시지 또는 응답 메시지의 *CorrelId*를 설정하는 방법을 제어하려면 나열된 옵션 중 하나를 지정하십시오.

#### **MQRO\_COPY\_MSG\_ID\_TO\_CORREL\_ID**

이는 기본 조치이며 보고서 또는 응답이 이 메시지의 결과로 생성되는 경우 이 메시지의 *MsgId*가 보고 메시지 또는 응답 메시지의 *CorrelId*에 복사됨을 표시합니다.

발행 메시지의 *MsgId*는 발행물의 사본을 수신하는 각 구독자마다 다르므로 보고 또는 응답 메시지의 *CorrelId*에 복사된 *MsgId*가 각각 다릅니다.

#### **MQRO\_PASS\_CORREL\_ID**

보고서 또는 응답이 이 메시지의 결과로 생성되는 경우 이 메시지의 *CorrelId*가 보고 메시지 또는 응답 메시지의 *CorrelId*에 복사됩니다.

발행 메시지의 *CorrelId*가 MQSO\_SET\_CORREL\_ID 옵션을 사용하고 MQSD에서 SubCorrelId 필드를 MQCI\_NONE으로 설정하지 않는 경우 한 구독자에 대해 한정됩니다. 따라서 보고 메시지 또는 응답 메시지의 *CorrelId*에 복사되는 *CorrelId*가 각각 다를 수 있습니다.

이 옵션을 지정하지 않으면 MQRO\_COPY\_MSG\_ID\_TO\_CORREL\_ID인 것으로 가정됩니다.

요청에 응답하거나 보고 메시지를 생성하는 서버는 원래 메시지에서 MQRO\_PASS\_MSG\_ID 또는 MQRO\_PASS\_CORREL\_ID 옵션이 설정되어 있는지 확인해야 합니다. 설정되어 있는 경우 서버는 해당 옵션에서 설명된 조치를 수행해야 합니다. 설정되어 있지 않은 경우 서버는 해당 기본 조치를 수행해야 합니다.

**처리 옵션:** 원래 메시지를 목적지 큐에 전달할 수 없는 경우 처리를 제어하려면 나열된 옵션 중 하나를 지정하십시오. 애플리케이션은 예외 보고서 요청과 상관없이 배치 옵션을 설정할 수 있습니다.

#### **MQRO\_DEAD\_LETTER\_Q**

이는 기본 조치이며 메시지를 목적지 큐에 배달할 수 없는 경우 데드-레터 큐에 메시지를 배치합니다. 이는 다음 상황에서 발생합니다.

- 기존 메시지를 넣은 애플리케이션이 MQPUT 또는 MQPUT1 호출에서 리턴된 이유 코드로 문제점에 대해 동기적으로 알림을 받을 수 없는 경우입니다. 송신자에 의해 요청된 경우, 예외 보고 메시지가 생성됩니다.
- 기존 메시지를 넣은 애플리케이션을 토픽에 넣는 경우

#### **MQRO\_DISCARD\_MSG**

메시지를 목적지 큐에 배달할 수 없는 경우 메시지를 제거합니다. 이는 다음 상황에서 발생합니다.

- 기존 메시지를 넣은 애플리케이션이 MQPUT 또는 MQPUT1 호출에서 리턴된 이유 코드로 문제점에 대해 동기적으로 알림을 받을 수 없는 경우입니다. 송신자에 의해 요청된 경우, 예외 보고 메시지가 생성됩니다.
- 기존 메시지를 넣은 애플리케이션을 토픽에 넣는 경우

원래 메시지를 데드-레터 큐에 배치하지 않고 송신자에게 리턴하려는 경우 송신자가 MQRO\_EXCEPTION\_WITH\_FULL\_DATA와 함께 MQRO\_DISCARD\_MSG를 지정해야 합니다.

#### **MQRO\_PASS\_DISCARD\_AND\_EXPIRY**

메시지에 이 옵션이 설정되고 보고서 또는 응답이 이로 인해 생성된 경우 보고서의 메시지 디스크립터가 상속됩니다.

- MQRO\_DISCARD\_MSG(설정된 경우)
- 메시지의 남아 있는 만기 날짜(만기 보고서가 아닌 경우). 만기 보고서인 경우 만기 시간이 60초로 설정됩니다.

#### **활동 옵션**

##### **MQRO\_ACTIVITY**

이 값을 사용하면 큐 관리자 네트워크에서 모든 메시지의 라우트가 추적되도록 설정할 수 있습니다. 보고서 옵션은 현재 사용자 메시지에서 지정할 수 있으며 지정하는 즉시 사용자가 네트워크를 통해 메시지의 라우트를 계산할 수 있습니다.

메시지를 생성하는 애플리케이션이 활동 보고서를 사용 가능하게 할 수 없는 경우, 큐 관리자가 제공하는 API 교차 엑시트를 사용하여 보고를 사용 가능하게 할 수 있습니다.

#### **참고:**

1. 활동 보고서를 생성할 수 있는 네트워크 내의 큐 관리자 수가 적을수록 라우트가 자세하지 않습니다.

2. 지나온 라우트를 판별하기 위해 활동 보고서를 올바른 순서로 배치하기 어려울 수 있습니다.
3. 활동 보고서가 요청된 목적지에 대한 라우트를 발견하지 못할 수 있습니다.
4. 이 보고서 옵션이 설정된 메시지는 모든 큐 관리자에 의해 승인되어야 합니다. 큐 관리자가 옵션을 이해하지 못하는 경우에도 해당됩니다. 이는 사용자 메시지를 IBM WebSphere MQ 6.0 이상이 아닌 큐 관리자가 처리하는 경우에도 이에 대해 보고서 옵션을 설정할 수 있게 해 줍니다.
5. 큐 관리자 또는 사용자 프로세스 등의 프로세스가 이 옵션이 설정된 메시지에서 활동을 수행하는 경우 활동 보고서를 생성하고 넣도록 선택할 수 있습니다.

**기본 옵션:** 보고서 옵션이 필요하지 않은 경우 다음을 지정하십시오.

#### **MQRO\_NONE**

기타 옵션을 지정하지 않도록 표시하려면 이 값을 사용하십시오. MQRO\_NONE은 프로그램 문서화를 지원하기 위해 정의됩니다. 이 옵션은 기타와 함께 사용하기 위한 용도가 아니지만, 해당 값이 0이므로 해당 사용을 감지할 수 없습니다.

#### **일반 정보:**

1. 필요한 모든 보고서 유형은 원래 메시지를 전송하는 애플리케이션에 의해 특별히 요청되어야 합니다. 예를 들어, COA 보고서는 요청되었으나 예외 보고서는 요청되지 않은 경우 메시지가 목적지 큐에 배치되면 COA 보고서가 생성되지만 메시지가 도달할 때 목적지 큐가 가득 찬 경우 예외 보고서가 생성되지 않습니다. Report 옵션이 설정되지 않은 경우 큐 관리자 또는 메시지 채널 에이전트(MCA)가 보고 메시지를 생성하지 않습니다.

로컬 큐 관리자가 인식하지 않는 경우에도 일부 보고서 옵션을 지정할 수 있습니다. 이는 옵션이 목적지 큐 관리자에 의해 처리되는 경우에 유용합니다. 자세한 내용은 865 페이지의 『보고 옵션 및 메시지 플래그』의 내용을 참조하십시오.

보고 메시지가 요청되는 경우 ReplyToQ 필드에서 보고서를 전송할 큐의 이름을 지정해야 합니다. 보고 메시지가 수신될 때 메시지 디스크립터에서 Feedback 필드를 검사하여 보고서의 네이처를 판별할 수 있습니다.

2. 보고 메시지를 생성하는 큐 관리자 또는 MCA가 보고 메시지를 응답 큐에 넣을 수 없는 경우(예를 들어, 응답 큐 또는 전송 큐가 가득 찬 경우) 보고 메시지가 대신 데드-레터 큐에 배치됩니다. 해당 작업도 실패하는 경우 또는 데드-레터 큐가 없는 경우 수행되는 조치는 보고 메시지의 유형에 따라 다릅니다.
  - 보고 메시지가 예외 보고서인 경우 예외 보고서를 생성한 메시지가 전송 큐에 남아 있습니다. 이로 인해 메시지가 손실되지 않습니다.
  - 모든 기타 보고서 유형의 경우, 보고 메시지가 제거되고 정상적으로 계속 처리됩니다. 이는 기존 메시지가 이미 안전하게 전달되었거나(COA 또는 COD 보고 메시지의 경우) 더 이상 관련되지 않으므로(만기 보고 메시지의 경우) 수행됩니다.

보고 메시지가 큐에 성공적으로 배치되었으면(목적지 큐 또는 중간 전송 큐), 메시지는 더 이상 특수 처리 대상이 아닙니다. 이는 다른 메시지처럼 처리됩니다.

3. 보고서가 생성될 때 ReplyToQ 큐가 열리고 보고서가 발생한 원인이 되는 메시지의 MQMD에 있는 UserIdentifier의 권한을 사용하여 보고 메시지를 넣습니다. 단, 다음 경우는 예외입니다.
  - 보고서의 원인이 되는 메시지를 넣으려고 시도했을 때 MCA가 사용한 권한에 상관없이, 수신 MCA에서 생성된 예외 보고서를 넣습니다.
  - 큐 관리자가 생성한 COA 보고서는 보고서 발생 원인이 되는 메시지를 보고서를 생성한 큐 관리자에 넣을 때 사용한 권한을 사용하여 넣습니다. 예를 들어, MCA의 사용자 ID를 통해 수신 MCA에서 메시지를 넣은 경우, 큐 관리자가 MCA의 사용자 ID를 통해 COA 보고서를 넣습니다.

보고서를 생성하는 애플리케이션은 응답을 생성하기 위해 사용한 것과 동일한 권한을 사용해야 합니다. 일반적으로 원래 메시지에 있는 사용자 ID의 권한입니다.

보고서가 원격 목적지로 이동해야 하는 경우, 송신자 및 수신자는 기타 메시지에 대해 수행하는 동일한 방식으로 이를 허용할지 여부를 결정할 수 있습니다.

4. 데이터가 있는 보고 메시지가 요청된 경우:
  - 보고 메시지는 기존 메시지의 송신자가 요청한 데이터의 양으로 항상 생성됩니다. 보고 메시지가 응답 큐에 비해 너무 큰 경우 위에서 설명한 처리가 발생합니다. 응답 큐에 맞게 보고 메시지를 자르는 경우는 없습니다.

- 원래 메시지의 *Format*이 MQFMT\_XMIT\_Q\_HEADER인 경우 보고서에 포함된 데이터가 MQXQH를 포함하지 않습니다. 보고서 데이터는 원래 메시지의 MQXQH 위에 있는 데이터의 첫 번째 바이트부터 시작됩니다. 이는 큐가 전송 큐인지 여부에 상관없이 발생합니다.
5. COA, COD 또는 만기 보고 메시지가 응답 큐에 수신되는 경우 기존 메시지가 적절한 대로 도달했고 전달되었거나 만기되었음이 확실합니다. 그러나 이러한 보고 메시지 중 하나 이상이 요청되었으나 수신되지 않은 경우 다음 중 하나가 발생했을 수 있으므로 반대로 가정할 수 없습니다.

- a. 링크가 끊어져 보고 메시지가 보류됩니다.
- b. 블로킹 조건이 중간 전송 큐 또는 응답 큐에 있으므로 보고 메시지가 보류됩니다(예를 들어, 큐가 가득 차거나 Put에 대해 상속됨).
- c. 보고 메시지가 데드-레터 큐에 있습니다.
- d. 큐 관리자가 보고 메시지를 생성하려고 시도할 때 적절한 큐에도 넣지 않고 데드-레터 큐에도 넣지 않아 보고 메시지를 생성할 수 없습니다.
- e. 보고되는 조치(도달, 배달 또는 만기) 및 해당 보고 메시지의 생성 간에 큐 관리자 실패가 발생했습니다. COD 보고 메시지가 동일한 작업 단위 내에 생성되어, 애플리케이션이 작업 단위 내에서 기존 메시지를 검색하는 경우, 이는 COD 보고 메시지에 대해 발생하지 않습니다.

위의 이유 1, 2, 3으로 인해 동일한 방법으로 예외 보고 메시지를 보류할 수 있습니다. 그러나 MCA가 예외 보고 메시지를 생성할 수 없는 경우(보고 메시지를 응답 큐 또는 데드-레터 큐에 넣을 수 없는 경우) 원래 메시지가 송신자에서 전송 큐에 남아 있게 되고 채널이 닫힙니다. 이는 보고 메시지가 채널의 송신 또는 수신 끝에서 생성되는지 여부에 상관없이 발생합니다.

6. 원래 메시지가 임시로 차단되었으나(예외 보고 메시지가 생성되고 원래 메시지를 데드-레터 큐에 넣은 경우) 차단이 해제된 다음 애플리케이션이 데드-레터 큐에서 원래 메시지를 읽고 다시 목적지에 넣는 경우 다음이 발생할 수 있습니다.
  - 예외 보고 메시지가 생성된 경우에도, 기존 메시지가 결국 해당 목적지에 성공적으로 도달합니다.
  - 하나의 원래 메시지에 대해 둘 이상의 예외 보고 메시지가 생성됩니다. 그 이유는 원래 메시지에 나중에 다른 차단이 발생할 수 있기 때문입니다.

#### 토픽에 넣을 때 보고 메시지:

1. 메시지를 토픽에 넣는 경우, 보고서가 생성될 수 있습니다. 이 메시지는 토픽의 모든 구독자에게 전송되어 0, 1 또는 다수가 될 수 있습니다. 이는 결과적으로 다수의 보고 메시지가 생성될 수 있어서 보고서 옵션을 사용하도록 선택할 때 고려되어야 합니다.
2. 토픽에 메시지를 넣을 때, 메시지의 사본이 제공되는 다수의 목적지 큐가 있을 수 있습니다. 이러한 목적지 큐 중 일부에 큐가 가득 차는 등의 문제점이 있는 경우, MQPUT의 성공적인 완료는 NPMSGDLV 또는 PMSGDLV의 설정에 따라 다릅니다(메시지의 지속성에 따라). 목적지 큐에 대한 메시지 전달이 성공적이어야 하도록 설정된 경우(예를 들어, 지속형 구독자에 대한 지속 메시지이며 PMSGDLV가 ALL 또는 ALLDUR로 설정됨), 다음 기준 중 하나가 충족되면 성공으로 정의됩니다.
  - 구독자 큐에 성공적으로 넣음
  - 구독자 큐가 메시지를 수용할 수 없을 때 MQRO\_DEAD\_LETTER\_Q를 사용하고 데드-레터 큐에 성공적으로 넣은 경우
  - 구독자 큐가 메시지를 수용할 수 없을 때 MQRO\_DISCARD\_MSG를 사용한 경우

#### 메시지 세그먼트에 대한 보고 메시지:

1. 세그먼트화가 허용된 메시지에 대해 보고 메시지를 요청할 수 있습니다. MQMF\_SEGMENTATION\_ALLOWED 플래그에 대한 설명을 참조하십시오. 큐 관리자가 메시지를 세그먼트화하는 것이 필요하다는 것을 발견하면 이후에 관련 조건이 발생하는 각 세그먼트에 대해 보고 메시지가 생성될 수 있습니다. 애플리케이션은 요청된 각 유형의 보고 메시지에 대해 다중 보고 메시지를 수신할 준비가 되어 있어야 합니다. 다중 보고서와 원래 메시지의 그룹 ID의 상관 관계를 지정하려면 보고 메시지의 *GroupId* 필드를 사용하고 각 보고 메시지의 유형을 식별하려면 *Feedback* 필드를 사용하십시오.
2. 세그먼트에 대한 보고 메시지를 검색하기 위해 MQGMO\_LOGICAL\_ORDER를 사용한 경우 다른 유형의 해당 보고서가 후속 MQGET 호출에 의해 리턴될 수 있습니다. 예를 들어, 큐 관리자가 세그먼트화하는 메시지에 대해 COA 및 COD 보고서가 둘 다 요청되면 보고 메시지에 대한 MQGET 호출이 예상치 못한 방법으로 COA 및 COD 메시지 인터리브를 리턴할 수 있습니다. MQGMO\_COMPLETE\_MSG 옵션을 사용하여(선택적으로 MQGMO\_ACCEPT\_TRUNCATED\_MSG 사용) 이를 방지하십시오. MQGMO\_COMPLETE\_MSG를 사용하면 큐

관리자가 보고서 유형이 동일한 보고 메시지를 리어셈블링합니다. 예를 들어, 첫 번째 MQGET 호출이 원래 메시지와 연관된 모든 COA 메시지를 리어셈블링하고 두 번째 MQGET 호출이 모든 COD 메시지를 리어셈블링할 수 있습니다. 처음 리어셈블링되는 대상은 큐에서 처음 발생하는 보고 메시지의 유형에 따라 결정됩니다.

3. 직접 세그먼트를 넣는 애플리케이션은 각 세그먼트에 대해 서로 다른 보고서 옵션을 지정할 수 있습니다. 그러나 다음 사항에 주의하십시오.
  - 세그먼트가 MQGMO\_COMPLETE\_MSG 옵션을 사용하여 검색되는 경우 첫 번째 세그먼트의 보고서 옵션만 큐 관리자에 의해 허용됩니다.
  - 세그먼트가 하나씩 검색되며 대부분 MQRO\_COD\_\* 옵션 중 하나를 갖고 있으나 하나 이상의 세그먼트는 갖고 있지 않은 경우 MQGMO\_COMPLETE\_MSG 옵션을 사용하여 단일 MQGET 호출이 있는 보고 메시지를 검색할 수 없거나 MQGMO\_ALL\_SEGMENTS\_AVAILABLE 옵션을 사용하여 모든 보고 메시지가 도착한 시기를 발견할 수 없습니다.
4. MQ 네트워크에서 큐 관리자가 다른 기능을 가질 수 있습니다. 세그먼트화를 지원하지 않는 큐 관리자 또는 MCA에 의해 세그먼트에 대한 보고 메시지가 생성되는 경우 큐 관리자 또는 MCA가 기본적으로 보고 메시지에 필수 세그먼트 정보를 포함하지 않으며 이로 인해 보고서가 생성되는 원인이 된 원래 메시지를 식별하기 어려울 수 있습니다. 보고 메시지가 있는 데이터를 요청하여, 즉, 적절한 MQRO\_\*\_WITH\_DATA 또는 MQRO\_\*\_WITH\_FULL\_DATA 옵션을 지정하여 이러한 어려움을 방지하십시오. 그러나 세그먼트화를 지원하지 않는 큐 관리자 또는 MCA에 의해 보고 메시지가 생성되었을 때 MQRO\_\*\_WITH\_DATA가 지정된 경우 100 바이트 미만인 애플리케이션 메시지 데이터가 보고 메시지를 검색하는 애플리케이션에 리턴될 수 있습니다.

**보고 메시지에 대한 메시지 디스크립터의 콘텐츠:** 큐 관리자 또는 메시지 채널 에이전트(MCA)가 보고 메시지를 생성하는 경우, 메시지 디스크립터의 필드를 다음 값으로 설정한 후 정상적인 방법으로 메시지를 넣습니다.

표 501. 보고 메시지가 시스템에서 생성되는 경우에 MQMD 필드에 대해 사용되는 값

MQMD의 필드	사용된 값
<i>StrucId</i>	MQMD_STRUC_ID
<i>Version</i>	MQMD_VERSION_2
<i>Report</i>	MQRO_NONE
<i>MsgType</i>	MQMT_REPORT
<i>Expiry</i>	MQEI_UNLIMITED
<i>Feedback</i>	보고서의 네이처에 대해 적절하게 사용(MQFB_COA, MQFB_COD, MQFB_EXPIRATION 또는 MQRC_* 값)
<i>Encoding</i>	원래 메시지 디스크립터로부터 복사됩니다.
<i>CodedCharSetId</i>	원래 메시지 디스크립터로부터 복사됩니다.
<i>Format</i>	원래 메시지 디스크립터로부터 복사됩니다.
<i>Priority</i>	원래 메시지 디스크립터로부터 복사됩니다.
<i>Persistence</i>	원래 메시지 디스크립터로부터 복사됩니다.
<i>MsgId</i>	원래 메시지 디스크립터의 보고서 옵션에 의해 지정됨
<i>CorrelId</i>	원래 메시지 디스크립터의 보고서 옵션에 의해 지정됨
<i>BackoutCount</i>	0
<i>ReplyToQ</i>	공백
<i>ReplyToQMgr</i>	큐 관리자의 이름
<i>UserIdentifier</i>	MQPMO_PASS_IDENTITY_CONTEXT 옵션에 의해 설정된 대로 사용됨
<i>AccountingToken</i>	MQPMO_PASS_IDENTITY_CONTEXT 옵션에 의해 설정된 대로 사용됨
<i>ApplIdentityData</i>	MQPMO_PASS_IDENTITY_CONTEXT 옵션에 의해 설정된 대로 사용됨
<i>PutApplType</i>	MQAT_QMGR 또는 메시지 채널 에이전트에 대해 적절하게 사용됨

MQMD의 필드	사용된 값
<i>PutApplName</i>	큐 관리자 이름 또는 메시지 채널 에이전트 이름의 첫 번째 28바이트입니다. IMS 브릿지에서 생성된 보고 메시지의 경우, 이 필드에는 메시지와 관련된 IMS 시스템의 XCF 그룹 이름 및 XCF 멤버 이름이 포함됩니다.
<i>PutDate</i>	보고 메시지가 송신된 날짜
<i>PutTime</i>	보고 메시지를 송신한 시간
<i>ApplOriginData</i>	공백
<i>GroupId</i>	원래 메시지 디스크립터로부터 복사됩니다.
<i>MsgSeqNumber</i>	원래 메시지 디스크립터로부터 복사됩니다.
<i>Offset</i>	원래 메시지 디스크립터로부터 복사됩니다.
<i>MsgFlags</i>	원래 메시지 디스크립터로부터 복사됩니다.
<i>OriginalLength</i>	MQOL_UNDEFINED가 아닌 경우 원래 메시지 디스크립터로부터 복사되며 그렇지 않은 경우 원래 메시지 데이터의 길이로 설정됩니다.

보고서를 생성하는 애플리케이션은 다음을 제외하고 유사한 값을 설정하도록 권장됩니다.

- *ReplyToQMgr* 필드를 공백으로 설정할 수 있습니다(메시지를 넣을 때 큐 관리자가 이를 로컬 큐 관리자의 이름으로 변경함).
- 응답에 대해 사용된 옵션(일반적으로 MQPMO\_PASS\_IDENTITY\_CONTEXT)을 사용하여 컨텍스트 필드를 설정하십시오.

**보고서 필드 분석:** *Report* 필드에는 하위 필드가 포함되어 있습니다. 이로 인해 메시지의 송신자가 특정 보고서를 요청했는지 여부를 확인해야 하는 애플리케이션은 867 페이지의 『보고서 필드 분석』에서 설명하는 기술 중 하나를 사용해야 합니다.

이 필드는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 초기값은 MQRO\_NONE입니다.

### MQMD의 경우 *MsgType* (MQLONG)

이는 메시지의 유형을 표시합니다. 다음과 같이 메시지 유형을 분류할 수 있습니다.

#### MQMT\_SYSTEM\_FIRST

시스템에서 정의된 메시지 유형에 대한 최저값입니다.

#### MQMT\_SYSTEM\_LAST

시스템에서 정의된 메시지 유형에 대한 최고값입니다.

현재 시스템 범위 내에서 정의된 값은 다음과 같습니다.

#### MQMT\_DATAGRAM

응답을 요청하지 않는 메시지입니다.

#### MQMT\_REQUEST

응답이 필요한 메시지입니다.

*ReplyToQ* 필드에서 응답을 전송하는 큐의 이름을 지정하십시오. *Report* 필드는 응답의 *MsgId* 및 *CorrelId*를 설정하는 방법을 표시합니다.

#### MQMT\_REPLY

메시지는 이전 요청 메시지(MQMT\_REQUEST)에 대한 응답입니다. 메시지는 요청 메시지의 *ReplyToQ* 필드가 표시하는 큐에 전송되어야 합니다. 요청의 *Report* 필드를 사용하여 응답의 *MsgId* 및 *CorrelId*를 설정하는 방법을 제어하십시오.

**참고:** 큐 관리자는 요청-응답 관계를 강요하지 않습니다. 이는 애플리케이션의 책임입니다.

## MQMT\_REPORT

메시지는 일반적으로 일부 다른 메시지에 관련된 예상된 발생 또는 예상치 못한 발생을 보고합니다. 예를 들어, 올바르지 않은 데이터를 포함한 요청 메시지가 수신되는 경우 등을 보고합니다. 원래 메시지의 메시지 디스크립터의 *ReplyToQ* 필드가 표시하는 큐에 메시지를 전송하십시오. 보고서의 네이처를 표시하려면 *Feedback* 필드를 설정하십시오. 원래 메시지의 *Report* 필드를 사용하여 보고 메시지의 *MsgId* 및 *CorrelId*를 설정하는 방법을 제어하십시오.

큐 관리자 또는 메시지 채널 에이전트가 생성하는 보고 메시지는 항상 *Feedback* 및 *CorrelId* 필드가 위에서 설명된 대로 설정된 상태로 *ReplyToQ* 큐에 전송됩니다.

애플리케이션에서 정의된 값을 사용할 수도 있습니다. 값은 다음 범위 내에 있어야 합니다.

## MQMT\_APPL\_FIRST

애플리케이션에서 정의된 메시지 유형에 대한 최저값입니다.

## MQMT\_APPL\_LAST

애플리케이션에서 정의된 메시지 유형에 대한 최고값입니다.

MQPUT 및 MQPUT1 호출의 경우 *MsgType* 값은 시스템 정의된 범위 또는 애플리케이션 정의된 범위 내에 있어야 합니다. 그렇지 않은 경우 이유 코드 MQRC\_MSG\_TYPE\_ERROR와 함께 호출이 실패합니다.

이는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 초기값은 MQMT\_DATAGRAM입니다.

## MQMD의 만기 (MQLONG)

이는 메시지를 넣는 애플리케이션이 설정한, 10분의 1초로 표시되는 일정 시간입니다. 이 기간이 경과하기 전에 목적지 큐에서 메시지를 제거하지 않은 경우, 이 메시지를 버릴 수 있습니다.

예를 들어, 만기 시간으로 1분을 설정하려면 MQMD를 설정해야 합니다. **Expiry -600**.

메시지가 목적지 큐에서 소비하는 시간 및 리모트 큐에 메시지를 넣는 경우 중간 전송 큐에서 소비하는 시간을 반영하기 위해 값이 감소됩니다. 이 값은 전송 횟수가 중요한 경우 이를 반영하기 위해 메시지 채널 에이전트에 의해서도 감소될 수 있습니다. 마찬가지로 이 메시지를 다른 큐에 전달하는 애플리케이션은 상당한 시간 동안 메시지를 보유한 경우 필요하면 해당 값을 감소시킬 수 있습니다. 그러나, 만기 시간이 근사치로 처리되며 작은 시간 간격을 반영하는 데 값을 감량할 필요는 없습니다.

MQGET 호출을 사용하여 애플리케이션이 메시지를 검색할 때 *Expiry* 필드는 아직 남아 있는 만기 시간을 나타냅니다.

메시지의 만기 시간이 경과된 후에는 큐 관리자가 이를 제거할 수 있습니다. 메시지는 이미 만료되지 않은 메시지를 리턴하는 찾아보기 또는 찾아보기를 하지 않는 MQGET 호출이 발생할 때 제거됩니다. 예를 들어, 찾아보기를 하지 않는 MQGET 호출(MQMO의 *MatchOptions* 필드가 MQMO\_NONE으로 설정됨)은 선입선출(FIFO)로 순서화된 큐에서 읽고 만료되지 않은 메시지를 첫 번째 만료되지 않은 메시지까지 모두 제거합니다. 우선순위 정렬 큐에서는 동일한 호출이 높은 우선순위의 만기된 메시지 및 첫 번째 만기되지 않은 메시지 이전에 큐에 도착한 동일한 우선순위의 메시지를 제거합니다.

만료된 메시지는 애플리케이션으로 리턴되지 않으므로(찾아보기 또는 찾아보기를 하지 않는 MQGET 호출에 의해) MQGET 호출을 성공한 후 메시지 디스크립터의 *Expiry* 필드 값은 0보다 크거나 특수 값 MQEI\_UNLIMITED입니다.

메시지를 리모트 큐에 넣는 경우 메시지는 메시지가 목적지 큐에 도달하기 전에 중간 전송 큐에 있는 동안 만료(및 제거)될 수 있습니다.

메시지에 MQRO\_EXPIRATION\_\* 보고 옵션 중 하나가 지정된 경우 만료된 메시지가 제거될 때 보고서가 생성됩니다. 이러한 옵션 중 지정된 옵션이 없으면, 해당 보고서가 생성되지 않습니다. 메시지는 이 기간 이후 더 이상 관련되지 않는 것으로 간주됩니다(이후 메시지가 이를 대체했기 때문일 수 있음).

동기점 내의 메시지 넣기의 경우 만기 간격은 동기점이 커밋되는 시간이 아니라 메시지를 넣는 시간에 시작합니다. 동기점이 커밋되기 전에 만기 간격이 지나갈 수 있습니다. 이 경우 메시지는 커밋 조작 후 어느 시점에 제거되고 메시지는 MQGET 조작에 대한 응답으로 애플리케이션에 리턴되지 않습니다.

만기 시간에 기준하여 메시지를 제거하는 기타 프로그램은 요청된 경우 적절한 보고 메시지도 전송해야 합니다.

**참고:**

1. 메시지를 0 또는 999,999,999보다 큰 수의 *Expiry* 시간으로 넣는 경우 MQPUT 또는 MQPUT1 호출이 이유 코드 MQRC\_EXPIRY\_ERROR와 함께 실패합니다. 이 경우 보고 메시지는 생성되지 않습니다.

이유 코드 2013, MQRC\_EXPIRY\_ERROR를 사용으로 설정하려면, 환경 변수 AMQ\_ENFORCE\_MAX\_EXPIRY\_ERROR를 사용으로 설정해야 합니다.

다음은 Linux용 예를 사용합니다.

```
$ export AMQ_ENFORCE_MAX_EXPIRY_ERROR=True
```

다음에 유의하십시오.

- 변수를 내보내기하는 것이 중요함
  - 실제 값은 무시되지만 설정 검토 시 True 사용이 도움이 될 수 있습니다.
2. 만기 시간이 경과된 메시지는 나중까지 제거되지 않을 수 있으므로 해당 만기 시간이 지나서 검색할 수 없는 메시지가 큐에 있을 수 있습니다. 그러나 이들 메시지는 용량 트리거를 포함하여 큐에 있는 메시지의 수에 가산됩니다.  
구독자/이용자(클라이언트)가 메시지를 가져오려고 시도하고 해당 메시지가 만기된 경우, 메시지가 너무 오래되어 메시지가 제거되었으므로 클라이언트가 아무 것도 수신하지 않습니다. 또한 클라이언트는 오류 메시지를 수신하지 않습니다.
  3. 만기 보고서는 메시지가 제거되었을 때(제거될 수 있을 때가 아님) 요청된 경우에 생성됩니다.
  4. 작업 단위 내에서 MQGET 호출 조작의 결과로 메시지가 제거되도록 스케줄되었더라도 만료된 메시지 제거 및 요청 시 만기 보고서 생성은 애플리케이션의 작업 단위에 포함되지 않습니다.
  5. 작업 단위 내에서 MQGET 호출에 의해 거의 만료된 메시지가 검색된 경우 작업 단위가 그 뒤에 백아웃되면 메시지는 다시 검색되기 전에 제거될 수 있습니다.
  6. 거의 만료된 메시지가 MQGMO\_LOCK을 사용한 MQGET 호출로 잠겨진 경우, 이 메시지는 MQGMO\_MSG\_UNDER\_CURSOR를 사용한 MQGET 호출로 검색되기 전에 제거될 수 있습니다. 이러한 경우 나중 MQGET 호출에 이유 코드 MQRC\_NO\_MSG\_UNDER\_CURSOR가 리턴됩니다.
  7. 만기 시간이 0보다 큰 요청 메시지가 검색된 경우 애플리케이션은 응답 메시지를 전송할 때 다음 조치 중 하나를 수행할 수 있습니다.
    - 요청 메시지에서 응답 메시지로 남은 만기 시간을 복사합니다.
    - 응답 메시지의 만기 시간을 0보다 큰 명확한 값으로 설정합니다.
    - 응답 메시지의 만기 시간을 MQEI\_UNLIMITED로 설정합니다.
 수행할 조치는 애플리케이션의 설계에 따라 다릅니다. 그러나 데드-레터(미배달 메시지) 큐에 메시지를 넣기 위한 기본 조치는 메시지의 남은 만기 시간을 유지하고 그 시간을 계속 감소시켜야 합니다.
  8. 트리거 메시지는 항상 MQEI\_UNLIMITED로 생성됩니다.
  9. MQFMT\_XMIT\_Q\_HEADER의 *Format* 이름을 갖는 메시지(일반적으로 전송 큐에 있음)는 MQXQH 내에 두 번째 메시지 디스크립터를 가집니다. 그러므로 이와 연관된 두 개의 *Expiry* 필드를 가집니다. 이 경우 추가적으로 다음과 같은 사항을 참고해야 합니다.
    - 애플리케이션이 리모트 큐에 메시지를 넣을 경우 큐 관리자는 이 메시지를 처음에 로컬 전송 큐에 넣고 애플리케이션 메시지 데이터의 접두부에 MQXQH 구조를 표시합니다. 큐 관리자는 두 *Expiry* 필드의 값을 애플리케이션이 지정한 것과 같은 값으로 설정합니다.  
애플리케이션이 메시지를 직접 로컬 전송 큐에 넣는 경우 메시지 데이터는 이미 MQXQH 구조로 시작해야 하며 형식 이름은 MQFMT\_XMIT\_Q\_HEADER여야 합니다. 이 경우 애플리케이션은 이 두 *Expiry* 필드의 값을 동일한 값으로 설정할 필요가 없습니다. (큐 관리자는 MQXQH 내의 *Expiry* 필드에 올바른 값이 포함되고 메시지 데이터가 이 값을 포함할 수 있는 길이인지 확인합니다.) 전송 큐에 직접 쓸 수 있는 애플리케이션의 경우 애플리케이션은 임베드된 메시지 디스크립터를 사용하여 전송 큐 헤더를 작성해야 합니다. 그러나 전송 큐에 쓰여진 메시지 디스크립터의 만기 값이 임베드된 메시지 디스크립터의 값과 일치하지 않은 경우 만기 오류 거부가 발생합니다.
    - 큐(정상 큐든 전송 큐든 상관없이)에서 MQFMT\_XMIT\_Q\_HEADER의 이름이 *Format*인 메시지를 검색할 때 큐 관리자는 큐에서 대기하면서 소비한 시간으로 이 두 개의 *Expiry* 필드를 감소시킵니다. 메시지 데이터 길이가 MQXQH의 *Expiry* 필드를 포함하기에 충분하지 않으면 오류가 발생하지 않습니다.



- 큐 관리자는 분리된 메시지 디스크립터에 있는 *Expiry* 필드(즉, MQXQH 구조 내에 임베드된 메시지 디스크립터에 있는 필드가 아님)를 사용하여 메시지가 제거될 수 있는지 여부를 테스트합니다.
- 두 *Expiry* 필드의 초기값이 서로 다른 경우 MQXQH에 있는 *Expiry* 필드에 따른 시간은 경과되었지만 메시지를 검색할 때 별도의 메시지 디스크립터에 있는 *Expiry* 시간이 0보다 클 수 있습니다(따라서 메시지가 제거 가능하지 않음). 이 경우 MQXQH에 있는 *Expiry* 필드는 0으로 설정됩니다.

10. IMS 브릿지에서 리턴된 응답 메시지의 만기 시간은 MQIIH\_PASS\_EXPIRATION이 MQIIH의 플래그 필드에서 설정된 경우를 제외하고는 무제한입니다. 자세한 정보는 Flags를 참조하십시오.

다음 특수 값이 인식됩니다.

#### **MQEI\_UNLIMITED**

메시지에 무제한 만기 시간이 있습니다.

이 필드는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 초기값은 MQEI\_UNLIMITED입니다.

#### *Expired messages on z/OS*

On IBM MQ for z/OS, messages that have expired are discarded by the next appropriate MQGET call.

However, if no such call occurs, the expired message is not discarded, and, for some queues, a large number of expired messages can accumulate. To remedy this, set the queue manager to scan queues periodically and discard expired messages on one or more queues in one of the following ways:

#### **Periodic scan**

You can specify a period using the EXPRYINT (expiry interval) queue manager attribute. Each time the expiry interval is reached, the queue manager looks for candidate queues that are worth scanning to discard expired messages.

The queue manager maintains information about the expired messages on each queue, and knows whether a scan for expired messages is worthwhile. So, only a selection of queues is scanned at any time.

Shared queues are scanned by only one queue manager in a queue sharing group. Generally, it is the first queue manager to restart, or the first to have EXPRYINT set. If this queue manager terminates, another queue manager in the queue sharing group takes over the queue scanning. Set the expiry interval value for all queue managers within a queue sharing group to the same value.

Note that expiry processing takes place for every queue when a queue manager restarts, regardless of the EXPRYINT setting.

#### **Explicit request**



Issue the REFRESH QMGR TYPE(EXPIRY) command, specifying the queue or queues that you want scanned.

더 짧은 만기 시간 적용

관리자는 **CAPEXPY** 속성을 사용하여 큐 또는 토픽에 넣는 메시지의 만기 시간을 제한할 수 있습니다.

CAPEXPY를 설정하는 두 가지 방법이 있습니다.

-   **CAPEXPY** 속성 지정
- **CUSTOM** 속성에서 **CAPEXPY** 속성 지정

  IBM MQ 9.4.0에서는 **CAPEXPY** 속성을 사용해야 합니다.

**CAPEXPY** 속성을 설정한 방법에 관계없이 다음 정보가 적용됩니다.

- CAPEXPY의 값은 10분의 1초로 표시되므로 600값은 1분을 나타냅니다.
- 큐 또는 토픽에 지정된 **CAPEXPY** 값보다 큰 애플리케이션에 의해 MQMD 구조의 **Expiry** 필드에 지정된 만기 시간이 해당 **CAPEXPY** 값으로 대체됩니다. **CAPEXPY** 값보다 낮은 애플리케이션에서 지정한 만기 시간이 사용됩니다.
- 둘 이상의 오브젝트가 해석 경로에서 사용되는 경우(예를 들어, 알리어스 또는 리모트 큐에 메시지를 넣는 경우) 모든 **CAPEXPY** 값 중 가장 낮은 값이 메시지 만기에 대한 상한으로 사용됩니다.

- **CAPEXPY** 값에 대한 변경사항은 바로 적용됩니다. 만기 값은 큐 또는 토픽에 대한 각 넣기에 대해 평가되므로 오브젝트 해석에 민감하며, 이는 각 넣기 조작 간에 다를 수 있습니다.

그러나 **CAPEXPY**의 변경 이전에 큐에 있는 기존 메시지는 변경의 영향을 받지 않습니다 (즉, 만기 시간이 동일하게 유지됨). **CAPEXPY**의 변경 후 큐 또는 토픽에 넣은 새 메시지에만 새 만기 시간이 있습니다.

- **MQOO\_BIND\_NOT\_FIXED**로 열린 큐에 넣기가 수행되는 클러스터에서는 선택된 대상 큐 관리자로 메시지를 보내는 채널에서 사용되는 전송 큐에 대해 설정된 **CAPEXPY** 값에 따라 각 넣기에서 메시지에 서로 다른 만기 값을 지정할 수 있습니다.



**주의: CAPEXPY** 는 IBM MQ 내부적으로 생성된 메시지 (예: **SYSTEM.CLUSTER.\*** 큐 및 **SYSTEM.PROTECTION.POLICY.QUEUE**)입니다.

## CUSTOM 속성에서 CAPEXPY 속성과 CAPEXPY 속성 간의 상호작용

V9.4.0 > V9.4.0

이전 버전의 제품에서 오브젝트를 마이그레이션할 때 **CAPEXPY** 값은 기본값 **NOLIMIT** (큐의 경우) 및 **ASPARENT** (토픽의 경우) 로 설정됩니다. 따라서 설정된 경우 **CUSTOM** 속성 내의 **CAPEXPY** 속성이 우선합니다.

큐 또는 토픽에 이미 **CUSTOM** 속성 내에 **CAPEXPY** 속성 세트가 있는 경우 새 **CAPEXPY** 속성이 기본값이 아닌 값으로 설정되기 전에 **CUSTOM** 속성에서 **CAPEXPY** 속성을 제거해야 합니다. 단일 명령으로 이를 수행할 수 있습니다. 예를 들어, 다음과 같습니다.

```
ALTER QLOCAL(Q1) CAPEXPY(1000) CUSTOM('')
```

큐 또는 토픽에 직접 설정하는 **CAPEXPY** 속성 (**CUSTOM** 속성 내부가 아님) 은 클러스터된 속성입니다. 클러스터된 큐 또는 클러스터된 토픽의 모든 인스턴스는 동일한 **CAPEXPY** 값을 사용해야 합니다. 전송 큐에 **CAPEXPY**가 설정되어 있고 값이 클러스터 큐 또는 토픽의 **CAPEXPY** 속성보다 낮은 경우에도 전송 큐가 메시지의 만기 시간을 줄일 수 있습니다.

### 관련 참조

[DEFINE QUEUES](#)

[DEFINE 토픽](#)

### MQMD에 대한 피드백 (MQLONG)

피드백 필드는 **MQMT\_REPORT** 유형의 메시지와 함께 사용되며 보고서의 네이처를 표시합니다. 또한 해당 유형의 메시지와 함께 사용되는 경우에만 의미가 있습니다.

필드에는 **MQFB\_\*** 값 중 하나 또는 **MQRC\_\*** 값 중 하나를 사용할 수 있습니다. 다음과 같이 피드백 코드를 분류할 수 있습니다.

#### MQFB\_NONE

피드백이 제공되지 않습니다.

#### MQFB\_SYSTEM\_FIRST

시스템에서 생성된 피드백에 대한 최저값입니다.

#### MQFB\_SYSTEM\_LAST

시스템에서 생성된 피드백에 대한 최고값입니다.

시스템에서 생성된 피드백 코드 **MQFB\_SYSTEM\_FIRST**에서 **MQFB\_SYSTEM\_LAST**까지의 범위에는 이 주제에서 나열된 일반 피드백 코드(**MQFB\_\***) 및 메시지를 목적지 큐에 넣지 못할 때 발생할 수 있는 이유 코드(**MQRC\_\***)가 포함됩니다.

#### MQFB\_APPL\_FIRST

애플리케이션에서 생성된 피드백에 대한 최저값입니다.

#### MQFB\_APPL\_LAST

애플리케이션에서 생성된 피드백에 대한 최고값입니다.

큐 관리자 또는 메시지 채널 에이전트에 의해 생성된 보고 메시지를 시뮬레이션하는 경우가 아니라면 보고 메시지를 생성하는 애플리케이션에서 시스템 범위의 피드백 코드(**MQFB\_QUIT** 이외)를 사용해서는 안 됩니다.

MQPUT 또는 MQPUT1 호출에서 지정된 값은 MQFB\_NONE이거나 시스템 범위 또는 애플리케이션 범위 안에 있어야 합니다. 이는 *MsgType*의 값에 상관없이 확인됩니다.

**일반 피드백 코드:**

**MQFB\_COA**

목적지 큐에서의 도착 확인입니다(MQRO\_COA 참조).

**MQFB\_COD**

수신 애플리케이션에 대한 전달 확인입니다(MQRO\_COD 참조).

**MQFB\_EXPIRATION**

만기 시간이 경과되기 전에 메시지가 목적지 큐에서 제거되지 않았으므로 제거되었습니다.

**MQFB\_PAN**

긍정 조치 알림입니다(MQRO\_PAN 참조).

**MQFB\_NAN**

부정 조치 알림입니다(MQRO\_NAN 참조).

**MQFB\_QUIT**

애플리케이션을 종료합니다.

이 코드는 실행 중인 애플리케이션 프로그램의 인스턴스 수를 제어하기 위한 목적으로 워크로드 스케줄 프로그램에서 사용됩니다. MQMT\_REPORT 메시지를 이 피드백 코드와 함께 애플리케이션 프로그램의 인스턴스로 송신하면 해당 인스턴스가 처리를 중지하라는 지시를 받습니다. 그러나, 이 규칙을 준수하는 것은 애플리케이션의 문제이며, 큐 관리자가 강제 실행하지 않습니다.

**채널 피드백 코드:**

**MQFB\_CHANNEL\_COMPLETED**

채널이 정상적으로 종료되었습니다.

**MQFB\_CHANNEL\_FAIL**

채널이 비정상적으로 종료되었고 STOPPED 상태가 됩니다.

**MQFB\_CHANNEL\_FAIL\_RETRY**

채널이 비정상적으로 종료되었고 RETRY 상태가 됩니다.

**IMS-브릿지 피드백 코드**

예상치 못한 IMS-OTMA 감지 코드가 수신되면 이러한 코드가 사용됩니다. 감지 코드(또는 감지 코드가 0x1A인 경우, 해당 감지 코드와 연관된 이유 코드는 *Feedback*에서 표시됩니다).

1. MQFB\_IMS\_FIRST(300)에서 MQFB\_IMS\_LAST(399)까지의 범위에 있는 *Feedback* 코드의 경우, 0x1A 외의 감지 코드가 수신됩니다. 감지 코드는 표현식(*Feedback* - MQFB\_IMS\_FIRST+1)으로 제공됩니다.
2. MQFB\_IMS\_NACK\_1A\_REASON\_FIRST(600)에서 MQFB\_IMS\_NACK\_1A\_REASON\_LAST(855)까지의 범위에 있는 *Feedback* 코드의 경우, 0x1A 감지 코드가 수신됩니다. 감지 코드와 연관된 이유 코드는 표현식(*Feedback* - MQFB\_IMS\_NACK\_1A\_REASON\_FIRST)으로 제공됩니다.

IMS-OTMA 감지 코드 및 해당 이유 코드의 의미는 *Open Transaction Manager Access Guide and Reference*에 설명되어 있습니다.

다음 피드백 코드가 IMS 브릿지에 의해 생성될 수 있습니다.

**MQFB\_DATA\_LENGTH\_ZERO**

메시지의 애플리케이션 데이터에서 세그먼트 길이가 0입니다.

**MQFB\_DATA\_LENGTH\_NEGATIVE**

메시지의 애플리케이션 데이터에서 세그먼트 길이가 음수입니다.

**MQFB\_DATA\_LENGTH\_TOO\_BIG**

메시지의 애플리케이션 데이터에서 세그먼트 길이가 너무 깁니다.

**MQFB\_BUFFER\_OVERFLOW**

길이 필드 중 하나의 값으로 인해 데이터가 메시지 버퍼 오버플로를 발생시킵니다.

**MQFB\_LENGTH\_OFF\_BY\_ONE**

길이 필드의 값 중 하나가 1바이트이며 너무 짧습니다.

**MQFB\_IIH\_ERROR**

MQMD의 *Format* 필드는 MQFMT\_IMS를 지정하지만 메시지가 올바른 MQIIH 구조로 시작되지 않습니다.

**MQFB\_NOT\_AUTHORIZED\_FOR\_IMS**

메시지 디스크립터 MQMD에 포함된 사용자 ID 또는 MQIIH 구조의 *Authenticator* 필드에 포함된 비밀 번호가 IMS 브릿지에 의해 수행되는 유효성 검증에 실패했습니다. 그 결과, 메시지가 IMS에 전달되지 않았습니다.

**MQFB\_IMS\_ERROR**

IMS에서 예상치 못한 오류가 리턴되었습니다. 오류에 대한 자세한 정보는 IMS 브릿지가 상주하는 시스템에서 IBM MQ 오류 로그를 참조하십시오.

**MQFB\_IMS\_FIRST**

IMS-OTMA 감지 코드가 0x1A가 아닌 경우, IMS-생성 피드백 코드는 MQFB\_IMS\_FIRST(300)에서 MQFB\_IMS\_LAST(399) 범위에 있습니다. IMS-OTMA 감지 코드 자체는 *Feedback*에서 MQFB\_IMS\_ERROR를 뺀 것입니다.

**MQFB\_IMS\_LAST**

감지 코드가 0x1A가 아닌 경우 IMS-생성 피드백에 대한 가장 높은 값입니다.

**MQFB\_IMS\_NACK\_1A\_REASON\_FIRST**

감지 코드가 0x1A인 경우 IMS-생성 피드백 코드는 MQFB\_IMS\_NACK\_1A\_REASON\_FIRST(600)에서 MQFB\_IMS\_NACK\_1A\_REASON\_LAST(855) 범위에 있습니다.

**MQFB\_IMS\_NACK\_1A\_REASON\_LAST**

감지 코드가 0x1A인 경우 IMS-생성 피드백에 대한 가장 높은 값입니다.

**CICS-브릿지 피드백 코드:** 다음 피드백 코드가 CICS bridge에서 생성될 수 있습니다.

**MQFB\_CICS\_APPL\_ABENDED**

메시지에 지정된 애플리케이션 프로그램이 비정상적으로 종료되었습니다. 이 피드백 코드는 MQDLH 구조의 *Reason* 필드에서만 발생합니다.

**MQFB\_CICS\_APPL\_NOT\_STARTED**

메시지에 지정된 애플리케이션 프로그램에 대한 EXEC CICS LINK가 실패했습니다. 이 피드백 코드는 MQDLH 구조의 *Reason* 필드에서만 발생합니다.

**MQFB\_CICS\_BRIDGE\_FAILURE**

CICS bridge가 정상 오류 처리를 완료하지 않고 비정상적으로 종료되었습니다.

**MQFB\_CICS\_CCSID\_ERROR**

문자 세트 ID가 올바르지 않습니다.

**MQFB\_CICS\_CIH\_ERROR**

CICS 정보 헤더 구조가 누락되었거나 올바르지 않습니다.

**MQFB\_CICS\_COMMAREA\_ERROR**

CICS commarea의 길이가 올바르지 않습니다.

**MQFB\_CICS\_CORREL\_ID\_ERROR**

상관 ID가 올바르지 않습니다.

**MQFB\_CICS\_DLQ\_ERROR**

CICS bridge 태스크가 이 요청에 대한 응답을 데드-레터 큐에 복사할 수 없습니다. 요청이 백아웃되었습니다.

**MQFB\_CICS\_ENCODING\_ERROR**

인코딩이 올바르지 않습니다.

**MQFB\_CICS\_INTERNAL\_ERROR**

CICS bridge에서 예상치 못한 오류가 발생했습니다.

이 피드백 코드는 MQDLH 구조의 *Reason* 필드에서만 발생합니다.

**MQFB\_CICS\_NOT\_AUTHORIZED**

사용자 ID에 권한이 없거나 비밀번호가 올바르지 않습니다.

이 피드백 코드는 MQDLH 구조의 *Reason* 필드에서만 발생합니다.

**MQFB\_CICS\_UOW\_BACKED\_OUT**

작업 단위가 다음 이유 중 하나로 백아웃되었습니다.

- 동일한 작업 단위 내에서 다른 요청을 처리하는 동안 실패가 감지되었습니다.
- 작업 단위가 진행 중인 동안 CICS 이상종료가 발생했습니다.

**MQFB\_CICS\_UOW\_ERROR**

작업 단위 제어 필드인 *UOWControl*이 올바르지 않습니다.

라우트 추적 메시지 피드백 코드:

**MQFB\_ACTIVITY**

활동 보고서에 따른 사용자 데이터의 옵션을 허용하기 위해 MQFMT\_EMBEDDED\_PCF 형식으로 사용됩니다.

**MQFB\_MAX\_ACTIVITIES**

메시지가 연관된 활동의 수가 최대 활동 수 한계를 초과하여 라우트 추적 메시지가 제거될 때 리턴됩니다.

**MQFB\_NOT\_FORWARDED**

라우트 추적 메시지를 지원하지 않는 리모트 큐 관리자에게 메시지가 송신하려고 하여 라우트 추적 메시지가 제거될 때 리턴됩니다.

**MQFB\_NOT\_DELIVERED**

라우트 추적 메시지를 로컬 큐에 넣으려고 하여 해당 메시지가 제거될 때 리턴됩니다.

**MQFB\_UNSUPPORTED\_FORWARDING**

전달 매개변수의 값이 인식되지 않고 거부되는 비트 마스크 내에 있어서 라우트 추적 메시지가 제거될 때 리턴됩니다.

**MQFB\_UNSUPPORTED\_DELIVERY**

전달 매개변수의 값이 인식되지 않고 거부되는 비트 마스크 내에 있어서 라우트 추적 메시지가 제거될 때 리턴됩니다.

**IBM MQ 이유 코드:** 예외 보고 메시지의 경우 *Feedback*에 IBM MQ 이유 코드가 포함됩니다. 가능한 이유 코드는 다음과 같습니다.

**MQRC\_PUT\_INHIBITED**

(2051, X'803') 큐에 대해 금지된 Put 호출입니다.

**MQRC\_Q\_FULL**

(2053, X'805') 큐에 이미 최대 수의 메시지가 포함되어 있습니다.

**MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

**MQRC\_Q\_SPACE\_NOT\_AVAILABLE**

(2056, X'808') 큐에 사용할 수 있는 공간이 디스크에 없습니다.

**MQRC\_PERSISTENT\_NOT\_ALLOWED**

(2048, X'800') 큐가 지속 메시지를 지원하지 않습니다.

**MQRC\_MSG\_TOO\_BIG\_FOR\_Q\_MGR**

(2031, X'7EF') 메시지 길이가 큐 관리자의 최대값보다 큼니다.

**MQRC\_MSG\_TOO\_BIG\_FOR\_Q**

(2030, X'7EE') 메시지 길이가 큐의 최대 길이보다 큼니다.

이유 코드의 전체 목록은 다음을 참조하십시오.

- IBM MQ for z/OS의 경우 [API 완료 및 이유 코드](#)를 참조하십시오.
- 기타 모든 플랫폼의 경우 [API 완료 및 이유 코드](#)를 참조하십시오.

이는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 초기값은 MQFB\_NONE입니다.

**MQMD의 인코딩 (MQLONG)**

이는 메시지에서 숫자 데이터의 숫자 인코딩을 지정합니다. 이는 MQMD 구조 자체에서 숫자 데이터에 적용되지 않습니다. 숫자 인코딩은 2진 정수, 팩형 10진수 정수 및 소수점수에 사용된 표시를 정의합니다.

z/OS에서 해당 문자 세트 ID가 문자 세트의 표현이 2진 정수에 사용된 인코딩에 따라 다름을 표시하는 경우 **Encoding** 필드의 2진 정수 부분도 메시지 본문에서 문자 데이터의 정수 인코딩을 지정하는 데 사용됩니다. 이는 특정 멀티바이트 문자 세트(예: UTF-16 문자 세트)에만 영향을 미칩니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 큐 관리자는 이 필드가 올바른지 검사하지 않습니다. 다음 특수 값이 정의됩니다.

#### **MQENC\_NATIVE**

인코딩은 애플리케이션이 실행 중인 프로그래밍 언어 및 시스템의 기본값입니다.

**참고:** 이 상수 값은 프로그래밍 언어 및 환경에 따라 다릅니다. 따라서, 애플리케이션은 실행될 환경에 적절한 헤더, 매크로, COPY 또는 INCLUDE 파일을 통해 컴파일되어야 합니다.

메시지를 넣는 애플리케이션은 일반적으로 MQENC\_NATIVE를 지정합니다. 메시지를 검색하는 애플리케이션은 이 필드를 MQENC\_NATIVE 값과 비교해야 합니다. 값이 다른 경우 애플리케이션은 메시지에 있는 숫자 데이터를 변환해야 합니다. MQGMO\_CONVERT 옵션을 사용하여 MQGET 호출 처리의 부분으로 메시지를 변환하도록 큐 관리자에 요청하십시오. **Encoding** 필드가 구성되는 방법에 대한 세부사항은 862 페이지의 『시스템 인코딩』의 내용을 참조하십시오.

MQGET 호출에서 MQGMO\_CONVERT 옵션을 지정하는 경우 이 필드는 입출력(I/O) 필드입니다. 애플리케이션이 지정하는 값은 필요한 경우 메시지 데이터를 변환하기 위한 인코딩입니다. 변환이 성공적이거나 불필요한 경우 값은 변경되지 않습니다. 변환이 실패하면 MQGET 호출 이후의 값은 애플리케이션에 리턴되는 변환되지 않은 메시지의 인코딩을 나타냅니다.

기타의 경우, 이는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 초기 값은 MQENC\_NATIVE입니다.

#### **MQMD의 경우 CodedCharSetId (MQLONG)**

이 필드는 메시지 본문 내에서 문자 데이터의 문자 세트 ID를 지정합니다.

**참고:** 호출의 매개변수인 MQMD 및 기타 MQ 데이터 구조의 문자 데이터는 큐 관리자의 문자 세트에 있어야 합니다. 이는 큐 관리자의 **CodedCharSetId** 속성에서 정의됩니다. 이 속성의 세부사항은 768 페이지의 『큐 관리자의 속성』의 내용을 참조하십시오.

옵션에 MQGMO\_CONVERT를 사용하여 MQGET을 호출할 때 이 필드가 MQCCSI\_Q\_MGR로 설정되는 경우 클라이언트와 서버 애플리케이션 사이의 작동이 다릅니다. 서버 애플리케이션의 경우 문자 변환에 사용된 코드 페이지는 큐 관리자의 **CodedCharSetId**이고, 클라이언트 애플리케이션의 경우 문자 변환에 사용되는 코드 페이지는 현재의 로케일 코드 페이지입니다.

클라이언트 애플리케이션의 경우 큐 관리자의 로케일이 아니라 클라이언트의 로케일을 기반으로 MQCCSI\_Q\_MGR이 채워집니다. 해당 규칙에 대한 예외는 IMS 브릿지 큐에 메시지를 넣는 시기이며, MQMD의 **CodedCharSetId** 필드에서 리턴되는 항목은 큐 관리자의 CCSID입니다.

다음 특수 값을 사용해서는 안 됩니다.

#### **MQCCSI\_APPL**

이는 MQMD의 **CodedCharSetId** 필드에서 올바르지 않은 값으로 인해 발행하며 이로 인해 MQGMO\_CONVERT 옵션과 함께 MQGET 호출을 사용하여 메시지를 수신할 때 MQRC\_SOURCE\_CCSDID\_ERROR(또는 z/OS의 경우 MQRC\_FORMAT\_ERROR)의 리턴 코드가 발생할 수 있습니다.

다음 특수 값을 사용할 수 있습니다.

#### **MQCCSI\_Q\_MGR**

메시지의 문자 데이터가 큐 관리자의 문자 세트에 있습니다.

MQPUT 및 MQPUT1 호출에서 큐 관리자는 메시지와 함께 전송된 MQMD의 이 값을 큐 관리자의 실제 문자 세트 ID로 변경합니다. 따라서 MQCCSI\_Q\_MGR 값은 MQGET 호출로 리턴되지 않습니다.



## MQCCSI\_DEFAULT

*String* 필드에 있는 데이터의 CodedCharSetId 는 MQCFH 구조에 선행하는 헤더 구조의 CodedCharSetId 필드 또는 MQCFH가 메시지의 시작 부분에 있는 경우 MQMD의 CodedCharSetId 필드에 의해 정의됩니다.

## MQCCSI\_INHERIT

메시지의 문자 데이터가 이 구조와 동일한 문자 세트에 있습니다. 이는 큐 관리자의 문자 세트입니다. (MQMD의 경우만 MQCCSI\_INHERIT가 MQCCSI\_Q\_MGR과 동일한 의미를 가집니다).

큐 관리자는 메시지와 함께 전송된 MQMD의 이 값을 MQMD의 실제 문자 세트 ID로 변경합니다. 오류가 발생하지 않으면 MQGET 호출에서 MQCCSI\_INHERIT 값을 리턴하지 않습니다.

MQMD의 PutApp1Type 필드 값이 MQAT\_BROKER인 경우 MQCCSI\_INHERIT를 사용하지 마십시오.

## MQCCSI\_EMBEDDED

메시지의 문자 데이터는 메시지 데이터 자체 내에 포함된 ID와 함께 문자 세트에 있습니다. 데이터의 서로 다른 부분에 적용될 경우 메시지 데이터 내에 임베드되는 문자 세트 ID 수는 제한되지 않습니다. 이 값은 혼합된 문자 세트에 데이터를 포함하는 PCF 메시지(MQFMT\_ADMIN, MQFMT\_EVENT 또는 MQFMT\_PCF의 형식)에 사용되어야 합니다. PCF 메시지 내에 포함된 각 MQCFST, MQCFSL, MQCFSF 구조에 명확한 문자 세트 ID를 지정해야 하며 MQCCSI\_DEFAULT로 지정하지 않아야 합니다.

MQFMT\_EMBEDDED\_PCF 형식의 메시지에 혼합된 문자 세트의 데이터가 포함될 경우 MQCCSI\_EMBEDDED를 사용하지 마십시오. 대신 MQEPH 구조의 플래그 필드에서 MQEPH\_CCSID\_EMBEDDED를 설정하십시오. 이는 선행 구조에서 MQCCSI\_EMBEDDED를 설정하는 것과 같습니다. PCF 메시지 내에 포함된 각 MQCFST, MQCFSL, MQCFSF 구조에 명확한 문자 세트 ID를 지정해야 하며 MQCCSI\_DEFAULT로 지정하지 않아야 합니다. MQEPH 구조에 대한 자세한 정보는 [357 페이지의 『MQEPH - 임베드된 PCF 헤더』](#)의 내용을 참조하십시오.

MQPUT 및 MQPUT1 호출에서만 이 값을 지정하십시오. MQGET 호출에서 지정된 경우, 이는 메시지의 변환을 막습니다.

MQPUT 및 MQPUT1 호출에서 큐 관리자는 위에서 설명된 대로 메시지와 함께 전송된 MQMD의 MQCCSI\_Q\_MGR 및 MQCCSI\_INHERIT 값을 변경하지만 MQPUT 또는 MQPUT1 호출에서 지정된 MQMD는 변경하지 않습니다. 지정된 값에 대해서는 다른 검사가 수행되지 않습니다.

메시지를 검색하는 애플리케이션은 이 필드를 애플리케이션이 예상하는 값과 비교해야 합니다. 값이 다른 경우 애플리케이션은 메시지에 있는 문자 데이터를 변환해야 합니다.

z/OS에서 MQMD의 CodedCharSetId 필드가 문자 세트의 표현이 2진 정수에 사용되는 인코딩에 종속됨을 표시하는 경우 MQMD의 Encoding 필드는 메시지 본문에서 문자 데이터의 정수 인코딩을 지정하는 데 사용됩니다. 멀티플랫폼에서 문자 데이터의 바이트 순서는 큐 관리자가 실행 중인 플랫폼의 고유 정수 인코딩과 동일하다고 가정합니다. 이는 특정 멀티바이트 문자 세트(예: UTF-16 문자 세트)에만 영향을 미칩니다.

MQGET 호출에서 MQGMO\_CONVERT 옵션을 지정하는 경우 이 필드는 입출력(I/O) 필드입니다. 애플리케이션이 지정하는 값은 필요한 경우 메시지 데이터를 변환하기 위해 코딩된 문자 세트입니다. 변환에 성공했거나 불필요한 경우 값은 변경되지 않습니다(단, 값 MQCCSI\_Q\_MGR 또는 MQCCSI\_INHERIT는 실제 값으로 변환됨). 변환에 성공하지 않은 경우 MQGET 호출 후의 값은 애플리케이션에 리턴된 변환되지 않은 메시지의 코딩된 문자 세트 ID를 나타냅니다.

그렇지 않으면, 이는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 초기값은 MQCCSI\_Q\_MGR입니다.

## MQMD의 형식 (MQCHAR8)

이는 메시지 송신자가 수신자에게 메시지에 있는 데이터의 네이처를 표시하기 위해 사용하는 이름입니다. 큐 관리자의 문자 세트에 있는 문자는 이름으로 지정될 수 있으나 이름을 다음과 같이 제한해야 합니다.

- 대문자 A - Z
- 숫자 0 - 9

기타 문자를 사용하면 송신 및 수신 큐 관리자의 문자 세트 간에 이름을 변환하지 못할 수도 있습니다.

공백으로 이름을 채워서 필드의 길이를 맞추거나 널 문자를 사용하여 필드의 끝 앞에서 이름을 종료하십시오. 널 및 후속 문자는 공백으로 처리됩니다. 선두 문자 또는 임베드된 공백으로 이름을 지정하지 마십시오. MQGET 호출의 경우, 큐 관리자는 필드의 길이까지 공백으로 채워진 이름을 리턴합니다.

큐 관리자는 이름이 위에서 설명한 권장사항을 준수하는지 확인하지 않습니다.

대문자, 소문자 및 대소문자가 혼합된 MQ로 시작하는 이름은 큐 관리자에 의해 정의된 의미를 갖습니다. 사용자의 자체 형식에 이러한 문자로 시작하는 이름을 사용하지 마십시오. 큐 관리자 내장 형식은 다음과 같습니다.

#### MQFMT\_NONE

데이터의 네이처가 정의되지 않았습니다. MQGMO\_CONVERT 옵션을 사용하여 큐에서 메시지를 검색할 때 데이터를 변환할 수 없습니다.

MQGET 호출에서 MQGMO\_CONVERT를 지정하고 메시지의 데이터 인코딩 또는 문자 세트가 **MsgDesc** 매개변수에서 지정된 항목과 다른 경우 메시지는 다음 완료 및 이유 코드와 함께 리턴됩니다(다른 오류가 없다고 가정).

- MQFMT\_NONE 데이터가 메시지의 시작에 있는 경우 완료 코드 MQCC\_WARNING 및 이유 코드 MQRC\_FORMAT\_ERROR입니다.
- MQFMT\_NONE 데이터가 메시지의 끝에 있는 경우(즉, 하나 이상의 MQ 헤더 구조가 앞에 올 때) 완료 코드 MQCC\_OK 및 이유 코드 MQRC\_NONE입니다. MQ 헤더 구조는 이 경우 요청된 문자 세트 및 인코딩으로 변환됩니다.

C 프로그래밍 언어의 경우 상수 MQFMT\_NONE\_ARRAY도 정의됩니다. 이는 MQFMT\_NONE과 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.


#### MQFMT\_ADMIN

메시지는 프로그램 가능 명령 형식(PCF)의 명령-서버 요청 또는 응답 메시지입니다. MQGMO\_CONVERT 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다. 프로그래밍 가능한 명령 형식 메시지 사용에 대한 자세한 정보는 [프로그램 가능한 명령 형식 사용](#)을 참조하십시오.

C 프로그래밍 언어의 경우 상수 MQFMT\_ADMIN\_ARRAY도 정의됩니다. 이는 MQFMT\_ADMIN과 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

#### MQFMT\_CICS

메시지 데이터는 애플리케이션 데이터 앞에 오는 CICS 정보 헤더 MQCIH로 시작합니다. 애플리케이션 데이터의 형식 이름은 MQCIH 구조의 Format 필드에 의해 제공됩니다.

 z/OS에서, MQGET 호출에서 MQGMO\_CONVERT 옵션을 지정하여 MQFMT\_CICS 형식이 있는 메시지를 변환하십시오.

C 프로그래밍 언어의 경우 상수 MQFMT\_CICS\_ARRAY도 정의됩니다. 이는 MQFMT\_CICS와 값이 동일하지만 문자열이 아닌 문자의 배열입니다.

#### MQFMT\_COMMAND\_1

메시지는 오브젝트 수, 완료 코드 및 이유 코드가 포함된 MQSC 명령-서버 응답 메시지입니다. MQGMO\_CONVERT 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.

C 프로그래밍 언어의 경우 상수 MQFMT\_COMMAND\_1\_ARRAY도 정의됩니다. 이는 MQFMT\_COMMAND\_1과 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

#### MQFMT\_COMMAND\_2

메시지는 요청된 오브젝트에 대한 정보를 포함하는 MQSC 명령 서버 응답 메시지입니다. MQGMO\_CONVERT 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.

C 프로그래밍 언어의 경우 상수 MQFMT\_COMMAND\_2\_ARRAY도 정의됩니다. 이는 MQFMT\_COMMAND\_2와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

#### MQFMT\_DEAD\_LETTER\_HEADER

메시지 데이터가 데드 레터 헤더 MQDLH로 시작합니다. 원래 메시지의 데이터 바로 뒤에 MQDLH 구조가 나옵니다. 원래 메시지 데이터의 형식 이름은 MQDLH 구조의 Format 필드에 의해 제공됩니다. 이 구조에 대한 세부사항은 346 페이지의 『MQDLH - 데드 레터 헤더』의 내용을 참조하십시오. MQGMO\_CONVERT 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.



Format이 MQFMT\_DEAD\_LETTER\_HEADER인 메시지에 대해 COA 및 COD 보고서가 생성되지 않았습니다.

C 프로그래밍 언어의 경우 상수 MQFMT\_DEAD\_LETTER\_HEADER\_ARRAY도 정의됩니다. 이는 MQFMT\_DEAD\_LETTER\_HEADER와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### MQFMT\_DIST\_HEADER

메시지 데이터가 분배 목록 헤더 MQDH로 시작합니다. 여기에는 MQOR 및 MQPMR 레코드의 배열이 포함됩니다. 분배 목록 헤더 뒤에는 추가 데이터가 올 수 있습니다. 추가 데이터의 형식은(있는 경우) MQDH 구조의 *Format* 필드에 의해 제공됩니다. 이 구조에 대한 세부사항은 340 페이지의 『MQDH - 분배 헤더』의 내용을 참조하십시오. MQGMO\_CONVERT 옵션이 MQGET 호출에서 지정되는 경우 형식이 MQFMT\_DIST\_HEADER인 메시지를 변환할 수 있습니다.

이 형식은 다음 환경에서 지원됩니다.

-  AIX
-  IBM i
-  Linux
-  Windows

및 해당 시스템에 연결된 IBM MQ MQI clients

C 프로그래밍 언어의 경우 상수 MQFMT\_DIST\_HEADER\_ARRAY도 정의됩니다. 이는 MQFMT\_DIST\_HEADER와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### MQFMT\_EMBEDDED\_PCF

PCF 명령 값이 MQCMD\_TRACE\_ROUTE로 설정된 경우 라우트 추적 메시지에 대한 형식입니다. 이 형식을 사용하면 사용자 데이터가 라우트 추적 메시지와 함께 전송될 수 있습니다. 단, 애플리케이션이 선행 PCF 매개변수와 대응할 수 있는 경우에 한합니다.

PCF 헤더가 반드시 첫 번째 헤더여야 합니다. 그렇지 않으면 메시지가 라우트 추적 메시지로 처리되지 않습니다. 이는 메시지가 그룹에 있을 수 없고 추적 라우트 메시지를 세그먼트화할 수 없음을 의미합니다. 추적 라우트 메시지가 그룹에서 전송되는 경우 이유 코드 MQRC\_MSG\_NOT\_ALLOWED\_IN\_GROUP과 함께 메시지가 거부됩니다.

MQFMT\_ADMIN을 추적 라우트 메시지의 형식에 사용할 수 있지만 이 경우 추적 라우트 메시지와 함께 전송할 수 있는 사용자 데이터가 없음을 참고하십시오.

### MQFMT\_EVENT

메시지가 발생한 이벤트를 보고하는 MQ 이벤트 메시지입니다. 이벤트 메시지에는 프로그램 가능한 명령과 동일한 구조가 있습니다. 이 구조에 대한 자세한 정보는 PCF 명령 메시지를 참조하고 이벤트에 대한 정보는 이벤트 모니터링을 참조하십시오.

MQGMO\_CONVERT 옵션이 MQGET 호출에서 지정되는 경우 모든 환경에서 버전-1 이벤트 메시지를 변환할 수 있습니다. 버전-2 이벤트 메시지는 z/OS에서만 변환할 수 있습니다.

C 프로그래밍 언어의 경우 상수 MQFMT\_EVENT\_ARRAY도 정의됩니다. 이는 MQFMT\_EVENT와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### MQFMT\_IMS

메시지 데이터는 IMS 정보 헤더 MQIIH로 시작하며, 이는 애플리케이션 데이터가 뒤따릅니다. 애플리케이션 데이터의 형식 이름은 MQIIH 구조의 *Format* 필드에 의해 제공됩니다.

MQGMO\_CONVERT와 함께 MQGET을 사용할 때 MQIIH 구조를 핸들링하는 방법에 대한 세부사항은 402 페이지의 『MQIIH의 형식 (MQCHAR8)』 및 402 페이지의 『MQIIH의 ReplyTo형식 (MQCHAR8)』의 내용을 참조하십시오.

C 프로그래밍 언어의 경우 상수 MQFMT\_IMS\_ARRAY도 정의됩니다. 이는 MQFMT\_IMS와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### MQFMT\_IMS\_VAR\_STRING

메시지는 IMS 변수 문자열이며, 이는 양식 11zzccc의 문자열입니다. 여기서, 다음과 같습니다.

## 11

IMS 변수 문자열 항목의 총 길이를 지정하는 2바이트 길이 필드입니다. 이 길이는 11의 길이(2바이트) 더하기 zz의 길이(2바이트) 더하기 문자열 자체의 길이와 같습니다. 11는 Encoding 필드에서 지정하는 인코딩의 2바이트 2진 정수입니다.

## zz

IMS에 중요한 플래그가 포함된 2바이트 필드입니다. zz는 두 개의 MQBYTE 필드로 구성된 바이트 문자열이며 전송자에서 수신자에게 변경 없이 전송됩니다(즉, zz는 변환 대상이 아님).

## ccc

11-4자가 포함된 변수-길이 문자열입니다. ccc는 CodedCharSetId 필드가 지정한 문자 세트에 있습니다.

z/OS에서 메시지 데이터는 함께 연결된 IMS 변수 문자열의 순서로 구성될 수 있으며 각 문자열은 11zzccc 양식으로 되어 있습니다. 후속 IMS 변수 문자열 간에 건너편 바이트가 없어야 합니다. 이는 첫 번째 문자열이 홀수 길이이면 두 번째 문자열이 불일치한다는 의미입니다. 즉, 2의 배수인 경계에서 시작하지 않습니다. 요소 데이터 유형의 맞추기가 필요한 시스템에서 해당 문자열을 구성할 때 주의하십시오.

MQGET 호출에서 MQGMO\_CONVERT 옵션을 사용하여 MQFMT\_IMS\_VAR\_STRING 형식이 있는 메시지를 변환하십시오.

C 프로그래밍 언어의 경우 상수 MQFMT\_IMS\_VAR\_STRING\_ARRAY도 정의됩니다. 이는 MQFMT\_IMS\_VAR\_STRING과 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

## MQFMT\_MD\_EXTENSION

메시지 데이터는 메시지-디스크립터 확장자 MQMDE로 시작하고 선택적으로 기타 데이터가 뒤따릅니다(일반적으로 애플리케이션 메시지 데이터). MQMDE 다음에 오는 데이터의 형식 이름, 문자 세트, 인코딩은 MQMDE의 Format, CodedCharSetId, Encoding 필드에 의해 제공됩니다. 이 구조의 세부사항은 [458 페이지의 『MQMDE - 메시지 디스크립터 확장』](#)의 내용을 참조하십시오. MQGMO\_CONVERT 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.

C 프로그래밍 언어의 경우 상수 MQFMT\_MD\_EXTENSION\_ARRAY도 정의됩니다. 이는 MQFMT\_MD\_EXTENSION과 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

## MQFMT\_PCF

메시지는 프로그램 가능 명령 형식(PCF) 메시지의 구조를 준수하는 사용자 정의 메시지입니다.

MQGMO\_CONVERT 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다. 프로그래밍 가능한 명령 형식 메시지 사용에 대한 자세한 정보는 [프로그램 가능한 명령 형식 사용](#)을 참조하십시오.

C 프로그래밍 언어의 경우 상수 MQFMT\_PCF\_ARRAY도 정의됩니다. 이는 MQFMT\_PCF와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

## MQFMT\_REF\_MSG\_HEADER

메시지 데이터는 참조 메시지 헤더 MQRMH로 시작되며 선택적으로 기타 데이터가 뒤따릅니다. 데이터의 형식 이름, 문자 세트, 인코딩은 MQRMH의 Format, CodedCharSetId, Encoding 필드에 의해 제공됩니다. 이 구조의 세부사항은 [531 페이지의 『MQRMH - 참조 메시지 헤더』](#)의 내용을 참조하십시오.

MQGMO\_CONVERT 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.

이 형식은 다음 환경에서 지원됩니다.

-  AIX
-  IBM i
-  Linux
-  Windows

및 해당 시스템에 연결된 IBM MQ MQI clients

C 프로그래밍 언어의 경우 상수 MQFMT\_REF\_MSG\_HEADER\_ARRAY도 정의됩니다. 이는 MQFMT\_REF\_MSG\_HEADER와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

## MQFMT\_RF\_HEADER

메시지 데이터는 규칙 및 형식화 헤더 MQRFH로 시작되며 선택적으로 기타 데이터가 뒤따릅니다. 데이터의 형식 이름, 문자 세트, 인코딩은(있는 경우) MQRFH의 Format, CodedCharSetId, Encoding 필드에 의해 제공됩니다. MQGMO\_CONVERT 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.

C 프로그래밍 언어의 경우 상수 MQFMT\_RF\_HEADER\_ARRAY도 정의됩니다. 이는 MQFMT\_RF\_HEADER와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

## MQFMT\_RF\_HEADER\_2

메시지 데이터는 버전-2 규칙 및 형식화 헤더 MQRFH2로 시작되며 선택적으로 기타 데이터가 뒤따릅니다. 선택적 데이터의 형식 이름, 문자 세트, 인코딩은(있는 경우) MQRFH2의 Format, CodedCharSetId, Encoding 필드에 의해 제공됩니다. MQGMO\_CONVERT 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.

C 프로그래밍 언어의 경우 상수 MQFMT\_RF\_HEADER\_2\_ARRAY도 정의됩니다. 이는 MQFMT\_RF\_HEADER\_2와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

## MQFMT\_STRING

애플리케이션 메시지 데이터는 SBCS 문자열(1바이트 문자 세트) 또는 DBCS 문자열(2바이트 문자 세트)일 수 있습니다. MQGMO\_CONVERT 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.

C 프로그래밍 언어의 경우 상수 MQFMT\_STRING\_ARRAY도 정의됩니다. 이는 MQFMT\_STRING과 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.


## MQFMT\_TRIGGER

메시지는 MQTM 구조에서 설명된 트리거 메시지입니다. 이 구조의 세부사항은 579 페이지의 『MQTM - 트리거 메시지』의 내용을 참조하십시오. MQGMO\_CONVERT 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.

C 프로그래밍 언어의 경우 상수 MQFMT\_TRIGGER\_ARRAY도 정의됩니다. 이는 MQFMT\_TRIGGER와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

## MQFMT\_WORK\_INFO\_HEADER

메시지 데이터는 작업 정보 헤더 MQWIH로 시작하며, 이는 애플리케이션 데이터가 뒤따릅니다. 애플리케이션 데이터의 형식 이름은 MQWIH 구조의 Format 필드에 의해 제공됩니다.

 z/OS에서, MQGET 호출에서 MQGMO\_CONVERT 옵션을 지정하여 MQFMT\_WORK\_INFO\_HEADER 형식이 있는 메시지의 사용자 데이터를 변환하십시오. 그러나 MQWIH 구조 자체는 항상 큐 관리자의 문자 세트 및 인코딩에서 리턴됩니다. 즉, MQWIH 구조는 MQGMO\_CONVERT 옵션의 지정 여부에 상관없이 변환됩니다.

C 프로그래밍 언어의 경우 상수 MQFMT\_WORK\_INFO\_HEADER\_ARRAY도 정의됩니다. 이는 MQFMT\_WORK\_INFO\_HEADER와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

## MQFMT\_XMIT\_Q\_HEADER

메시지 데이터가 전송 큐 헤더 MQXQH로 시작합니다. 원래 메시지의 데이터 바로 뒤에 MQXQH 구조가 나옵니다. 원래 메시지 데이터의 형식 이름은 전송 큐 헤더 MQXQH의 부분인 MQMD 구조의 Format 필드에 의해 제공됩니다. 이 구조의 세부사항은 597 페이지의 『MQXQH - 전송 큐 헤더』의 내용을 참조하십시오.

Format이 MQFMT\_XMIT\_Q\_HEADER인 메시지에 대해 COA 및 COD 보고서가 생성되지 않았습니다.

C 프로그래밍 언어의 경우 상수 MQFMT\_XMIT\_Q\_HEADER\_ARRAY도 정의됩니다. 이는 MQFMT\_XMIT\_Q\_HEADER와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

이 필드는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 길이는 MQ\_FORMAT\_LENGTH에 지정되어 있습니다. 이 필드의 초기값은 MQFMT\_NONE입니다.

## MQMD의 우선순위 (MQLONG)

MQPUT 및 MQPUT1 호출의 경우, 값은 0 이상이어야 합니다. 0은 우선순위가 가장 낮습니다. 다음 특수 값을 사용할 수도 있습니다.

## MQPRI\_PRIORITY\_AS\_Q\_DEF

- 큐가 클러스터 큐인 경우 메시지의 우선순위는 메시지가 배치된 큐의 특정 인스턴스를 소유하는 목적지 큐 관리자에 정의된 **DefPriority** 속성에서 가져옵니다.

클러스터 큐의 인스턴스가 여러 개 있는데 이들의 속성이 서로 다를 경우, 이 중 하나의 값이 선택되며 어떤 값이 사용될지 예측할 수는 없습니다. 따라서 이 속성을 모든 인스턴스에서 동일한 값으로 설정해야 합니다. 그렇지 않을 경우 큐 관리자 로그에 오류 메시지 AMQ9407이 발행됩니다. 별명, 리모트 및 클러스터 큐에 대해 목적지 오브젝트 속성을 해석하는 방법도 참조하십시오.

메시지가 목적지 큐에 배치될 때 *DefPriority* 값이 *Priority* 필드에 복사됩니다. *DefPriority*이(가) 나중에 변경되면 큐에 이미 있는 메시지는 영향을 받지 않습니다.

- 큐가 클러스터 큐가 아닌 경우, 메시지의 지속성은 목적지 큐 관리자가 원격인 경우에도 로컬 큐 관리자에 정의된 **DefPriority** 속성에서 가져옵니다.

큐 이름 해석 경로에 정의가 둘 이상인 경우 기본 우선순위는 경로에서 첫 번째 정의에 있는 이 속성의 값에서 가져옵니다. 다음과 같습니다.

- 알리어스 큐
- 로컬 큐
- 리모트 큐의 로컬 정의
- 큐 관리자 알리어스
- 전송 큐(예: *DefXmitQName* 큐)

메시지를 넣을 때 *DefPriority* 값이 *Priority* 필드에 복사됩니다. *DefPriority*이(가) 나중에 변경되면 이미 넣은 메시지는 영향을 받지 않습니다.

MQGET 호출로 리턴되는 값은 항상 0 이상입니다. MQPRI\_PRIORITY\_AS\_Q\_DEF 값은 리턴되지 않습니다.

로컬 큐 관리자가 지원하는 최대값보다 큰 우선순위로 메시지를 넣는 경우(이 최대값은 **MaxPriority** 큐 관리자 속성으로 지정됨), 큐 관리자가 메시지를 허용하지만 큐 관리자의 최대 우선순위에 있는 큐에 배치됩니다. MQPUT 또는 MQPUT1 호출은 MQCC\_WARNING 및 이유 코드 MQRC\_PRIORITY\_EXCEEDS\_MAXIMUM과 함께 완료됩니다. 그러나 *Priority* 필드는 메시지를 넣은 애플리케이션에서 지정된 값을 보유합니다.

z/OS에서 MsgSeqNumber가 1인 메시지를 메시지 전달 순서가 MQMDS\_PRIORITY이고 색인 유형이 MQIT\_GROUP\_ID인 큐에 넣는 경우 큐는 메시지를 다른 우선순위로 처리할 수 있습니다. 우선순위가 0 또는 1인 큐에 메시지가 배치된 경우 우선순위가 2인 것처럼 처리됩니다. 이 유형의 큐에 배치된 메시지의 순서가 효율적인 그룹 완전성 테스트를 사용하도록 최적화되었기 때문입니다. 메시지 전달 순서 MQMDS\_PRIORITY 및 색인 유형 MQIT\_GROUP\_ID에 대한 자세한 정보는 [MsgDeliverySequence](#) 속성을 참조하십시오.

메시지에 응답할 때 애플리케이션이 응답 메시지에 대한 요청 메시지의 우선순위를 사용해야 합니다. 다른 상황에서 MQPRI\_PRIORITY\_AS\_Q\_DEF를 지정하면 애플리케이션을 변경하지 않고 우선순위 조정을 수행할 수 있습니다.

이 필드는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 초기값은 MQPRI\_PRIORITY\_AS\_Q\_DEF입니다.

## MQMD의 지속성 (MQLONG)

이는 메시지가 시스템 실패 및 큐 관리자의 재시작 이후 남아 있는지 여부를 나타냅니다. MQPUT 및 MQPUT1 호출의 경우, 값은 다음 중 하나여야 합니다.

### MQPER\_PERSISTENT

메시지가 시스템 실패 이후에 남아 있으며 큐 관리자를 재시작합니다. 메시지를 넣고 메시지를 넣은 작업 단위가 커밋되면(메시지를 작업 단위의 일부로 넣은 경우) 메시지가 보조 기억장치에 보존됩니다. 메시지가 작업 단위의 일부로 검색되는 경우 메시지가 큐 및 커밋된 작업 단위에서 제거될 때까지 남아 있습니다.

지속 메시지가 리모트 큐에 전송되는 경우 메시지가 다음 큐 관리자에 도착했음이 알려질 때까지 저장 및 전달 메커니즘이 목적지에 대한 라우트와 함께 각 큐 관리자에서 메시지를 보유합니다.

다음 큐에는 지속 메시지를 넣을 수 없습니다.

- 임시 동적 큐

- CFLEVEL(2) 이하의 CFSTRUCT 오브젝트에 맵핑되는 공유 큐 또는 CFSTRUCT 오브젝트가 RECOVER(NO)로 정의되는 공유 큐.

지속 메시지는 영구적 동적 큐 및 사전정의된 큐에 배치할 수 있습니다.

### MQPER\_NOT\_PERSISTENT

메시지가 일반적으로 시스템 실패 또는 큐 관리자 재시작 시에 남아 있지 않습니다. 이는 큐 관리자가 재시작 될 때 메시지의 원래 사본이 보조 기억장치에서 발견되는 경우에도 적용됩니다.

NPMCLASS(HIGH) 큐의 경우 비지속 메시지가 정상적인 큐 관리자 시스템 종료 및 재시작 시에 남아 있습니다.

공유 큐의 경우 큐 공유 그룹에서 큐 관리자 재시작 시 비지속 메시지가 남아 있으나 공유 큐에 메시지를 저장 하기 위해 사용되는 커플링 기능의 실패 시에는 남아 있지 않습니다.

### MQPER\_PERSISTENCE\_AS\_Q\_DEF

- 큐가 클러스터 큐인 경우 메시지의 지속성은 메시지가 있는 큐의 특정 인스턴스를 소유하는 목적지 큐 관리자에 정의된 **DefPersistence** 속성에서 가져옵니다.

클러스터 큐의 인스턴스가 여러 개 있는데 이들의 속성이 서로 다를 경우, 이 중 하나의 값이 선택되며 어떤 값이 사용될지 예측할 수는 없습니다. 따라서 이 속성을 모든 인스턴스에서 동일한 값으로 설정해야 합니다. 그렇지 않을 경우 큐 관리자 로그에 오류 메시지 AMQ9407이 발행됩니다. 별명, 리모트 및 클러스터 큐에 대해 목적지 오브젝트 속성을 해석하는 방법도 참조하십시오.

메시지가 목적지 큐에 배치될 때 **DefPersistence** 값이 **Persistence** 필드에 복사됩니다. **DefPersistence**이(가) 나중에 변경되면 큐에 이미 있는 메시지는 영향을 받지 않습니다.

- 큐가 클러스터 큐가 아닌 경우, 목적지 큐 관리자가 리모트인 경우에도 로컬 큐 관리자에 정의된 **DefPersistence** 속성에서 메시지의 지속성을 가져옵니다.

큐 이름 해석 경로에 정의가 둘 이상인 경우 기본 지속성은 경로에서 첫 번째 정의에 있는 이 속성의 값에서 가져옵니다. 다음과 같습니다.

- 알리어스 큐
- 로컬 큐
- 리모트 큐의 로컬 정의
- 큐 관리자 알리어스
- 전송 큐(예: *DefXmitQName* 큐)

메시지를 넣을 때 **DefPersistence** 값이 **Persistence** 필드에 복사됩니다. **DefPersistence**이(가) 나중에 변경되면 이미 넣은 메시지는 영향을 받지 않습니다.

지속 및 비지속 메시지 둘 다 동일한 큐에 존재할 수 있습니다.

메시지에 응답할 때 애플리케이션이 응답 메시지에 대한 요청 메시지의 지속성을 사용해야 합니다.

MQGET 호출의 경우 리턴된 값은 MQPER\_PERSISTENT 또는 MQPER\_NOT\_PERSISTENT입니다.

이 필드는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 초기값은 MQPER\_PERSISTENCE\_AS\_Q\_DEF입니다.

### MQMD의 경우 *MsgId* (MQBYTE24)

한 메시지를 다른 메시지와 구분하기 위해 사용되는 바이트 문자열입니다. 일반적으로, 큐 관리자가 허용하지 않는 것은 아니지만 동일한 메시지 ID를 갖는 2개의 메시지가 없어야 합니다. 메시지 ID는 메시지의 영구적 특성이며 큐 관리자를 재시작해도 지속됩니다. 메시지 ID가 바이트 문자열이며 문자열이 아니므로, 메시지가 하나의 큐 관리자에서 다른 큐 관리자로 이동할 때 메시지 ID는 문자 세트 간에 변환되지 않습니다.

MQPUT와 MQPUT1 호출의 경우, 애플리케이션에서 MQMI\_NONE 또는 MQPMO\_NEW\_MSG\_ID를 지정하면 큐 관리자가 고유 메시지 ID를 생성합니다.<sup>3</sup> 메시지를 넣을 때 메시지와 함께 전송되는 메시지 디스크립터에 이를

<sup>3</sup> 큐 관리자가 생성하는 *MsgId* 는 4바이트제품 ID (ASCII 또는 EBCDIC로 된 AMQ- 또는 CSQ-, 여기서 - 는 공백 문자를 나타냄) 와 그 뒤에 오는 고유 문자열의 제품 특정 구현으로 구성됩니다. IBM MQ에서 이는 큐 관리자 이름의 첫 번째 12자 및 시스템 클럭에서 나온 값을 포함합니다. 그러므로 메시지 ID가 고유하기 위해서 소통할 수 있는



배치합니다. 또한 큐 관리자가 전송 중인 애플리케이션에 속하는 메시지 디스크립터에서 이 메시지 ID를 리턴합니다. 애플리케이션은 이 값을 사용하여 특정한 메시지에 대한 정보를 레코딩하고 애플리케이션의 기타 부분에서 조회에 응답할 수 있습니다.

메시지를 토픽에 넣는 경우 큐 관리자가 발행되는 각 메시지에 필요한 대로 고유한 메시지 ID를 생성합니다. 애플리케이션이 MQPMO\_NEW\_MSG\_ID를 지정하면 큐 관리자가 출력을 리턴하기 위해 고유한 메시지 ID를 생성합니다. 애플리케이션에서 MQMI\_NONE을 지정하면 MQMD의 *MsgId* 필드 값이 호출에서 리턴되는 경우 변경되지 않습니다.

보유된 발행에 대한 자세한 정보는 490 페이지의 『MQPMO에 대한 옵션 (MQLONG)』에서 MQPMO\_RETAIN에 대한 설명을 참조하십시오.

메시지를 분배 목록에 넣는 경우 큐 관리자가 필요에 따라 고유한 메시지 ID를 생성하지만 MQMI\_NONE 또는 MQPMO\_NEW\_MSG\_ID가 지정된 경우에도 MQMD의 *MsgId* 필드 값이 호출에서 리턴 시 변경되지 않습니다. 애플리케이션이 큐 관리자에 의해 생성된 메시지 ID를 알아야 하는 경우 애플리케이션이 *MsgId* 필드를 포함하는 MQPMR 레코드를 제공해야 합니다.

또한 전송 애플리케이션이 MQMI\_NONE 외의 메시지 ID에 대해 값을 지정할 수 있습니다. 이로 인해 큐 관리자가 고유 메시지 ID 생성을 중지합니다. 메시지를 전달하는 애플리케이션이 이를 사용하여 원래 메시지의 메시지 ID를 전파할 수 있습니다.

다음 경우를 제외하고 큐 관리자는 이 필드를 사용하지 않습니다.

- 위에서 설명한 대로 요청된 경우 고유 값 생성
- 메시지에 대한 Get 요청을 실행하는 애플리케이션에 값 전달
- *Report* 옵션에 따라 이 메시지에 대해 생성하는 모든 보고 메시지의 *CorrelId* 필드에 값 복사

큐 관리자 또는 메시지 채널 에이전트가 보고 메시지를 생성할 때 원래 메시지 MQRO\_NEW\_MSG\_ID 또는 MQRO\_PASS\_MSG\_ID의 *Report* 필드에 의해 지정된 방식으로 *MsgId* 필드를 설정합니다. 보고 메시지를 생성하는 애플리케이션은 이 작업도 수행해야 합니다.

MQGET 호출의 경우 *MsgId*는 큐에서 특정 메시지를 검색하는 데 사용할 수 있는 다섯 필드 중 하나입니다. 일반적으로 MQGET 호출이 큐에서 다음 메시지를 리턴하지만 어떠한 결합으로든 다섯 개의 선택 기준 중 하나 이상을 지정하는 방법으로 특정 메시지를 얻을 수 있습니다. 다섯 개의 필드는 다음과 같습니다.

- *MsgId*
- *CorrelId*
- *GroupId*
- *MsgSeqNumber*
- *Offset*

해당 필드를 선택 기준으로 사용하기 위해 애플리케이션이 이러한 필드 중 하나 이상을 필요한 값으로 설정한 다음 MQGMO의 *MatchOptions* 필드에서 해당 MQMO\_\* 일치 옵션을 설정합니다. 해당 필드에 지정된 값이 있는 메시지만이 검색 후보입니다. *MatchOptions* 필드에 대한 기본값(애플리케이션에 의해 변경되지 않은 경우)은 메시지 ID 및 상관 ID를 모두 일치시키는 것입니다.

z/OS에서 사용할 수 있는 선택 기준은 큐에 사용된 색인 유형으로 제한됩니다. 추가 세부사항은 **IndexType** 큐 속성을 참조하십시오.

일반적으로 리턴된 메시지는 선택 기준을 만족시키는 큐에 대한 첫 번째 메시지입니다. 그러나 MQGMO\_BROWSE\_NEXT가 지정되는 경우 리턴된 메시지는 선택 기준을 충족하는 다음 메시지입니다. 이 메시지에 대한 스캔이 현재 커서 위치 다음에 오는 메시지와 함께 시작됩니다.

---

모든 큐 관리자는 처음 12자가 다른 이름이 있어야 합니다. 또한 고유 문자열을 생성할 수 있는 기능은 뒤로 변경되지 않는 시스템 클럭에 의존합니다. 큐 관리자에 의해 생성되는 메시지 ID가 애플리케이션에 의해 생성되는 메시지 ID와 중복될 가능성을 제거하려면 애플리케이션이 ASCII 또는 EBCDIC에서 A - I의 범위 내에 있는 초기 문자를 사용하여 ID를 생성하지 않아야 합니다(X'41'에서 X'49' 및 X'C1'에서 X'C9'). 그러나 애플리케이션은 이러한 범위의 초기 문자를 사용하여 ID를 생성하는 것이 방지되지 않습니다.

**참고:** 선택 기준을 충족하는 메시지에 대해 큐를 연속적으로 스캔하므로 선택 기준을 지정하지 않은 경우보다 검색 시간이 느려집니다. 특히, 적당한 메시지를 찾기 전에 많은 메시지를 스캔해야 하는 경우 시간이 느려집니다. 예외는 다음과 같습니다.

- ▶ **Multi** 64비트 Multiplatforms에서 *CorrelId*에 의한 MQGET 호출. 여기서, *CorrelId* 색인은 true 순차 스캔을 수행할 필요성을 제거합니다.
- ▶ **z/OS** z/OS에서 *IndexType*에 의한 MQGET 호출.

두 경우 모두 검색 성능이 향상됩니다.

선택 기준이 다양한 상황에서 사용되는 방법에 대한 자세한 정보는 [386 페이지의 표 495](#)의 내용을 참조하십시오.

메시지 ID로 MQMI\_NONE을 지정하면 MQMO\_MATCH\_MSG\_ID를 지정하지 것과 동일한 영향을 미칩니다. 즉, 임의의 메시지 ID가 일치합니다.

MQGET 호출의 **GetMsgOpts** 매개변수에서 MQGMO\_MSG\_UNDER\_CURSOR 옵션이 지정되면 이 필드가 무시됩니다.

MQGET 호출에서 리턴 시 *MsgId* 필드가 리턴된 메시지(있는 경우)의 메시지 ID로 설정됩니다.

다음 특수 값을 사용할 수 있습니다.

#### **MQMI\_NONE**

메시지 ID가 지정되지 않습니다.

이 값은 필드 길이 만큼의 2진 0입니다.

C 프로그래밍 언어의 경우 상수 MQMI\_NONE\_ARRAY도 정의됩니다. 이는 MQMI\_NONE과 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

이는 MQGET, MQPUT 또는 MQPUT1 호출의 입출력(I/O) 필드입니다. 이 필드의 길이는 MQ\_MSG\_ID\_LENGTH로 지정됩니다. 이 필드의 초기값은 MQMI\_NONE입니다.

#### **MQMD의 경우 CorrelId (MQBYTE24)**

CorrelId 필드는 특정 메시지 또는 메시지 그룹을 식별하는 데 사용될 수 있는 메시지 헤더의 특성입니다.

이는 하나의 메시지를 다른 메시지에 관련시키거나 애플리케이션이 수행 중인 다른 작업에 메시지를 관련시키는 데 애플리케이션이 사용할 수 있는 바이트 문자열입니다. 상관 ID는 메시지의 지속적 특성이며 큐 관리자를 재시작해도 지속됩니다. 상관 ID가 바이트 문자열이며 문자열이 아니므로, 메시지가 하나의 큐 관리자에서 다른 큐 관리자로 이동할 때 상관 ID는 문자 세트 간에 변환되지 않습니다.

MQPUT 및 MQPUT1 호출의 경우, 애플리케이션은 임의의 값을 지정할 수 있습니다. 큐 관리자가 메시지와 함께 이 값을 전송하며 메시지에 대해 get 요청을 실행하는 애플리케이션에 이를 전달합니다.

애플리케이션이 MQPMO\_NEW\_CORREL\_ID를 지정하면 큐 관리자는 메시지와 함께 전송되고 또한 MQPUT 또는 MQPUT1 호출에서 출력 시 전송 애플리케이션으로 리턴되는 고유 상관 ID를 생성합니다.

큐 관리자가 생성하는 상관 ID는 3바이트 제품 ID(ASCII 또는 EBCDIC에서 AMQ 또는 CSQ)와 그 뒤에 오는 한 개의 예약 바이트 및 제품별 고유 구현 문자열로 구성됩니다. IBM MQ에서 이 제품별 구현 문자열은 큐 관리자 이름의 처음 12자 및 시스템 클럭에서 파생된 값을 포함합니다. 그러므로 메시지 ID가 고유하기 위해서 소통할 수 있는 모든 큐 관리자는 처음 12자가 다른 이름이 있어야 합니다. 또한 고유 문자열을 생성할 수 있는 기능은 뒤로 변경되지 않는 시스템 클럭에 의존합니다. 큐 관리자에 의해 생성되는 메시지 ID가 애플리케이션에 의해 생성되는 메시지 ID와 중복될 가능성을 제거하려면 애플리케이션이 ASCII 또는 EBCDIC에서 A - I의 범위 내에 있는 초기 문자를 사용하여 ID를 생성하지 않아야 합니다(X'41'에서 X'49' 및 X'C1'에서 X'C9'). 그러나 애플리케이션은 이러한 범위의 초기 문자를 사용하여 ID를 생성하는 것이 방지되지 않습니다.

이렇게 생성된 상관 ID는 메시지가 보유되는 경우 메시지와 함께 보관되고 메시지가 MQSUB 호출에 전달된 MQSD의 SubCorrelId 필드에서 MQCI\_NONE을 지정하는 구독자에게 발행물로 전송될 때 상관 ID로 사용됩니다. 보유한 구독에 대한 자세한 정보는 [MQPMO](#) 옵션을 참조하십시오.

큐 관리자 또는 메시지 채널 에이전트가 보고 메시지를 생성할 때 원래 메시지

MQRO\_COPY\_MSG\_ID\_TO\_CORREL\_ID 또는 MQRO\_PASS\_CORREL\_ID의 *Report* 필드에 의해 지정된 방식으로 *CorrelId* 필드를 설정합니다. 보고 메시지를 생성하는 애플리케이션은 이 작업도 수행해야 합니다.

MQGET 호출의 경우 *CorrelId*는 큐에서 검색할 특정 메시지를 선택하는 데 사용할 수 있는 다섯 필드 중 하나입니다. 이 필드에 값을 지정하는 방법에 대한 세부사항은 *MsgId* 필드에 대한 설명을 참조하십시오.

상관 ID로 MQCI\_NONE을 지정하면 MQMO\_MATCH\_CORREL\_ID를 지정하지 않는 것과 동일한 영향을 미칩니다. 즉, 임의의 상관 ID가 일치합니다.

MQGET 호출의 **GetMsgOpts** 매개변수에서 MQGMO\_MSG\_UNDER\_CURSOR 옵션이 지정되면 이 필드가 무시됩니다.

MQGET 호출에서 리턴 시 *CorrelId* 필드가 리턴된 메시지(있는 경우)의 상관 ID로 설정됩니다.

다음 특수 값을 사용할 수 있습니다.

#### **MQCI\_NONE**

상관 ID가 지정되어 있지 않습니다.

이 값은 필드 길이만큼의 2진 0입니다.

C 프로그래밍 언어의 경우 상수 MQCI\_NONE\_ARRAY도 정의됩니다. 이는 MQCI\_NONE과 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

#### **MQCI\_NEW\_SESSION**

메시지는 새 세션의 시작입니다.

이 값은 새 세션의 시작을 나타내는 것으로 CICS bridge에서 인식합니다. 즉, 메시지의 새 시퀀스의 시작입니다.

C 프로그래밍 언어의 경우 상수 MQCI\_NEW\_SESSION\_ARRAY도 정의됩니다. 이는 MQCI\_NEW\_SESSION과 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

MQGET 호출의 경우 이는 입출력(I/O) 필드입니다. MQPUT 및 MQPUT1 호출의 경우, 이는 MQPMO\_NEW\_CORREL\_ID가 지정되지 않으면 입력 필드이고, MQPMO\_NEW\_CORREL\_ID가 지정되면 출력 필드입니다. 이 필드의 길이는 MQ\_CORREL\_ID\_LENGTH로 지정됩니다. 이 필드의 초기값은 MQCI\_NONE입니다.

#### **참고:**

한 계층에서 발행물의 상관 ID를 전달할 수 없습니다. 이 필드는 큐 관리자에서 사용됩니다.

#### **MQMD의 경우 BackoutCount (MQLONG)**

이 수는 메시지가 이전에 작업 단위의 일부로 MQGET 호출에 의해 리턴되고 그 뒤에 백업된 횟수입니다. 이는 애플리케이션이 메시지 콘텐츠에 근거한 오류 처리를 감지하는 데 도움이 됩니다. 이 수는 MQGMO\_BROWSE\_\* 옵션을 지정하는 MQGET 호출은 제외합니다.

이 수의 정확성은 **HardenGetBackout** 큐 속성의 영향을 받습니다. 804 페이지의 『큐의 속성』의 내용을 참조하십시오.

z/OS에서 값 255는 메시지가 255번 이상 백아웃됨을 의미합니다. 리턴된 값은 255를 초과하지 않습니다.

MQGET 호출의 출력 필드입니다. 이는 MQPUT 및 MQPUT1 호출에 대해 무시됩니다. 이 필드의 초기값은 0입니다.

#### **MQMD의 경우 ReplyToQ (MQCHAR48)**

이는 메시지에 대한 가져오기 요청을 발행한 애플리케이션이 MQMT\_REPLY 및 MQMT\_REPORT 메시지를 전송하는 메시지 큐의 이름입니다. 이름은 *ReplyToQMgr*이 식별한 큐 관리자에서 정의되는 큐의 로컬 이름입니다. 이 큐는 전송 큐 관리자가 메시지를 넣을 때 확인하지 않더라도 모델 큐가 아니어야 합니다.

MQPUT 및 MQPUT1 호출의 경우 *MsgType* 필드의 값이 MQMT\_REQUEST이거나 *Report* 필드에 의해 보고 메시지가 요청되는 경우 이 필드가 공백이 아니어야 합니다. 그러나 지정된(또는 대체된) 값은 메시지 유형에 상관 없이 메시지에 대해 Get 요청을 실행하는 애플리케이션으로 전달됩니다.

*ReplyToQMgr* 필드가 공백이면 로컬 큐 관리자가 소유하고 있는 큐 정의에서 *ReplyToQ* 이름을 검색합니다. 리모트 큐의 로컬 정의가 이 이름과 함께 존재하는 경우 전송된 메시지의 *ReplyToQ* 값은 리모트 큐의 정의에서 **RemoteQName** 속성의 값으로 대체되며 수신 애플리케이션이 메시지에 대해 MQGET 호출을 발행할 때 메시지 디스크립터에서 이 값이 리턴됩니다. 리모트 큐의 로컬 정의가 없으면 *ReplyToQ*가 변경되지 않습니다.



이름이 지정된 경우 후미 문자 공백을 포함할 수 있습니다. 첫 번째 널 문자 및 널 문자 뒤에 오는 문자는 공백으로 처리됩니다. 그렇지 않으면 이름이 큐에 대해 이름 지정 규칙을 준수하는지 확인하지 않습니다. 전송된 메시지에서 *ReplyToQ*가 대체되는 경우 전송되는 이름에 대해서도 위 설명이 적용됩니다. 상황에 따라, 이름이 지정되었는지만 확인됩니다.

응답 대상 큐가 필요하지 않은 경우 *ReplyToQ* 필드를 공백, 널 문자열(C 프로그래밍 언어인 경우) 또는 널 문자 앞에 오는 하나 이상의 공백으로 설정하십시오. 필드를 초기화되지 않은 상태로 두지 마십시오.

MQGET 호출의 경우, 큐 관리자는 항상 필드 길이까지 공백으로 채워진 이름을 리턴합니다.

보고 메시지를 요구하는 메시지를 전달할 수 없고 보고 메시지도 지정된 큐에 전달할 수 없는 경우, 기존 메시지 및 보고 메시지 둘 다 데드-레터(미발송-메시지) 큐로 이동합니다(768 페이지의 『큐 관리자의 속성』에 설명된 **DeadLetterQName** 속성 참조).

이 필드는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백입니다.

### **MQMD의 경우 ReplyToQMgr (MQCHAR48)**

이는 응답 메시지 또는 보고 메시지를 전송하는 큐 관리자의 이름입니다. *ReplyToQ*는 이 큐 관리자에서 정의되는 큐의 로컬 이름입니다.

*ReplyToQMgr* 필드가 공백이면 로컬 큐 관리자가 큐 정의에서 *ReplyToQ* 이름을 검색합니다. 리모트 큐의 로컬 정의가 이 이름과 함께 존재하는 경우 전송된 메시지의 *ReplyToQMgr* 값은 리모트 큐의 정의에서 **RemoteQMgrName** 속성의 값으로 대체되며 수신 애플리케이션이 메시지에 대해 MQGET 호출을 발행할 때 메시지 디스크립터에서 이 값이 리턴됩니다. 리모트 큐의 로컬 정의가 없으면 메시지와 함께 전송되는 *ReplyToQMgr*이 로컬 큐 관리자의 이름입니다.

이름이 지정된 경우 후미 문자 공백을 포함할 수 있습니다. 첫 번째 널 문자 및 널 문자 뒤에 오는 문자는 공백으로 처리됩니다. 그렇지 않으면 이름이 큐 관리자에 대한 이름 지정 규칙을 준수하는지 또는 이 이름이 전송 큐 관리자에 알려지는지 여부를 확인하지 않습니다. *ReplyToQMgr*이 전송된 메시지에서 대체되는 경우 이는 전송된 이름에 대해서도 적용됩니다.

응답 대상 큐가 필요하지 않은 경우 *ReplyToQMgr* 필드를 공백, 널 문자열(C 프로그래밍 언어인 경우) 또는 널 문자 앞에 오는 하나 이상의 공백으로 설정하십시오. 필드를 초기화되지 않은 상태로 두지 마십시오.

MQGET 호출의 경우, 큐 관리자는 항상 필드 길이까지 공백으로 채워진 이름을 리턴합니다.

이 필드는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 길이는 MQ\_Q\_MGR\_NAME\_LENGTH로 제공됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백입니다.

### **MQMD의 경우 UserIdentifier (MQCHAR12)**

메시지의 **ID 컨텍스트**의 부분입니다. 메시지 컨텍스트에 대한 자세한 정보는 412 페이지의 『MQMD - 메시지 디스크립터』 및 **메시지 컨텍스트**를 참조하십시오.

*UserIdentifier*는 메시지를 생성한 애플리케이션의 사용자 ID를 지정합니다. 큐 관리자는 이 정보를 문자 데이터로 처리하지만 그 형식은 정의하지 않습니다.

메시지를 수신한 후 후속 MQOPEN 또는 MQPUT1 호출의 **ObjDesc** 매개변수에 있는 *AlternateUserId* 필드에서 *UserIdentifier*를 사용하여 열기를 수행하는 애플리케이션 대신 *UserIdentifier* 사용자에 대한 권한 검사를 수행하십시오.

큐 관리자가 MQPUT 또는 MQPUT1 호출에 대해 이 정보를 생성하는 경우:

- z/OS에서 큐 관리자는 MQOO\_ALTERNATE\_USER\_AUTHORITY 또는 MQPMO\_ALTERNATE\_USER\_AUTHORITY 옵션이 지정된 경우 MQOPEN 또는 MQPUT1 호출의 **ObjDesc** 매개변수에서 *AlternateUserId*를 사용합니다. 관련 옵션이 지정되지 않은 경우 큐 관리자는 환경에서 판별된 사용자 ID를 사용합니다.
- 다른 환경에서는 큐 관리자가 항상 환경에서 판별된 사용자 ID를 사용합니다.

사용자 ID가 환경에서 결정되는 경우:

- z/OS에서 큐 관리자는 다음을 사용합니다.
  - MVS(배치)의 경우 JES JOB 카드 또는 시작된 태스크의 사용자 ID
  - TSO의 경우 작업 제출 동안 작업에 전파된 사용자 ID
  - CICS의 경우, 태스크와 연관된 사용자 ID
  - IMS의 경우 사용자 ID는 애플리케이션 유형에 따라 다릅니다.

- 다음의 경우:

- Nonmessage BMP 리전
- Nonmessage IFP 리전
- 성공적인 GU 호출을 발행하지 않은 메시지 BMP 및 메시지 IFP 영역

큐 관리자가 JES JOB 카드 영역 또는 TSO 사용자 ID로부터 사용자 ID를 사용합니다. 공백이거나 널인 경우, 이는 프로그램 스펙 블록(PSB)의 이름을 사용합니다.

- 다음의 경우:

- 성공적인 GU 호출을 발행한 메시지 BMP 및 메시지 IFP 영역
- MPP 리전

큐 관리자가 다음 중 하나를 사용합니다.

- 메시지와 연관된 사인온한 사용자 ID
- 논리 터미널(LTERM) 이름
- 영역 JES JOB 카드의 사용자 ID
- TSO 사용자 ID
- PSB 이름

- IBM i에서, 큐 관리자는 애플리케이션 작업과 연관된 사용자 프로파일의 이름을 사용합니다.

- AIX and Linux에서 큐 관리자는 다음을 사용합니다.

- 애플리케이션의 로그인 이름
- 사용 가능한 로그인이 없는 경우, 프로세스의 유효한 사용자 ID
- 애플리케이션이 CICS 트랜잭션인 경우, 트랜잭션과 연관된 사용자 ID

- Windows 시스템에서 큐 관리자는 처음 12자의 로그인된 사용자 이름을 사용합니다.

일반적으로 이 필드는 큐 관리자에 의해 생성되는 출력 필드이지만 MQPUT 또는 MQPUT1 호출의 경우 이 필드를 입/출력(I/O) 필드로 만들고 큐 관리자가 이 정보를 생성하도록 하는 대신 UserIdentifier 필드를 지정할 수 있습니다. 큐 관리자가 MQPUT 또는 MQPUT1 호출에 대해 UserIdentifier 필드를 생성하지 않도록 하려면 PutMsgOpts 매개변수에서 MQPMO\_SET\_IDENTITY\_CONTEXT 또는 MQPMO\_SET\_ALL\_CONTEXT를 지정하고 UserIdentifier 필드에서 사용자 ID를 지정하십시오.

MQPUT 및 MQPUT1 호출의 경우 MQPMO\_SET\_IDENTITY\_CONTEXT 또는 MQPMO\_SET\_ALL\_CONTEXT가 PutMsgOpts 매개변수에서 지정되면 이는 입출력(I/O) 필드입니다. 필드 내에서 널 문자 이후의 정보는 제거됩니다. 큐 관리자가 널 문자 및 다음 문자를 공백으로 변환합니다. MQPMO\_SET\_IDENTITY\_CONTEXT 또는 MQPMO\_SET\_ALL\_CONTEXT가 지정되지 않으면 이 필드는 입력에서 무시되며 출력 전용 필드입니다.

MQPUT 또는 MQPUT1 호출이 성공적으로 완료되면, 이 필드에는 큐에 넣은 경우 메시지와 함께 전송된 UserIdentifier가 포함됩니다. 이는 보유되는 경우에 메시지와 함께 보유되는 UserIdentifier의 값이 됩니다. 보유된 발행에 대한 세부사항은 MQPMO\_RETAIN에 대한 설명을 참조하십시오. 그러나 송신되는 모든 발행에서 UserIdentifier를 대체하기 위한 값을 제공하므로 메시지가 구독자에게 발행으로 송신되는 경우에는 UserIdentifier로 사용되지 않습니다. 메시지에 컨텍스트가 없는 경우 필드 전체가 공백입니다.

MQGET 호출의 출력 필드입니다. 이 필드의 길이는 MQ\_USER\_ID\_LENGTH에 지정되어 있습니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 12자의 공백 문자입니다.

### MQMD의 경우 AccountingToken (MQBYTE32)

이는 메시지의 ID 컨텍스트 부분인 계정 토큰입니다. 메시지 컨텍스트에 대한 자세한 정보는 [412 페이지의 『MQMD - 메시지 디스크립터』](#) 및 메시지 컨텍스트를 참조하십시오.

AccountingToken을 사용하면 애플리케이션이 메시지의 결과로 완료된 작업에 적절히 청구할 수 있습니다. 큐 관리자는 이 정보를 비트 문자열로 처리하며 그 콘텐츠를 검사하지 않습니다.

큐 관리자는 다음과 같이 이 정보를 생성합니다.

- 필드의 첫 번째 바이트는 뒤따르는 바이트 단위로 제공된 회계 정보의 길이로 설정됩니다. 이 길이의 범위는 0 부터 30까지이며 2진 정수로 첫 번째 바이트에 저장됩니다.
- 두 번째 및 후속 바이트(길이 필드에서 지정된 대로)는 환경에 적절한 회계 정보로 설정됩니다.
  - **z/OS** z/OS에서 회계 정보는 다음으로 설정됩니다.
    - z/OS 배치의 경우, JES JOB 카드 또는 EXEC 카드의 JES ACCT 명령문에서 나온 회계 정보입니다(섬표 구분 기호는 X'FF'로 변경됨). 이 정보는 필요 시 31바이트로 잘립니다.
    - TSO의 경우, 사용자의 회계 번호입니다.
    - CICS의 경우, LU 6.2 작업 단위 ID입니다(UEPUOWDS)(26바이트).
    - IMS의 경우, 16자 IMS 복구 토큰과 함께 병합된 8자 PSB 이름입니다.
  - **IBM i** IBM i에서, 회계 정보는 작업의 회계 코드로 설정됩니다.
  - **Linux** **AIX** AIX and Linux에서 계정 정보는 ASCII 문자에서 숫자 사용자 ID로 설정됩니다.
  - **Windows** Windows에서, 회계 정보는 압축 형식으로 Windows 보안 ID(SID)로 설정됩니다. 이 SID는 *UserIdentifier* 필드에 저장된 사용자 ID를 식별합니다. SID가 *AccountingToken* 필드에 저장될 때 6바이트 ID 권한(SID의 세 번째 및 후속 바이트에 있음)이 생략됩니다. 예를 들어, Windows SID가 28바이트 길이인 경우 22바이트의 SID 정보는 *AccountingToken* 필드에 저장됩니다.
- 계정 필드의 마지막 바이트(32바이트)는 계정 토큰 유형으로 설정됩니다(이 경우 MQACTT\_NT\_SECURITY\_ID, x'0b').

#### **MQACTT\_CICS\_LUOW\_ID**

CICS LUOW ID입니다.

#### **Windows MQACTT\_NT\_SECURITY\_ID**

Windows 보안 ID.

#### **IBM i MQACTT\_OS400\_ACCOUNT\_TOKEN**

IBM i 회계 토큰입니다.

#### **UNIX MQACTT\_UNIX\_NUMERIC\_ID**

UNIX 숫자 ID입니다.

#### **MQACTT\_USER**

사용자 정의 회계 토큰입니다.

#### **MQACTT\_UNKNOWN**

알 수 없는 회계-토큰 유형입니다.

이 회계-토큰 유형은 다음 환경의 명시적 값으로만 설정됩니다.

- **AIX** AIX
- **IBM i** IBM i
- **Linux** Linux
- **Windows** Windows

및 해당 시스템에 연결된 IBM MQ MQI clients 다른 환경에서는 계정 토큰 유형이 값 MQACTT\_UNKNOWN으로 설정됩니다. 이러한 환경에서는 *PutApplType* 필드를 사용하여 수신된 계정 토큰의 유형을 추정하십시오.

- 기타 모든 바이트는 2진 0으로 설정됩니다.

MQPUT 및 MQPUT1 호출의 경우 MQPMO\_SET\_IDENTITY\_CONTEXT 또는 MQPMO\_SET\_ALL\_CONTEXT가 **PutMsgOpts** 매개변수에서 지정되면 이는 입출력(I/O) 필드입니다. MQPMO\_SET\_IDENTITY\_CONTEXT 또는 MQPMO\_SET\_ALL\_CONTEXT가 지정되지 않으면 이 필드는 입력 시 무시되며 출력 전용 필드입니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트를 참조하십시오](#).

MQPUT 또는 MQPUT1 호출이 성공적으로 완료되면, 이 필드에는 큐에 넣은 경우 메시지와 함께 전송된 AccountingToken이 포함됩니다. 이는 보유되는 경우에 메시지와 함께 보유되는 AccountingToken의 값이 됩니다. 보유된 발행에 대한 세부사항은 490 페이지의 『MQPMO에 대한 옵션 (MQLONG)』에서 MQPMO\_RETAIN에 대한 설명을 참조하십시오. 그러나 송신되는 모든 발행에서 AccountingToken을 대체하기 위한 값을 제공하므로 메시지가 발행으로 구독자에게 송신되는 경우에는 AccountingToken으로 사용되지 않습니다. 메시지에 컨텍스트가 없는 경우 필드 전체가 2진 0입니다.

MQGET 호출의 출력 필드입니다.

이 필드는 큐 관리자의 문자 세트를 기반으로 변환되지 않습니다. 이 필드는 비트 문자열로 처리되며 문자의 문자열로 처리되지 않습니다.

큐 관리자는 이 필드의 정보로 수행하는 작업이 없습니다. 애플리케이션은 회계 용도로 정보를 사용하려는 경우 정보를 해석해야 합니다.

AccountingToken 필드에 대해 다음 특수 값을 사용할 수 있습니다.

#### **MQACT\_NONE**

계정 토큰이 지정되지 않습니다.

이 값은 필드 길이만큼의 2진 0입니다.

C 프로그래밍 언어의 경우 상수 MQACT\_NONE\_ARRAY도 정의됩니다. 이는 MQACT\_NONE과 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

이 필드의 길이는 MQ\_ACCOUNTING\_TOKEN\_LENGTH로 지정됩니다. 이 필드의 초기값은 MQACT\_NONE입니다.

#### **MQMD의 ApplIdentity데이터 (MQCHAR32)**

메시지의 **ID** 컨텍스트의 부분입니다. 메시지 컨텍스트에 대한 자세한 정보는 412 페이지의 『MQMD - 메시지 디스크립터』 및 [메시지 컨텍스트를 참조하십시오](#).

*ApplIdentityData*는 애플리케이션 스위트가 정의하는 정보이며 메시지 또는 해당 진원지에 대한 추가 정보를 제공하는 데 사용할 수 있습니다. 큐 관리자는 이 정보를 문자 데이터로 처리하지만 그 형식은 정의하지 않습니다. 큐 관리자가 이 정보를 생성할 때 이는 전부 공백입니다.

MQPUT 및 MQPUT1 호출의 경우 MQPMO\_SET\_IDENTITY\_CONTEXT 또는 MQPMO\_SET\_ALL\_CONTEXT가 **PutMsgOpts** 매개변수에서 지정되면 이는 입출력(I/O) 필드입니다. 널 문자가 있는 경우 널 및 다음에 오는 임의의 문자는 큐 관리자에 의해 공백으로 변환됩니다. MQPMO\_SET\_IDENTITY\_CONTEXT 또는 MQPMO\_SET\_ALL\_CONTEXT가 지정되지 않으면 이 필드는 입력 시 무시되며 출력 전용 필드입니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트를 참조하십시오](#).

MQPUT 또는 MQPUT1 호출이 성공적으로 완료되면, 이 필드에는 큐에 넣은 경우 메시지와 함께 전송된 *ApplIdentityData*가 포함됩니다. 이는 메시지가 보유되는 경우 메시지와 함께 보관되는 *ApplIdentityData*의 값이 됩니다(보유된 발행에 대한 자세한 내용은 MQPMO\_RETAIN의 설명 참조). 그러나 구독자는 자신에게 송신되는 모든 발행에 있는 *ApplIdentityData*를 대체하기 위한 값을 제공하므로 메시지가 구독자에게 발행으로 송신되는 경우에는 *ApplIdentityData*로 사용되지 않습니다. 메시지에 컨텍스트가 없는 경우 필드 전체가 공백입니다.

MQGET 호출의 출력 필드입니다. 이 필드의 길이는 MQ\_APPL\_IDENTITY\_DATA\_LENGTH로 지정됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 32자의 공백 문자입니다.

#### **PutApplMQMD의 유형 (MQLONG)**

이는 메시지를 넣는 애플리케이션의 유형이며 메시지의 **원본 컨텍스트**의 일부입니다. 메시지 컨텍스트에 대한 자세한 정보는 412 페이지의 『MQMD - 메시지 디스크립터』 및 [메시지 컨텍스트를 참조하십시오](#).

*PutApplType*는 다음 표준 유형 중 하나를 가질 수 있습니다. 또한 고유한 유형을 정의할 수 있지만 MQAT\_USER\_FIRST에서 MQAT\_USER\_LAST 범위의 값만 가능합니다.

**MQAT\_AIX**

AIX 애플리케이션(MQAT\_UNIX와 동일한 값).

**MQAT\_AMQP**

AMQP 프로토콜 애플리케이션

**MQAT\_BROKER**

브로커입니다.

**MQAT\_CICS**

CICS 트랜잭션.

**MQAT\_CICS\_BRIDGE**

CICS bridge.

**MQAT\_CICS\_VSE**

CICS/VSE 트랜잭션.

**MQAT\_DOS**

PC DOS용 IBM MQ MQI client 애플리케이션입니다.

**MQAT\_DQM**

분산 큐 관리자 에이전트.

**MQAT\_GUARDIAN**

Tandem Guardian 애플리케이션입니다(MQAT\_NSK와 동일한 값).

**MQAT\_IMS**

IMS 애플리케이션입니다.

**MQAT\_IMS\_BRIDGE**

IMS 브릿지.

**MQAT\_JAVA**

Java.

**MQAT\_MVS**

MVS 또는 TSO 애플리케이션(MQAT\_ZOS와 동일한 값)입니다.

**MQAT\_NOTES\_AGENT**

Lotus Notes® Agent 애플리케이션입니다.

**MQAT\_OS390**

OS/390® 애플리케이션(MQAT\_ZOS와 동일한 값)입니다.

**MQAT\_OS400**

IBM i 애플리케이션입니다.

**MQAT\_QMGR**

큐 관리자.

**MQAT\_UNIX**

UNIX 애플리케이션입니다.

**MQAT\_VOS**

Stratus VOS 애플리케이션입니다.

**MQAT\_WINDOWS**

16비트 Windows 애플리케이션입니다.

**MQAT\_WINDOWS\_NT**

32비트 Windows 애플리케이션입니다.

**MQAT\_WLM**

z/OS 워크로드 관리자 애플리케이션입니다.

**MQAT\_XCF**

XCF.

## MQAT\_ZOS

z/OS 애플리케이션입니다.

## MQAT\_DEFAULT

기본 애플리케이션 유형입니다.

이는 애플리케이션이 실행되는 플랫폼의 기본 애플리케이션 유형입니다.

**참고:** 이 상수의 값은 환경에 따라 달라집니다. 이로 인해 항상 애플리케이션이 실행될 플랫폼에 적절한 헤더, 포함 파일 또는 COPY 파일을 사용하여 애플리케이션을 컴파일하십시오.

## MQAT\_UNKNOWN

기타 컨텍스트 정보가 있는 경우에도 애플리케이션 유형을 알 수 없음을 표시하려면 이 값을 사용하십시오.

## MQAT\_USER\_FIRST

사용자 정의된 애플리케이션 유형의 최저값입니다.

## MQAT\_USER\_LAST

사용자 정의된 애플리케이션 유형의 최고값입니다.

다음은 발생할 수 있는 특수 값입니다.

## MQAT\_NO\_CONTEXT

이 값은 컨텍스트 없이 메시지를 넣을 때, 즉, MQPMO\_NO\_CONTEXT 컨텍스트 옵션이 지정된 경우에 큐 관리자에 의해 설정됩니다.

메시지를 검색할 때 메시지에 컨텍스트가 있는지 파악하기 위해 이 값에 대해 *PutApplType*을 테스트할 수 있습니다. 기타 컨텍스트 필드 중 공백이 아닌 것이 있는 경우에는 MQPMO\_SET\_ALL\_CONTEXT를 사용하는 애플리케이션으로 *PutApplType*을 MQAT\_NO\_CONTEXT로 설정하지 않는 것이 좋습니다.

애플리케이션 넣기의 결과로 큐 관리자가 이 정보를 생성하는 경우 필드는 환경이 판별하는 값으로 설정됩니다. IBM i에서는 MQAT\_OS400으로 설정됩니다. 큐 관리자는 IBM i에서 MQAT\_CICS를 사용하지 않습니다.

MQPUT 및 MQPUT1 호출의 경우, MQPMO\_SET\_ALL\_CONTEXT가 **PutMsgOpts** 매개변수에 지정되어 있으면 이는 입출력(I/O) 필드입니다. MQPMO\_SET\_ALL\_CONTEXT를 지정하지 않으면 입력에서 이 필드가 무시되며 출력 전용 필드가 됩니다.

MQGET 호출의 출력 필드입니다. 이 필드의 초기값은 MQAT\_NO\_CONTEXT입니다.

## PutApplMQMD의 이름 (MQCHAR28)

이는 메시지를 넣는 애플리케이션의 이름이며 메시지의 원본 컨텍스트의 일부입니다. 콘텐츠는 플랫폼에 따라 다르며 릴리스에 따라 다를 수 있습니다.

메시지 컨텍스트에 대한 자세한 정보는 412 페이지의 『MQMD - 메시지 디스크립터』 및 메시지 컨텍스트를 참조하십시오.

추가 프로그래밍 언어로 애플리케이션 이름을 지정할 수 있습니다. 자세한 정보는 지원되는 프로그래밍 언어로 애플리케이션 이름 지정을 참조하십시오.

*PutApplName*의 형식은 *PutApplType*의 값에 따라 다르며 한 릴리스에서 다른 릴리스로 변경될 수 있습니다. 변경은 자주 발생하지 않지만 환경이 변경되면 발생합니다.

큐 관리자가 이 필드를 설정하는 경우(즉, MQPMO\_SET\_ALL\_CONTEXT를 제외한 다른 모든 옵션의 경우) 환경이 판별하는 값으로 필드를 설정합니다.

- ▶ **z/OS** z/OS에서 큐 관리자는 다음을 사용합니다.
  - z/OS 배치의 경우, JES JOB 카드의 8자 작업 이름
  - TSO의 경우, 7자 TSO 사용자 ID
  - CICS의 경우, 4자 tranid가 뒤따르는 8자 applid
  - IMS의 경우, 8자 PSB 이름이 뒤따르는 8자 IMS 시스템 ID
  - XCF의 경우, 16자 XCF 멤버 이름이 뒤따르는 8자 XCF 그룹 이름
  - 큐 관리자에서 생성된 메시지의 경우, 큐 관리자 이름의 첫 번째 28자



- CICS 없이 분산 큐잉의 경우, 8자 태스크 ID가 뒤따르는 데드-레터 큐에 넣는 모듈의 8자 이름이 뒤따르는 채널 시작기의 8자 작업 이름

각 이름은 필드의 나머지에 공백이 있는 것처럼 오른쪽까지 공백으로 채워집니다. 둘 이상의 이름이 있는 경우 해당 구분자가 없습니다.

- **Windows** Windows 시스템에서 큐 관리자는 다음 이름을 사용합니다.

- CICS 애플리케이션의 경우, CICS 트랜잭션 이름
- 비CICS 애플리케이션의 경우, 실행 파일의 완전한 이름 중 가장 오른쪽 28자

- **IBM i** IBM i에서, 큐 관리자는 완전한 작업 이름을 사용합니다.

- **Linux** **AIX** AIX and Linux에서 큐 관리자는 다음 이름을 사용합니다.

- CICS 애플리케이션의 경우, CICS 트랜잭션 이름
- CICS 애플리케이션의 경우 MQ는 프로세스 이름에 대해 운영 체제를 요청합니다. 이는 전체 경로 없이 프로그램 파일 이름으로 리턴됩니다. 그런 다음 MQ는 다음과 같이 이 프로세스 이름을 MQMD.PutAppName 필드에 위치시킵니다.

#### **AIX** **AIX**

이름이 28바이트 이하인 경우 이름이 삽입되고 오른쪽까지 공백으로 채워집니다.

이름이 28바이트보다 큰 경우 이름의 가장 왼쪽 28바이트가 삽입됩니다.

#### **Linux** **Linux**

이름이 15바이트 이하인 경우 이름이 삽입되고 오른쪽까지 공백으로 채워집니다.

이름이 15바이트보다 큰 경우 이름의 가장 왼쪽 15바이트가 삽입됩니다.

예를 들어, /opt/mqm/samp/bin/amqsput QNAME QMNAME을 실행하면 PutAppName은 'amqsput'입니다. 이 CHAR28 필드에는 21개의 패딩 공백 문자가 있습니다. /opt/mqm/samp/bin이 포함된 전체 경로가 PutAppName에 포함되지 않음을 참고하십시오.

MQPUT 및 MQPUT1 호출의 경우, MQPMO\_SET\_ALL\_CONTEXT가 **PutMsgOpts** 매개변수에 지정되어 있으면 이는 입출력(I/O) 필드입니다. 필드 내에서 널 문자 이후의 정보는 제거됩니다. 널 문자 및 다음 문자는 큐 관리자에 의해 공백으로 변환됩니다. MQPMO\_SET\_ALL\_CONTEXT를 지정하지 않으면 입력에서 이 필드가 무시되며 출력 전용 필드가 됩니다.

### **MQMD의 경우 PutDate (MQCHAR8)**

이는 메시지를 넣은 날짜이며 메시지의 **원본 컨텍스트**의 일부입니다. 메시지 컨텍스트에 대한 자세한 정보는 412 페이지의 『MQMD - 메시지 디스크립터』 및 메시지 컨텍스트를 참조하십시오.

이 필드가 큐 관리자에 의해 생성될 때 날짜에 사용되는 형식입니다.

- YYYYMMDD

여기서 각 문자는 다음을 나타냅니다.

#### **YYYY**

연도(4자리 숫자)

#### **MM**

연 중 월(01에서 12)

#### **DD**

월 중 일(01에서 31)

그리니치 표준시(GMT)에 맞게 정확하게 설정된 시스템 시계에 따라 **PutDate** 및 **PutTime** 필드에 GMT가 사용 됩니다.

메시지를 작업 단위의 일부로 넣은 경우, 날짜는 작업 단위가 커밋된 날짜가 아니라 메시지를 넣은 날짜입니다.

MQPUT 및 MQPUT1 호출의 경우, MQPMO\_SET\_ALL\_CONTEXT가 **PutMsgOpts** 매개변수에 지정되어 있으면 이는 입출력(I/O) 필드입니다. 필드의 콘텐츠는 필드 내에서 널 문자 다음 정보가 제거되는 것을 제외하고 큐 관

리자에서 확인되지 않습니다. 큐 관리자가 널 문자 및 다음 문자를 공백으로 변환합니다.  
MQPMO\_SET\_ALL\_CONTEXT를 지정하지 않으면 입력에서 이 필드가 무시되며 출력 전용 필드가 됩니다.

MQGET 호출의 출력 필드입니다. 이 필드의 길이는 MQ\_PUT\_DATE\_LENGTH로 지정됩니다. 이 필드의 초기값은 C에서는 널 문자열이고 기타 프로그래밍 언어에서는 8자의 공백 문자입니다.

### **MQMD의 경우 PutTime (MQCHAR8)**

메시지를 넣은 시간이며 메시지의 **원본 컨텍스트**의 일부입니다. 메시지 컨텍스트에 대한 자세한 정보는 [412 페이지의 『MQMD - 메시지 디스크립터』](#) 및 [메시지 컨텍스트](#)를 참조하십시오.

이 필드가 큐 관리자에 의해 생성될 때 시간에 사용되는 형식입니다.

- HHMMSSSTH

여기서 문자는 다음을 나타냅니다(순서대로).

#### **HH**

시간(00에서 23까지)

#### **MM**

분(00에서 59까지)

#### **SS**

초(00 - 59, 참고 참조)

#### **T**

1/10초(0에서 9)

#### **H**

1/100초(0에서 9)

**참고:** 시스템 시계가 매우 정확한 시간 표준에 동기화되는 경우 매우 드물게 *PutTime*에서 초에 대해 60 또는 61이 리턴될 수 있습니다. 이는 윤초가 글로벌 시간 표준으로 삽입되는 경우 발생합니다.

그리니치 표준시(GMT)에 맞게 정확하게 설정된 시스템 시계에 따라 *PutDate* 및 *PutTime* 필드에 GMT가 사용 됩니다.

메시지를 작업 단위의 일부로 넣은 경우, 시간은 작업 단위가 커밋된 시간이 아니라 메시지를 넣은 시간입니다.

MQPUT 및 MQPUT1 호출의 경우, MQPMO\_SET\_ALL\_CONTEXT가 **PutMsgOpts** 매개변수에 지정되어 있으면 이는 입출력(I/O) 필드입니다. 큐 관리자는 필드의 콘텐츠를 확인하지 않습니다. 단, 필드 내에서 널 문자 다음에 오는 정보는 제거됩니다. 큐 관리자가 널 문자 및 다음 문자를 공백으로 변환합니다.

MQPMO\_SET\_ALL\_CONTEXT를 지정하지 않으면 입력에서 이 필드가 무시되며 출력 전용 필드가 됩니다.

MQGET 호출의 출력 필드입니다. 이 필드의 길이는 MQ\_PUT\_TIME\_LENGTH로 지정됩니다. 이 필드의 초기값은 C에서는 널 문자열이고 기타 프로그래밍 언어에서는 8자의 공백 문자입니다.

### **MQMD의 ApplOrigin데이터 (MQCHAR4)**

이는 메시지의 원본 컨텍스트의 일부입니다. 메시지 컨텍스트에 대한 자세한 정보는 [412 페이지의 『MQMD - 메시지 디스크립터』](#) 및 [메시지 컨텍스트](#)를 참조하십시오.

*ApplOriginData*는 메시지 원본에 대한 추가 정보를 제공하는 데 사용할 수 있는 애플리케이션 스위트에 의해 정의된 정보입니다. 예를 들어, ID 데이터가 신뢰되는지 여부를 표시하기 위해 적절한 사용자 권한을 갖고 실행하는 애플리케이션이 이를 설정할 수 있습니다.

큐 관리자는 이 정보를 문자 데이터로 처리하지만 그 형식은 정의하지 않습니다. 큐 관리자가 이 정보를 생성할 때 이는 전부 공백입니다.

MQPUT 및 MQPUT1 호출의 경우, MQPMO\_SET\_ALL\_CONTEXT가 **PutMsgOpts** 매개변수에 지정되어 있으면 이는 입출력(I/O) 필드입니다. 필드 내에서 널 문자 이후의 정보는 제거됩니다. 큐 관리자가 널 문자 및 다음 문자를 공백으로 변환합니다. MQPMO\_SET\_ALL\_CONTEXT를 지정하지 않으면 입력에서 이 필드가 무시되며 출력 전용 필드가 됩니다.

MQGET 호출의 출력 필드입니다. 이 필드의 길이는 MQ\_APPL\_ORIGIN\_DATA\_LENGTH로 지정됩니다. 이 필드의 초기값은 C에서는 널 문자열이고 기타 프로그래밍 언어에서는 4자의 공백 문자입니다.



ApplOriginData를 설정해도 메시지가 발행되는 경우 수신하는 구독이 비어 있습니다.

## MQMD의 경우 GroupId (MQBYTE24)

이는 실제 메시지가 속한 논리 메시지 또는 특정 메시지 그룹을 식별하는 데 사용되는 바이트 문자열입니다. GroupId는 또한 메시지에 대해 세그먼트화가 허용되는 경우에도 사용됩니다. 이러한 모든 경우에서 GroupId에는 널이 아닌 값이 있고 하나 이상의 다음 플래그는 MsgFlags 필드에서 설정됩니다.

- MQMF\_MSG\_IN\_GROUP
- MQMF\_LAST\_MSG\_IN\_GROUP
- MQMF\_SEGMENT
- MQMF\_LAST\_SEGMENT
- MQMF\_SEGMENTATION\_ALLOWED

해당 플래그가 설정되지 않는 경우 GroupId에는 특수 널 값 MQGI\_NONE이 있습니다.

다음 경우에는 애플리케이션이 MQPUT 또는 MQGET 호출에서 이 필드를 설정할 필요가 없습니다.

- MQPUT 호출에서 MQPMO\_LOGICAL\_ORDER가 지정됩니다.
- MQGET 호출에서 MQMO\_MATCH\_GROUP\_ID가 지정되지 않습니다.

이는 보고 메시지가 아닌 메시지에 대해 이러한 호출을 사용하는 권장 방법입니다. 그러나 애플리케이션이 추가 제어가 필요하거나 호출이 MQPUT1인 경우 애플리케이션은 GroupId가 적절한 값으로 설정되었는지 확인해야 합니다.

메시지 그룹 및 세그먼트는 그룹 ID가 고유한 경우에만 올바르게 처리될 수 있습니다. 이러한 이유로 애플리케이션은 고유한 그룹 ID를 생성할 수 없습니다. 대신 애플리케이션은 다음 중 하나를 수행해야 합니다.

- MQPMO\_LOGICAL\_ORDER가 지정되는 경우 큐 관리자는 논리 메시지의 그룹 또는 세그먼트에서 첫 번째 메시지에 대해 고유한 그룹 ID를 자동으로 생성하고 논리 메시지의 그룹 또는 세그먼트에서 나머지 메시지에 대해 해당 그룹 ID를 사용하므로 애플리케이션이 특별한 조치를 수행할 필요가 없습니다. 이는 권장되는 프로시저입니다.
- MQPMO\_LOGICAL\_ORDER가 지정되지 않는 경우, 애플리케이션은 논리 메시지의 그룹 또는 세그먼트에서 메시지에 대해 첫 번째 MQPUT 또는 MQPUT1 호출에서 GroupId를 MQGI\_NONE으로 설정하여 그룹 ID를 생성하도록 큐 관리자에 요청해야 합니다. 해당 호출에서 출력 시 큐 관리자가 리턴하는 그룹 ID는 논리 메시지의 그룹 또는 세그먼트에서 나머지 메시지에 사용되어야 합니다. 메시지 그룹이 세그먼트화된 메시지를 포함하는 경우 그룹의 모든 세그먼트 및 메시지에 대해 동일한 그룹 ID를 사용해야 합니다.

MQPMO\_LOGICAL\_ORDER가 지정되지 않은 경우 논리 메시지의 그룹 및 세그먼트에 있는 메시지는 역순을 포함하여 어떤 순서로든 넣을 수 있지만 그룹 ID는 해당 메시지에 대해 발행되는 첫 번째 MQPUT 또는 MQPUT1 호출로 할당해야 합니다.

MQPUT 및 MQPUT1 호출에 대한 입력에서 큐 관리자는 큐의 실제 순서에서 설명하는 값을 사용합니다. MQPUT 및 MQPUT1 호출에서 출력 시 큐 관리자는 열린 오브젝트가 단일 큐이지만 분배 목록이 아닌 경우 메시지와 함께 전송된 값으로 이 필드를 설정하지만 열린 오브젝트가 분배 목록인 경우 이를 변경되지 않은 상태로 둡니다. 후자의 경우, 애플리케이션이 생성된 그룹 ID를 알아야 하는 경우 애플리케이션은 GroupId 필드가 포함된 MQPMR 레코드를 제공해야 합니다.

MQGET 호출에 대한 입력에서 큐 관리자는 386 페이지의 표 495에서 설명하는 값을 사용합니다. MQGET 호출에서 출력 시 큐 관리자는 이 필드를 검색된 메시지의 값으로 설정합니다.

다음 특수 값이 정의됩니다.

### MQGI\_NONE

지정된 그룹 ID가 없습니다.

이 값은 필드 길이만큼의 2진 0입니다. 이는 그룹에 없고 논리 메시지의 세그먼트가 아니며 세그먼트가 허용되지 않는 메시지에 사용되는 값입니다.

C 프로그래밍 언어의 경우 상수 MQGI\_NONE\_ARRAY도 정의됩니다. 이는 MQGI\_NONE과 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

이 필드의 길이는 MQ\_GROUP\_ID\_LENGTH로 지정됩니다. 이 필드의 초기값은 MQGI\_NONE입니다. *Version* 이(가) MQMD\_VERSION\_2보다 작으면 이 필드는 무시됩니다.

### **MsgSeqMQMD의 수 (MQLONG)**

그룹에 있는 논리 메시지의 순번입니다.

순서 번호는 1에서 시작하여, 그룹의 각 새 논리 메시지마다 1씩 최대 999 999 999까지 증가합니다. 그룹에 없는 실제 메시지의 순서 번호는 1입니다.

다음 경우에는 애플리케이션이 MQPUT 또는 MQGET 호출에서 이 필드를 설정할 필요가 없습니다.

- MQPUT 호출에서 MQPMO\_LOGICAL\_ORDER가 지정됩니다.
- MQGET 호출에서 MQMO\_MATCH\_MSG\_SEQ\_NUMBER가 지정되지 않습니다.

이는 보고 메시지가 아닌 메시지에 대해 이러한 호출을 사용하는 권장 방법입니다. 그러나 애플리케이션이 추가 제어가 필요하거나 호출이 MQPUT1인 경우 애플리케이션은 *MsgSeqNumber*가 적절한 값으로 설정되었는지 확인해야 합니다.

MQPUT 및 MQPUT1 호출에 대한 입력에서 큐 관리자는 큐의 실제 순서에서 설명하는 값을 사용합니다. MQPUT 및 MQPUT1 호출에서 출력 시 큐 관리자는 이 필드를 메시지와 함께 전송된 값으로 설정합니다.

MQGET 호출에 대한 입력에서 큐 관리자는 386 페이지의 표 495에 표시된 값을 사용합니다. MQGET 호출에서 출력 시 큐 관리자는 이 필드를 검색된 메시지의 값으로 설정합니다.

이 필드의 초기값은 1입니다. *Version*이(가) MQMD\_VERSION\_2보다 작으면 이 필드는 무시됩니다.

### **MQMD의 오프셋 (MQLONG)**

이는 데이터가 일부를 형성하는 논리 메시지의 시작부터 실제 메시지의 데이터 오프셋(바이트)입니다. 이 데이터를 세그먼트라고 합니다. 오프셋의 범위는 0 - 999,999,999입니다. 논리적 메시지의 세그먼트가 아닌 물리적 메시지의 오프셋은 0입니다.

다음 경우에는 애플리케이션이 MQPUT 또는 MQGET 호출에서 이 필드를 설정할 필요가 없습니다.

- MQPUT 호출에서 MQPMO\_LOGICAL\_ORDER가 지정됩니다.
- MQGET 호출에서 MQMO\_MATCH\_OFFSET이 지정되지 않습니다.

이는 보고 메시지가 아닌 메시지에 대해 이러한 호출을 사용하는 권장 방법입니다. 그러나 애플리케이션이 해당 조건을 조건을 준수하지 않거나 호출이 MQPUT1인 경우 애플리케이션은 *Offset*이 적절한 값으로 설정되었는지 확인해야 합니다.

MQPUT 및 MQPUT1 호출에 대한 입력에서 큐 관리자는 큐의 실제 순서에서 설명하는 값을 사용합니다. MQPUT 및 MQPUT1 호출에서 출력 시 큐 관리자는 이 필드를 메시지와 함께 전송된 값으로 설정합니다.

논리적 메시지의 세그먼트를 보고하는 보고 메시지의 경우 *OriginalLength* 필드가 MQOL\_UNDEFINED가 아닌 경우에만 큐 관리자가 보유하는 세그먼트 정보에서 오프셋을 업데이트하는 데 사용됩니다.

MQGET 호출에 대한 입력에서 큐 관리자는 386 페이지의 표 495에 표시된 값을 사용합니다. MQGET 호출에서 출력 시 큐 관리자는 이 필드를 검색된 메시지의 값으로 설정합니다.

이 필드의 초기값은 0입니다. *Version*이(가) MQMD\_VERSION\_2보다 작으면 이 필드는 무시됩니다.

### **MQMD의 경우 MsgFlags (MQLONG)**

MsgFlags는 메시지의 속성을 지정하거나 처리를 제어하는 플래그입니다.

MsgFlags는 다음 범주로 나뉩니다.

- 세그먼트화 플래그
- 상태 플래그

**세그먼트화 플래그:** 메시지가 큐에 비해 너무 큰 경우 일반적으로 메시지를 큐에 넣으려는 시도가 실패합니다. 세그먼트화는 큐 관리자 또는 애플리케이션이 메시지를 일명 세그먼트라는 더 작은 조각으로 분할하고 별도의 실제 메시지로 큐에 각 세그먼트를 배치하는 기술입니다. 메시지를 검색하는 애플리케이션은 세그먼트를 하나씩

검색하거나 큐 관리자가 세그먼트를 MQGET 호출에 의해 리턴되는 단일 메시지로 리어셈블링하도록 요청할 수 있습니다. 후자의 경우 MQGET 호출에서 MQGMO\_COMPLETE\_MSG 옵션을 지정하고 완전한 메시지를 수용하기에 충분히 큰 버퍼를 제공하는 방법으로 수행할 수 있습니다. (MQGMO\_COMPLETE에 대한 세부사항은 362 페이지의 『MQGMO - 메시지 가져오기 옵션』의 내용을 참조하십시오.) 메시지는 전송 큐 관리자, 중간 큐 관리자 또는 목적지 큐 관리자에서 세그먼트화할 수 있습니다.

메시지의 세그먼트화를 제어하기 위해 다음 중 하나를 지정할 수 있습니다.

#### **MQMF\_SEGMENTATION\_INHIBITED**

이 옵션은 메시지가 큐 관리자에 의해 세그먼트로 분할되지 않도록 합니다. 이미 세그먼트화된 메시지에 대해 이 옵션을 지정하면 이 옵션으로 인해 세그먼트가 더 작은 세그먼트로 구분되지 않습니다.

이 플래그의 값은 2진 0입니다. 기본값입니다.

#### **MQMF\_SEGMENTATION\_ALLOWED**

이 옵션은 메시지가 큐 관리자에 의해 세그먼트로 구분되는 것을 허용합니다. 이미 세그먼트화된 메시지에 대해 이 옵션을 지정하면 이 옵션으로 인해 세그먼트가 더 작은 세그먼트로 구분됩니다.

MQMF\_SEGMENTATION\_ALLOWED는 MQMF\_SEGMENT 또는 MQMF\_LAST\_SEGMENT를 설정하지 않고 설정할 수 있습니다.

- z/OS에서 큐 관리자는 메시지의 세그먼트를 지원하지 않습니다. 메시지가 큐에 너무 큰 경우 MQPUT 또는 MQPUT1 호출이 이유 코드 MQRC\_MSG\_TOO\_BIG\_FOR\_Q와 함께 실패합니다. 그러나 MQMF\_SEGMENTATION\_ALLOWED 옵션을 여전히 지정할 수 있고 메시지가 리모트 큐 관리자에서 세그먼트화되도록 허용합니다.

큐 관리자가 메시지를 세그먼트화할 때 큐 관리자는 각 세그먼트와 함께 전송되는 MQMD의 사본에서 MQMF\_SEGMENT 플래그를 켭니다. 그러나 MQPUT 또는 MQPUT1 호출에서 애플리케이션이 제공하는 MQMD에서 해당 플래그의 설정을 변경하지 않습니다. 논리 메시지의 마지막 세그먼트의 경우 큐 관리자가 세그먼트와 함께 전송되는 MQMD에서 MQMF\_LAST\_SEGMENT 플래그를 켭니다.

**참고:** MQMF\_SEGMENTATION\_ALLOWED는 사용하지만 MQPMO\_LOGICAL\_ORDER는 사용하지 않고 메시지를 넣는 경우 주의해야 합니다. 메시지가

- 세그먼트가 아니며
- 그룹에 없으며
- 전달되지 않는 경우

큐 관리자가 각 메시지에 대해 고유 그룹 ID를 생성할 수 있도록 각 MQPUT 또는 MQPUT1 호출 앞에서 *GroupId* 필드를 MQGI\_NONE으로 재설정해야 합니다. 이 작업이 수행되지 않으면 관련이 없는 메시지가 동일한 그룹 ID를 가질 수 있으며 이로 인해 연속적으로 잘못된 처리가 발생할 수 있습니다. *GroupId* 필드를 재설정하는 시기에 대한 자세한 정보는 *GroupId* 필드 및 MQPMO\_LOGICAL\_ORDER 옵션에 대한 설명을 참조하십시오.

큐 관리자는 세그먼트 및 필수 헤더 데이터가 큐에 맞도록 필요에 따라 메시지를 세그먼트로 분할합니다. 하지만 큐 관리자가 생성하는 세그먼트의 크기에 대한 하한 값이 있으며 메시지에서 작성된 마지막 세그먼트만 이 한계보다 작을 수 있습니다. 애플리케이션이 생성하는 세그먼트의 크기에 대한 하한 값은 1바이트입니다. 큐 관리자가 생성하는 세그먼트는 길이가 동일하지 않을 수 있습니다. 큐 관리자는 다음과 같이 메시지를 처리합니다.

- 사용자 정의 형식은 16바이트의 배수인 경계에서 분할됩니다. 큐 관리자는 마지막 세그먼트를 제외하고는 16바이트보다 작은 세그먼트를 생성하지 않습니다.
- MQFMT\_STRING 외의 내장 형식은 제공된 데이터의 네이처에 적절한 위치에서 분할됩니다. 그러나 큐 관리자는 IBM MQ 헤더 구조의 중간에서 메시지를 분할하지 않습니다. 즉, 단일 MQ 헤더 구조를 포함하는 세그먼트는 큐 관리자에 의해 추가적으로 분할될 수 없으며 결과적으로 해당 메시지에 대해 가능한 최소 크기는 16바이트보다 큼니다.

큐 관리자가 생성하는 두 번째 또는 그 이후의 세그먼트는 다음 중 하나로 시작됩니다.

- MQ 헤더 구조
- 애플리케이션 메시지 데이터의 시작 부분
- 애플리케이션 메시지 데이터를 통과하는 부분

- MQFMT\_STRING은 제공된 데이터의 네이처(SBCS, DBCS 또는 혼합 SBCS/DBCS)에 상관없이 분할됩니다. 문자열이 DBCS 또는 혼합 SBCS/DBCS인 경우, 이로 인해 한 문자 세트에서 다른 문자 세트로 변환될 수 없는 세그먼트로 분할될 수 있습니다. 큐 관리자는 MQFMT\_STRING 메시지를 마지막 세그먼트를 제외하고는 16바이트보다 작은 세그먼트로 분할하지 않습니다.
- 큐 관리자는 세그먼트의 시작에서 제공된 데이터를 올바르게 설명하기 위해 각 세그먼트의 MQMD에서 *Format, CodedCharSetId, Encoding* 필드를 설정합니다. 형식 이름은 내장 형식의 이름 또는 사용자 정의된 형식의 이름입니다.
- 0보다 큰 *Offset*이 있는 세그먼트의 MQMD에 있는 *Report* 필드는 수정됩니다. 각 보고서 유형의 경우 보고서 옵션이 MQRO\_\*\_WITH\_DATA이지만 세그먼트가 처음 100바이트의 사용자 데이터를 포함할 수 없는 경우(즉, 존재할 수 있는 IBM MQ 헤더 구조 다음에 오는 데이터) 보고서 옵션은 MQRO\_\*로 변경됩니다.

큐 관리자는 위의 규칙을 준수하지만 그렇지 않은 경우 메시지를 예상치 못하게 분할합니다. 메시지가 분할되는 위치에 대해 가정 하지 마십시오.

지속 메시지의 경우 큐 관리자가 작업 단위 내에서만 세그먼트화를 수행할 수 있습니다.

- MQPUT 또는 MQPUT1 호출이 사용자 정의 작업 단위 내에서 작동 중인 경우 해당 작업 단위가 사용된다. 세그먼트화 프로세스 중에 호출이 실패하는 경우 호출 실패의 결과로 큐 관리자가 큐에 배치된 세그먼트를 제거합니다. 그러나 이 실패와 상관없이 작업 단위는 성공적으로 커밋됩니다.
- 호출이 사용자 정의 작업 단위 외부에서 작동 중이고 사용자 정의 작업 단위가 없는 경우 큐 관리자는 호출이 지속되는 동안 작업 단위를 작성합니다. 호출이 성공하면 큐 관리자가 자동으로 작업 단위를 커밋합니다. 호출이 실패하면 큐 관리자가 작업 단위를 백아웃합니다.
- 호출이 사용자 정의 작업 단위 외부에서 작동 중이지만 사용자 정의 작업 단위가 있는 경우 큐 관리자는 세그먼트화를 수행할 수 없습니다. 메시지에서 세그먼트화가 필요하지 않은 경우, 호출은 여전히 성공할 수 있습니다. 그러나 메시지를 세그먼트화해야 하는 경우 이유 코드 MQRC\_UOW\_NOT\_AVAILABLE과 함께 호출이 실패합니다.

비지속 메시지의 경우 큐 관리자는 세그먼트화를 수행하기 위해 작업 단위를 사용할 필요가 없습니다.

세그먼트화될 수 있는 메시지에서 데이터를 변환할 때 주의하십시오.

- 수신 애플리케이션이 MQGET 호출에서 데이터를 변환하고 MQGMO\_COMPLETE\_MSG 옵션을 지정하는 경우 데이터 변환 엑시트가 변환할 엑시트에 대한 완료 메시지에 전달되고 메시지가 세그먼트화되었다는 사실이 엑시트에 명확해집니다.
- 수신 애플리케이션이 한 번에 하나의 세그먼트를 검색하면 데이터 변환 엑시트가 호출되어 한 번에 하나의 세그먼트를 변환합니다. 그러므로 엑시트가 세그먼트의 데이터를 기타 세그먼트의 데이터와 별도로 변환해야 합니다.

메시지 내의 데이터의 네이처가 16바이트 경계에서 데이터의 임의 세그먼트화이면 엑시트에 의해 변환될 수 없는 세그먼트이거나 형식이 MQFMT\_STRING이며 문자 세트가 DBCS 또는 혼합 SBCS/DBCS이므로 송신 애플리케이션이 추가 세그먼트화를 억제하기 위해 MQMF\_SEGMENTATION\_INHIBITED를 지정한 상태에서 세그먼트를 작성하고 넣어야 합니다. 이런 방식으로, 송신 애플리케이션은 데이터-변환 엑시트가 성공적으로 세그먼트를 변환할 수 있도록 각 세그먼트에 충분한 정보가 포함되는지 확인할 수 있습니다.

- 송신자 변환이 전송 메시지 채널 에이전트(MCA)에 대해 지정된 경우 MCA는 논리 메시지의 세그먼트가 아닌 메시지만 변환합니다. MCA는 세그먼트인 메시지를 변환하려고 시도하지 않습니다.

이 플래그는 MQPUT 및 MQPUT1 호출에서는 입력 플래그이며 MQGET 호출에서는 출력 플래그입니다. 또한 후자의 호출인 경우 큐 관리자가 플래그의 값을 MQGMO의 *Segmentation* 필드로 화면표시합니다.

이 플래그의 초기값은 MQMF\_SEGMENTATION\_INHIBITED입니다.

**상태 플래그:** 이는 실제 메시지가 메시지 그룹에 속하는지, 논리 메시지의 세그먼트인지, 둘 다이거나 둘 다 아닌지 여부를 나타내는 플래그입니다. 다음 중 하나 이상이 MQPUT 또는 MQPUT1 호출 시 지정되거나 MQGET 호출에서 리턴될 수 있습니다.

#### **MQMF\_MSG\_IN\_GROUP**

메시지가 그룹의 멤버입니다.

#### **MQMF\_LAST\_MSG\_IN\_GROUP**

메시지가 그룹의 마지막 논리적 메시지입니다.

이 플래그가 설정되면 큐 관리자가 메시지와 함께 전송된 MQMD의 사본에서 MQMF\_MSG\_IN\_GROUP을 커지만 MQPUT 또는 MQPUT1 호출에서 애플리케이션이 제공하는 MQMD에서 이러한 플래그의 설정이 변경되지는 않습니다.

이는 그룹이 하나의 논리 메시지로만 구성되는 데 유효합니다. 이런 경우에 MQMF\_LAST\_MSG\_IN\_GROUP이 설정되지만 *MsgSeqNumber* 필드는 값 1을 가집니다.

### MQMF\_SEGMENT

메시지가 논리적 메시지의 세그먼트입니다.

MQMF\_SEGMENT가 MQMF\_LAST\_SEGMENT없이 지정되면 세그먼트에 있는 애플리케이션 메시지 데이터의 길이(존재할 수 있는 IBM MQ 헤더 구조의 길이 제외)는 1이상이어야 합니다. 길이가 0인 경우 MQPUT 또는 MQPUT1 호출이 이유 코드 MQRC\_SEGMENT\_LENGTH\_ZERO와 함께 실패합니다.

z/OS에서 색인 유형이 MQIT\_GROUP\_ID인 큐에 메시지를 넣는 경우 이 옵션이 지원되지 않습니다.

### MQMF\_LAST\_SEGMENT

메시지가 논리적 메시지의 마지막 세그먼트입니다.

이 플래그가 설정되면 큐 관리자가 메시지와 함께 전송된 MQMD의 사본에서 MQMF\_SEGMENT를 커지만 MQPUT 또는 MQPUT1 호출에서 애플리케이션이 제공하는 MQMD에서 이러한 플래그의 설정이 변경되지는 않습니다.

논리 메시지는 단 하나의 세그먼트로 구성될 수 있습니다. 그런 경우 MQMF\_LAST\_SEGMENT가 설정되지만 *Offset* 필드의 값은 0이 됩니다.

MQMF\_LAST\_SEGMENT가 지정되면 세그먼트의 애플리케이션 메시지 데이터 길이(존재할 수 있는 헤더 구조의 길이 제외)는 0일 수 있습니다.

z/OS에서 색인 유형이 MQIT\_GROUP\_ID인 큐에 메시지를 넣는 경우 이 옵션이 지원되지 않습니다.

메시지를 넣을 때 애플리케이션이 이러한 플래그가 올바르게 설정되었는지 확인해야 합니다.

MQPMO\_LOGICAL\_ORDER가 지정되거나 큐 핸들에 대해 선행 MQPUT 호출에서 지정된 경우 플래그의 설정이 큐 핸들에 대해 큐 관리자가 보유한 그룹 및 세그먼트 정보와 일치해야 합니다. 다음 조건은 MQPMO\_LOGICAL\_ORDER가 지정된 경우 큐 핸들에 대해 연속 MQPUT 호출에 적용됩니다.

- 현재 그룹 또는 논리 메시지가 없는 경우 모든 해당 플래그(및 그 조합)가 유효합니다.
- MQMF\_MSG\_IN\_GROUP이 지정되면 MQMF\_LAST\_MSG\_IN\_GROUP이 지정될 때까지 남아 있어야 합니다. 이 조건이 충족되지 않으면 이유 코드 MQRC\_INCOMPLETE\_GROUP으로 호출이 실패합니다.
- MQMF\_SEGMENT가 지정되면 MQMF\_LAST\_SEGMENT가 지정될 때까지 남아 있어야 합니다. 이 조건이 충족되지 않으면 이유 코드 MQRC\_INCOMPLETE\_MSG와 함께 호출이 실패합니다.
- MQMF\_SEGMENT가 MQMF\_MSG\_IN\_GROUP 없이 지정되면 MQMF\_LAST\_SEGMENT가 지정된 후까지 MQMF\_MSG\_IN\_GROUP이 꺼짐 상태여야 합니다. 이 조건이 충족되지 않으면 이유 코드 MQRC\_INCOMPLETE\_MSG와 함께 호출이 실패합니다.

큐의 물리적 순서에서는 플래그의 올바른 조합 및 다양한 필드에 사용되는 값을 표시합니다.

이러한 플래그는 MQPUT 및 MQPUT1 호출의 입력 플래그이며 MQGET 호출의 출력 플래그입니다. 후자의 호출에서 큐 관리자가 플래그의 값을 MQGMO의 *GroupStatus* 및 *SegmentStatus* 필드로 화면표시합니다.

발행/구독과 함께 그룹화되거나 세그먼트화된 메시지를 사용할 수 없습니다.

**기본 플래그:** 다음을 지정하여 메시지에 기본 속성이 있는지 나타낼 수 있습니다.

### MQMF\_NONE

메시지 플래그가 없습니다(기본 메시지 속성).

이는 세그먼트화를 방지하며, 메시지가 그룹에 없고 논리적 메시지의 세그먼트가 아님을 표시합니다.

MQMF\_NONE은 프로그램 문서화를 지원하기 위해 정의됩니다. 이 플래그는 다른 플래그와 함께 사용하도록 의도되지 않았지만, 해당 값이 0이므로 그와 같은 사용을 감지할 수 없습니다.

*MsgFlags* 필드는 하위 필드로 파티션화됩니다. 세부사항은 865 페이지의 『보고 옵션 및 메시지 플래그』의 내용을 참조하십시오.

이 필드의 초기값은 MQMF\_NONE입니다. *Version*이(가) MQMD\_VERSION\_2보다 작으면 이 필드는 무시됩니다.



## MQMD의 경우 *OriginalLength* (MQLONG)

이 필드는 세그먼트인 보고 메시지에만 관련됩니다. 또한 보고 메시지와 관련된 메시지 세그먼트의 길이를 지정하며 세그먼트가 파트를 구성하는 논리적 메시지의 길이 또는 보고 메시지의 데이터의 길이를 지정하지 않습니다.

**참고:** 세그먼트인 메시지에 대한 보고 메시지를 생성할 때 큐 관리자 및 메시지 채널 에이전트는 원래 메시지의 *GroupId*, *MsgSeqNumber*, *Offset* 및 *MsgFlags* 필드를 보고 메시지의 MQMD에 복사합니다. 결과적으로 보고 메시지 역시 세그먼트입니다. 보고 메시지를 생성하는 애플리케이션은 동일하게 수행해야 하며 *OriginalLength* 필드를 올바르게 설정해야 합니다.

다음 특수 값이 정의됩니다.

### MQOL\_UNDEFINED

메시지의 원래 길이가 정의되지 않습니다.

*OriginalLength*은(는) MQPUT 및 MQPUT1 호출의 입력 필드이지만 애플리케이션이 제공하는 값은 특정 상황에서만 허용됩니다.

- 넣는 메시지가 세그먼트이며 또한 보고 메시지인 경우, 큐 관리자가 지정된 값을 승인합니다. 값은 다음과 같아야 합니다.
  - 세그먼트가 마지막 세그먼트가 아닌 경우 0보다 큼
  - 세그먼트가 마지막 세그먼트인 경우 0 이상이어야 함
  - 메시지 내에 있는 데이터의 길이 이상이어야 함이러한 조건이 충족되지 않으면 이유 코드 MQRC\_ORIGINAL\_LENGTH\_ERROR와 함께 호출이 실패합니다.
- 넣는 메시지가 세그먼트이지만 보고 메시지가 아닌 경우, 큐 관리자는 필드를 무시하고 대신 애플리케이션 메시지 데이터의 길이를 사용합니다.
- 기타 모든 경우에서 큐 관리자가 필드를 무시하고 대신 MQOL\_UNDEFINED를 사용합니다.

이는 MQGET 호출의 출력 필드입니다.

이 필드의 초기값은 MQOL\_UNDEFINED입니다. *Version*이(가) MQMD\_VERSION\_2보다 작으면 이 필드는 무시됩니다.

## MQMDE - 메시지 디스크립터 확장

MQMDE 구조는 애플리케이션 메시지 데이터에 선행하여 가끔 발생하는 데이터를 설명합니다. 구조는 버전-1 MQMD가 아닌 버전-2 MQMD에 있는 MQMD 필드를 포함합니다.

### 가용성

모든 IBM MQ 시스템 및 해당 시스템에 연결된 IBM MQ MQI clients

### 형식 이름

MQFMT\_MD\_EXTENSION

### 문자 세트 및 인코딩

MQMDE의 데이터는 로컬 큐 관리자의 문자 세트 및 인코딩에 있어야 합니다. 이는 C 프로그래밍 언어의 경우 **CodedCharSetId** 큐 관리자 속성 및 MQENC\_NATIVE에서 제공됩니다.

MQMDE의 문자 세트 및 인코딩을 다음을 사용하여 *CodedCharSetId* 및 *Encoding* 필드로 설정하십시오.

- MQMD(MQMDE 구조가 메시지 데이터의 시작에 있는 경우) 또는
- MQMDE 구조에 선행하는 헤더 구조(다른 모든 경우).

MQMDE가 큐 관리자의 문자 세트 및 인코딩에 없는 경우 MQMDE는 승인되지만 적용되지 않습니다 즉, MQMDE는 메시지 데이터로 처리됩니다.

**참고:** Windows에서 Micro Focus COBOL과 함께 컴파일된 애플리케이션은 큐 관리자의 인코딩과 다른 MQENC\_NATIVE의 값을 사용합니다. MQPUT, MQPUT1 및 MQGET 호출 시 MQMD 구조의 숫자 필드가 Micro Focus COBOL 인코딩에 있어야 하지만 MQMDE 구조의 숫자 필드는 큐 관리자의 인코딩에 있어야 합니다. 이 후자는 C 프로그래밍 언어의 경우 MQENC\_NATIVE에서 지정하며 값 546을 가집니다.

## 사용법

version-2 MQMD를 사용하는 애플리케이션은 MQMDE 구조를 만나지 않습니다. 그러나 특수화된 애플리케이션 및 버전-1 MQMD를 계속 사용하는 애플리케이션의 경우 일부 상황에서 MQMDE가 발생할 수 있습니다. MQMDE 구조는 다음 환경에서 발생할 수 있습니다.

- MQPUT 및 MQPUT1 호출에 지정됨
- MQGET 호출로 리턴됨
- 전송 큐의 메시지에서

## MQPUT 및 MQPUT1 호출에 지정된 MQMDE

MQPUT 및 MQPUT1 호출에서 MQPUT 및 MQPUT1 호출에서 애플리케이션이 버전-1 MQMD를 제공하는 경우 MQMDE가 있음을 표시하기 위해 MQMD의 *Format* 필드를 MQFMT\_MD\_EXTENSION으로 설정하여 애플리케이션이 선택적으로 메시지 데이터에 MQMDE를 접두부로 설정할 수 있습니다. 애플리케이션이 MQMDE를 제공하지 않는 경우 큐 관리자는 MQMDE의 필드에 대해 기본값 가정합니다. 큐 관리자가 사용하는 기본값은 구조의 초기값과 동일합니다. [460 페이지의 표 503](#)의 내용을 참조하십시오.

애플리케이션이 version-2 MQMD 를 제공하고 가 애플리케이션 메시지 데이터에 MQMDE를 접두부로 추가하는 경우 구조는 다음 표에 표시된 대로 처리됩니다.

표 502. MQMDE가 MQMDE의 MQPUT 또는 MQPUT1에서 지정된 경우 큐 관리자 조치			
MQMD 버전	버전-2 필드의 값	MQMDE의 해당하는 필드의 값	큐 관리자에 의해 수행되는 조치
1	-	올바름	MQMDE가 적용됨
2	기본값	올바름	MQMDE가 적용됨
2	기본값이 아님	올바름	MQMDE가 메시지 데이터로 처리됨
1 또는 2	임의	올바르지 않음	호출이 적절한 이유 코드로 실패함
1 또는 2	임의	MQMDE가 잘못된 문자 세트 또는 인코딩이거나 지원되지 않는 버전임	MQMDE가 메시지 데이터로 처리됨

**참고:** z/OS에서 애플리케이션이 MQMDE와 함께 버전-1 MQMD를 지정하면 큐에 MQIT\_GROUP\_ID의 *IndexType*이 있는 경우에만 큐 관리자가 MQMDE를 유효성 검증합니다.

특별한 경우가 하나 있습니다. 애플리케이션이 세그먼트인 메시지를 넣기 위해 버전-2 MQMD를 사용하고(즉, MQMF\_SEGMENT 또는 MQMF\_LAST\_SEGMENT 플래그가 설정됨) MQMD의 형식 이름이 MQFMT\_DEAD\_LETTER\_HEADER인 경우 큐 관리자는 MQMDE 구조를 생성하여 이를 다음에 오는 데이터 및 MQDLH 구조 사이에 삽입합니다. 큐 관리자가 메시지로 유지하는 MQMD에서 버전-2 필드는 해당 기본값으로 설정됩니다.

버전 2 MQMD에 존재하지만 버전 1 MQMD에 존재하지 않는 몇몇 필드가 MQPUT 및 MQPUT1 호출에서 입/출력(I/O) 필드로 사용됩니다. 그러나 큐 관리자는 MQPUT 및 MQPUT1 호출의 출력에서 MQMDE의 동일 필드에는 아무 값도 리턴하지 않습니다. 애플리케이션은 이들 출력 값이 필요한 경우 버전-2 MQMD를 사용해야 합니다.

## MQGET 호출로 리턴된 MQMDE

MQGET 호출에서 애플리케이션이 버전-1 MQMD를 제공하는 경우 큐 관리자는 리턴된 메시지에 MQMDE로 접두부를 붙입니다. 단, MQMDE의 필드 중 하나 이상이 기본값이 아닌 값을 가진 경우에만 이를 수행합니다. 큐 관

리자는 MQMDE가 있음을 표시하기 위해 MQMD의 *Format* 필드를 값 MQFMT\_MD\_EXTENSION으로 설정합니다.

애플리케이션이 **Buffer** 매개변수의 시작에서 MQMDE를 제공하는 경우 MQMDE가 무시됩니다. MQGET 호출에서 리턴 시 메시지의 MQMDE에 의해 대체되거나(하나가 필요한 경우) 애플리케이션 메시지 데이터에 의해 덮어씁니다(MQMDE가 필요하지 않은 경우).

MQGET 호출이 MQMDE를 리턴하는 경우 MQMDE의 데이터는 일반적으로 큐 관리자의 문자 세트 및 인코딩에 있습니다. 그러나 다음의 경우 MQMDE는 다른 문자 세트 및 인코딩에 있을 수 있습니다.

- MQMDE는 MQPUT 또는 MQPUT1 호출의 데이터로 처리되었습니다(이를 발생시킬 수 있는 환경은 [459 페이지의 표 502](#) 참조).
- 메시지가 TCP 연결로 연결된 리모트 큐 관리자에서 수신되었고 수신 메시지 채널 에이전트(MCA)가 올바르게 설정되지 않았습니다.

**참고:** Windows에서 Micro Focus COBOL과 함께 컴파일된 애플리케이션은 큐 관리자의 인코딩과 다른 MQENC\_NATIVE의 값을 사용합니다(위의 항목 참조).

## 전송 큐 메시지의 MQMDE

전송 큐의 메시지는 MQXQH 구조 앞에 붙이며 버전-1 MQMD 내에 포함합니다. MQMDE도 존재할 수 있으며 MQXQH 구조와 애플리케이션 메시지 데이터 사이에 위치할 수 있지만, 일반적으로 MQMDE에 있는 하나 이상의 필드에 기본값이 아닌 값이 있는 경우에만 존재합니다.

다른 MQ 헤더 구조는 MQXQH 구조 및 애플리케이션 메시지 데이터 사이에서도 발생할 수 있습니다. 예를 들어, 데드 레터 헤더 MQDLH가 존재하면 메시지는 세그먼트가 아닙니다. 순서는 다음과 같습니다.

- MQXQH(버전-1 MQMD 포함)
- MQMDE
- MQDLH
- 애플리케이션 메시지 데이터

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 503. MQMDE에 대한 MQMDE의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQMDE_STRUC_ID	'MDE-'
버전 (구조 버전 번호)	MQMDE_VERSION_2	2
StrucLength (MQMDE 구조의 길이)	MQMDE_LENGTH_2	72
인코딩 (MQMDE뒤에 오는 데이터의 숫자 인코딩)	MQENC_NATIVE	환경에 따라 다름
CodedCharSetId (MQMDE뒤에 오는 데이터의 문자 세트 ID)	MQCCSI_UNDEFINED	0
형식 (MQMDE뒤에 오는 데이터의 형식 이름)	MQFMT_NONE	공백
플래그 (일반 플래그)	MQMDEF_NONE	0
GroupId (그룹 ID)	MQGI_NONE	널
MsgSeqNumber (그룹 내 논리 메시지의 순서 번호)	없음	1
오프셋 (논리 메시지의 시작부터 실제 메시지의 데이터 오프셋)	없음	0
MsgFlags (메시지 플래그)	MQMF_NONE	0



표 503. MQMDE에 대한 MQMDE의 필드 (계속)

필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
OriginalLength (원래 메시지의 길이)	MQOL_UNDEFINED	-1

**참고:**

1. ~ 기호는 단일 공백 문자를 나타냅니다.
2. C 프로그래밍 언어의 매크로 변수MQMDE\_DEFAULT는 테이블에 나열되는 값을 포함합니다. 즉, 구조 내의 필드에 대한 초기값을 제공하기 위해 다음과 같은 방법으로 사용될 수 있습니다.

```
MQMDE MyMDE = {MQMDE_DEFAULT};
```

## 언어 선언

### MQMDE의 C 선언

```
typedef struct tagMQMDE MQMDE;
struct tagMQMDE {
    MQCHAR4   StruId;           /* Structure identifier */
    MQLONG    Version;         /* Structure version number */
    MQLONG    StruLength;      /* Length of MQMDE structure */
    MQLONG    Encoding;        /* Numeric encoding of data that follows
                               MQMDE */
    MQLONG    CodedCharSetId; /* Character-set identifier of data that
                               follows MQMDE */
    MQCHAR8   Format;          /* Format name of data that follows
                               MQMDE */
    MQLONG    Flags;           /* General flags */
    MQBYTE24  GroupId;         /* Group identifier */
    MQLONG    MsgSeqNumber;    /* Sequence number of logical message
                               within group */
    MQLONG    Offset;          /* Offset of data in physical message from
                               start of logical message */
    MQLONG    MsgFlags;        /* Message flags */
    MQLONG    OriginalLength; /* Length of original message */
};
```

### MQMDE의 COBOL 선언

```
** MQMDE structure
10 MQMDE.
** Structure identifier
15 MQMDE-STRUCID PIC X(4).
** Structure version number
15 MQMDE-VERSION PIC S9(9) BINARY.
** Length of MQMDE structure
15 MQMDE-STRUCLNGTH PIC S9(9) BINARY.
** Numeric encoding of data that follows MQMDE
15 MQMDE-ENCODING PIC S9(9) BINARY.
** Character-set identifier of data that follows MQMDE
15 MQMDE-CODEDCHARSETID PIC S9(9) BINARY.
** Format name of data that follows MQMDE
15 MQMDE-FORMAT PIC X(8).
** General flags
15 MQMDE-FLAGS PIC S9(9) BINARY.
** Group identifier
15 MQMDE-GROUPID PIC X(24).
** Sequence number of logical message within group
15 MQMDE-MSGSEQNUMBER PIC S9(9) BINARY.
** Offset of data in physical message from start of logical message
15 MQMDE-OFFSET PIC S9(9) BINARY.
** Message flags
15 MQMDE-MSGFLAGS PIC S9(9) BINARY.
** Length of original message
15 MQMDE-ORIGINALLENGTH PIC S9(9) BINARY.
```

## MQMDE의 PL/I 선언

```
dc1
1 MQMDE based,
3 StrucId          char(4),          /* Structure identifier */
3 Version          fixed bin(31),   /* Structure version number */
3 StrucLength      fixed bin(31),   /* Length of MQMDE structure */
3 Encoding         fixed bin(31),   /* Numeric encoding of data that
                                     follows MQMDE */
3 CodedCharSetId  fixed bin(31),   /* Character-set identifier of data
                                     that follows MQMDE */
3 Format           char(8),          /* Format name of data that follows
                                     MQMDE */
3 Flags           fixed bin(31),   /* General flags */
3 GroupId         char(24),         /* Group identifier */
3 MsgSeqNumber    fixed bin(31),   /* Sequence number of logical message
                                     within group */
3 Offset          fixed bin(31),   /* Offset of data in physical message
                                     from start of logical message */
3 MsgFlags        fixed bin(31),   /* Message flags */
3 OriginalLength  fixed bin(31); /* Length of original message */
```

## MQMDE의 상위 레벨 어셈블러 선언

```
MQMDE          DSECT
MQMDE_STRUCID  DS   CL4   Structure identifier
MQMDE_VERSION  DS   F     Structure version number
MQMDE_STRUCLNGTH DS  F     Length of MQMDE structure
MQMDE_ENCODING DS  F     Numeric encoding of data that follows
*              MQMDE
MQMDE_CODEDCHARSETID DS  F   Character-set identifier of data that
*              follows MQMDE
MQMDE_FORMAT   DS   CL8   Format name of data that follows MQMDE
MQMDE_FLAGS    DS   F     General flags
MQMDE_GROUPID  DS   XL24  Group identifier
MQMDE_MSGSEQNUMBER DS  F   Sequence number of logical message
*              within group
MQMDE_OFFSET   DS   F     Offset of data in physical message from
*              start of logical message
MQMDE_MSGFLAGS DS   F     Message flags
MQMDE_ORIGINALLENGTH DS  F   Length of original message
*
MQMDE_LENGTH   EQU  *-MQMDE
               ORG  MQMDE
MQMDE_AREA     DS   CL(MQMDE_LENGTH)
```

## MQMDE의 Visual Basic 선언

```
Type MQMDE
StrucId          As String*4 'Structure identifier'
Version          As Long     'Structure version number'
StrucLength      As Long     'Length of MQMDE structure'
Encoding         As Long     'Numeric encoding of data that follows'
                  'MQMDE'
CodedCharSetId  As Long     'Character-set identifier of data that'
                  'follows MQMDE'
Format          As String*8 'Format name of data that follows MQMDE'
Flags           As Long     'General flags'
GroupId         As MQBYTE24 'Group identifier'
MsgSeqNumber    As Long     'Sequence number of logical message within'
                  'group'
Offset          As Long     'Offset of data in physical message from'
                  'start of logical message'
MsgFlags        As Long     'Message flags'
OriginalLength  As Long     'Length of original message'
End Type
```

### MQMDE의 경우 StrucId (MQCHAR4)

메시지 디스크립터 확장 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQMDE\_STRUC\_ID입니다. 값은 다음과 같아야 합니다.

## MQMDE\_STRUC\_ID

메시지 디스크립터 확장 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQMDE\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQMDE\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

## MQMDE용 버전 (MQLONG)

구조 버전 번호이며 값은 다음과 같아야 합니다.

### MQMDE\_VERSION\_2

버전2 메시지 디스크립터 확장자 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

### MQMDE\_CURRENT\_VERSION

메시지 디스크립터 확장자 구조의 현재 버전.

이 필드의 초기값은 MQMDE\_VERSION\_2입니다.

## MQMDE의 경우 StrucLength (MQLONG)

이는 MQMDE 구조의 길이입니다. 다음 값이 정의됩니다.

### MQMDE\_LENGTH\_2

버전-2 메시지 디스크립터 확장자 구조의 길이.

이 필드의 초기값은 MQMDE\_LENGTH\_2입니다.

## MQMDE에 대한 인코딩 (MQLONG)

MQMDE 구조 뒤에 오는 데이터의 숫자 인코딩을 지정합니다. MQMDE 구조 자체의 숫자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 큐 관리자는 이 필드가 올바른지 검사하지 않습니다. 데이터 인코딩에 대한 자세한 정보는 [412 페이지의 『MQMD - 메시지 디스크립터』](#)에서 설명하는 *Encoding* 필드를 참조하십시오.

이 필드의 초기값은 MQENC\_NATIVE입니다.

## MQMDE의 경우 CodedCharSetId (MQLONG)

MQMDE 구조 뒤에 오는 데이터의 문자 세트 ID를 지정합니다. MQMDE 구조 자체의 문자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 큐 관리자는 이 필드가 올바른지 검사하지 않습니다. 다음 특수 값을 사용할 수 있습니다.

### MQCCSI\_INHERIT

이 구조 다음에 오는 데이터의 문자 데이터는 이 구조와 동일한 문자 세트로 되어 있습니다.

큐 관리자가 구조의 실제 문자 세트 ID에 메시지로 송신한 구조에서 이 값을 변경합니다. 오류가 발생하지 않으면 MQGET 호출에서 MQCCSI\_INHERIT 값을 리턴하지 않습니다.

MQMD의 *PutApplType* 필드 값이 MQAT\_BROKER인 경우 MQCCSI\_INHERIT를 사용할 수 없습니다.

이 값은 다음 환경에서 지원됩니다.

-  AIX
-  IBM i
-  Linux
-  Windows

및 해당 시스템에 연결된 IBM MQ 클라이언트

이 필드의 초기값은 MQCCSI\_UNDEFINED입니다.

## **MQMDE의 형식 (MQCHAR8)**

MQMDE 구조 뒤에 오는 데이터의 형식 이름을 지정합니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 큐 관리자는 이 필드가 올바른지 검사하지 않습니다. 형식명에 대한 자세한 정보는 412 페이지의 『MQMD - 메시지 디스크립터』에 설명된 *Format* 필드를 참조하십시오.

이 필드의 초기값은 MQFMT\_NONE입니다.

## **MQMDE에 대한 플래그 (MQLONG)**

다음 플래그가 지정될 수 있습니다.

### **MQMDEF\_NONE**

플래그가 없습니다.

이 필드의 초기값은 MQMDEF\_NONE입니다.

## **MQMDE의 경우 GroupId (MQBYTE24)**

412 페이지의 『MQMD - 메시지 디스크립터』에 설명된 *GroupId* 필드를 참조하십시오. 이 필드의 초기값은 MQGI\_NONE입니다.

## **MsgSeqMQMDE의 번호 (MQLONG)**

412 페이지의 『MQMD - 메시지 디스크립터』에 설명된 *MsgSeqNumber* 필드를 참조하십시오. 이 필드의 초기값은 1입니다.

## **MQMDE의 오프셋 (MQLONG)**

412 페이지의 『MQMD - 메시지 디스크립터』에 설명된 *Offset* 필드를 참조하십시오. 이 필드의 초기값은 0입니다.

## **MQMDE의 경우 MsgFlags (MQLONG)**

412 페이지의 『MQMD - 메시지 디스크립터』에서 설명하는 *MsgFlags* 필드를 참조하십시오. 이 필드의 초기값은 MQMF\_NONE입니다.

## **MQMDE의 경우 OriginalLength (MQLONG)**

412 페이지의 『MQMD - 메시지 디스크립터』에 설명된 *OriginalLength* 필드를 참조하십시오. 이 필드의 초기값은 MQOL\_UNDEFINED입니다.

## **MQMHBO - 버퍼에 대한 메시지 핸들 옵션**

MQMHBO 구조에서는 애플리케이션이 메시지 핸들에서 버퍼가 생성되는 방법을 제어하는 옵션을 지정할 수 있습니다. 구조는 MQMHBUF 호출의 입력 매개변수입니다.

## **문자 세트 및 인코딩**

MQMHBO의 데이터는 애플리케이션의 문자 세트 및 애플리케이션의 인코딩 (MQENC\_NATIVE)에 있어야 합니다.

## **필드**

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 504. MQMHBO의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQMHBO_STRUC_ID	'MHBO'

표 504. MQMHBO의 필드 (계속)		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
버전 (구조 버전 번호)	MQMHBO_VERSION_1	1
옵션 (MQMHBUF의 조치를 제어하는 옵션)	MQMHBO_PROPERTIES_I N_MQRFH2	

**참고:**

- 널 문자열 값 또는 공백은 C의 널 문자열과 기타 프로그래밍 언어의 공백 문자를 나타냅니다.
- C 프로그래밍 언어의 매크로 변수 MQMHBO\_DEFAULT는 테이블에 나열되는 값을 포함합니다. 구조의 필드에 초기 값을 제공하기 위해 다음 방법으로 사용하십시오.

```
MQMHBO MyMHBO = {MQMHBO_DEFAULT};
```

## 언어 선언

### MQMHBO의 C 선언

```
typedef struct tagMQMHBO MQMHBO;
struct tagMQMHBO {
    MQCHAR4  StrucId;          /* Structure identifier */
    MQLONG   Version;        /* Structure version number */
    MQLONG   Options;        /* Options that control the action of
                             MQMHBUF */
};
```

### MQMHBO의 COBOL 선언

```
** MQMHBO structure
10 MQMHBO.
**   Structure identifier
15 MQMHBO-STRUCID          PIC X(4).
**   Structure version number
15 MQMHBO-VERSION        PIC S9(9) BINARY.
**   Options that control the action of MQMHBUF
15 MQMHBO-OPTIONS        PIC S9(9) BINARY.
```

### MQMHBO의 PL/I 선언

```
Dcl
1 MQMHBO based,
3 StrucId      char(4),      /* Structure identifier */
3 Version      fixed bin(31), /* Structure version number */
3 Options      fixed bin(31), /* Options that control the action
                             of MQMHBUF */
```

### MQMHBO의 상위 레벨 어셈블러 선언

```
MQMHBO          DSECT
MQMHBO_STRUCID  DS   CL4  Structure identifier
MQMHBO_VERSION  DS   F    Structure version number
MQMHBO_OPTIONS  DS   F    Options that control the
*                  action of MQMHBUF
MQMHBO_LENGTH   EQU   *-MQMHBO
MQMHBO_AREA     DS   CL(MQMHBO_LENGTH)
```

### MQMHBO의 경우 StrucId (MQCHAR4)

이는 버퍼 옵션 구조에 대한 메시지 핸들의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQMHBO\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

#### **MQMHBO\_STRUC\_ID**

버퍼에 대한 메시지 핸들 옵션 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQMHBO\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQMHBO\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

#### **MQMHBO의 버전 (MQLONG)**

버퍼에 대한 메시지 핸들 옵션 구조 - 버전 필드

구조 버전 번호입니다. 값은 다음과 같아야 합니다.

#### **MQMHBO\_VERSION\_1**

버퍼 옵션 구조에 대한 메시지 핸들의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### **MQMHBO\_CURRENT\_VERSION**

버퍼 옵션 구조에 대한 메시지 핸들의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQMHBO\_VERSION\_1입니다.

#### **MQMHBO에 대한 옵션 (MQLONG)**

버퍼에 대한 메시지 핸들 옵션 구조 - 옵션 필드

이러한 옵션은 MQMHBUF의 조치를 제어합니다.

다음 옵션을 지정해야 합니다.

#### **MQMHBO\_PROPERTIES\_IN\_MQRFH2**

메시지 핸들의 특성을 버퍼로 변환할 때 MQRFH2 형식으로 변환하십시오.

선택적으로 다음 옵션을 지정할 수도 있습니다. 둘 이상의 옵션을 지정하려면 값을 함께 추가하거나(동일한 상수를 두 번 이상 추가하지 않음), 비트 단위 OR 연산을 사용하여 값을 결합하십시오(프로그래밍 언어가 비트 연산을 지원하는 경우).

#### **MQMHBO\_DELETE\_PROPERTIES**

버퍼에 추가되는 특성은 메시지 핸들에서 삭제됩니다. 호출이 실패하면 어떤 특성도 삭제되지 않습니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQMHBO\_PROPERTIES\_IN\_MQRFH2입니다.

#### **MQOD - 오브젝트 디스크립터**

MQOD 구조는 이름별로 오브젝트를 지정하는 데 사용됩니다. 구조는 MQOPEN 및 MQPUT1 호출의 입/출력 매개변수입니다.

올바른 오브젝트의 유형은 다음과 같습니다.

- 큐 또는 분배 목록
- 이름 목록
- 프로세스 정의
- 큐 관리자
- 토픽

#### **가용성**

모든 IBM MQ 시스템 및 해당 시스템에 연결된 IBM MQ MQI clients .

#### **버전**

MQOD의 현재 버전은 MQOD\_VERSION\_4입니다. 여러 환경 간에 이식하려는 애플리케이션은 MQOD의 필수 버전이 관련된 모든 환경에서 지원되는지 확인해야 합니다. 최신 버전의 구조에만 있는 필드는 다음에 오는 설명에서 최신 필드로 식별됩니다.

지원되는 프로그래밍 언어에 제공되는 헤더, COPY, INCLUDE 파일은 MQOD\_VERSION\_1로 설정된, *Version* 필드의 초기값과 함께 환경이 지원하는 MQOD의 최신 버전을 포함합니다. *version-1* 구조에 없는 필드를 사용하면 애플리케이션이 *Version* 필드를 필요한 버전의 버전 번호로 설정해야 합니다.

분배 목록을 열려면 *Version*이 MQOD\_VERSION\_2 이상이어야 합니다.

## 문자 세트 및 인코딩

MQOD의 데이터는 MQENC\_NATIVE로 지정된 로컬 큐 관리자의 인코딩 및 **CodedCharSetId** 큐 관리자 속성으로 지정된 문자 세트에 있어야 합니다. 그러나 애플리케이션이 MQ MQI 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트와 인코딩으로 작성되어야 합니다.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQOD_STRUC_ID	'0D--'
버전 (구조 버전 번호)	MQOD_VERSION_1	1
ObjectType (오브젝트 유형)	MQOT_Q	1
ObjectName (오브젝트 이름)	없음	널 문자열 또는 공백
ObjectQMgr이름 (오브젝트 큐 관리자 이름)	없음	널 문자열 또는 공백
DynamicQName (동적 큐 이름)	없음	z/OS의 경우 'CSQ.*' 'AMQ.*' 그 외의 경우
AlternateUserID (대체 사용자 ID)	없음	널 문자열 또는 공백
<b>참고:</b> <i>Version</i> 이 MQOD_VERSION_2 미만인 경우 나머지 필드는 무시됩니다.		
RecsPresent (존재하는 오브젝트 레코드 수)	없음	0
KnownDest수 (성공적으로 열린 로컬 큐 수)	없음	0
UnknownDest수 (성공적으로 열린 리모트 큐 수)	없음	0
InvalidDest수 (열기에 실패한 큐 수)	없음	0
ObjectRec오프셋 (MQOD 시작부터 첫 번째 오브젝트 레코드의 오프셋)	없음	0
ResponseRec오프셋 (MQOD 시작 이후 첫 번째 응답 레코드의 오프셋)	없음	0
ObjectRecPtr (첫 번째 오브젝트 레코드의 주소)	없음	널 포인터 또는 널 바이트
ResponseRecPtr (첫 번째 응답 레코드의 주소)	없음	널 포인터 또는 널 바이트
<b>참고:</b> <i>Version</i> 이 MQOD_VERSION_3 미만인 경우 나머지 필드는 무시됩니다.		
AlternateSecurityID (대체 보안 ID)	MQSID_NONE	널
ResolvedQName (해석된 큐 이름)	없음	널 문자열 또는 공백
ResolvedQMgr이름 (해석된 큐 관리자 이름)	없음	널 문자열 또는 공백
<b>참고:</b> <i>Version</i> 이 MQOD_VERSION_4 미만인 경우 나머지 필드는 무시됩니다.		
ObjectString (긴 오브젝트 이름)	MQCHARV_DEFAULT	MQCHARV에 대해 정의된 대로

필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
SelectionString (선택 문자열)	MQCHARV_DEFAULT	MQCHARV에 대해 정의된 대로
ResObjectString (해석된 긴 오브젝트 이름)	MQCHARV_DEFAULT	MQCHARV에 대해 정의된 대로
ResolvedType (분석된 오브젝트 유형)	MQOT_NONE	0

**참고:**

- ~ 기호는 단일 공백 문자를 나타냅니다.
- 널 문자열 값 또는 공백은 C의 널 문자열과 기타 프로그래밍 언어의 공백 문자를 나타냅니다.
- C 프로그래밍 언어의 매크로 변수MQOD\_DEFAULT는 테이블에 나열되는 값을 포함합니다. 즉, 구조 내의 필드에 대한 초기값을 제공하기 위해 다음과 같은 방법으로 사용될 수 있습니다.

```
MQOD MyOD = {MQOD_DEFAULT};
```

## 언어 선언

### MQOD의 C 선언

```
typedef struct tagMQOD MQOD;
struct tagMQOD {
    MQCHAR4      StrucId;           /* Structure identifier */
    MQLONG       Version;          /* Structure version number */
    MQLONG       ObjectType;       /* Object type */
    MQCHAR48     ObjectName;       /* Object name */
    MQCHAR48     ObjectQMgrName;   /* Object queue manager name */
    MQCHAR48     DynamicQName;    /* Dynamic queue name */
    MQCHAR12     AlternateUserId;  /* Alternate user identifier */
    /* Ver:1 */
    MQLONG       RecsPresent;      /* Number of object records present */
    MQLONG       KnownDestCount;  /* Number of local queues opened
    successfully */
    MQLONG       UnknownDestCount; /* Number of remote queues opened
    successfully */
    MQLONG       InvalidDestCount; /* Number of queues that failed to
    open */
    MQLONG       ObjectRecOffset; /* Offset of first object record from
    start of MQOD */
    MQLONG       ResponseRecOffset; /* Offset of first response record
    from start of MQOD */
    MQPTR        ObjectRecPtr;    /* Address of first object record */
    MQPTR        ResponseRecPtr; /* Address of first response record */
    /* Ver:2 */
    MQBYTE40     AlternateSecurityId; /* Alternate security identifier */
    MQCHAR48     ResolvedQName;     /* Resolved queue name */
    MQCHAR48     ResolvedQMgrName; /* Resolved queue manager name */
    /* Ver:3 */
    MQCHARV      ObjectString;      /* Object Long name */
    MQCHARV      SelectionString;   /* Message Selector */
    MQCHARV      ResObjectString;  /* Resolved Long object name*/
    MQLONG       ResolvedType       /* Alias queue resolved
    object type */
    /* Ver:4 */
};
```

### MQOD의 COBOL 선언

```
** MQOD structure
10 MQOD.
** Structure identifier
15 MQOD-STRUCID          PIC X(4).
** Structure version number
15 MQOD-VERSION        PIC S9(9) BINARY.
** Object type
```



```

15 MQOD-OBJECTTYPE          PIC S9(9) BINARY.
** Object name
15 MQOD-OBJECTNAME         PIC X(48).
** Object queue manager name
15 MQOD-OBJECTQMGRNAME     PIC X(48).
** Dynamic queue name
15 MQOD-DYNAMICQNAME       PIC X(48).
** Alternate user identifier
15 MQOD-ALTERNATEUSERID    PIC X(12).
** Number of object records present
15 MQOD-RECSPRESENT        PIC S9(9) BINARY.
** Number of local queues opened successfully
15 MQOD-KNOWNDDESTCOUNT   PIC S9(9) BINARY.
** Number of remote queues opened successfully
15 MQOD-UNKNOWNDDESTCOUNT PIC S9(9) BINARY.
** Number of queues that failed to open
15 MQOD-INVALIDDESTCOUNT PIC S9(9) BINARY.
** Offset of first object record from start of MQOD
15 MQOD-OBJECTRECOFFSET    PIC S9(9) BINARY.
** Offset of first response record from start of MQOD
15 MQOD-RESPONSERECOFFSET  PIC S9(9) BINARY.
** Address of first object record
15 MQOD-OBJECTRECPTTR      POINTER.
** Address of first response record
15 MQOD-RESPONSERECPTTR    POINTER.
** Alternate security identifier
15 MQOD-ALTERNATESECURITYID PIC X(40).
** Resolved queue name
15 MQOD-RESOLVEDQNAME      PIC X(48).
** Resolved queue manager name
15 MQOD-RESOLVEDQMGRNAME   PIC X(48).
** Object Long name
15 MQOD-OBJECTSTRING.
** Address of variable length string
20 MQOD-OBJECTSTRING-VSPTR  POINTER.
** Offset of variable length string
20 MQOD-OBJECTSTRING-VSOFFSET PIC S9(9) BINARY.
** size of buffer
20 MQOD-OBJECTSTRING-VSBUFSIZE PIC S9(9) BINARY.
** Length of variable length string
20 MQOD-OBJECTSTRING-VSLENGTH PIC S9(9) BINARY.
** CCSID of variable length string
20 MQOD-OBJECTSTRING-VSCCSID PIC S9(9) BINARY.
** Message Selector
15 MQOD-SELECTIONSTRING.
** Address of variable length string
20 MQOD-SELECTIONSTRING-VSPTR  POINTER.
** Offset of variable length string
20 MQOD-SELECTIONSTRING-VSOFFSET PIC S9(9) BINARY.
** size of buffer
20 MQOD-SELECTIONSTRING-VSBUFSIZE PIC S9(9) BINARY.
** Length of variable length string
20 MQOD-SELECTIONSTRING-VSLENGTH PIC S9(9) BINARY.
** CCSID of variable length string
20 MQOD-SELECTIONSTRING-VSCCSID PIC S9(9) BINARY.
** Resolved Long object name
15 MQOD-RESOBJECTSTRING.
** Address of variable length string
20 MQOD-RESOBJECTSTRING-VSPTR  POINTER.
** Offset of variable length string
20 MQOD-RESOBJECTSTRING-VSOFFSET PIC S9(9) BINARY.
** size of buffer
20 MQOD-RESOBJECTSTRING-VSBUFSIZE PIC S9(9) BINARY.
** Length of variable length string
20 MQOD-RESOBJECTSTRING-VSLENGTH PIC S9(9) BINARY.
** CCSID of variable length string
20 MQOD-RESOBJECTSTRING-VSCCSID PIC S9(9) BINARY.
** Alias queue resolved object type
15 MQOD-RESOLVEDTYPE        PIC S9(9) BINARY.

```

## MQOD의 PL/I 선언

```

dcl
1 MQOD based,
3 StructId          char(4),          /* Structure identifier */
3 Version           fixed bin(31),    /* Structure version number */
3 ObjectType        fixed bin(31),    /* Object type */
3 ObjectName        char(48),         /* Object name */
3 ObjectQMgrName    char(48),         /* Object queue manager name */

```

```

3 DynamicQName      char(48),      /* Dynamic queue name */
3 AlternateUserId   char(12),     /* Alternate user identifier */
3 RecsPresent       fixed bin(31), /* Number of object records
present */
3 KnownDestCount    fixed bin(31), /* Number of local queues opened
successfully */
3 UnknownDestCount  fixed bin(31), /* Number of remote queues opened
successfully */
3 InvalidDestCount  fixed bin(31), /* Number of queues that failed to
open */
3 ObjectRecOffset   fixed bin(31), /* Offset of first object record
from start of MQOD */
3 ResponseRecOffset fixed bin(31), /* Offset of first response record
from start of MQOD */
3 ObjectRecPtr      pointer,      /* Address of first object record */
3 ResponseRecPtr    pointer,      /* Address of first response
record */
3 AlternateSecurityId char(40),     /* Alternate security identifier */
3 ResolvedQName     char(48),     /* Resolved queue name */
3 ResolvedQMgrName  char(48),     /* Resolved queue manager name */
3 ObjectString,     /* Object Long name */
5 VSPtr            pointer,      /* Address of variable length string */
5 VSOffset         fixed bin(31), /* Offset of variable length string */
5 VSBufSize        fixed bin(31), /* size of buffer */
5 VSLength         fixed bin(31), /* Length of variable length string */
5 VSCCSID          fixed bin(31), /* CCSID of variable length string */
3 SelectionString, /* Message Selection */
5 VSPtr            pointer,      /* Address of variable length string */
5 VSOffset         fixed bin(31), /* Offset of variable length string */
5 VSBufSize        fixed bin(31), /* size of buffer */
5 VSLength         fixed bin(31), /* Length of variable length string */
5 VSCCSID          fixed bin(31), /* CCSID of variable length string */
3 ResObjectString, /* Resolved Long object name */
5 VSPtr            pointer,      /* Address of variable length string */
5 VSOffset         fixed bin(31), /* Offset of variable length string */
5 VSBufSize        fixed bin(31), /* size of buffer */
5 VSLength         fixed bin(31), /* Length of variable length string */
5 VSCCSID          fixed bin(31), /* CCSID of variable length string */
3 ResolvedType     fixed bin(31); /* Alias queue resolved object type */

```

### MQOD의 상위 레벨 어셈블러 선언

```

MQOD                DSECT
MQOD_STRUCID        DS    CL4    Structure identifier
MQOD_VERSION        DS    F      Structure version number
MQOD_OBJECTTYPE     DS    F      Object type
MQOD_OBJECTNAME     DS    CL48   Object name
MQOD_OBJECTQMGRNAME DS    CL48   Object queue manager name
MQOD_DYNAMICQNAME   DS    CL48   Dynamic queue name
MQOD_ALTERNATEUSERID DS    CL12  Alternate user identifier
MQOD_RECSPRESENT    DS    F      Number of object records present
MQOD_KNOWNDESTCOUNT DS    F      Number of local queues opened
*                   successfully
MQOD_UNKNOWNDSTCOUNT DS    F      Number of remote queues opened
*                   successfully
MQOD_INVALIDDESTCOUNT DS    F      Number of queues that failed to
*                   open
MQOD_OBJECTRECOFFSET DS    F      Offset of first object record from
*                   start of MQOD
MQOD_RESPONSERECOFFSET DS    F      Offset of first response record
*                   from start of MQOD
MQOD_OBJECTRECPTTR  DS    F      Address of first object record
MQOD_RESPONSERECPTR DS    F      Address of first response record
MQOD_ALTERNATESECURITYID DS    XL40  Alternate security identifier
MQOD_RESOLVEDQNAME  DS    CL48   Resolved queue name
MQOD_RESOLVEDQMGRNAME DS    CL48   Resolved queue manager name
MQOD_OBJECTSTRING   DS    F      Object Long name
MQOD_OBJECTSTRING_VSPTR DS    F      Address of variable length string
MQOD_OBJECTSTRING_VSOFFSET DS    F      Offset of variable length string
MQOD_OBJECTSTRING_VSBUFSIZE DS    F      size of buffer
MQOD_OBJECTSTRING_VSLENGTH DS    F      Length of variable length string
MQOD_OBJECTSTRING_VSCCSID DS    F      CCSID of variable length string
MQOD_OBJECTSTRING_LENGTH EQU    *- MQOD_OBJECTSTRING
*                   ORG    MQOD_OBJECTSTRING
MQOD_OBJECTSTRING_AREA DS    CL(MQOD_OBJECTSTRING_LENGTH)
*
MQOD_SELECTIONSTRING DS    F      Message Selector
MQOD_SELECTIONSTRING_VSPTR DS    F      Address of variable length string
MQOD_SELECTIONSTRING_VSOFFSET DS    F      Offset of variable length string

```

```

MQOD_SELECTIONSTRING_VSBUFFSIZE DS F size of buffer
MQOD_SELECTIONSTRING_VSLENGTH DS F Length of variable length string
MQOD_SELECTIONSTRING_VSCCSID DS F CCSID of variable length string
MQOD_SELECTIONSTRING_LENGTH EQU *- MQOD_SELECTIONSTRING
                                ORG MQOD_SELECTIONSTRING
MQOD_SELECTIONSTRING_AREA DS CL(MQOD_SELECTIONSTRING_LENGTH)
*
MQOD_RESOBJECTSTRING DS F Resolved Long object name
MQOD_RESOBJECTSTRING_VSPTR DS F Address of variable length string
MQOD_RESOBJECTSTRING_VSOFFSET DS F Offset of variable length string
MQOD_RESOBJECTSTRING_VSBUFFSIZE DS F size of buffer
MQOD_RESOBJECTSTRING_VSLENGTH DS F Length of variable length string
MQOD_RESOBJECTSTRING_VSCCSID DS F CCSID of variable length string
MQOD_RESOBJECTSTRING_LENGTH EQU *- MQOD_RESOBJECTSTRING
                                ORG MQOD_RESOBJECTSTRING
MQOD_RESOBJECTSTRING_AREA DS CL(MQOD_RESOBJECTSTRING_LENGTH)
MQOD_RESOLVEDTYPE DS F Alias queue object resolved type
*
MQOD_LENGTH EQU *-MQOD
                                ORG MQOD
MQOD_AREA DS CL(MQOD_LENGTH)

```

## MQOD의 Visual Basic 선언

```

Type MQOD
  StrucId As String*4 'Structure identifier'
  Version As Long 'Structure version number'
  ObjectType As Long 'Object type'
  ObjectName As String*48 'Object name'
  ObjectQMgrName As String*48 'Object queue manager name'
  DynamicQName As String*48 'Dynamic queue name'
  AlternateUserId As String*12 'Alternate user identifier'
  RecsPresent As Long 'Number of object records present'
  KnownDestCount As Long 'Number of local queues opened
                          'successfully'
  UnknownDestCount As Long 'Number of remote queues opened
                          'successfully'
  InvalidDestCount As Long 'Number of queues that failed to
                          'open'
  ObjectRecOffset As Long 'Offset of first object record from
                          'start of MQOD'
  ResponseRecOffset As Long 'Offset of first response record
                          'from start of MQOD'
  ObjectRecPtr As MQPTR 'Address of first object record'
  ResponseRecPtr As MQPTR 'Address of first response record'
  AlternateSecurityId As MQBYTE40 'Alternate security identifier'
  ResolvedQName As String*48 'Resolved queue name'
  ResolvedQMgrName As String*48 'Resolved queue manager name'
End Type

```

## MQOD의 경우 StrucId (MQCHAR4)

오브젝트 디스크립터 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQOD\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

### MQOD\_STRUC\_ID

오브젝트 디스크립터 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQOD\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQOD\_STRUC\_ID와 동일 한 값을 갖지만 문자열이 아닌 문자의 배열입니다.

## MQOD의 버전 (MQLONG)

이는 구조 버전 번호입니다. 값은 다음 중 하나여야 합니다.

### MQOD\_VERSION\_1

버전 1 오브젝트 디스크립터 구조.

### MQOD\_VERSION\_2

버전 2 오브젝트 디스크립터 구조.

### MQOD\_VERSION\_3

버전 3 오브젝트 디스크립터 구조.

## **MQOD\_VERSION\_4**

버전 4 오브젝트 디스크립터 구조.

모든 버전이 모든 IBM MQ V7.0 환경에서 지원됩니다.

구조의 최신 버전에만 있는 필드는 필드 설명에 그렇게 식별되어 있습니다. 다음 상수는 현재 버전의 버전 번호를 지정합니다.

## **MQOD\_CURRENT\_VERSION**

현재 버전의 오브젝트 디스크립터 구조.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQOD\_VERSION\_1입니다.

## **MQOD의 경우 ObjectType (MQLONG)**

오브젝트 디스크립터에서 이름 지정되는 오브젝트의 유형입니다. 가능한 값은 다음과 같습니다.

### **MQOT\_CLNTCONN\_CHANNEL**

클라이언트 연결 채널. 오브젝트의 이름은 *ObjectName* 필드에서 찾을 수 있습니다.

### **MQOT\_Q**

큐. 오브젝트의 이름은 *ObjectName* 필드에서 찾을 수 있습니다.

### **MQOT\_NAMELIST**

이름 목록. 오브젝트의 이름은 *ObjectName* 필드에서 찾을 수 있습니다.

### **MQOT\_PROCESS**

process definition. 오브젝트의 이름은 *ObjectName* 필드에서 찾을 수 있습니다.

### **MQOT\_Q\_MGR**

큐 관리자. 오브젝트의 이름은 *ObjectName* 필드에서 찾을 수 있습니다.

### **MQOT\_TOPIC**

있습니다. 전체 토픽 이름은 두 개의 서로 다른 필드 *ObjectName* 및 *ObjectString*에서 빌드될 수 있습니다.

이러한 두 필드를 사용하는 방법에 대한 세부사항은 [토픽 문자열 결합](#)을 참조하십시오.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQOT\_Q입니다.

## **MQOD의 경우 ObjectName (MQCHAR48)**

이는 *ObjectQMgrName*으로 식별된 큐 관리자에서 정의된 오브젝트의 로컬 이름입니다. 이름에는 다음 문자가 포함될 수 있습니다.

- 대문자 알파벳(A - Z)
- 소문자 알파벳(a - z)
- 숫자(0 - 9)
- 마침표(.), 슬래시(/), 밑줄(\_), 퍼센트(%)

이름에 선두 문자 또는 임베드된 공백을 사용할 수 없으나 후미 문자 공백은 포함할 수 있습니다. 이름에서 중요한 데이터의 끝을 표시하기 위해 널 문자를 사용하십시오. 널 및 그 뒤에 오는 모든 문자는 공백으로 처리됩니다. 다음 제한사항이 표시된 환경에서 적용됩니다.

- EBCDIC 가타카나를 사용하는 시스템에서는 소문자를 사용할 수 없습니다.
- z/OS의 경우:
  - 밑줄로 시작하거나 끝나는 이름을 피하십시오. 조작 및 제어판에서 이를 처리할 수 없습니다.
  - 퍼센트 문자에는 RACF에 대한 특별한 의미가 있습니다. RACF가 외부 보안 관리자로 사용되는 경우 이름은 퍼센트를 포함할 수 없습니다. 퍼센트를 포함하는 경우 RACF 일반 프로파일이 사용될 때 보안 검사에 해당 이름이 포함되지 않습니다.
- IBM i에서 소문자, 슬래시 또는 퍼센트를 포함하는 이름은 명령에 지정할 때 따옴표로 묶어야 합니다. 이 인용 부호는 구조의 필드로서 또는 호출의 매개변수로서 발생하는 이름에는 지정될 수 없습니다.

전체 토픽 이름은 두 개의 서로 다른 필드 *ObjectName* 및 *ObjectString*에서 빌드될 수 있습니다. 이 두 필드를 사용하는 방법에 대한 자세한 내용은 [토픽 문자열 결합](#)을 참조하십시오.

다음 지점은 표시된 오브젝트의 유형에 적용됩니다.

- *ObjectName*이 모델 큐의 이름인 경우 큐 관리자는 모델 큐의 속성으로 동적 큐를 작성하며 작성된 큐의 이름을 *ObjectName* 필드에서 리턴합니다. 모델 큐는 MQOPEN 호출에서만 지정될 수 있습니다. 모델 큐는 MQPUT1 호출에서 유효하지 않습니다.
- *ObjectName*이 TARGTYPE(TOPIC)이 있는 알리어스 큐의 이름인 경우 이름 지정된 알리어스 큐에서 보안 검사가 먼저 수행됩니다. 알리어스 큐가 사용될 때 이는 정상입니다. 보안 검사가 완료되면 MQOPEN 호출이 계속되고 MQOT\_TOPIC의 MQOPEN 호출과 같이 동작합니다. 여기에는 관리 토픽 오브젝트에 대한 보안 검사 수행도 포함됩니다.
- *ObjectName* 및 *ObjectQMgrName*이 로컬 큐 관리자가 속한 큐 공유 그룹에서 소유한 공유 큐를 식별하는 경우 로컬 큐 관리자에서 동일한 이름의 큐 정의가 없어야 합니다. 해당 정의가 있는 경우(로컬 큐, 알리어스 큐, 리모트 큐 또는 모델 큐) 이유 코드 MQRC\_OBJECT\_NOT\_UNIQUE와 함께 호출이 실패합니다.
- 열리는 오브젝트가 분배 목록인 경우(즉, *RecsPresent*가 있고 0보다 큼) *ObjectName*은 공백이거나 널 문자열이어야 합니다. 이 조건이 충족되지 않으면 이유 코드 MQRC\_OBJECT\_NAME\_ERROR와 함께 호출이 실패합니다.
- *ObjectType*이 MQOT\_Q\_MGR인 경우 특수 규칙이 적용됩니다. 이 경우 이름은 첫 번째 널 문자 또는 필드의 끝까지 완전히 공백이어야 합니다.

이는 *ObjectName*이 모델 큐의 이름인 경우 MQOPEN 호출에 대한 입출력(I/O) 필드이고 다른 모든 경우에는 입력 전용 필드입니다. 이 필드의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백입니다.

### **ObjectQMgrMQOD의 이름 (MQCHAR48)**

다음은 *ObjectName* 오브젝트가 정의되어 있는 큐 관리자의 이름입니다. 이름에서 올바른 문자는 *ObjectName*의 문자와 동일합니다(472 페이지의 『MQOD의 경우 ObjectName (MQCHAR48)』 참조). 첫 번째 널 문자 또는 필드의 마지막까지 완전히 비어 있는 이름은 애플리케이션이 연결된 큐 관리자를 나타냅니다(로컬 큐 관리자).

다음 지점은 표시된 오브젝트의 유형에 적용됩니다.

- *ObjectType*이 MQOT\_TOPIC, MQOT\_NAMELIST, MQOT\_PROCESS 또는 MQOT\_Q\_MGR인 경우 *ObjectQMgrName*은 공백이거나 로컬 큐 관리자의 이름이어야 합니다.
- *ObjectName*이 모델 큐의 이름인 경우 큐 관리자는 모델 큐의 속성으로 동적 큐를 작성하고 *ObjectQMgrName* 필드에서 큐가 작성된 큐 관리자의 이름을 리턴합니다. 이는 로컬 큐 관리자의 이름입니다. 모델 큐는 MQOPEN 호출에서만 지정될 수 있습니다. 모델 큐는 MQPUT1 호출에서 유효하지 않습니다.
- *ObjectName*이 클러스터 큐의 이름이고 *ObjectQMgrName*이 공백인 경우 MQOPEN 호출로 리턴된 큐 핸들을 사용하여 전송된 메시지의 목적지는 다음과 같이 큐 관리자(또는 설치되어 있는 경우 클러스터 워크로드 엑시트)가 선택합니다.
  - MQOO\_BIND\_ON\_OPEN이 지정되면 MQOPEN 호출을 처리하는 동안 큐 관리자가 클러스터 큐의 특정 인스턴스를 선택하고 이 큐 핸들을 사용하여 넣는 모든 메시지는 해당 인스턴스로 전송됩니다.
  - MQOO\_BIND\_NOT\_FIXED가 지정되면 큐 관리자는 이 큐 핸들을 사용하는 각 초과 MQPUT 호출에 대해 클러스터의 다른 큐 관리자에 상주하는 목적지 큐의 다른 인스턴스를 선택할 수 있습니다.

애플리케이션이 클러스터 큐의 특정 인스턴스(즉, 클러스터의 특정 큐 관리자에 상주하는 큐 인스턴스)에 메시지를 전송해야 하는 경우 애플리케이션은 *ObjectQMgrName* 필드에서 해당 큐 관리자의 이름을 지정해야 합니다. 이는 로컬 큐 관리자가 강제로 지정된 대상 큐 관리자에 메시지를 전송하도록 합니다.

- *ObjectName*이 로컬 큐 관리자가 속한 큐 공유 그룹에서 소유한 공유 큐의 이름인 경우 *ObjectQMgrName*은 큐 공유 그룹의 이름, 로컬 큐 관리자의 이름 또는 공백일 수 있습니다. 메시지는 해당 값이 지정된 큐와 동일한 큐에 위치합니다.

큐 공유 그룹은 z/OS에서만 지원됩니다.

- *ObjectName*이 리모트 큐 공유 그룹(즉, 로컬 큐 관리자가 속하지 않은 큐 공유 그룹)이 소유한 공유 큐의 이름인 경우 *ObjectQMGrName*은 큐 공유 그룹의 이름이어야 합니다. 해당 그룹에 속한 큐 관리자의 이름을 사용할 수 있지만 이는 메시지가 큐 공유 그룹에 도달할 때 해당 특정 큐 관리자가 사용 불가능한 경우 메시지를 지연시킬 수 있습니다.
- 열리는 오브젝트가 분배 목록인 경우(즉, *RecsPresent*가 0보다 큼) *ObjectQMGrName*은 공백이거나 널 문자열이어야 합니다. 이 조건이 충족되지 않으면 이유 코드 MQRC\_OBJECT\_Q\_MGR\_NAME\_ERROR와 함께 호출이 실패합니다.

이는 *ObjectName*이 모델 큐의 이름인 경우 MQOPEN 호출에 대한 입출력(I/O) 필드이고 다른 모든 경우에는 입력 전용 필드입니다. 이 필드의 길이는 MQ\_Q\_MGR\_NAME\_LENGTH로 제공됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백입니다.

### **MQOD의 경우 *DynamicQName* (MQCHAR48)**

이는 MQOPEN 호출에서 작성하는 동적 큐의 이름입니다. 이는 *ObjectName*이 모델 큐의 이름을 지정하는 경우에만 관련성을 가집니다. 다른 모든 경우에서 *DynamicQName*은 무시됩니다.

이름에서 올바른 문자는 *ObjectName*의 문자와 동일합니다(별표도 유효함은 제외). *ObjectName*이 모델 큐의 이름인 경우 공백인 이름 또는 첫 번째 널 문자 전에 공백만 발생하는 이름은 올바르지 않습니다.

이름에서 공백이 아닌 마지막 문자가 별표 (\*) 인 경우, 큐 관리자는 별표를 큐에 대해 생성된 이름이 로컬 큐 관리자에서 고유하도록 보장하는 문자열로 대체합니다. 이에 대한 충분한 문자 수를 허용하기 위해 별표는 위치 1-33에서만 유효합니다. 별표 뒤에는 공백 또는 널 문자 이외의 문자가 있을 수 없습니다.

이는 별표가 첫 번째 문자 위치에서 발생하는 경우 유효하며 이 이름은 큐 관리자가 생성한 문자로만 구성됩니다.

z/OS에서 자동으로 생성되는 전체 이름으로 큐에서 수행되는 보안 검사가 없을 수 있으므로 첫 번째 문자 위치에서 별표가 있는 이름을 사용하지 마십시오.

입력 필드입니다. 이 필드의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다. 이 필드의 초기값은 환경에 의해 판별됩니다.

- z/OS에서 값은 'CSQ.\*'입니다.
- 기타 플랫폼에서 값은 'AMQ.\*'입니다.

값은 C에서는 널(Null) 종료 문자열이며 기타 프로그래밍 언어에서는 공백으로 채워진 문자열입니다.

### **MQOD에 대한 *AlternateUserID* (MQCHAR12)**

MQOPEN 호출에 대해 MQOO\_ALTERNATE\_USER\_AUTHORITY를 지정하거나 MQPUT1 호출에 대해 MQPMO\_ALTERNATE\_USER\_AUTHORITY를 지정하는 경우 애플리케이션이 현재 실행 중인 사용자 ID 대신 이 폴더는 열기를 위한 권한을 검사하는 데 사용되는 대체 사용자 ID를 포함합니다. 그러나 일부 검사는 현재 사용자 ID(예: 컨텍스트 검사)로 실행됩니다.

MQOO\_ALTERNATE\_USER\_AUTHORITY 또는 MQPMO\_ALTERNATE\_USER\_AUTHORITY가 지정되고 이 필드가 필드의 첫 번째 널 문자 또는 끝까지 완전히 공백인 경우 지정된 옵션으로 이 오브젝트를 여는 데 사용자 권한이 필요하지 않은 경우에만 열기가 성공할 수 있습니다.

MQOO\_ALTERNATE\_USER\_AUTHORITY 및 MQPMO\_ALTERNATE\_USER\_AUTHORITY 모두 지정되지 않으면 이 필드는 무시됩니다.

표시된 환경에서 다음 차이점이 존재합니다.

- z/OS에서 *AlternateUserId* 중 처음 8자는 열기의 권한을 검사하는 데 사용됩니다. 그러나 현재 사용자 ID에 이 특정 대체 사용자 ID를 지정할 권한이 부여되어야 합니다. 대체 사용자 ID 중 12자 모두 이 검사에 사용됩니다. 사용자 ID는 외부 보안 관리자가 허용하는 문자만 포함해야 합니다.

*AlternateUserId*가 큐에 지정되면 메시지를 넣을 때 큐 관리자가 값을 다음에 사용할 수 있습니다.

MQPUT 또는 MQPUT1 호출에서 지정된 MQPMO\_\*\_CONTEXT 옵션으로 인해 큐 관리자가 ID 컨텍스트 정보를 생성하는 경우 큐 관리자가 현재 사용자 ID 대신 *AlternateUserId*를 메시지의 MQMD에 있는 *UserIdentifier* 필드에 배치합니다.

- 다른 환경에서 *AlternateUserId*는 여는 오브젝트의 액세스 제어 검사에만 사용됩니다. 오브젝트가 큐인 경우 *AlternateUserId*는 해당 핸들을 사용하여 전송된 메시지의 MQMD에 있는 *UserIdentifier* 필드의 콘텐츠에 영향을 미치지 않습니다.

입력 필드입니다. 이 필드의 길이는 MQ\_USER\_ID\_LENGTH에 지정되어 있습니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 12자의 공백 문자입니다.

### **MQOD의 경우 RecsPresent (MQLONG)**

이는 애플리케이션에서 제공한 MQOR 오브젝트 레코드의 수입입니다. 이 숫자가 0보다 큰 경우 이는 *RecsPresent*가 목록에서 목적지 큐의 수가 되고 분배 목록이 열림을 표시합니다. 분배 목록은 단 하나의 목적지만 포함할 수 있습니다.

*RecsPresent*의 값이 0 미만일 수 없으며 0보다 큰 경우 *ObjectType*은 MQOT\_Q여야 합니다. 해당 조건이 충족되지 않으면 이유 코드 MQRC\_RECS\_PRESENT\_ERROR와 함께 호출이 실패합니다.

z/OS에서 이 필드는 0이어야 합니다.

입력 필드입니다. 이 필드의 초기값은 0입니다. 이 필드는 *Version*이 MQOD\_VERSION\_2 미만인 경우 무시됩니다.

### **KnownDestMQOD의 경우 (MQLONG)**

분배 목록에서 로컬 큐로 해석되고 성공적으로 열린 큐의 수입입니다. 이 수는 리모트 큐로 해석되는 큐는 포함하지 않습니다(초기에 메시지를 저장하는 데 로컬 전송 큐가 사용된 경우라도). 존재하는 경우 분배 목록에 없는 단일 큐를 열 때 이 필드도 설정됩니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 0입니다. *Version*이 MQOD\_VERSION\_1미만이면 이 필드가 무시됩니다.

### **MQOD에 대한 UnknownDest수 (MQLONG)**

분배 목록에서 리모트 큐로 해석되고 성공적으로 열린 큐의 수입입니다. 존재하는 경우 분배 목록에 없는 단일 큐를 열 때 이 필드도 설정됩니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 0입니다. *Version*이 MQOD\_VERSION\_1미만이면 이 필드가 무시됩니다.

### **MQOD에 대한 InvalidDest수 (MQLONG)**

이는 여는 데 실패한 분배 목록의 큐 수입입니다. 존재하는 경우 분배 목록에 없는 단일 큐를 열 때 이 필드도 설정됩니다.

**참고:** 존재하는 경우, 이 필드는 MQOPEN 또는 MQPUT1 호출의 **CompCode** 매개변수가 MQCC\_OK 또는 MQCC\_WARNING인 경우에만 설정됩니다. **CompCode** 매개변수가 MQCC\_FAILED인 경우 설정되지 않습니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 0입니다. *Version*이 MQOD\_VERSION\_1미만이면 이 필드가 무시됩니다.

### **ObjectRecMQOD의 오프셋 (MQLONG)**

MQOD 구조의 시작에서 처음 MQOR 오브젝트 레코드의 오프셋(바이트)입니다. 오프셋은 양수 또는 음수일 수 있습니다. *ObjectRecOffset*은 분배 목록이 열릴 때만 사용됩니다. *RecsPresent*이(가) 0인 경우 필드는 무시됩니다.

분배 목록이 열리고 있을 때, 분배 목록에 목적지 큐의 이름을 지정하기 위해 하나 이상의 MQOR 오브젝트 레코드의 배열을 제공해야 합니다. 다음 두 가지 방법 중 하나로 완료할 수 있습니다.

- 오프셋 필드 *ObjectRecOffset* 사용.

이 경우 애플리케이션은 필요한 만큼의 배열 요소가 있는 MQOR 레코드 배열 앞에 오는 MQOD가 포함된 고유한 구조를 선언하고 MQOD의 시작에서부터 배열의 첫 번째 요소의 오프셋으로 *ObjectRecOffset*을 설정해야 합니다. 이 오프셋이 정확하고 MQLONG 내에서 수용할 수 있는 값을 가지는지 확인하십시오(가장 제한적인 프로그래밍 언어는 올바른 범위가 -999,999,999 - +999,999,999인 COBOL임).



포인터 데이터 유형을 지원하지 않거나 다른 환경에 적용 가능하지 않은 방법으로 포인터 데이터 유형을 구현하는 프로그래밍 언어(예: COBOL 프로그래밍 언어)에 대해 *ObjectRecOffset*을 사용하십시오.

- 포인터 필드 *ObjectRecPtr* 사용.

이 경우, 애플리케이션은 MQOD 구조와 별도로 MQOR 구조의 배열을 선언하고 *ObjectRecPtr*을 배열의 주소로 설정할 수 있습니다.

다른 환경에 적용 가능한 방법으로 포인터 데이터 유형을 지원하는 프로그래밍 언어(예: C 프로그래밍 언어)에 대해 *ObjectRecPtr*을 사용하십시오.

사용자가 선택하는 기술에 상관없이 *ObjectRecOffset* 및 *ObjectRecPtr* 중 하나를 사용하십시오. 둘 모두 0이거나 0이 아닌 경우 이유 코드 MQRC\_OBJECT\_RECORDS\_ERROR와 함께 호출이 실패합니다.

입력 필드입니다. 이 필드의 초기값은 0입니다. 이 필드는 *Version*이 MQOD\_VERSION\_2 미만인 경우 무시됩니다.

### **ResponseRecMQOD의 오프셋 (MQLONG)**

MQOD 구조의 시작에서 처음 MQRR 응답 레코드의 오프셋(바이트)입니다. 오프셋은 양수 또는 음수일 수 있습니다. *ResponseRecOffset*은 분배 목록이 열릴 때만 사용됩니다. *RecsPresent*이(가) 0인 경우 필드는 무시됩니다.

분배 목록이 열릴 때 여는 데 실패한 큐(MQRR의 *CompCode* 필드) 및 각 실패에 대한 이유(MQRR의 *Reason* 필드)를 식별하기 위해 하나 이상의 MQRR 응답 레코드의 배열을 제공할 수 있습니다. 큐 이름이 오브젝트 레코드 배열에 나오는 동일한 순서대로 응답 레코드 배열에 데이터가 리턴됩니다. 큐 관리자는 호출의 결과가 혼합된 경우에만 응답 레코드를 설정합니다(즉, 기타 큐가 실패한 동안 일부 큐가 성공적으로 열림 또는 다른 이유로 인해 모두 실패함). 호출에서 이유 코드 MQRC\_MULTIPLE\_REASONS가 이 경우를 표시합니다. 동일한 이유 코드가 모든 큐에 적용되는 경우 해당 이유는 MQOPEN 또는 MQPUT1 호출의 **Reason** 매개변수에서 리턴되며 응답 레코드가 설정되지 않습니다. 응답 레코드는 선택사항이지만 제공되는 경우 *RecsPresent*가 있어야 합니다.

응답 레코드는 *ResponseRecOffset*에서 오프셋을 지정하거나 *ResponseRecPtr*에서 주소를 지정하여 오브젝트 레코드와 동일한 방법으로 제공할 수 있습니다. 이를 수행하는 방법에 대한 세부사항은 475 페이지의 『ObjectRecMQOD의 오프셋 (MQLONG)』의 내용을 참조하십시오. 그러나 *ResponseRecOffset* 및 *ResponseRecPtr* 중 둘 이상을 사용할 수 없습니다. 둘 모두 0이 아닌 경우 이유 코드 MQRC\_RESPONSE\_RECORDS\_ERROR와 함께 호출이 실패합니다.

MQPUT1 호출의 경우 이러한 응답 레코드는 메시지가 분배 목록에 있는 큐에 전송될 때 발생하는 오류 및 큐가 열릴 때 발생하는 오류에 대한 정보를 리턴하는 데 사용됩니다. 큐에 대한 넣기 조작의 완료 코드 및 이유 코드는 해당 큐에 대한 열기 조작의 완료 코드가 MQCC\_OK 또는 MQCC\_WARNING인 경우에만 이들 항목을 대체합니다.

입력 필드입니다. 이 필드의 초기값은 0입니다. 이 필드는 *Version*이 MQOD\_VERSION\_2 미만인 경우 무시됩니다.

### **MQOD의 경우 ObjectRecPtr (MQPTR)**

첫 번째 MQOR 오브젝트 레코드의 주소입니다. *ObjectRecPtr*은 분배 목록이 열릴 때만 사용됩니다. *RecsPresent*이(가) 0인 경우 필드는 무시됩니다.

*ObjectRecPtr* 또는 *ObjectRecOffset* 중 하나만 사용하여 오브젝트 레코드를 지정할 수 있습니다. *ObjectRecOffset* 필드에 대한 설명은 475 페이지의 『ObjectRecMQOD의 오프셋 (MQLONG)』의 내용을 참조하십시오. *ObjectRecPtr*을 사용하지 않는 경우 이를 널 포인터 또는 널 바이트로 설정하십시오.

입력 필드입니다. 이 필드의 초기값은 포인터를 지원하는 프로그래밍 언어로 된 널 포인터이며, 그렇지 않은 경우 모두 널 바이트 문자열입니다. 이 필드는 *Version*이 MQOD\_VERSION\_2 미만인 경우 무시됩니다.

**참고:** 프로그래밍 언어가 포인터 데이터 유형을 지원하지 않는 플랫폼에서 이 필드는 적절한 길이의 바이트 문자열로 선언되며, 초기값은 모두 널 바이트 문자열입니다.

### **MQOD의 경우 ResponseRecPtr (MQPTR)**



이는 첫 번째 MQRR 응답 레코드의 주소입니다. *ResponseRecPtr*은 분배 목록이 열릴 때만 사용됩니다. *RecsPresent*이(가) 0인 경우 필드는 무시됩니다.

*ResponseRecPtr* 또는 *ResponseRecOffset* 중 하나만 사용하여 응답 레코드를 지정하십시오. 세부사항은 476 페이지의 『*ResponseRecMQOD*의 오프셋 (MQLONG)』의 내용을 참조하십시오. *ResponseRecPtr*을 사용하지 않는 경우 이를 널 포인터 또는 널 바이트로 설정하십시오.

입력 필드입니다. 이 필드의 초기값은 포인터를 지원하는 프로그래밍 언어로 된 널 포인터이며, 그렇지 않은 경우 모두 널 바이트 문자열입니다. 이 필드는 *Version*이 MQOD\_VERSION\_2 미만인 경우 무시됩니다.

**참고:** 프로그래밍 언어가 포인터 데이터 유형을 지원하지 않는 플랫폼에서 이 필드는 적절한 길이의 바이트 문자열로 선언되며, 초기값은 모두 널 바이트 문자열입니다.

### **MQOD에 대한 *AlternateSecurityID* (MQBYTE40)**

이는 적절한 권한 검사를 수행할 수 있도록 *AlternateUserId*와 함께 권한 서비스에 전달되는 보안 ID입니다. *AlternateSecurityID*는 다음과 같은 경우에만 사용됩니다.

- MQOO\_ALTERNATE\_USER\_AUTHORITY는 MQOPEN 호출에서 제공됩니다. 또는
- MQPMO\_ALTERNATE\_USER\_AUTHORITY는 MQPUT1 호출에서 지정됩니다.

그리고 *AlternateUserId* 필드는 첫 번째 널 문자 또는 필드의 끝까지 완전히 공백이 아닙니다.

Windows에서 *AlternateSecurityId*를 사용하여 *AlternateUserId*를 고유하게 식별하는 Windows 보안 ID (SID)를 제공할 수 있습니다. 사용자의 SID는 LookupAccountName() Windows API 호출을 사용하여 Windows 시스템에서 얻을 수 있습니다.

z/OS에서 이 필드는 무시됩니다.

*AlternateSecurityId* 필드에는 다음 구조가 있습니다.

- 첫 번째 바이트는 뒤에 오는 중요한 데이터의 길이를 포함하는 2진 정수이며 이 값은 길이 바이트 자체를 제외합니다. 보안 ID가 존재하지 않으면 길이는 0입니다.
- 두 번째 바이트는 존재하는 보안 ID의 유형을 표시합니다. 다음 값이 가능합니다.

#### **MQSIDT\_NT\_SECURITY\_ID**

Windows 보안 ID.

#### **MQSIDT\_NONE**

보안 ID가 없음.

- 첫 번째 바이트에 의해 정의된 길이까지의 세 번째 및 후속 바이트는 보안 ID 자체를 포함합니다.
- 필드에 남아 있는 바이트는 2진수 영으로 설정됩니다.

다음 특수 값을 사용할 수 있습니다.

#### **MQSID\_NONE**

보안 ID가 지정되지 않음.

이 값은 필드 길이 만큼의 2진 0입니다.

C 프로그래밍 언어의 경우 상수 MQSID\_NONE\_ARRAY도 정의됩니다. 이는 MQSID\_NONE과 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

입력 필드입니다. 이 필드 길이는 MQ\_SECURITY\_ID\_LENGTH에서 제공합니다. 이 필드의 초기값은 MQSID\_NONE입니다. *Version*이(가) MQOD\_VERSION\_3보다 작으면 이 필드는 무시됩니다.

### **MQOD의 경우 *ResolvedQName* (MQCHAR48)**

이는 로컬 큐 관리자가 이름을 해석한 후 목적지 큐의 이름입니다. 리턴되는 이름은 *ResolvedQMgrName*(으)로 식별되는 큐 관리자에 존재하는 큐의 이름입니다.

오브젝트가 찾아보기, 입력 또는 출력(또는 결합)에 열린 단일 큐인 경우 비어 있는 값이 리턴됩니다. 열려 있는 오브젝트가 다음 중 하나인 경우 *ResolvedQName*은(는) 공백으로 설정됩니다.

- 큐가 아님

- 큐(단, 찾아보기, 입력 또는 출력을 위해 열려 있지 않음)
- 분배 목록
- 토픽 오브젝트를 참조하는 알리어스 큐입니다(대신 [ResObjectString](#) 참조).
- 토픽 오브젝트에 해석되는 알리어스 큐입니다.

이 필드는 출력 필드입니다. 이 필드의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백입니다. *Version*이(가) MQOD\_VERSION\_3보다 작으면 이 필드는 무시됩니다.

### **ResolvedQMGrMQOD의 이름 (MQCHAR48)**

로컬 큐 관리자가 이름을 해석한 후 목적지 큐 관리자의 이름입니다. 리턴되는 이름은 *ResolvedQName*(으)로 식별되는 큐를 소유하는 큐 관리자의 이름입니다. *ResolvedQMGrName*은(는) 로컬 큐 관리자의 이름일 수 있습니다.

*ResolvedQName*이(가) 로컬 큐 관리자가 속한 큐 공유 그룹이 소유하는 공유 큐인 경우, *ResolvedQMGrName*은(는) 큐 공유 그룹의 이름입니다. 다른 큐 공유 그룹이 큐를 소유하는 경우, *ResolvedQName*은(는) 큐 공유 그룹의 이름이거나 큐 공유 그룹의 구성원인 큐 관리자의 이름이 될 수 있습니다(리턴되는 값의 특성은 로컬 큐 관리자에 존재하는 큐 정의에 의해 결정됩니다).

오브젝트가 찾아보기, 입력 또는 출력(또는 결합)에 열린 단일 큐인 경우 비어 있는 값이 리턴됩니다. 열려 있는 오브젝트가 다음 중 하나인 경우 *ResolvedQMGrName*은(는) 공백으로 설정됩니다.

- 큐가 아님
- 큐(단, 찾아보기, 입력 또는 출력을 위해 열려 있지 않음)
- MQOO\_BIND\_NOT\_FIXED가 지정된 클러스터 큐(또는 **DefBind** 큐 속성에 MQBND\_BIND\_NOT\_FIXED값이 있는 경우 MQOO\_BIND\_AS\_Q\_DEF가 적용됨)
- 분배 목록

이 필드는 출력 필드입니다. 이 필드의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백입니다. *Version*이(가) MQOD\_VERSION\_3보다 작으면 이 필드는 무시됩니다.

### **MQOD의 경우 ObjectString (MQCHARV)**

*ObjectString* 필드는 긴 오브젝트 이름을 지정합니다.

이는 사용할 긴 오브젝트 이름을 지정합니다. 이 필드는 *ObjectType*의 특정 값에 대해서만 참조되며 모든 기타 값에 대해 무시됩니다. 이 필드를 사용함을 표시하는 값에 대한 세부사항은 *ObjectType*에 대한 설명을 참조하십시오.

*MQCHARV* 구조를 사용하는 방법에 대한 설명에 따라 *ObjectString*이 올바르지 않게 지정되는 경우 또는 최대 길이를 초과하는 경우 이유 코드 MQRC\_OBJECT\_STRING\_ERROR와 함께 호출이 실패합니다.

입력 필드입니다. 이 구조에 있는 필드의 초기값은 MQCHARV 구조의 값과 동일합니다.

전체 토픽 이름은 두 개의 서로 다른 필드 *ObjectName* 및 *ObjectString*에서 빌드될 수 있습니다. 이 두 필드를 사용하는 방법에 대한 자세한 내용은 [토픽 문자열 결합](#)을 참조하십시오.

### **MQOD의 경우 SelectionString (MQCHARV)**

이는 큐에서 메시지를 검색할 때 사용되는 선택 기준을 제공하는 데 사용되는 문자열입니다.

*SelectionString*은 다음 경우 제공되지 않아야 합니다.

- *ObjectType*이 MQOT\_Q가 아닌 경우
- 열리는 큐가 MQOO\_BROWSE 또는 MQOO\_INPUT\_\* 옵션 중 하나를 사용하여 열리지 않는 경우

위 경우에 *SelectionString*이 제공되면 호출은 MQRC\_SELECTOR\_INVALID\_FOR\_TYPE 이유 코드로 실패합니다.

291 페이지의 『MQCHARV - 가변 길이 문자열』 구조의 사용 방법 설명에 의하면 *SelectionString*이 올바르게 제공되었거나 최대 길이를 초과하는 경우 호출은 MQRC\_SELECTION\_STRING\_ERROR 이유 코드로 실패합니다. *SelectionString*의 최대 길이가 MQ\_SELECTOR\_LENGTH입니다.

*SelectionString* 사용법은 [Selectors](#)에서 설명됩니다.

### **MQOD에 대한 ResObject문자열 (MQCHARV)**

*ResObjectString* 필드는 큐 관리자가 *ObjectName* 필드에 제공된 이름을 해석한 후의 긴 오브젝트 이름입니다.

이 필드는 토픽 오브젝트를 참조하는 토픽 및 큐 알리어스에 대해서만 리턴됩니다.

긴 오브젝트 이름을 *ObjectString*에 제공하고 *ObjectName*에는 아무 것도 입력하지 않은 경우, 이 필드에 리턴되는 값은 *ObjectString*에 제공한 값과 동일합니다.

이 필드가 생략되면(즉, *ResObjectString.VSBufSize*가 0임) *ResObjectString*이 리턴되지 않지만 길이가 *ResObjectString.VSLength*에서 리턴됩니다.

버퍼 길이(*ResObjectString.VSBufSize*에서 제공됨)가 전체 *ResObjectString*보다 짧은 경우 문자열이 잘리고 제공된 버퍼에 최대한 맞도록 리턴됩니다.

MQCHARV 구조의 사용 방법 설명에 의하면 *ResObjectString*이 올바르게 지정되었거나 최대 길이를 초과하는 경우 호출은 MQRC\_RES\_OBJECT\_STRING\_ERROR 이유 코드로 실패합니다.

### **MQOD의 경우 ResolvedType (MQLONG)**

여는 해석된(기본) 오브젝트의 유형입니다.

가능한 값은 다음과 같습니다.

#### **MQOT\_Q**

해석된 오브젝트는 큐입니다. 큐를 직접 열거나 큐를 가리키는 알리어스 큐를 열 때 이 값이 적용됩니다.

#### **MQOT\_TOPIC**

해석된 오브젝트는 토픽입니다. 토픽을 직접 열거나 토픽 오브젝트를 가리키는 알리어스 큐를 열 때 이 값이 적용됩니다.

#### **MQOT\_NONE**

해석된 유형은 큐 및 토픽이 아닙니다.

## **MQOR - 오브젝트 레코드**

MQOR 구조를 사용하여 단일 목적지 큐의 큐 이름 및 큐 관리자 이름을 지정하십시오. MQOR은 MQOPEN 및 MQPUT1 호출의 입력 구조입니다.

### **가용성**

MQOR 구조는 다음 플랫폼에서 사용 가능합니다.

-  AIX
-  IBM i
-  Linux
-  Windows

및 해당 시스템에 연결된 IBM MQ MQI clients

### **문자 세트 및 인코딩**

MQOR의 데이터는 MQENC\_NATIVE로 지정된 로컬 큐 관리자의 인코딩 및 **CodedCharSetId** 큐 관리자 속성으로 지정된 문자 세트에 있어야 합니다. 그러나 애플리케이션이 MQ MQI 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트와 인코딩으로 작성되어야 합니다.

## 사용법

MQOPEN 호출에서 이러한 구조의 배열을 제공하여 큐 목록을 열 수 있습니다. 이 목록을 분배 목록이라고 합니다. 큐가 열린 경우 해당 MQOPEN 호출이 리턴한 큐 핸들을 사용한 각 메시지 넣기가 목록의 각 큐에 배치됩니다.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 505. MQOR에 대한 MQOR의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
ObjectName (오브젝트 이름)	없음	널 문자열 또는 공백
ObjectQMgr이름 (오브젝트 큐 관리자 이름)	없음	널 문자열 또는 공백

**참고:**

- 널 문자열 값 또는 공백은 C의 널 문자열과 기타 프로그래밍 언어의 공백 문자를 나타냅니다.
- C 프로그래밍 언어의 매크로 변수MQOR\_DEFAULT는 테이블에 나열되는 값을 포함합니다. 즉, 구조 내의 필드에 대한 초기값을 제공하기 위해 다음과 같은 방법으로 사용될 수 있습니다.

```
MQOR MyOR = {MQOR_DEFAULT};
```

## 언어 선언

### MQOR의 C 선언

```
typedef struct tagMQOR MQOR;  
struct tagMQOR {  
    MQCHAR48  ObjectName;    /* Object name */  
    MQCHAR48  ObjectQMgrName; /* Object queue manager name */  
};
```

### MQOR의 COBOL 선언

```
** MQOR structure  
10 MQOR.  
** Object name  
15 MQOR-OBJECTNAME PIC X(48).  
** Object queue manager name  
15 MQOR-OBJECTQMGRNAME PIC X(48).
```

### MQOR의 PL/I 선언

```
dcl  
1 MQOR based,  
3 ObjectName char(48), /* Object name */  
3 ObjectQMgrName char(48); /* Object queue manager name */
```

### MQOR의 Visual Basic 선언

```
Type MQOR  
    ObjectName As String*48 'Object name'  
    ObjectQMgrName As String*48 'Object queue manager name'  
End Type
```

## MQOR의 경우 ObjectName (MQCHAR48)

이는 다음을 제외하고는 MQOD 구조의 ObjectName 필드와 동일합니다(자세한 내용은 MQOD 참조).

- 큐의 이름이어야 합니다.
- 모델 큐의 이름이 아니어야 합니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백입니다.

### MQOR에 대한 ObjectQMgr이름 (MQCHAR48)

MQOD 구조의 *ObjectQMgrName* 필드와 동일합니다(자세한 내용은 MQOD 참조).

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백입니다.

### MQPD - 특성 디스크립터

MQPD 구조는 특성의 속성을 정의하는 데 사용됩니다. 구조는 MQSETMP 호출에서 입출력(I/O) 매개변수이며 MQINQMP 호출에서 출력 필드입니다.

### 가용성

MQPD 구조는 다음 플랫폼에서 사용 가능합니다.

-  AIX
-  IBM i
-  Linux
-  Windows
-  z/OS

및 IBM MQ MQI clients.

### 문자 세트 및 인코딩

MQPD의 데이터는 애플리케이션의 문자 세트 및 애플리케이션의 인코딩 (**MQENC\_NATIVE**)에 있어야 합니다.

### 필드

참고: 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 506. MQPD의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQPD_STRUC_ID	' PD '
버전 (구조 버전 번호)	MQPD_VERSION_1	1
옵션 (옵션)	MQPD_NONE	0
지원 (메시지 특성에 대한 필수 지원)	MQPD_SUPPORT_OPTIONAL	0
컨텍스트 (특성이 속한 메시지 컨텍스트)	MQPD_NO_CONTEXT	0
CopyOptions (특성이 속한 복사 옵션)	MQCOPY_DEFAULT	0

표 506. MQPD의 필드 (계속)

필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
<b>참고:</b>		
1. C 프로그래밍 언어에서 매크로 변수 MQPD_DEFAULT는 테이블에 나열되는 값을 포함합니다. 즉, 구조 내의 필드에 대한 초기값을 제공하기 위해 다음과 같은 방법으로 사용될 수 있습니다.		
<pre>MQPD MyPD = {MQPD_DEFAULT};</pre>		

## 언어 선언

### MQPD의 C 선언

```
typedef struct tagMQPD MQPD;
struct tagMQPD {
    MQCHAR4  StructId;      /* Structure identifier */
    MQLONG   Version;      /* Structure version number */
    MQLONG   Options;      /* Options that control the action of
                           MQSETMP and MQINQMP */
    MQLONG   Support;      /* Property support option */
    MQLONG   Context;      /* Property context */
    MQLONG   CopyOptions;  /* Property copy options */
};
```

### MQPD의 COBOL 선언

```
** MQPD structure
10 MQPD.
**   Structure identifier
15 MQPD-STRUCID PIC X(4).
**   Structure version number
15 MQPD-VERSION PIC S9(9) BINARY.
**   Options that control the action of MQSETMP and
**   MQINQMP
15 MQPD-OPTIONS PIC S9(9) BINARY.
**   Property support option
15 MQPD-SUPPORT PIC S9(9) BINARY.
**   Property context
15 MQPD-CONTEXT PIC S9(9) BINARY.
**   Property copy options
15 MQPD-COPYOPTIONS PIC S9(9) BINARY.
```

### MQPD의 PL/I 선언

```
dcl
1 MQPD based,
3 StructId   char(4),      /* Structure identifier */
3 Version    fixed bin(31), /* Structure version number */
3 Options    fixed bin(31), /* Options that control the action
                           of MQSETMP and MQINQMP */
3 Support    fixed bin(31), /* Property support option */
3 Context    fixed bin(31), /* Property context */
3 CopyOptions fixed bin(31); /* Property copy options */
```

### MQPD의 상위 레벨 어셈블러 선언

MQPD	DSECT		
MQPD_STRUCID	DS	CL4	Structure identifier
MQPD_VERSION	DS	F	Structure version number
MQPD_OPTIONS	DS	F	Options that control the action of MQSETMP and MQINQMP
*			
MQPD_SUPPORT	DS	F	Property support option
MQPD_CONTEXT	DS	F	Property context
MQPD_COPYOPTIONS	DS	F	Property copy options

MQPD_LENGTH	EQU	*-MQPD
MQPD_AREA	DS	CL(MQPD_LENGTH)

### **MQPD의 경우 StrucId (MQCHAR4)**

특성 디스크립터 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQPD\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

#### **MQPD\_STRUC\_ID**

특성 디스크립터 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 **MQPD\_STRUC\_ID\_ARRAY**도 정의됩니다. 이는 **MQPD\_STRUC\_ID**와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### **MQPD의 버전 (MQLONG)**

구조 버전 번호이며 값은 다음과 같아야 합니다.

#### **MQPD\_VERSION\_1**

버전 1 특성 디스크립터 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### **MQPD\_CURRENT\_VERSION**

특성 디스크립터 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 **MQPD\_VERSION\_1**입니다.

### **MQPD에 대한 옵션 (MQLONG)**

값은 다음과 같아야 합니다.

#### **MQPD\_NONE**

옵션이 지정되지 않음

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQPD\_NONE입니다.

### **MQPD에 대한 지원 (MQLONG)**

이 필드는 큐 관리자가 이 특성을 포함하는 메시지를 큐에 넣기 위해 메시지 특성에 대한 어떤 지원 레벨이 필요한지 설명합니다. 이는 IBM MQ 정의 특성에만 적용됩니다. 다른 모든 특성에 대한 지원은 선택사항입니다.

큐 관리자가 IBM MQ 정의 특성을 알고 있는 경우 필드는 올바른 값으로 자동 설정됩니다. 특성이 인식되지 않는 경우 MQPD\_SUPPORT\_OPTIONAL이 지정됩니다. 큐 관리자에서 큐 관리자가 올바르지 않다고 인식하는 IBM MQ 정의 특성이 포함된 메시지를 수신하는 경우 큐 관리자는 *Support* 필드의 값을 조정합니다.

MQCMMHO\_NO\_VALIDATION 옵션이 설정된 메시지 핸들에서 MQSETMP 호출을 사용하여 IBM MQ 정의 특성을 설정할 때 *Support*가 입력 필드가 됩니다. 이를 통해 애플리케이션은 연결된 큐 관리자에서 특성이 지원되지 않지만 메시지가 다른 큐 관리자에서 처리되어야 하는 경우 IBM MQ 정의 특성을 올바른 값과 함께 넣을 수 있습니다.

값 MQPD\_SUPPORT\_OPTIONAL은 항상 IBM MQ 정의 특성이 아닌 특성에 지정됩니다.

메시지 특성을 지원하는 IBM WebSphere MQ 7.0 큐 관리자가 인식되지 않는 *Support* 값을 포함하는 특성을 수신하는 경우, 해당 특성은 다음과 같이 처리됩니다.

- 인식되지 않는 값이 MQPD\_REJECT\_UNSUP\_MASK에 포함된 경우 MQPD\_SUPPORT\_REQUIRED가 지정되었습니다.
- 인식되지 않는 값이 MQPD\_ACCEPT\_UNSUP\_IF\_XMIT\_MASK에 포함된 경우 MQPD\_SUPPORT\_REQUIRED\_IF\_LOCAL이 지정되었습니다.
- 그렇지 않으면 MQPD\_SUPPORT\_OPTIONAL이 지정되었습니다.

MQCMHO\_NO\_VALIDATION 옵션이 설정된 메시지 핸들에서 MQSETMP 호출을 사용할 때 다음 값 중 하나가 MQINQMP 호출로 리턴되거나 값 중 하나가 지정될 수 있습니다.

#### **MQPD\_SUPPORT\_OPTIONAL**

지원되지 않는 경우라도 특성은 큐 관리자에 의해 허용됩니다. 특성은 메시지 특성을 지원하지 않는 큐 관리자 메시지가 플로우하기 위해 특성을 제거할 수 있습니다. 또한 이 값은 IBM MQ에서 정의하지 않은 특성에 지정됩니다.

#### **MQPD\_SUPPORT\_REQUIRED**

특성에 대한 지원은 필수입니다. 메시지는 IBM MQ 정의 특성을 지원하지 않는 큐 관리자에 의해 거부됩니다. MQPUT 또는 MQPUT1 호출은 MQCC\_FAILED 완료 코드 및 MQRC\_UNSUPPORTED\_PROPERTY 이유 코드로 실패합니다.

#### **MQPD\_SUPPORT\_REQUIRED\_IF\_LOCAL**

메시지가 로컬 큐로 예정되는 경우 메시지는 IBM MQ 정의 특성을 지원하지 않는 큐 관리자에 의해 거부됩니다. MQPUT 또는 MQPUT1 호출은 MQCC\_FAILED 완료 코드 및 MQRC\_UNSUPPORTED\_PROPERTY 이유 코드로 실패합니다.

MQPUT 또는 MQPUT1 호출은 메시지가 리모트 큐 관리자로 예정되는 경우 성공합니다.

메시지 핸들이 MQCMHO\_NO\_VALIDATION 옵션 세트에 작성된 경우 이는 MQINQMP 호출에서 출력 필드이고 MQSETMP 호출에서 입력 필드입니다. 이 필드의 초기값은 MQPD\_SUPPORT\_OPTIONAL입니다.

### **MQPD의 컨텍스트 (MQLONG)**

특성이 속한 메시지 컨텍스트에 대해 설명합니다.

큐 관리자가 올바르게 있다고 인식하는 IBM MQ 정의 특성이 포함된 메시지를 큐 관리자가 수신하는 경우, 큐 관리자는 *Context* 필드의 값을 정정합니다.

다음 옵션을 지정할 수 있습니다.

#### **MQPD\_USER\_CONTEXT**

특성은 사용자 컨텍스트와 연관됩니다.

MQSETMP 호출을 사용하여 사용자 컨텍스트와 연관된 특성을 설정할 수 있는 특수 권한은 필요하지 않습니다.

IBM WebSphere MQ 7.0 큐 관리자에서 사용자 컨텍스트와 연관된 특성은 MQOO\_SAVE\_ALL\_CONTEXT에 대해 설명된 대로 저장됩니다. 지정된 MQPMO\_PASS\_ALL\_CONTEXT가 있는 MQPUT 호출로 인해 저장된 컨텍스트에서 새 메시지로 특성이 복사될 수 있습니다.

이전에 설명된 옵션이 필요하지 않다면 다음 옵션을 사용할 수 있습니다.

#### **MQPD\_NO\_CONTEXT**

특성은 메시지 컨텍스트와 연관되지 않습니다.

인식되지 않은 값은 MQRC\_PD\_ERROR의 *Reason* 코드로 거부됩니다.

이는 MQSETMP 호출에 대한 입출력(I/O) 필드이며 MQINQMP 호출의 출력 필드입니다. 이 필드의 초기값은 MQPD\_NO\_CONTEXT입니다.

### **MQPD의 경우 CopyOptions (MQLONG)**

속성이 복사되어야 하는 메시지 유형에 대해 설명합니다. 이는 인식된 IBM MQ 정의된 특성에 대한 출력 유일 필드입니다. IBM MQ는 적절한 값을 설정합니다.

큐 관리자가 올바르게 않은 것으로 인식하는 IBM MQ 정의 특성을 포함하는 메시지를 큐 관리자가 수신하면 큐 관리자가 *CopyOptions* 필드의 값을 정정합니다.

이 옵션 중 하나 이상을 지정할 수 있습니다. 둘 이상의 옵션을 지정하려면 값을 함께 추가하거나(동일한 상수를 두 번 이상 추가하지 않음), 비트 단위 OR 연산을 사용하여 값을 결합하십시오(프로그래밍 언어가 비트 연산을 지원하는 경우).

#### **MQCOPY\_FORWARD**

이 특성은 전달되는 메시지에 복사됩니다.



## **MQCOPY\_PUBLISH**

이 특성은 메시지가 발행되는 경우 구독자가 수신하는 메시지에 복사됩니다.

## **MQCOPY\_REPLY**

이 특성은 응답 메시지에 복사됩니다.

## **MQCOPY\_REPORT**

이 특성은 보고서 메시지에 복사됩니다.

## **MQCOPY\_ALL**

이 특성은 모든 유형의 후속 메시지에 복사됩니다.

**기본 옵션:** 기본 복사 옵션 세트를 제공하기 위해 다음 옵션을 지정할 수 있습니다.

## **MQCOPY\_DEFAULT**

이 특성은 전달되는 메시지, 보고서 메시지 또는 메시지가 발행되는 경우 구독자가 수신하는 메시지에 복사됩니다.

이는 옵션 MQCOPY\_FORWARD 및 MQCOPY\_REPORT 및 MQCOPY\_PUBLISH의 결합을 지정하는 것과 같습니다.

이전에 설명된 옵션이 필요하지 않은 경우 다음 옵션을 사용하십시오.

## **MQCOPY\_NONE**

이 값을 사용하여 기타 복사 옵션이 지정되지 않음을 표시하십시오. 이 특성 및 후속 메시지 간에 프로그래밍 방식으로 관계가 없습니다. 이는 항상 메시지 디스크립터 특성에 대해 리턴됩니다.

이는 MQSETMP 호출에 대한 입출력(I/O) 필드이며 MQINQMP 호출의 출력 필드입니다. 이 필드의 초기값은 MQCOPY\_DEFAULT입니다.

## **MQPMO - 메시지 넣기 옵션**

MQPMO 구조를 사용하면 애플리케이션이 메시지가 큐에 배치되거나 토픽에 발행되는 방법을 제어하는 옵션을 지정할 수 있습니다. 구조는 MQPUT 및 MQPUT1 호출의 입/출력 매개변수입니다.

## **버전**

MQPMO의 현재 버전은 MQPMO\_VERSION\_3입니다. 특정 필드는 MQPMO의 특정 버전에서만 사용 가능합니다. 여러 환경 간에 애플리케이션을 이식해야 하는 경우 MQPMO의 버전이 모든 환경에서 지속적인지 확인해야 합니다. 특정 버전의 구조에만 존재하는 필드는 이 주제 및 필드 설명에서와 같이 식별됩니다.

지원되는 프로그래밍 언어에 제공되는 헤더, COPY, INCLUDE 파일은 MQPMO\_VERSION\_1로 설정된, *Version* 필드의 초기값과 함께 환경이 지원하는 MQPMO의 최신 버전을 포함합니다. version-1 구조에 없는 필드를 사용하려면 애플리케이션이 *Version* 필드를 필요한 버전의 버전 번호로 설정해야 합니다.

## **문자 세트 및 인코딩**

MQPMO의 데이터는 MQENC\_NATIVE로 지정된 로컬 큐 관리자의 인코딩 및 **CodedCharSetId** 큐 관리자 속성으로 지정된 문자 세트에 있어야 합니다. 그러나 애플리케이션이 MQ MQI 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트와 인코딩으로 작성되어야 합니다.

## **필드**

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 507. MQPMO의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQPMO_STRUC_ID	'PMO~'
버전 (구조 버전 번호)	MQPMO_VERSION_1	1
옵션 (MQPUT 및 MQPUT1의 조치를 제어하는 옵션)	MQPMO_NONE	0

표 507. MQPMO의 필드 (계속)		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
제한시간 (예약됨)	없음	-1
Context (입력 큐의 오브젝트 핸들)	없음	0
KnownDest수 (로컬 큐로 보낸 메시지 수)	없음	0
UnknownDest수 (원격 큐로 보낸 메시지 수)	없음	0
InvalidDest수 (보낼 수 없는 메시지 수)	없음	0
ResolvedQName (목적지 큐의 해석된 이름)	없음	널 문자열 또는 공백
ResolvedQMgr이름 (목적지 큐 관리자의 해석된 이름)	없음	널 문자열 또는 공백
참고: <i>Version</i> 이 MQPMO_VERSION_2 미만인 경우 나머지 필드는 무시됩니다.		
RecsPresent (존재하는 Put 메시지 레코드 또는 응답 레코드 수)	없음	0
PutMsgRecFields (존재하는 MQPMR 필드를 표시하는 플래그)	MQPMRF_NONE	0
PutMsgRecOffset (MQPMO의 시작에서 첫 번째 넣기 메시지 레코드의 오프셋)	없음	0
ResponseRec오프셋 (MQPMO 시작 이후 첫 번째 응답 레코드의 오프셋)	없음	0
PutMsgRecPtr (첫 번째 넣기 메시지 레코드의 주소)	없음	널 포인터 또는 널 바이트
ResponseRecPtr (첫 번째 응답 레코드의 주소)	없음	널 포인터 또는 널 바이트
참고: <i>Version</i> 이 MQPMO_VERSION_3 미만인 경우 나머지 필드는 무시됩니다.		
OriginalMsgHandle (원래 메시지 핸들)	MQHM_NONE	0
NewMsg핸들 (새 메시지 핸들)	MQHM_NONE	0
조치 (수행 중인 넣기 유형 및 <i>OriginalMsgHandle</i> 필드에 지정된 원래 메시지와 <i>NewMsgHandle</i> 필드에 지정된 새 메시지 간의 관계)	MQACTP_NEW	0
PubLevel (발행물의 대상이 되는 구독 레벨)	없음	9
<p>참고:</p> <ol style="list-style-type: none"> <li>~ 기호는 단일 공백 문자를 나타냅니다.</li> <li>널 문자열 값 또는 공백은 C의 널 문자열과 기타 프로그래밍 언어의 공백 문자를 나타냅니다.</li> <li>C 프로그래밍 언어의 매크로 변수MQPMO_DEFAULT는 테이블에 나열되는 값을 포함합니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오.</li> </ol> <pre>MQPMO MyPMO = {MQPMO_DEFAULT};</pre>		

## 언어 선언

### MQPMO의 C 선언

```
typedef struct tagMQPMO MQPMO;
struct tagMQPMO {
```

```

MQCHAR4  StrucId;          /* Structure identifier */
MQLONG   Version;         /* Structure version number */
MQLONG   Options;        /* Options that control the action of
                          MQPUT and MQPUT1 */

MQLONG   Timeout;        /* Reserved */
MQHOBJS  Context;        /* Object handle of input queue */
MQLONG   KnownDestCount; /* Number of messages sent
                          successfully to local queues */
MQLONG   UnknownDestCount; /* Number of messages sent
                          successfully to remote queues */
MQLONG   InvalidDestCount; /* Number of messages that could not
                          be sent */
MQCHAR48 ResolvedQName;  /* Resolved name of destination
                          queue */
MQCHAR48 ResolvedQMGrName; /* Resolved name of destination queue
                          manager */

/* Ver:1 */
MQLONG   RecsPresent;    /* Number of put message records or
                          response records present */
MQLONG   PutMsgRecFields; /* Flags indicating which MQPMR fields
                          are present */
MQLONG   PutMsgRecOffset; /* Offset of first put message record
                          from start of MQPMO */
MQLONG   ResponseRecOffset; /* Offset of first response record
                          from start of MQPMO */
MQPTR    PutMsgRecPtr;   /* Address of first put message
                          record */
MQPTR    ResponseRecPtr; /* Address of first response record */

/* Ver:2 */
MQHMSG   OriginalMsgHandle; /* Original message handle */
MQHMSG   NewMsgHandle;     /* New message handle */
MQLONG   Action;          /* The action being performed */
MQLONG   PubLevel;        /* Subscription level */

/* Ver:3 */
};

```

## MQPMO의 COBOL 선언

```

**  MQPMO structure
10  MQPMO.
**  Structure identifier
15  MQPMO-STRUCID      PIC X(4).
**  Structure version number
15  MQPMO-VERSION     PIC S9(9) BINARY.
**  Options that control the action of MQPUT and MQPUT1
15  MQPMO-OPTIONS     PIC S9(9) BINARY.
**  Reserved
15  MQPMO-TIMEOUT     PIC S9(9) BINARY.
**  Object handle of input queue
15  MQPMO-CONTEXT     PIC S9(9) BINARY.
**  Number of messages sent successfully to local queues
15  MQPMO-KNOWNDSTCOUNT PIC S9(9) BINARY.
**  Number of messages sent successfully to remote queues
15  MQPMO-UNKNOWNDSTCOUNT PIC S9(9) BINARY.
**  Number of messages that could not be sent
15  MQPMO-INVALIDDSTCOUNT PIC S9(9) BINARY.
**  Resolved name of destination queue
15  MQPMO-RESOLVEDQNAME  PIC X(48).
**  Resolved name of destination queue manager
15  MQPMO-RESOLVEDQMGRNAME PIC X(48).
**  Number of put message records or response records present
15  MQPMO-RECSPRESENT   PIC S9(9) BINARY.
**  Flags indicating which MQPMR fields are present
15  MQPMO-PUTMSGRECFIELDS PIC S9(9) BINARY.
**  Offset of first put message record from start of MQPMO
15  MQPMO-PUTMSGRECOFFSET PIC S9(9) BINARY.
**  Offset of first response record from start of MQPMO
15  MQPMO-RESPONSERECOFFSET PIC S9(9) BINARY.
**  Address of first put message record
15  MQPMO-PUTMSGRECPTTR  POINTER.
**  Address of first response record
15  MQPMO-RESPONSERECPTTR POINTER.
**  Original message handle
15  MQPMO-ORIGINALMSGHANDLE PIC S9(18) BINARY.
**  New message handle
15  MQPMO-NEWMSGHANDLE   PIC S9(18) BINARY.
**  The action being performed
15  MQPMO-ACTION        PIC S9(9) BINARY.

```

```
**      Publish level
15 MQPMO-PUBLEVEL          PIC S9(9) BINARY.
```

## MQPMO의 PL/I 선언

```
dcl
1 MQPMO based,
3 StructId          char(4),          /* Structure identifier */
3 Version           fixed bin(31),    /* Structure version number */
3 Options           fixed bin(31),    /* Options that control the action
of MQPUT and MQPUT1 */

3 Timeout           fixed bin(31),    /* Reserved */
3 Context           fixed bin(31),    /* Object handle of input queue */
3 KnownDestCount   fixed bin(31),    /* Number of messages sent
successfully to local queues */
3 UnknownDestCount fixed bin(31),    /* Number of messages sent
successfully to remote queues */
3 InvalidDestCount fixed bin(31),    /* Number of messages that could
not be sent */
3 ResolvedQName    char(48),         /* Resolved name of destination
queue */
3 ResolvedQMgrName char(48),         /* Resolved name of destination
queue manager */
3 RecsPresent      fixed bin(31),    /* Number of put message records or
response records present */
3 PutMsgRecFields  fixed bin(31),    /* Flags indicating which MQPMR
fields are present */
3 PutMsgRecOffset  fixed bin(31),    /* Offset of first put message
record from start of MQPMO */
3 ResponseRecOffset fixed bin(31),  /* Offset of first response record
from start of MQPMO */
3 PutMsgRecPtr     pointer,          /* Address of first put message
record */
3 ResponseRecPtr   pointer,          /* Address of first response
record */

3 OriginalMsgHandle fixed bin(63),  /* Original message handle */
3 NewMsgHandle      fixed bin(63);   /* New message handle */
3 Action            fixed bin(31);   /* The action being performed */
3 PubLevel          fixed bin(31);   /* Publish level */
```

## MQPMO의 상위 레벨 어셈블러 선언

MQPMO	DSECT		
MQPMO_STRUCID	DS	CL4	Structure identifier
MQPMO_VERSION	DS	F	Structure version number
MQPMO_OPTIONS	DS	F	Options that control the action of
*			MQPUT and MQPUT1
MQPMO_TIMEOUT	DS	F	Reserved
MQPMO_CONTEXT	DS	F	Object handle of input queue
MQPMO_KNOWNDSTCOUNT	DS	F	Number of messages sent successfully
*			to local queues
MQPMO_UNKNOWNDSTCOUNT	DS	F	Number of messages sent successfully
*			to remote queues
MQPMO_INVALIDDESTCOUNT	DS	F	Number of messages that could not be
*			sent
MQPMO_RESOLVEDQNAME	DS	CL48	Resolved name of destination queue
MQPMO_RESOLVEDQMGRNAME	DS	CL48	Resolved name of destination queue
*			manager
MQPMO_RECSPRESENT	DS	F	Number of put message records or
*			response records present
MQPMO_PUTMSGRECFIELDS	DS	F	Flags indicating which MQPMR
*			fields are present
MQPMO_PUTMSGRECOFFSET	DS	F	Offset of first put message record
*			from start of MQPMO
MQPMO_RESPONSERECOFFSET	DS	F	Offset of first response record
*			from start of MQPMO
MQPMO_PUTMSGRECPtr	DS	F	Address of first put message
*			record
MQPMO_RESPONSERECPtr	DS	F	Address of first response record
MQPMO_ORIGINALMSGHANDLE	DS	D	Original message handle
MQPMO_NEWMSGHANDLE	DS	D	New message handle
MQPMO_ACTION	DS	F	The action being performed
MQPMO_PUBLEVEL	DS	F	Publish level
*			
MQPMO_LENGTH	EQU	**MQPMO	

MQPMO_AREA	ORG DS	MQPMO CL(MQPMO_LENGTH)
------------	--------	------------------------

## MQPMO의 Visual Basic 선언

```

Type MQPMO
  StrucId      As String*4  'Structure identifier'
  Version      As Long      'Structure version number'
  Options      As Long      'Options that control the action of'
                                     'MQPUT and MQPUT1'
  Timeout      As Long      'Reserved'
  Context      As Long      'Object handle of input queue'
  KnownDestCount As Long      'Number of messages sent successfully'
                                     'to local queues'
  UnknownDestCount As Long      'Number of messages sent successfully'
                                     'to remote queues'
  InvalidDestCount As Long      'Number of messages that could not be'
                                     'sent'
  ResolvedQName As String*48 'Resolved name of destination queue'
  ResolvedQMGrName As String*48 'Resolved name of destination queue'
                                     'manager'
  RecsPresent   As Long      'Number of put message records or'
                                     'response records present'
  PutMsgRecFields As Long      'Flags indicating which MQPMR fields'
                                     'are present'
  PutMsgRecOffset As Long      'Offset of first put message record'
                                     'from start of MQPMO'
  ResponseRecOffset As Long      'Offset of first response record from'
                                     'start of MQPMO'
  PutMsgRecPtr   As MQPTR     'Address of first put message record'
  ResponseRecPtr As MQPTR     'Address of first response record'
End Type

```

### MQPMO의 경우 StrucId (MQCHAR4)

메시지 넣기 옵션 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQPMO\_STRUC\_ID입니다. 값은 다음과 같아야 합니다.

#### MQPMO\_STRUC\_ID

메시지 넣기 옵션 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQPMO\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQPMO\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### MQPMO의 버전 (MQLONG)

구조 버전 번호.

값은 다음 중 하나여야 합니다.

#### MQPMO\_VERSION\_1

버전 1 Put 메시지 옵션 구조.

이 버전은 모든 환경에서 지원됩니다.

#### MQPMO\_VERSION\_2

버전 2 Put 메시지 옵션 구조.

이 버전은 다음 환경에서 지원됩니다.

-  AIX
-  IBM i
-  Linux
-  Windows

및 해당 시스템에 연결된 IBM MQ MQI clients

#### MQPMO\_VERSION\_3

버전 3 Put 메시지 옵션 구조.

이 버전은 모든 환경에서 지원됩니다.

최신 버전의 구조에만 있는 필드는 필드의 설명에서 최신 필드로 식별됩니다. 다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### **MQPMO\_CURRENT\_VERSION**

현재 버전의 Put 메시지 옵션 구조.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQPMO\_VERSION\_1입니다.

#### **MQPMO에 대한 옵션 (MQLONG)**

옵션 필드는 MQPUT 및 MQPUT1 호출의 조작을 제어합니다.

**범위 옵션.** 지정할 수 있는 MQPMO 옵션의 수에는 제한이 없습니다. 올바르지 않은 둘 이상의 옵션을 지정하려면 값을 함께 추가하거나(동일한 상수를 두 번 이상 추가하지 않음), 비트 단위 OR 연산을 사용하여 값을 결합하십시오(프로그래밍 언어가 비트 연산을 지원하는 경우). 결합이 설명되어 있습니다. 다른 모든 결합은 유효합니다.

다음 옵션은 전송된 발행의 범위를 제어합니다.

#### **MQPMO\_SCOPE\_QMGR**

발행은 이 큐 관리자를 구독한 구독자에게만 전송됩니다. 발행은 이 큐 관리자에 대해 구독을 작성한 리모트 발행/구독 큐 관리자로 전달되지 않으며 PUBSCOPE 토픽 속성을 사용하여 설정된 모든 작동을 대체합니다.

**참고:** 설정되지 않으면 발행 범위는 PUBSCOPE 토픽 속성으로 판별됩니다.

**발행 옵션.** 다음 옵션은 메시지가 토픽에 발행되는 방식을 제어합니다.

#### **MQPMO\_SUPPRESS\_REPLYTO**

이 발행의 MQMD의 *ReplyToQ* 및 *ReplyToQMGR* 필드에 지정된 정보는 구독자에게 전달되지 않습니다. 이 옵션이 *ReplyToQ*이(가) 필요한 보고 옵션과 함께 사용되는 경우 호출은 MQRC\_MISSING\_REPLY\_TO\_Q와 함께 실패합니다.

#### **MQPMO\_RETAIN**

송신되는 발행물을 큐 관리자가 보유하게 됩니다. 이렇게 보유되므로 구독자는 발행된 시간 이후에 MQSUBRQ 호출을 사용하여 이 발행의 사본을 요청할 수 있습니다. 또한 발행된 시간 후에 구독을 작성하는 애플리케이션에 발행을 전송할 수도 있습니다(MQSO\_NEW\_PUBLICATIONS\_ONLY 옵션을 사용하여 전송하도록 선택하지 않는 경우). 애플리케이션이 보유된 발행을 전송한 경우 이는 해당 발행의 MQIsRetained 메시지 특성으로 표시됩니다.

주제 트리의 각 노드는 단 하나의 발행물만 보유할 수 있습니다. 그러므로 이 토픽에 대해 보유된 발행이 이미 있는 경우 다른 애플리케이션이 발행하면 이 발행으로 바뀝니다. 그러므로 동일한 주제에 대해 둘 이상의 발행자가 메시지를 보유하지 않도록 하는 것이 좋습니다.

구독자가 보유된 발행을 요청하는 경우 사용된 구독이 토픽에서 와일드카드를 포함할 수 있으며 이 경우 많은 보유된 발행이 일치할 수 있고(토픽 트리의 다양한 노드에서) 요청하는 애플리케이션으로 여러 발행이 전송될 수 있습니다. 자세한 정보는 [765 페이지의 『MQSUBRQ - 구독 요청』](#) 호출에 대한 설명을 참조하십시오.

보유된 발행이 구독 레벨과 상호작용하는 방법에 대한 정보는 [발행 인터셉트](#)를 참조하십시오.

이 옵션이 사용되고 발행물을 보유할 수 없는 경우에는 메시지가 발행되지 않고 호출은 MQRC\_PUT\_NOT\_RETAINED가 발생하여 실패합니다.

#### **MQPMO\_NOT\_OWN\_SUBS**

애플리케이션이 소유한 구독으로 발행을 전송하지 않으려 한다고 큐 관리자에게 알립니다. 연결 핸들이 동일한 경우 구독은 동일한 애플리케이션이 소유한 것으로 간주됩니다.

#### **MQPMO\_WARN\_IF\_NO\_SUBS\_MATCHED**

발행과 일치하는 구독이 없으면 MQCC\_WARNING 및 이유 코드 MQRC\_NO\_SUBS\_MATCHED의 완료 코드(CompCode)를 리턴합니다.

넣기 조작이 MQRC\_NO\_SUBS\_MATCHED를 리턴하는 경우 발행이 구독에 전달되지 않았습니다. 그러나 넣기 조작에서 MQPMO\_RETAIN 옵션이 지정된 경우 메시지가 보유하고 이후에 정의된 일치하는 구독으로 전달됩니다.

토픽의 구독은 다음 조건을 충족하는 경우 발행과 일치합니다.

- 메시지가 구독 큐에 전달됨
- 메시지가 구독 큐에 전달되었지만 큐에서 메시지를 큐에 넣을 수 없음을 의미하는 문제점이 발생하여 데드-레터 큐에 배치되거나 제거됨
- 구독에으로의 메시지 전달을 억제하는 라우팅 엑시트가 정의됨

토픽의 구독은 다음 조건을 충족하는 경우 발행과 일치하지 않습니다.

- 구독에 발행과 일치하지 않는 선택 문자열이 있음
- 구독이 MQSO\_PUBLICATION\_ON\_REQUEST 옵션을 지정함
- 넣기 조작에서 MQPMO\_NOT\_OWN\_SUBS 옵션이 지정되고 구독이 발행자의 ID와 일치하므로 발행이 전달되지 않음

**동기점 옵션.** 다음 옵션은 작업 단위 내에서 MQPUT 또는 MQPUT1 호출의 참여와 관련됩니다.

#### **MQPMO\_SYNCPOINT**

요청이 일반 작업 단위 프로토콜 내에서 조작됩니다. 작업 단위가 커밋될 때까지 작업 단위의 외부에는 메시지가 표시되지 않습니다. 작업 단위가 백아웃되면 메시지가 삭제됩니다.

MQPMO\_SYNCPOINT 및 MQPMO\_NO\_SYNCPOINT가 지정되지 않은 경우 작업 단위 프로토콜에서 넣기 요청의 포함은 애플리케이션을 실행 중인 환경이 아니라 큐 관리자를 실행 중인 환경에 의해 판별됩니다. z/OS에서 Put 요청은 작업 단위 내에 있습니다. 다른 모든 환경의 경우 Put 요청은 작업 단위 내에 있지 않습니다.

이러한 차이점 때문에 포팅하려는 애플리케이션은 이 옵션에 기본값을 허용하지 않아야 합니다.

MQPMO\_SYNCPOINT 또는 MQPMO\_NO\_SYNCPOINT를 명시적으로 지정하십시오.

MQPMO\_SYNCPOINT를 MQPMO\_NO\_SYNCPOINT와 함께 지정하지 마십시오.

#### **MQPMO\_NO\_SYNCPOINT**

요청은 일반 작업 단위 프로토콜의 외부에서 조작하는 것입니다. 메시지는 즉시 사용 가능하며 작업 단위를 백아웃하여 삭제할 수는 없습니다.

MQPMO\_NO\_SYNCPOINT 및 MQPMO\_SYNCPOINT가 지정되지 않은 경우 작업 단위 프로토콜에서 넣기 요청의 포함은 애플리케이션을 실행 중인 환경이 아니라 큐 관리자를 실행 중인 환경에 의해 판별됩니다. z/OS에서 Put 요청은 작업 단위 내에 있습니다. 다른 모든 환경의 경우 Put 요청은 작업 단위 내에 있지 않습니다.

이러한 차이점 때문에 포팅하려는 애플리케이션은 이 옵션에 기본값을 허용하지 않아야 합니다.

MQPMO\_SYNCPOINT 또는 MQPMO\_NO\_SYNCPOINT를 명시적으로 지정하십시오.

MQPMO\_NO\_SYNCPOINT를 MQPMO\_SYNCPOINT와 함께 지정하지 마십시오.

**메시지 메시지 및 상관 ID 옵션.** 다음 옵션은 큐 관리자가 새 메시지 ID 또는 연관 ID를 생성하도록 요청합니다.

#### **MQPMO\_NEW\_MSG\_ID**

큐 관리자는 MQMD에 있는 *MsgId* 필드의 콘텐츠를 새 메시지 ID로 바꿉니다. 이 메시지 ID는 메시지와 함께 송신되며 MQPUT 또는 MQPUT1 호출 출력 시 애플리케이션으로 리턴됩니다.

메시지를 분배 목록에 넣을 때 MQPMO\_NEW\_MSG\_ID 옵션을 지정할 수도 있습니다. 자세한 정보는 MQPMR 구조의 *MsgId* 필드에 대한 설명을 참조하십시오.

이 옵션을 사용하면 애플리케이션이 각 MQPUT 또는 MQPUT1 호출 전에 *MsgId* 필드를 MQMI\_NONE으로 재설정할 필요가 없습니다.

#### **MQPMO\_NEW\_CORREL\_ID**

큐 관리자가 MQMD에 있는 *CorrelId* 필드의 콘텐츠를 새 상관 ID로 바꿉니다. 이 상관 ID는 메시지와 함께 송신되며 MQPUT 또는 MQPUT1 호출 출력 시 애플리케이션으로 리턴됩니다.

메시지를 분배 목록에 넣을 때 MQPMO\_NEW\_CORREL\_ID 옵션을 지정할 수도 있습니다. 자세한 정보는 MQPMR 구조의 *CorrelId* 필드에 대한 설명을 참조하십시오.

MQPMO\_NEW\_CORREL\_ID는 애플리케이션에 고유한 상관 ID가 필요한 상황에서 유용합니다.

**그룹 및 세그먼트 옵션.** 다음 옵션은 논리 메시지의 그룹 및 세그먼트에서 메시지 처리와 관련됩니다. 다음 정의를 읽으면 옵션을 이해하는 데 도움이 됩니다.





**주의:** 발행/구독으로 세그먼트되거나 그룹화된 메시지를 사용할 수 없습니다.

### 실제 메시지

이는 큐에서 제거하거나 배치할 수 있는 가장 작은 정보 단위입니다. 이는 대개 단일 MQPUT, MQPUT1 또는 MQGET 호출에서 검색되거나 지정된 정보에 해당합니다. 모든 실제 메시지에는 자신의 메시지 디스크립터(MQMD)가 있습니다. 일반적으로 실제 메시지는 메시지 ID(MQMD의 *MsgId* 필드)에 대한 값을 서로 다르게 하여 구별되지만, 큐 관리자에 의해 강제 적용되지는 않습니다.

### 논리 메시지

논리 메시지는 biz/OS 플랫폼 전용의 단일 단위의 애플리케이션 정보입니다. 시스템 제한 조건이 없는 경우 논리 메시지는 실제 메시지와 동일합니다. 그러나 논리 메시지가 매우 큰 경우 시스템 제한 조건에 따라 논리 메시지를 세그먼트라는 둘 이상의 실제 메시지로 분할해야 할 수 있습니다.

하나의 세그먼트화된 논리 메시지는 널이 아닌 그룹 ID(MQMD의 *GroupId* 필드)와 메시지 순서 번호(MQMD의 *MsgSeqNumber* 필드)가 동일한 둘 이상의 실제 메시지로 구성됩니다. 세그먼트는 세그먼트 오프셋(MQMD의 *Offset* 필드)에 대한 값을 서로 다르게 하여 구별되며, 이 값은 논리 메시지의 데이터 시작부터의 실제 메시지의 데이터 오프셋을 제공합니다. 각 세그먼트는 실제 메시지이므로 논리 메시지의 세그먼트는 일반적으로 메시지 ID가 다릅니다.

세그먼트화되지 않았지만 전송하는 애플리케이션이 세그먼트화할 수 있는 논리 메시지에 널이 아닌 그룹 ID가 있습니다. 다만, 이 경우 논리 메시지가 메시지 그룹에 속하지 않으면 해당 그룹 ID의 실제 메시지가 하나뿐입니다. 논리 메시지가 메시지 그룹에 속하지 않는 경우 전송하는 애플리케이션이 세그먼트화할 수 없는 논리 메시지에는 널 그룹 ID(MQGI\_NONE)가 있습니다.

### 메시지 그룹

메시지 그룹은 널이 아닌 동일한 그룹 ID가 있는 하나 이상의 논리 메시지 세트입니다. 그룹의 논리 메시지는 메시지 시퀀스 번호의 값을 다르게 하여 구분되며 이 값은 1 - *n* 범위의 정수입니다(여기서 *n*은 그룹의 논리 메시지 수). 하나 이상의 논리 메시지가 세그먼트화된 경우 그룹에 *n*개를 초과하는 실제 메시지가 있습니다.

### MQPMO\_LOGICAL\_ORDER

이 옵션은 큐 관리자에게 애플리케이션이 논리 메시지의 그룹 및 세그먼트에 메시지를 넣는 방법을 알려줍니다. 이 옵션은 MQPUT 호출에만 지정할 수 있으며 MQPUT1 호출에서는 유효하지 않습니다.

MQPMO\_LOGICAL\_ORDER를 지정할 경우 애플리케이션이 후속 MQPUT 호출을 사용하여 다음을 수행합니다.

1. 증가하는 세그먼트 오프셋(0부터 시작) 순으로 간격 없이 세그먼트를 각 논리 메시지에 넣습니다.
2. 세그먼트를 다음 논리 메시지에 넣기 전에 모든 세그먼트를 하나의 논리 메시지에 넣습니다.
3. 증가하는 세그먼트 순서 번호(1부터 시작) 순으로 간격 없이 논리 메시지를 각 메시지 그룹에 넣습니다. IBM MQ에서는 메시지 순서 번호가 자동으로 증가됩니다.
4. 논리 메시지를 다음 메시지 그룹에 넣기 전에 모든 논리 메시지를 하나의 메시지 그룹에 넣습니다.

MQPMO\_LOGICAL\_ORDER에 대한 자세한 정보는 [논리 및 실제 순서 지정](#)을 참조하십시오.

**컨텍스트 옵션.** 다음 옵션은 메시지 컨텍스트 처리를 제어합니다.

### MQPMO\_NO\_CONTEXT

컨텍스트가 없음을 표시하기 위해 ID와 원본 컨텍스트를 둘 다 설정합니다. 즉, MQMD의 컨텍스트 필드가 다음과 같이 설정됩니다.

- 문자 필드의 경우 공백
- 바이트 필드의 경우 널
- 숫자 필드의 경우 0

### MQPMO\_DEFAULT\_CONTEXT

메시지는 ID와 원본 둘 다에 대해 연관된 기본 컨텍스트 정보를 갖게 됩니다. 큐 관리자는 메시지 디스크립터의 컨텍스트 필드를 다음과 같이 설정합니다.

표 508. MQMD 필드에 대한 기본 컨텍스트 정보 값

MQMD의 필드	사용된 값
<i>UserIdentifier</i>	가능한 경우 환경에 따라 결정됩니다. 그밖의 경우에는 공백으로 설정됩니다.



MQMD의 필드	사용된 값
<i>AccountingToken</i>	가능한 경우 환경에서 결정됩니다. 그렇지 않으면 MQACT_NONE으로 설정됩니다.
<i>ApplIdentityData</i>	공백으로 설정합니다.
<i>PutApplType</i>	환경에 따라 결정됩니다.
<i>PutAppName</i>	가능한 경우 환경에 따라 결정됩니다. 그밖의 경우에는 공백으로 설정됩니다.
<i>PutDate</i>	메시지를 넣은 날짜로 설정합니다.
<i>PutTime</i>	메시지를 넣은 시간으로 설정합니다.
<i>ApplOriginData</i>	공백으로 설정합니다.

메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트](#)를 참조하십시오.

이는 컨텍스트 옵션이 지정되지 않은 경우의 기본값 및 조치입니다.

#### MQPMO\_PASS\_IDENTITY\_CONTEXT

메시지에는 연관된 컨텍스트 정보가 있습니다. ID 컨텍스트는 *Context* 필드에 지정된 큐 핸들에서 가져옵니다. 원본 컨텍스트 정보는 MQPMO\_DEFAULT\_CONTEXT와 동일한 방식으로 큐 관리자가 생성합니다(값은 위의 표 참조). 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트](#)를 참조하십시오.

MQPUT 호출의 경우 큐는 MQOO\_PASS\_IDENTITY\_CONTEXT 옵션(또는 이를 암시하는 옵션)을 사용하여 열어야 합니다. MQPUT1 호출의 경우 MQOO\_PASS\_IDENTITY\_CONTEXT 옵션이 있는 MQOPEN 호출과 동일한 권한 검사가 수행됩니다.

#### MQPMO\_PASS\_ALL\_CONTEXT

메시지에는 연관된 컨텍스트 정보가 있습니다. *Context* 필드에 지정된 큐 핸들에서 컨텍스트를 가져옵니다. 메시지 컨텍스트에 대한 자세한 정보는 [컨텍스트 정보 제어](#)를 참조하십시오.

MQPUT 호출의 경우 큐는 MQOO\_PASS\_ALL\_CONTEXT 옵션(또는 이를 암시하는 옵션)을 사용하여 열어야 합니다. MQPUT1 호출의 경우 MQOO\_PASS\_ALL\_CONTEXT 옵션이 있는 MQOPEN 호출과 동일한 권한 검사가 수행됩니다.

#### MQPMO\_SET\_IDENTITY\_CONTEXT

메시지에는 연관된 컨텍스트 정보가 있습니다. 애플리케이션은 MQMD 구조에 ID 컨텍스트를 지정합니다. 원본 컨텍스트 정보는 MQPMO\_DEFAULT\_CONTEXT와 동일한 방식으로 큐 관리자가 생성합니다(값은 위의 표 참조). 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트](#)를 참조하십시오.

MQPUT 호출의 경우 큐는 MQOO\_SET\_IDENTITY\_CONTEXT 옵션(또는 이를 암시하는 옵션)을 사용하여 열어야 합니다. MQPUT1 호출의 경우 MQOO\_SET\_IDENTITY\_CONTEXT 옵션이 있는 MQOPEN 호출과 동일한 권한 검사가 수행됩니다.

#### MQPMO\_SET\_ALL\_CONTEXT

메시지에는 연관된 컨텍스트 정보가 있습니다. 애플리케이션은 MQMD 구조에서 ID, 원본, 사용자 컨텍스트를 지정합니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트](#)를 참조하십시오.

MQPUT 호출의 경우 큐는 MQOO\_SET\_ALL\_CONTEXT 옵션을 사용하여 열어야 합니다. MQPUT1 호출의 경우 MQOO\_SET\_ALL\_CONTEXT 옵션이 있는 MQOPEN 호출과 동일한 권한 검사가 수행됩니다.

MQPMO\*\_CONTEXT 컨텍스트 옵션 중 하나만 지정할 수 있습니다. 지정하지 않으면 MQPMO\_DEFAULT\_CONTEXT로 가정합니다.

**특성 옵션.** 다음 옵션은 메시지의 특성과 관련됩니다.

#### MQPMO\_MD\_FOR\_OUTPUT\_ONLY

메시지 디스크립터 매개변수는 넣은 메시지의 메시지 디스크립터를 리턴하는 출력에만 사용되어야 합니다. MQPMO 구조의 *NewMsgHandle*, *OriginalMsgHandle* 또는 두 필드 모두와 연관된 메시지 디스크립터 필드가 입력에 사용되어야 합니다.

올바른 메시지 핸들이 제공되지 않으면 이유 코드 MQRC\_MD\_ERROR와 함께 호출이 실패합니다.

**응답 넣기 옵션.** 다음 옵션은 MQPUT 또는 MQPUT1 호출로 리턴되는 응답을 제어합니다. 이러한 옵션 중 하나만 지정할 수 있습니다. MQPMO\_ASYNC\_RESPONSE 및 MQPMO\_SYNC\_RESPONSE가 지정되지 않으면 MQPMO\_RESPONSE\_AS\_Q\_DEF 또는 MQPMO\_RESPONSE\_AS\_TOPIC\_DEF로 가정합니다.

### MQPMO\_ASYNC\_RESPONSE

MQPMO\_ASYNC\_RESPONSE 옵션은 애플리케이션이 큐 관리자가 호출을 완료하기를 기다리지 않고 MQPUT 또는 MQPUT1 조작이 완료되도록 요청합니다. 이 옵션을 사용하여 특히, 클라이언트 바인딩을 사용하는 애플리케이션의 경우 메시징 성능을 향상시킬 수 있습니다. 애플리케이션은 MQSTAT verb를 사용하여 이전 비동기 호출 중에 오류가 발생했는지 주기적으로 확인할 수 있습니다.

이 옵션을 사용하면 MQMD에서 다음 필드만 완료됩니다.

- ApplIdentityData
- PutApplType
- PutApplName
- ApplOriginData

또한 MQPMO\_NEW\_MSG\_ID나 MQPMO\_NEW\_CORREL\_ID 중 하나 또는 둘 다가 옵션으로 지정된 경우 리턴되는 MsgId 및 CorrelId도 완료됩니다. (MQPMO\_NEW\_MSG\_ID는 공백 MsgId 필드를 지정하여 암시적으로 지정할 수 있습니다.)

이전에 지정한 필드만 완료됩니다. 일반적으로 MQMD 또는 MQPMO 구조에서 리턴되는 기타 정보는 정의되지 않습니다.

MQPUT1에 대해 비동기 넣기 응답을 요청하는 경우 MQOD 구조에서 리턴된 ResolvedQName 및 ResolvedQMGrName은 정의되지 않습니다.

MQPUT 또는 MQPUT1에 대한 비동기 넣기 응답을 요청하는 경우, CompCode와 MQCC\_OK 및 MQRC\_NONE의 이유가 반드시 큐에 메시지가 성공적으로 넣어졌음을 의미하지는 않습니다. 비동기 넣기 응답을 사용하는 MQI 애플리케이션을 개발 중이고 메시지를 큐에 넣었다는 확인이 필요한 경우 넣기 조작의 완료 코드 및 이유 코드를 둘 다 확인하고 MQSTAT를 사용하여 비동기 오류 정보도 조회해야 합니다.

각 개별 MQPUT 또는 MQPUT1 호출의 성공 또는 실패는 즉시 리턴되지 않을 수 있지만 비동기 호출에서 발생한 첫 번째 오류는 나중에 MQSTAT 호출을 통해 판별할 수 있습니다.

비동기 넣기 응답을 사용하여 동기점 아래의 지속 메시지 전달에 실패하고 트랜잭션을 커밋하려고 시도하는 경우 커밋이 실패하며 트랜잭션은 완료 코드 MQCC\_FAILED 및 이유 코드 MQRC\_BACKED\_OUT와 함께 백아웃됩니다. 애플리케이션은 이전 MQPUT 또는 MQPUT1 실패의 원인을 판별하기 위해 MQSTAT를 호출할 수 있습니다.

### MQPMO\_SYNC\_RESPONSE

이 넣기 응답 유형을 지정하면 MQPUT 또는 MQPUT1 조작이 항상 동기적으로 실행됩니다. 넣기 조작이 성공하는 경우 MQMD 및 MQPMO의 모든 필드가 완료됩니다.

이 옵션을 사용하면 동기 응답이 큐 또는 주제 오브젝트에 정의된 기본 넣기 응답 값과 무관함을 확인할 수 있습니다.

### MQPMO\_RESPONSE\_AS\_Q\_DEF

MQPUT 호출에 대해 이 값이 지정된 경우 사용되는 넣기 응답 유형은 애플리케이션에서 처음 열 때 큐에 지정된 DEFPRESP 값에서 가져옵니다.

- 큐가 클러스터 큐이고 이 값이 MQPUT 호출에 지정된 경우, 메시지가 있는 큐의 특정 인스턴스를 소유하는 목적지 큐 관리자에 정의된 **DEFPRESP** 속성에서 사용되는 넣기 응답 유형을 가져옵니다.

클러스터 큐의 인스턴스가 여러 개 있는데 이들의 속성이 서로 다를 경우, 이 중 하나의 값이 선택되며 어떤 값이 사용될지 예측할 수는 없습니다. 따라서 이 속성을 모든 인스턴스에서 동일한 값으로 설정해야 합니다. 그렇지 않을 경우 큐 관리자 로그에 오류 메시지 AMQ9407이 발행됩니다. 별명, 리모트 및 클러스터 큐에 대해 목적지 오브젝트 속성을 해석하는 방법도 참조하십시오.

- 큐가 클러스터 큐가 아니고 이 값이 MQPUT 호출에 지정된 경우, 목적지 큐 관리자가 리모트인 경우에도 로컬 큐 관리자에 정의된 **DEFPRESP** 속성에서 사용되는 넣기 응답 유형을 가져옵니다.

클라이언트 애플리케이션이 IBM WebSphere MQ 7.0 이전의 큐 관리자에 연결되는 경우 MQPMO\_SYNC\_RESPONSE가 지정된 것처럼 작동합니다.

MQPUT1 호출에 대해 이 옵션이 지정된 경우 DEFPRESP 속성의 값은 요청이 서버로 전송되기 전에는 알 수 없습니다. 기본적으로 MQPUT1 호출이 MQPMO\_SYNCPOINT를 사용 중인 경우

MQPMO\_ASYNC\_RESPONSE의 경우와 같이 작동하고 MQPMO\_NO\_SYNCPOINT를 사용 중인 경우에는 MQPMO\_SYNC\_RESPONSE의 경우와 같이 작동합니다. 그러나 클라이언트 구성 파일에서 Put1DefaultAlwaysSync 특성을 설정하여 이 기본 작동을 대체할 수 있습니다. 클라이언트 구성 파일의 CHANNELS 스탠자를 참조하십시오.

#### **MQPMO\_RESPONSE\_AS\_TOPIC\_DEF**

MQPMO\_RESPONSE\_AS\_TOPIC\_DEF는 주제 오브젝트에 대해 사용하는 경우 MQPMO\_RESPONSE\_AS\_Q\_DEF의 동의어입니다.

**기타 옵션.** 다음 옵션은 권한 검사, 큐 관리자가 정지 중일 때 발생하는 사항, 큐 및 큐 관리자 이름 해석을 제어합니다.

#### **MQPMO\_ALTERNATE\_USER\_AUTHORITY**

MQPMO\_ALTERNATE\_USER\_AUTHORITY는 MQPUT1 호출의 **ObjDesc** 매개변수에 있는 *AlternateUserId* 필드에 큐에 메시지를 넣을 권한을 유효성 검증하는 데 사용되는 사용자 ID가 포함되어 있음을 표시합니다. 애플리케이션이 실행 중인 사용자 ID에 권한이 부여되었는지 여부에 관계없이 지정된 옵션으로 큐를 열 수 있는 권한이 *AlternateUserId*에 부여된 경우에만 호출이 성공할 수 있습니다. (지정된 컨텍스트 옵션에는 적용되지 않지만 애플리케이션을 실행하는 사용자 ID에 대해서는 항상 확인됩니다.)

이 옵션은 MQPUT1 호출과 함께 사용하는 경우에만 유효합니다.

#### **MQPMO\_FAIL\_IF QUIESCING**

이 옵션은 큐 관리자가 정지 중인 경우 MQPUT 또는 MQPUT1 호출을 실패하게 합니다.

z/OS에서 이 옵션은 또한 CICS 또는 IMS 애플리케이션에 대한 연결이 정지 상태인 경우 MQPUT 또는 MQPUT1 호출을 실패하게 합니다.

호출은 완료 코드 MQCC\_FAILED와 이유 코드 MQRC\_Q\_MGR QUIESCING 또는 MQRC\_CONNECTION QUIESCING을 리턴합니다.

#### **MQPMO\_RESOLVE\_LOCAL\_Q**

MQPMO 구조의 *ResolvedQName*을(를) 메시지가 넣은 로컬 큐의 이름으로 채우고 *ResolvedQMgrName*을(를) 로컬 큐를 호스팅하는 로컬 큐 관리자의 이름으로 채우려면 이 옵션을 사용하십시오. MQPMO\_RESOLVE\_LOCAL\_Q에 대한 자세한 정보는 MQOO\_RESOLVE\_LOCAL\_Q 토픽을 참조하십시오.

큐에 넣을 권한이 부여된 경우 MQPUT 호출에서 이 플래그를 지정하는 데 필요한 권한이 있습니다. 특수 권한은 필요하지 않습니다.

**기본 옵션.** 설명된 옵션 중 필요한 옵션이 없는 경우 다음 옵션을 사용할 수 있습니다.

#### **MQPMO\_NONE**

기타 옵션이 지정되지 않았음을 표시하려면 이 값을 사용하십시오. 모든 옵션은 기본 값을 가정합니다. MQPMO\_NONE은 프로그램 문서화를 돕기 위해 정의되었습니다. 이 옵션은 다른 옵션과 함께 사용하도록 의도되지 않았지만 값이 0인 경우에는 그러한 사용을 발견할 수 없습니다.

MQPMO\_NON은 입력 필드입니다. *Options* 필드의 초기값은 MQPMO\_NONE입니다.

#### **MQPMO의 제한시간 (MQLONG)**

이 필드는 예약 필드이며 값은 중요하지 않습니다. 이 필드의 초기값은 -1입니다.

#### **MQPMO의 컨텍스트 (MQHOBJ)**

MQPMO\_PASS\_IDENTITY\_CONTEXT 또는 MQPMO\_PASS\_ALL\_CONTEXT가 지정되면 이 필드는 넣는 메시지와 연관되는 컨텍스트 정보를 가져오는 입력 큐 핸들을 포함해야 합니다.

MQPMO\_PASS\_IDENTITY\_CONTEXT 및 MQPMO\_PASS\_ALL\_CONTEXT 모두 지정되지 않으면 이 필드는 무시됩니다.

입력 필드입니다. 이 필드의 초기값은 0입니다.

#### **KnownDestMQPMO의 수 (MQLONG)**

이는 현재 MQPUT 또는 MQPUT1 호출에서 로컬 큐인 분배 목록의 큐에 송신한 메시지의 수입입니다. 이 수는 리모트 큐로 분석되는 큐에 송신된 메시지는 포함하지 않습니다(초기에 메시지를 저장하는 데 로컬 전송 큐가 사용된 경우라도). 이 필드는 분배 목록에 없는 단일 큐에 메시지를 넣는 경우에도 설정됩니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 0입니다. *Version* 이 MQPMO\_VERSION\_1미만이면 이 필드가 설정되지 않습니다.

분배 목록이 지원되지 않아 이 필드가 z/OS에서 정의되지 않았습니다.

### **MQPMO에 대한 UnknownDest수 (MQLONG)**

이는 현재 MQPUT 또는 MQPUT1 호출에서 리모트 큐로 분석되는 분배 목록의 큐에 송신한 메시지의 수입입니다. 큐 관리자가 분배 목록 양식에 임의로 보유하는 메시지는 해당 분배 목록에 포함되어 있는 개별 목적지의 수로 간주됩니다. 이 필드는 분배 목록에 없는 단일 큐에 메시지를 넣는 경우에도 설정됩니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 0입니다. *Version* 이 MQPMO\_VERSION\_1미만이면 이 필드가 설정되지 않습니다.

분배 목록이 지원되지 않아 이 필드가 z/OS에서 정의되지 않았습니다.

### **MQPMO에 대한 InvalidDest수 (MQLONG)**

이는 분배 목록의 큐에 송신할 수 없는 메시지의 수입입니다. 이 수에는 여는 데 실패한 큐와 성공적으로 열었지만 넣기 조작이 실패한 큐가 포함됩니다. 이 필드는 분배 목록에 없는 단일 큐에 메시지를 넣는 경우에도 설정됩니다.

**참고:** MQPUT 또는 MQPUT1 호출의 **CompCode** 매개변수가 MQCC\_OK 또는 MQCC\_WARNING인 경우 이 필드가 설정되며 **CompCode** 매개변수가 MQCC\_FAILED인 경우 이 필드가 설정될 수도 있지만 애플리케이션 코드에서는 여기에 의존하지 않습니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 0입니다. *Version* 이 MQPMO\_VERSION\_1미만이면 이 필드가 설정되지 않습니다.

분배 목록이 지원되지 않아 이 필드가 z/OS에서 정의되지 않았습니다.

### **MQPMO의 경우 ResolvedQName (MQCHAR48)**

이는 이름 해석이 로컬 큐 관리자에 의해 수행된 이후 목적지 큐의 이름입니다. 리턴되는 이름은 *ResolvedQMgrName*(으)로 식별되는 큐 관리자에 존재하는 큐의 이름입니다.

오브젝트가 단일 큐인 경우에만 공백이 아닌 값이 리턴됩니다. 오브젝트가 분배 목록이거나 주제인 경우 리턴된 값이 정의되지 않습니다.

이 필드는 출력 필드입니다. 이 필드의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백입니다.

### **MQPMO에 대한 ResolvedQMgr이름 (MQCHAR48)**

이는 이름 해석이 로컬 큐 관리자에 의해 수행된 이후 목적지 큐 관리자의 이름입니다. 리턴되는 이름은 *ResolvedQName*(으)로 식별되는 큐를 소유하는 큐 관리자의 이름이며 로컬 큐 관리자의 이름이 될 수 있습니다.

*ResolvedQName*이(가) 로컬 큐 관리자가 속한 큐 공유 그룹이 소유하는 공유 큐인 경우, *ResolvedQMgrName*은(는) 큐 공유 그룹의 이름입니다. 다른 큐 공유 그룹이 큐를 소유하는 경우, *ResolvedQName*은(는) 큐 공유 그룹의 이름이거나 큐 공유 그룹의 구성원인 큐 관리자의 이름이 될 수 있습니다(리턴되는 값의 특성은 로컬 큐 관리자에 존재하는 큐 정의에 의해 결정됩니다).

오브젝트가 단일 큐인 경우에만 공백이 아닌 값이 리턴됩니다. 오브젝트가 분배 목록이거나 주제인 경우 리턴된 값이 정의되지 않습니다.

이 필드는 출력 필드입니다. 이 필드의 길이는 MQ\_Q\_MGR\_NAME\_LENGTH로 제공됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백입니다.

### **MQPMO의 경우 RecsPresent (MQLONG)**

애플리케이션에서 제공하는 MQPMR Put 메시지 레코드 또는 MQRR 응답 레코드의 수입니다. 이 숫자는 메시지가 분배 목록에 넣기 되는 경우에만 0보다 클 수 있습니다. 넣기 메시지 레코드 및 응답 레코드는 선택사항입니다. 애플리케이션이 레코드를 제공할 필요가 없거나 하나의 유형의 레코드만 제공하도록 선택할 수 있습니다. 그러나 애플리케이션이 두 유형 모두의 레코드를 제공하는 경우 각 유형의 *RecsPresent* 레코드를 제공해야 합니다.

*RecsPresent*의 값은 분배 목록의 목적지 수와 동일할 필요는 없습니다. 너무 많은 레코드가 제공되면 초과량은 사용되지 않습니다. 너무 적은 레코드가 제공되면 넣기 메시지 레코드가 없는 목적지의 메시지 특성에 기본값이 사용됩니다(*PutMsgRecOffset* 참조).

*RecsPresent*가 0 미만이거나 0보다 크지만 분배 목록에 메시지를 넣지 않는 경우 이유 코드 MQRC\_RECS\_PRESENT\_ERROR와 함께 호출이 실패합니다.

입력 필드입니다. 이 필드의 초기값은 0입니다. *Version*이(가) MQPMO\_VERSION\_2보다 작으면 이 필드는 무시됩니다.

### **MQPMO의 PutMsgRecFields (MQLONG)**

이 필드는 MQPMR 필드가 애플리케이션이 제공하는 넣기 메시지 레코드에 있음을 표시하는 플래그를 포함합니다. 메시지를 분배 목록에 넣는 경우에만 *PutMsgRecFields*를 사용하십시오. *RecsPresent*이(가) 0이거나 *PutMsgRecOffset* 및 *PutMsgRecPtr* 둘 다 0인 경우 필드는 무시됩니다.

존재하는 필드의 경우 큐 관리자는 각 목적지에 대해 해당하는 Put 메시지 레코드에 있는 필드의 값을 사용합니다. 없는 필드의 경우 큐 관리자는 MQMD 구조의 값을 사용합니다.

다음 플래그 중 하나 이상을 사용하여 필드가 넣기 메시지 레코드에 있음을 표시하십시오.

#### **MQPMRF\_MSG\_ID**

메시지 ID 필드가 존재합니다.

#### **MQPMRF\_CORREL\_ID**

상관 ID 필드가 존재합니다.

#### **MQPMRF\_GROUP\_ID**

그룹 ID 필드가 존재합니다.

#### **MQPMRF\_FEEDBACK**

피드백 필드가 존재합니다.

#### **MQPMRF\_ACCOUNTING\_TOKEN**

Accounting-token 필드가 존재합니다.

이 플래그를 지정하는 경우 *Options* 필드에서 MQPMO\_SET\_IDENTITY\_CONTEXT 또는 MQPMO\_SET\_ALL\_CONTEXT를 지정하십시오. 이 조건이 충족되지 않는 경우 이유 코드 MQRC\_PMO\_RECORD\_FLAGS\_ERROR와 함께 호출이 실패합니다.

MQPMR 필드가 없는 경우 다음을 지정할 수 있습니다.

#### **MQPMRF\_NONE**

넣기 메시지 레코드 필드가 없습니다.

이 값이 지정된 경우, *RecsPresent*은(는) 0이거나, *PutMsgRecOffset* 및 *PutMsgRecPtr* 모두 0이어야 합니다.

MQPMRF\_NONE은 프로그램 문서화를 지원하기 위해 정의됩니다. 이 상수는 다른 상수와 함께 사용하기 위한 용도는 아니지만, 값이 0이므로 그러한 사용을 감지할 수 없습니다.

*PutMsgRecFields*가 올바르지 않은 플래그를 포함하거나 넣기 메시지 레코드가 제공되지만 *PutMsgRecFields*에 값 MQPMRF\_NONE이 있는 경우 이유 코드 MQRC\_PMO\_RECORD\_FLAGS\_ERROR와 함께 호출이 실패합니다.

입력 필드입니다. 이 필드의 초기값은 MQPMRF\_NONE입니다. *Version*이(가) MQPMO\_VERSION\_2보다 작으면 이 필드는 무시됩니다.

### **PutMsgMQPMO의RecOffset (MQLONG)**



이는 MQPMO 구조의 시작에서 첫 번째 MQPMR put 메시지 레코드의 오프셋(바이트)입니다. 오프셋은 양수 또는 음수일 수 있습니다. *PutMsgRecOffset*은(는) 메시지가 분배 목록에 놓이는 경우에만 사용됩니다. *RecsPresent*이(가) 0인 경우 필드는 무시됩니다.

메시지를 분배 목록에 넣을 때 개별적으로 각 목적지의 메시지에 대한 특정 특성을 지정하기 위해 하나 이상의 MQPMR 넣기 메시지 레코드의 배열을 제공할 수 있습니다. 이러한 특성은 다음과 같습니다.

- 메시지 ID
- 상관 ID
- 그룹 ID
- 피드백 값
- 계정 토큰

이러한 모든 특성을 지정할 필요는 없지만 어떤 서브세트를 선택하든지 간에 필드를 올바른 순서로 지정하십시오. 자세한 정보는 MQPMR 구조에 대한 설명을 참조하십시오.

분배 목록을 열 때 일반적으로 MQOD가 지정된 오브젝트 레코드만큼의 넣기 메시지 레코드가 있어야 합니다. 각 넣기 메시지 레코드는 해당 오브젝트 레코드가 식별한 큐의 메시지 특성을 제공합니다. 이 경우 메시지 특성이 무시되어도 여는 데 실패한 분배 목록의 큐에는 배열에서 적절한 위치에서 할당된 넣기 메시지 레코드가 있어야 합니다.

넣기 메시지 레코드의 수는 오브젝트 레코드의 수와 다를 수 있습니다. 오브젝트 레코드보다 적은 넣기 메시지 레코드가 있는 경우 넣기 메시지 레코드가 없는 목적지의 메시지 특성을 메시지 디스크립터 MQMD의 해당 필드에서 가져옵니다. 오브젝트 레코드보다 Put 메시지 레코드가 많은 경우 초과분은 사용되지 않습니다(여전히 액세스해야 하는 경우에도). 넣기 메시지 레코드는 선택사항이지만 제공되는 경우 *RecsPresent*이(가) 있어야 합니다.

*PutMsgRecOffset*에서 오프셋을 지정하거나 *PutMsgRecPtr*에 주소를 지정하여 MQOD의 오브젝트 레코드와 유사한 방법으로 넣기 메시지 레코드를 제공할 수 있습니다. 이를 수행하는 방법에 대한 자세한 내용은 466 페이지의 『MQOD - 오브젝트 디스크립터』에 설명된 *ObjectRecOffset* 필드를 참조하십시오.

*PutMsgRecOffset* 및 *PutMsgRecPtr* 중 하나만 사용할 수 있습니다. 둘 다 0이 아닌 경우 이유 코드 MQRC\_PUT\_MSG\_RECORDS\_ERROR와 함께 호출이 실패합니다.

입력 필드입니다. 이 필드의 초기값은 0입니다. *Version*이(가) MQPMO\_VERSION\_2보다 작으면 이 필드는 무시됩니다.

## ResponseRecMQPMO에 대한 오프셋 (MQLONG)

이는 MQPMO 구조의 시작에서 첫 번째 MQRR 응답 레코드의 오프셋(바이트)입니다. 오프셋은 양수 또는 음수일 수 있습니다. *ResponseRecOffset*은(는) 메시지가 분배 목록에 놓이는 경우에만 사용됩니다. *RecsPresent*이(가) 0인 경우 필드는 무시됩니다.

분배 목록에 메시지를 넣을 때 메시지가 전송되지 않은 큐(MQRR의 *CompCode* 필드) 및 각 실패에 대한 이유(MQRR의 *Reason* 필드)를 식별하기 위해 하나 이상의 MQRR 응답 레코드의 배열을 제공할 수 있습니다. 큐를 여는 데 실패했거나 Put 조작이 실패했기 때문에 메시지가 송신되지 않았을 수 있습니다. 큐 관리자는 호출의 결과가 혼합된 경우에만 응답 레코드를 설정합니다(즉, 기타 메시지가 실패한 동안 일부 메시지가 성공적으로 전송된 또는 다른 이유로 인해 모두 실패함). 호출에서 이유 코드 MQRC\_MULTIPLE\_REASONS가 이 경우를 표시합니다. 동일한 이유 코드가 모든 큐에 적용되면 해당 이유가 MQPUT 또는 MQPUT1 호출의 **Reason** 매개변수에 리턴되고 응답 레코드가 설정되지 않습니다.

일반적으로 분배 목록이 열려 있을 때 MQOD가 지정한 오브젝트 레코드 수만큼의 응답 레코드가 있으며 필요한 경우 각 응답 레코드는 해당 오브젝트 레코드가 식별한 큐에 넣기를 위해 완료 코드 및 이유 코드로 설정됩니다. 여는 데 실패한 분배 목록의 큐가 넣기 조작이 아닌 열기 조작으로 인해 발생한 완료 코드 및 이유 코드로 설정된 경우에도 이러한 큐에는 배열의 적절한 위치에서 응답 레코드가 계속 할당되어 있어야 합니다.

응답 레코드의 수는 오브젝트 레코드의 수와 다를 수 있습니다. 응답 레코드 수가 오브젝트 레코드 수보다 적은 경우 애플리케이션이 넣기 조작이 실패한 모든 대상 또는 실패에 대한 이유를 식별하지 못할 수도 있습니다. 오브젝트 레코드보다 응답 레코드가 많은 경우 초과분은 사용되지 않습니다(여전히 액세스해야 하는 경우에도). 응답 레코드는 선택사항이지만 제공되는 경우 *RecsPresent*이어야 합니다.

*ResponseRecOffset*에서 오프셋을 지정하거나 *ResponseRecPtr*에서 주소를 지정하여 MQOD의 오브젝트 레코드에 유사한 방법으로 응답 레코드를 제공하십시오. 이를 수행하는 방법에 대한 세부사항은 466 페이지의 『MQOD - 오브젝트 디스크립터』에서 설명하는 *ObjectRecOffset* 필드를 참조하십시오. 그러나 *ResponseRecOffset* 및 *ResponseRecPtr* 중 둘 이상을 사용하지 마십시오. 둘 모두 0이 아닌 경우 오류 코드 MQRC\_RESPONSE\_RECORDS\_ERROR와 함께 호출이 실패합니다.

MQPUT1 호출의 경우 이 필드는 0이어야 합니다. 이는 응답 정보(필요하면)가 오브젝트 디스크립터 MQOD에서 지정한 응답 레코드에서 리턴되기 때문입니다.

입력 필드입니다. 이 필드의 초기값은 0입니다. *Version*이(가) MQPMO\_VERSION\_2보다 작으면 이 필드는 무시됩니다.

### **PutMsgMQPMO의RecPtr (MQPTR)**

첫 번째 MQPMR Put 메시지 레코드의 주소입니다. 메시지를 분배 목록에 넣는 경우에만 *PutMsgRecPtr*을 사용하십시오. *RecsPresent*이(가) 0인 경우 필드는 무시됩니다.

*PutMsgRecPtr* 또는 *PutMsgRecOffset* 중 하나만 사용하여 넣기 메시지 레코드를 지정할 수 있습니다. 세부사항은 497 페이지의 『PutMsgMQPMO의RecOffset (MQLONG)』의 내용을 참조하십시오. *PutMsgRecPtr*을 사용하지 않는 경우 이를 널 포인터 또는 널 바이트로 설정하십시오.

입력 필드입니다. 이 필드의 초기값은 포인터를 지원하는 프로그래밍 언어로 된 널 포인터이며, 그렇지 않은 경우 모두 널 바이트 문자열입니다. *Version*이(가) MQPMO\_VERSION\_2보다 작으면 이 필드는 무시됩니다.

**참고:** 프로그래밍 언어가 포인터 데이터 유형을 지원하지 않는 플랫폼에서 이 필드는 적절한 길이의 바이트 문자열로 선언되며, 초기값은 모두 널 바이트 문자열입니다.

### **MQPMO의 경우 ResponseRecPtr (MQPTR)**

이는 첫 번째 MQRR 응답 레코드의 주소입니다. *ResponseRecPtr*은(는) 메시지가 분배 목록에 놓이는 경우에만 사용됩니다. *RecsPresent*이(가) 0인 경우 필드는 무시됩니다.

*ResponseRecPtr* 또는 *ResponseRecOffset* 중 하나만 사용하여 응답 레코드를 지정하십시오. 세부사항은 498 페이지의 『ResponseRecMQPMO에 대한 오프셋 (MQLONG)』의 내용을 참조하십시오. *ResponseRecPtr*을 사용하지 않는 경우 이를 널 포인터 또는 널 바이트로 설정하십시오.

MQPUT1 호출의 경우 이 필드는 널 포인터 또는 널 바이트여야 합니다. 이는 응답 정보(필요하면)가 오브젝트 디스크립터 MQOD에서 지정한 응답 레코드에서 리턴되기 때문입니다.

입력 필드입니다. 이 필드의 초기값은 포인터를 지원하는 프로그래밍 언어로 된 널 포인터이며, 그렇지 않은 경우 모두 널 바이트 문자열입니다. *Version*이(가) MQPMO\_VERSION\_2보다 작으면 이 필드는 무시됩니다.

**참고:** 프로그래밍 언어가 포인터 데이터 유형을 지원하지 않는 플랫폼에서 이 필드는 적절한 길이의 바이트 문자열로 선언되며, 초기값은 모두 널 바이트 문자열입니다.

### **OriginalMsgMQPMO의 핸들 (MQHMSG)**

이는 메시지에 대한 선택적 핸들입니다. 이는 이전에 큐에서 검색되었을 수 있습니다. 이 핸들의 사용은 *Action* 필드의 값에 종속됩니다. *NewMsgHandle*도 참조하십시오.

원래 메시지 핸들의 콘텐츠는 MQPUT or MQPUT1 호출로 대체되지 않습니다.

입력 필드입니다. 이 필드의 초기값은 MQHM\_NONE입니다. 버전이 MQPMO\_VERSION\_3 미만인 경우 이 필드는 무시됩니다.

### **NewMsgMQPMO의 핸들 (MQHMSG)**

이는 Action 필드의 값에 따라 넣는 메시지의 선택적 핸들입니다. 지정되면 이는 메시지의 특성을 정의하고 *OriginalMsgHandle*의 값을 대체합니다.

MQPUT 또는 MQPUT1 호출에서 리턴 시 핸들의 콘텐츠는 실제로 넣은 메시지를 반영합니다.

입력 필드입니다. 이 필드의 초기값은 **MQHM\_NONE**입니다. 버전이 **MQPMO\_VERSION\_3** 미만인 경우 이 필드는 무시됩니다.

### **MQPMO에 대한 조치 (MQLONG)**

이는 OriginalMsgHandle 필드에서 지정된 원래 메시지 및 NewMsgHandle 필드에서 지정된 새 메시지 간의 관계 및 수행되는 넣기의 유형을 지정합니다. 메시지의 특성은 지정된 조치의 값에 따라 큐 관리자가 선택합니다.

MQPUT 또는 MQPUT1 호출에서 MsgDesc 매개변수를 사용하여 메시지 디스크립터의 콘텐츠를 제공하도록 선택할 수 있습니다. 또는 MQPMO 구조의 옵션 필드에서 MQPMO\_MD\_FOR\_OUTPUT\_ONLY를 포함하여 출력 전용임을 지정하거나 MsgDesc 매개변수를 제공할 수 없습니다.

MsgDesc 매개변수가 제공되지 않거나 출력 전용으로 지정되는 경우 새 메시지의 메시지 디스크립터가 이 토폰에서 설명하는 규칙에 따라 MQPMO의 메시지 핸들 필드에서 채워집니다.

컨텍스트 정보 제어에서 설명하는 컨텍스트 설정 및 전달 활동은 메시지 디스크립터가 작성된 후에 적용됩니다.

올바르지 않은 조치 값이 지정되면 이유 코드 MQRC\_ACTION\_ERROR와 함께 호출이 실패합니다.

다음 조치 중 하나를 지정할 수 있습니다.

#### **MQACTP\_NEW**

새 메시지를 넣고 프로그램이 이전 메시지에 대한 관계를 지정하지 않습니다. 메시지 디스크립터는 다음과 같이 구성됩니다.

- MsgDesc가 MQPUT 또는 MQPUT1 호출에서 제공되며 MQPMO\_MD\_FOR\_OUTPUT\_ONLY가 MQPMO.Options에 없는 경우, 이는 수정되지 않은 메시지 디스크립터로 사용됩니다.
- MsgDesc가 제공되지 않거나 MQPMO.Options에 MQPMO\_MD\_FOR\_OUTPUT\_ONLY가 있는 경우에는 큐 관리자가 OriginalMsgHandle 및 NewMsgHandle에서 특성 조합을 사용하여 메시지 디스크립터를 생성합니다. 새 메시지 핸들에 명시적으로 설정된 메시지 디스크립터 필드는 원래 메시지 핸들의 필드보다 우선권을 갖습니다.

메시지 데이터는 MQPUT 또는 MQPUT1 버퍼 매개변수에서 가져옵니다.

#### **MQACTP\_FORWARD**

이전에 검색된 메시지가 전달됩니다. 원래 메시지 핸들은 이전에 검색된 메시지를 지정합니다.

새 메시지 핸들은 원래 메시지 핸들에서 특성에 수정사항을 지정합니다(메시지 디스크립터의 사항 포함).

메시지 디스크립터는 다음과 같이 구성됩니다.

- MsgDesc가 MQPUT 또는 MQPUT1 호출에서 제공되며 MQPMO\_MD\_FOR\_OUTPUT\_ONLY가 MQPMO.Options에 없는 경우, 이는 수정되지 않은 메시지 디스크립터로 사용됩니다.
- MsgDesc가 제공되지 않거나 MQPMO.Options에 MQPMO\_MD\_FOR\_OUTPUT\_ONLY가 있는 경우에는 큐 관리자가 OriginalMsgHandle 및 NewMsgHandle에서 특성 조합을 사용하여 메시지 디스크립터를 생성합니다. 새 메시지 핸들에 명시적으로 설정된 메시지 디스크립터 필드는 원래 메시지 핸들의 필드보다 우선권을 갖습니다.
- MQPMO\_NEW\_MSG\_ID 또는 MQPMO\_NEW\_CORREL\_ID가 MQPMO.Options에서 지정되면 준수됩니다.

메시지 특성은 다음과 같이 작성됩니다.

- MQPD.CopyOptions에서 MQCOPY\_FORWARD가 있는 원래 메시지 핸들의 모든 특성
- 새 메시지 핸들의 모든 특성. 원래 메시지 핸들의 특성과 동일한 이름이 있는 새 메시지 핸들에 있는 각 특성의 경우 값은 새 메시지 핸들에서 가져옵니다. 새 메시지 핸들의 특성에 원래 메시지 핸들의 특성과 동일한 이름이 있는 경우 특성의 값이 널인 경우 이 규칙에 대한 예외만 특수 경우입니다. 이 경우 특성은 메시지에서 제거됩니다.

전달되는 메시지 데이터는 MQPUT 또는 MQPUT1 버퍼 매개변수에서 가져옵니다.

#### **MQACTP\_REPLY**

이전에 검색된 메시지에 대해 응답이 작성됩니다. 원래 메시지 핸들은 이전에 검색된 메시지를 지정합니다.

새 메시지 핸들은 원래 메시지 핸들에서 특성에 수정사항을 지정합니다(메시지 디스크립터의 사항 포함).

메시지 디스크립터는 다음과 같이 구성됩니다.



- MsgDesc가 MQPUT 또는 MQPUT1 호출에서 제공되며 MQPMO\_MD\_FOR\_OUTPUT\_ONLY가 MQPMO.Options에 없는 경우, 이는 수정되지 않은 메시지 디스크립터로 사용됩니다.
- MsgDesc를 제공하지 않거나 MQPMO\_MD\_FOR\_OUTPUT\_ONLY가 MQPMO.Options에 있는 경우에는 초기 메시지 디스크립터 필드가 다음과 같이 선택됩니다.

표 509. 응답 메시지 핸들 변환	
MQMD의 필드	사용된 값
보고서	MQRO_PASS_DISCARD_AND_EXPIRY 및 MQRO_DISCARD_MSG가 설정되는 경우: MQRO_DISCARD_MSG 그렇지 않으면 MQRO_NONE
MsgType	MQMT_REPLY
만기	MQRO_PASS_DISCARD_AND_EXPIRY 설정되는 경우: 입력 메시지에서 복사됩니다. 그렇지 않으면 MQEI_UNLIMITED
Feedback	MQFB_NONE
MsgId	MQPMO_NEW_MSG_ID가 설정되는 경우: 새 메시지 ID가 생성됩니다. 그렇지 않고 MQRO_PASS_MSG_ID가 설정되는 경우: 입력 메시지에서 복사됩니다. 그렇지 않으면 MQMI_NONE
CorrelId	MQPMO_NEW_CORREL_ID가 설정되는 경우: 새 상관 ID가 생성됩니다. 그렇지 않고 MQRO_COPY_MSG_ID_TO_CORREL_ID가 설정되는 경우: 입력 메시지의 MsgId 필드에서 복사됩니다. 그렇지 않고 MQRO_PASS_CORREL_ID가 설정되는 경우: 입력 메시지의 CorrelId 필드에서 복사됩니다. 그렇지 않으면 MQCI_NONE
BackoutCount	0
ReplyToQ	공백
ReplyToQMgr	공백
GroupId	MQGI_NONE
MsgSeqNumber	1
오프셋	0
MsgFlags	MQMF_NONE
OriginalLength	MQOL_UNDEFINED

- 그러면 메시지 디스크립터는 새 메시지 핸들에 의해 수정됩니다. 명시적으로 새 메시지 핸들에 특성으로 설정된 메시지 디스크립터 필드는 이전에 설명된 바와 같이 메시지 디스크립터 필드보다 우선권을 갖습니다.

메시지 특성은 다음과 같이 작성됩니다.

- MQPD.CopyOptions에서 MQCOPY\_REPLY가 있는 원래 메시지 핸들의 모든 특성
- 새 메시지 핸들의 모든 특성. 원래 메시지 핸들의 특성과 동일한 이름이 있는 새 메시지 핸들에 있는 각 특성의 경우 값은 새 메시지 핸들에서 가져옵니다. 새 메시지 핸들의 특성에 원래 메시지 핸들의 특성과 동일한 이름이 있는 경우 특성의 값이 널인 경우 이 규칙에 대한 예외만 특수 경우입니다. 이 경우 특성은 메시지에서 제거됩니다.

전달되는 메시지 데이터는 MQPUT/MQPUT1 버퍼 매개변수에서 가져옵니다.

### MQACTP\_REPORT

보고서는 이전에 검색된 메시지의 결과로 생성됩니다. 원래 메시지 핸들은 보고서가 생성되도록 하는 메시지를 지정합니다.

새 메시지 핸들은 원래 메시지 핸들에서 특성에 수정사항을 지정합니다(메시지 디스크립터의 사항 포함).

메시지 디스크립터는 다음과 같이 구성됩니다.

- MsgDesc가 MQPUT 또는 MQPUT1 호출에서 제공되며 MQPMO\_MD\_FOR\_OUTPUT\_ONLY가 MQPMO.Options에 없는 경우, 이는 수정되지 않은 메시지 디스크립터로 사용됩니다.
- MsgDesc가 제공되지 않거나 MQPMO\_MD\_FOR\_OUTPUT\_ONLY가 MQPMO.Options에 있는 경우에는 필드에 대한 초기 메시지 디스크립터가 다음과 같이 선택됩니다.

표 510. 보고 메시지 핸들 변환	
MQMD의 필드	사용된 값
보고서	MQRO_PASS_DISCARD_AND_EXPIRY 및 MQRO_DISCARD_MSG가 설정되는 경우: MQRO_DISCARD_MSG 그렇지 않으면 MQRO_NONE
MsgType	MQMT_REPORT
만기	MQRO_PASS_DISCARD_AND_EXPIRY 설정되는 경우: 입력 메시지에서 복사됩니다. 그렇지 않으면 MQEI_UNLIMITED
MsgId	MQPMO_NEW_MSG_ID가 설정되는 경우: 새 메시지 ID가 생성됩니다. 그렇지 않고 MQRO_PASS_MSG_ID가 설정되는 경우: 입력 메시지에서 복사됩니다. 그렇지 않으면 MQMI_NONE

표 510. 보고 메시지 핸들 변환 (계속)	
MQMD의 필드	사용된 값
CorrelId	MQPMO_NEW_CORREL_ID가 설정되는 경우: 새 상관 ID가 생성됩니다. 그렇지 않고 MQRO_COPY_MSG_ID_TO_CORREL_ID가 설정되는 경우: 입력 메시지의 MsgId 필드에서 복사됩니다. 그렇지 않고 MQRO_PASS_CORREL_ID가 설정되는 경우: 입력 메시지의 CorrelId 필드에서 복사됩니다. 그렇지 않으면 MQCI_NONE
BackoutCount	0
ReplyToQ	공백
ReplyToQMgr	공백
OriginalLength	BufferLength로 설정됩니다.

• 그러면 메시지 디스크립터는 새 메시지 핸들에 의해 수정됩니다. 명시적으로 새 메시지 핸들에 특성으로 설정된 메시지 디스크립터 필드는 이전에 설명된 바와 같이 메시지 디스크립터 필드보다 우선권을 갖습니다.

메시지 특성은 다음과 같이 작성됩니다.

- MQCOPY\_REPORT in the MQPD.CopyOptions가 있는 원래 메시지 핸들의 모든 특성
- 새 메시지 핸들의 모든 특성. 원래 메시지 핸들의 특성과 동일한 이름이 있는 새 메시지 핸들에 있는 각 특성의 경우 값은 새 메시지 핸들에서 가져옵니다. 새 메시지 핸들의 특성에 원래 메시지 핸들의 특성과 동일한 이름이 있는 경우 특성의 값이 널인 경우 이 규칙에 대한 예외만 특수 경우입니다. 이 경우 특성은 메시지에서 제거됩니다.

결과적인 MQMD의 피드백 필드는 생성되는 보고서를 나타냅니다. MQFB\_NONE의 피드백 값으로 인해 MQPUT 또는 MQPUT1 호출이 이유 코드 MQRC\_FEEDBACK\_ERROR와 함께 실패할 수 있습니다.

보고 메시지의 사용자 데이터를 선택하기 위해 IBM MQ는 MQPUT 또는 MQPUT1 호출의 결과적인 MQMD 및 Buffer와 BufferLength 매개변수에서 보고서 및 피드백 필드를 참조합니다.

- 피드백이 MQFB\_COA, MQFB\_COD 또는 MQFB\_EXPIRATION인 경우 보고서의 값을 조사합니다.
- 다음 경우가 참인 경우 BufferLength 길이의 버퍼에서 전체 메시지 데이터가 사용됩니다.
  - 피드백은 MQFB\_EXPIRATION이며 보고서는 MQRO\_EXPIRATION\_WITH\_FULL\_DATA를 포함합니다.
  - 피드백은 MQFB\_COD이며 보고서는 MQRO\_COD\_WITH\_FULL\_DATA를 포함합니다.
  - 피드백은 MQFB\_COA이며 보고서는 MQRO\_COA\_WITH\_FULL\_DATA를 포함합니다.
- 다음 경우가 참인 경우 버퍼에서 처음 100바이트의 메시지(또는 100바이트 미만인 경우 BufferLength)가 사용됩니다.
  - 피드백은 MQFB\_EXPIRATION이고 보고서는 MQRO\_EXPIRATION\_WITH\_DATA를 포함합니다.
  - 피드백은 MQFB\_COD이고 보고서는 MQRO\_COD\_WITH\_DATA를 포함합니다.
  - 피드백은 MQFB\_COA이고 보고서는 MQRO\_COA\_WITH\_DATA를 포함합니다.
- 피드백이 MQFB\_EXPIRATION, MQFB\_COD 또는 MQFB\_COA이고 보고서가 해당 피드백 값과 관련된 \*\_WITH\_FULL\_DATA 또는 \*\_WITH\_DATA 옵션을 포함하지 않는 경우 사용자 데이터가 메시지에 포함되지 않습니다.
- 피드백이 위에 나열된 값과 다른 값을 사용하는 경우 Buffer 및 BufferLength가 보통으로 사용됩니다.

이전 목록에서 설명된 사용자 데이터의 파생은 다음 표에서도 표시됩니다.

표 511. 사용자 데이터의 소스			
	MQFB_COA	MQFB_COD	MQFB_EXPIRATION
MQRO_EXPIRATION_WITH_FULL_DATA	없음	없음	버퍼(Bufferlength)
MQRO_COD_WITH_FULL_DATA	없음	버퍼(Bufferlength)	없음
MQRO_COA_WITH_FULL_DATA	버퍼(Bufferlength)	없음	없음
MQRO_EXPIRATION_WITH_DATA	없음	없음	버퍼(처음 100바이트)
MQRO_COD_WITH_DATA	없음	버퍼(처음 100바이트)	없음
MQRO_COA_WITH_DATA	버퍼(처음 100바이트)	없음	없음

### MQPMO의 경우 PubLevel (MQLONG)

이 필드의 초기값은 9입니다. 이 책의 대상이 되는 등록 레벨입니다. 이 값 이하의 가장 높은 SubLevel 레벨이 있는 구독만이 발행을 수신합니다. 이 값은 0 - 9 범위에 있어야 하며, 0이 가장 낮은 레벨입니다. 발행물을 보유한 경우 PubLevel 1에서 다시 발행되기 때문에 상위 레벨의 구독자에게 더 이상 사용 가능하지 않습니다.

정보는 발행 인터셉트를 참조하십시오.

### MQPMR - Put 메시지 레코드

MQPMR 구조를 사용하여 분배 목록에 메시지를 넣을 때 단일 목적지에 대한 다양한 메시지 특성을 지정하십시오. MQPMR은 MQPUT 및 MQPUT1 호출의 입출력(I/O) 구조입니다.

### 가용성

MQPMR 구조는 다음 플랫폼에서 사용 가능합니다.

-  AIX
-  IBM i
-  Linux
-  Windows

및 해당 시스템에 연결된 IBM MQ 클라이언트

### 문자 세트 및 인코딩

MQPMR의 데이터는 MQENC\_NATIVE로 지정된 로컬 큐 관리자의 인코딩 및 **CodedCharSetId** 큐 관리자 속성으로 지정된 문자 세트에 있어야 합니다. 그러나 애플리케이션이 MQ 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트 및 인코딩에 있어야 합니다.

### 사용법

MQPUT 또는 MQPUT1 호출에서 이러한 구조의 배열을 제공하여 분배 목록의 각 목적지 큐에 대해 서로 다른 값을 지정할 수 있습니다. 일부 필드는 입력 전용이며 다른 필드는 입출력(I/O)용입니다.

**참고:** 이 구조는 고정 레이아웃이 없다는 점에서 혼치 않습니다. 이 구조의 필드는 선택사항이며 각 필드의 존재 또는 부재는 MQPMO의 *PutMsgRecFields* 필드에서 플래그로 표시됩니다. 존재하는 필드는 다음 순서대로 발생해야 합니다.

- *MsgId*
- *CorrelId*
- *GroupId*
- *Feedback*
- *AccountingToken*

없는 필드는 레코드에서 공백을 차지하지 않습니다.

MQPMR에 고정된 레이아웃이 없으므로 지원되는 프로그래밍 언어에 대해 헤더, COPY, INCLUDE 파일에서 제공되는 해당 정의가 없습니다. 애플리케이션 프로그래머는 애플리케이션이 필요로 하는 필드가 포함된 선언을 작성해야 하며 *PutMsgRecFields*에서 플래그를 설정하여 존재하는 필드를 표시해야 합니다.

## 필드

지원되는 프로그래밍 언어에 대한 헤더, COPY 및 INCLUDE 파일에 구조 선언이 제공되지 않으므로 이 구조에 대해 정의된 초기값이 없습니다. 샘플 선언은 모든 필드가 요구되면 구조를 선언하는 방법을 표시합니다.

표 512. MQPMR의 필드	
필드 이름	필드 설명
<u>MsgId</u>	메시지 ID
<u>CorrelId</u>	상관 ID
<u>GroupId</u>	그룹 ID
<u>피드백</u>	피드백 또는 이유 코드
<u>AccountingToken</u>	계정 토큰

## 언어 선언

### MQPMR의 C 선언

```
typedef struct tagMQPMR MQPMR;
struct tagMQPMR {
    MQBYTE24  MsgId;           /* Message identifier */
    MQBYTE24  CorrelId;       /* Correlation identifier */
    MQBYTE24  GroupId;        /* Group identifier */
    MQLONG    Feedback;       /* Feedback or reason code */
    MQBYTE32  AccountingToken; /* Accounting token */
};
```

### MQPMR의 COBOL 선언

```
** MQPMR structure
10 MQPMR.
** Message identifier
15 MQPMR-MSGID PIC X(24).
** Correlation identifier
15 MQPMR-CORRELID PIC X(24).
** Group identifier
15 MQPMR-GROUPID PIC X(24).
** Feedback or reason code
15 MQPMR-FEEDBACK PIC S9(9) BINARY.
** Accounting token
15 MQPMR-ACCOUNTINGTOKEN PIC X(32).
```

### MQPMR의 PL/I 선언

```
dcl
1 MQPMR based,
3 MsgId char(24), /* Message identifier */
```

```

3 CorrelId      char(24),      /* Correlation identifier */
3 GroupId      char(24),      /* Group identifier */
3 Feedback     fixed bin(31), /* Feedback or reason code */
3 AccountingToken char(32);  /* Accounting token */

```

## MQPMR의 Visual Basic 선언

```

Type MQPMR
  MsgId      As MQBYTE24 'Message identifier'
  CorrelId   As MQBYTE24 'Correlation identifier'
  GroupId    As MQBYTE24 'Group identifier'
  Feedback   As Long     'Feedback or reason code'
  AccountingToken As MQBYTE32 'Accounting token'
End Type

```

### MQPMR의 경우 MsgId (MQBYTE24)

이는 MQOPEN 또는 MQPUT1 호출에 제공된 MQOR 구조의 배열에서 해당 요소가 지정한 이름이 있는 큐에 전송된 메시지에 사용할 메시지 ID입니다. 이는 단일 큐에 넣기 위해 MQMD에서 *MsgId* 필드와 동일한 방법으로 처리됩니다.

이 필드가 MQPMR 레코드에 없거나 MQPMR 레코드 수가 목적지보다 적은 경우, *MsgId* 필드가 포함된 MQPMR 레코드가 없는 목적지에 MQMD의 값이 사용됩니다. 해당 값이 MQMI\_NONE인 경우 새 메시지 ID가 각 해당 목적지에 대해 생성됩니다(즉, 두 목적지가 동일한 메시지 ID를 가지지 않음).

MQPMO\_NEW\_MSG\_ID가 지정되면 MQPMR 레코드가 있는지 여부에 상관없이 새 메시지 ID가 분배 목록의 모든 목적지에 생성됩니다. 이는 MQPMO\_NEW\_CORREL\_ID가 처리되는 방법과 다릅니다(*CorrelId* 필드 참조).

이 필드는 입출력(I/O) 필드입니다.

### MQPMR의 경우 CorrelId (MQBYTE24)

이는 MQOPEN 또는 MQPUT1 호출에 제공된 MQOR 구조의 배열에서 해당 요소가 지정한 이름이 있는 큐에 전송된 메시지에 사용할 상관 ID입니다. 이는 단일 큐에 넣기 위해 MQMD에서 *CorrelId* 필드와 동일한 방법으로 처리됩니다.

이 필드가 MQPMR 레코드에 없거나 목적지보다 적은 MQPMR 레코드가 있는 경우 *CorrelId* 필드가 포함된 MQPMR 레코드가 없는 해당 목적지에 대해 MQMD의 값이 사용됩니다.

MQPMO\_NEW\_CORREL\_ID가 지정되면 MQPMR 레코드가 있는지 여부에 상관없이 분배 목록에서 모든 목적지에 대해 하나의 새 상관 ID가 생성되고 사용됩니다. 이는 MQPMO\_NEW\_MSG\_ID가 처리되는 방법과 다릅니다(*MsgId* 필드 참조).

이 필드는 입출력(I/O) 필드입니다.

### MQPMR의 경우 GroupId (MQBYTE24)

*GroupId*는 MQOPEN 또는 MQPUT1 호출에 제공된 MQOR 구조의 배열에서 해당 요소가 지정한 이름이 있는 큐에 전송된 메시지에 사용할 그룹 ID입니다. 이는 단일 큐에 넣기 위해 MQMD에서 *GroupId* 필드와 동일한 방법으로 처리됩니다.

이 필드가 MQPMR 레코드에 없거나 MQPMR 레코드 수가 목적지보다 적은 경우, *GroupId* 필드가 포함된 MQPMR 레코드가 없는 목적지에 MQMD의 값이 사용됩니다. 큐의 실제 순서에 설명한 대로 값이 처리되지만 다음 차이가 있습니다.

- *GroupId*는 QMName 및 시간소인에서 작성됩니다. 그러므로 *GroupId*를 고유하게 유지하려면 큐 관리자 이름도 고유하게 유지해야 합니다. 또한 큐 관리자 시스템에서 클럭을 다시 설정하지 마십시오.
- 새 그룹 ID가 사용될 경우 큐 관리자는 각 목적지에 대해 서로 다른 그룹 ID를 생성합니다. (즉, 동일한 그룹 ID를 갖는 두 개의 목적지는 없습니다.)
- 필드의 값이 사용되는 경우 이유 코드 MQRC\_GROUP\_ID\_ERROR와 함께 호출이 실패합니다.

이 필드는 입출력(I/O) 필드입니다.

### MQPMR에 대한 피드백 (MQLONG)

MQOPEN 또는 MQPUT1 호출에 제공된 MQOR 구조 배열에서 해당하는 요소가 지정한 이름의 큐에 메시지를 송신하는 데 사용될 피드백 코드입니다. 이는 단일 큐에 넣기 위해 MQMD에서 *Feedback* 필드와 동일한 방법으로 처리됩니다.

이 필드가 없는 경우 MQMD의 값이 사용됩니다.

입력 필드입니다.

### MQPMR의 경우 *AccountingToken* (MQBYTE32)

이는 MQOPEN 또는 MQPUT1 호출에 제공된 MQOR 구조의 배열에서 해당 요소가 지정한 이름이 있는 큐에 전송된 메시지에 사용할 회계 토큰입니다. 이는 단일 큐에 넣기 위해 MQMD에서 *AccountingToken* 필드와 동일한 방법으로 처리됩니다. 이 필드의 콘텐츠에 대한 정보는 412 페이지의 『MQMD - 메시지 디스크립터』에서 *AccountingToken*에 대한 설명을 참조하십시오.

이 필드가 없는 경우 MQMD의 값이 사용됩니다.

입력 필드입니다.

## MQRFH - 규칙 및 형식화 헤더

MQRFH 구조는 규칙 및 형식화 헤더의 레이아웃을 정의합니다. 이 헤더를 사용하여 이름-값 쌍의 형식으로 문자열 데이터를 전송하십시오.

### 가용성

모든 IBM MQ 시스템 및 해당 시스템에 연결된 IBM MQ MQI clients

### 형식 이름

MQFMT\_RF\_HEADER

### 문자 세트 및 인코딩

MQRFH 구조의 필드 (*NameValueString* 포함)는 MQRFH에 선행하는 헤더 구조의 *CodedCharSetId* 및 *Encoding* 필드 또는 MQRFH가 애플리케이션 메시지 데이터의 시작에 있는 경우 MQMD 구조의 해당 필드에서 제공되는 문자 세트 및 인코딩에 있습니다.

문자 세트는 큐 이름에 유효한 문자에 사용되는 단일 바이트 문자가 있어야 합니다.

### 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 513. MQRFH에서 MQRFH의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
<u>StrucId</u> (구조 ID)	MQRFH_STRUC_ID	'RFH~'
<u>버전</u> (구조 버전 번호)	MQRFH_VERSION_1	1
<u>StrucLength</u> (MQRFH 구조의 길이 (바이트))	MQRFH_STRUC_LENGTH_FIXED	32
<u>인코딩</u> ( <i>NameValueString</i> 뒤에 오는 데이터의 문자 인코딩)	MQENC_NATIVE	환경에 따라 다름
<u>CodedCharSetId</u> ( <i>NameValueString</i> 뒤에 오는 데이터의 문자 세트 ID 지정)	MQCCSI_UNDEFINED	0
<u>형식</u> ( <i>NameValueString</i> 뒤에 오는 데이터의 형식 이름)	MQFMT_NONE	공백

표 513. MQRFH에서 MQRFH의 필드 (계속)		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
플래그 (플래그)	MQRFH_NONE	0
NameValueString (이름-값 쌍을 포함하는 가변 길이 문자열)	없음	없음
<b>참고:</b> 1. ~ 기호는 단일 공백 문자를 나타냅니다. 2. C 프로그래밍 언어의 매크로 변수 MQRFH_DEFAULT는 테이블에 나열되는 값을 포함합니다. 즉, 구조 내의 필드에 대한 초기값을 제공하기 위해 다음과 같은 방법으로 사용될 수 있습니다. <pre>MQRFH MyRFH = {MQRFH_DEFAULT};</pre>		

## 언어 선언

### MQRFH의 C 선언

```
typedef struct tagMQRFH MQRFH;
struct tagMQRFH {
    MQCHAR4  StrucId;          /* Structure identifier */
    MQLONG   Version;         /* Structure version number */
    MQLONG   StrucLength;     /* Total length of MQRFH including
                             NameValueString */
    MQLONG   Encoding;        /* Numeric encoding of data that follows
                             NameValueString */
    MQLONG   CodedCharSetId; /* Character set identifier of data that
                             follows NameValueString */
    MQCHAR8  Format;          /* Format name of data that follows
                             NameValueString */
    MQLONG   Flags;           /* Flags */
};
```

### MQRFH의 COBOL 선언

```
** MQRFH structure
10 MQRFH.
** Structure identifier
15 MQRFH-STRUCID PIC X(4).
** Structure version number
15 MQRFH-VERSION PIC S9(9) BINARY.
** Total length of MQRFH including NAMEVALUESTRING
15 MQRFH-STRUCLENGTH PIC S9(9) BINARY.
** Numeric encoding of data that follows NAMEVALUESTRING
15 MQRFH-ENCODING PIC S9(9) BINARY.
** Character set identifier of data that follows NAMEVALUESTRING
15 MQRFH-CODEDCHARSETID PIC S9(9) BINARY.
** Format name of data that follows NAMEVALUESTRING
15 MQRFH-FORMAT PIC X(8).
** Flags
15 MQRFH-FLAGS PIC S9(9) BINARY.
```

### MQRFH의 PL/I 선언

```
dcl
1 MQRFH based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 StrucLength fixed bin(31), /* Total length of MQRFH including
                             NameValueString */
3 Encoding fixed bin(31), /* Numeric encoding of data that
                             follows NameValueString */
3 CodedCharSetId fixed bin(31), /* Character set identifier of data
                             that follows NameValueString */
```



```

3 Format      char(8),      /* Format name of data that follows
                          NameValueString */
3 Flags      fixed bin(31); /* Flags */

```

### MQRFH의 상위 레벨 어셈블러 선언

```

MQRFH          DSECT
MQRFH_STRUCID  DS    CL4  Structure identifier
MQRFH_VERSION  DS    F    Structure version number
MQRFH_STRUCLNGTH DS    F    Total length of MQRFH including
*              NAMEVALUESTRING
MQRFH_ENCODING DS    F    Numeric encoding of data that follows
*              NAMEVALUESTRING
MQRFH_CODEDCHARSETID DS    F    Character set identifier of data that
*              follows NAMEVALUESTRING
MQRFH_FORMAT   DS    CL8  Format name of data that follows
*              NAMEVALUESTRING
MQRFH_FLAGS    DS    F    Flags
*
MQRFH_LENGTH   EQU    *-MQRFH
                ORG    MQRFH
MQRFH_AREA     DS    CL(MQRFH_LENGTH)

```

### MQRFH의 Visual Basic 선언

```

Type MQRFH
  StrucId      As String*4 'Structure identifier'
  Version      As Long     'Structure version number'
  StrucLength  As Long     'Total length of MQRFH including
                          'NameValueString'
  Encoding     As Long     'Numeric encoding of data that follows'
                          'NameValueString'
  CodedCharSetId As Long   'Character set identifier of data that
                          'follows NameValueString'
  Format       As String*8 'Format name of data that follows'
                          'NameValueString'
  Flags       As Long     'Flags'
End Type

```

### **MQRFH용 StrucId (MQCHAR4)**

규칙 및 형식화 헤더 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQRFH\_STRUC\_ID입니다. 값은 다음과 같아야 합니다.

#### **MQRFH\_STRUC\_ID**

규칙 및 형식화 헤더 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQRFH\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQRFH\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### **MQRFH의 버전 (MQLONG)**

구조 버전 번호이며 값은 다음과 같아야 합니다.

#### **MQRFH\_VERSION\_1**

버전-1 규칙 및 형식화 헤더 구조 ID.

이 필드의 초기값은 MQRFH\_VERSION\_1입니다.

### **MQRFH용 StrucLength (MQLONG)**

이는 구조의 끝에 있는 *NameValueString* 필드를 포함하여 MQRFH 구조의 길이(바이트)입니다. 길이에 *NameValueString* 필드 다음에 오는 사용자 데이터가 포함되지 않습니다.

일부 환경에서 사용자 데이터를 변환하는 데 문제가 발생하지 않도록 하려면 *StrucLength*이(가) 4의 배수여야 합니다.

다음 상수는 구조의 고정된 부분의 길이(즉, *NameValueString* 필드를 제외한 길이)를 제공합니다.

## **MQRFH\_STRUC\_LENGTH\_FIXED**

MQRFH 구조에서 고정 부분의 길이.

이 필드의 초기값은 MQRFH\_STRUC\_LENGTH\_FIXED입니다.

## **MQRFH용 인코딩 (MQLONG)**

이는 *NameValueString* 다음에 오는 데이터의 숫자 인코딩을 지정합니다. MQRFH 구조 자체의 숫자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다.

이 필드의 초기값은 MQENC\_NATIVE입니다.

## **MQRFH용 CodedCharSetId (MQLONG)**

이는 *NameValueString* 다음에 오는 데이터의 문자 세트 ID를 지정합니다. MQRFH 구조 자체의 문자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 다음 특수 값을 사용할 수 있습니다.

### **MQCCSI\_INHERIT**

이 구조 다음에 오는 데이터의 문자 데이터는 이 구조와 동일한 문자 세트로 되어 있습니다.

큐 관리자가 구조의 실제 문자 세트 ID에 메시지로 송신한 구조에서 이 값을 변경합니다. 오류가 발생하지 않으면 MQGET 호출에서 MQCCSI\_INHERIT 값을 리턴하지 않습니다.

MQMD의 *PutApplType* 필드 값이 MQAT\_BROKER인 경우 MQCCSI\_INHERIT를 사용할 수 없습니다.

이 필드의 초기값은 MQCCSI\_UNDEFINED입니다.

## **MQRFH의 형식 (MQCHAR8)**

*NameValueString* 뒤에 오는 데이터의 형식명을 지정합니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 이 필드의 코딩 규칙은 MQMD의 *Format* 필드에 대한 규칙과 동일합니다.

이 필드의 초기값은 MQFMT\_NONE입니다.

## **MQRFH의 플래그 (MQLONG)**

다음을 지정할 수 있습니다.

### **MQRFH\_NONE**

플래그가 없습니다.

이 필드의 초기값은 MQRFH\_NONE입니다.

## **NameValueMQRFH의 문자열 (MQCHARn)**

다음 양식으로 이름-값 쌍을 포함하는 가변 길이 문자열입니다.

```
name1 value1 name2 value2 name3 value3 ...
```

각 이름 또는 값은 하나 이상의 공백 문자열을 사용하여 근접한 이름 또는 값과 분리해야 합니다. 이 때 사용된 공백은 중요하지 않습니다. 이름 또는 값은 이름 또는 값에 접두부 및 접미부로 큰따옴표 표시를 사용하여 중요한 공백을 포함할 수 있습니다. 여는 큰따옴표 표시 및 일치하는 닫는 큰따옴표 표시 간의 모든 문자는 중요하게 처리됩니다. 다음 예에서 이름은 FAMOUS\_WORDS이고 값은 Hello World입니다.

```
FAMOUS_WORDS "Hello World"
```

이름 및 값은 널 문자가 아닌 문자를 포함할 수 있습니다(*NameValueString*)에 대해 구분 기호의 역할을 함). 그러나 상호 운용성을 지원하기 위해 애플리케이션은 이름을 다음 문자로 제한할 수 있습니다.

- 첫 번째 문자: 대문자 또는 소문자 알파벳(A - Z 또는 a - z) 또는 밑줄
- 후속 문자: 대문자 또는 소문자 영문자, 10진수(0 - 9), 밑줄, 하이픈 또는 점.

이름 또는 값이 하나 이상의 큰따옴표 표시를 포함하는 경우 이름 또는 값은 큰따옴표 표시 안에 있어야 하고 문자열 내의 각 큰따옴표 표시는 두 번 표시되어야 합니다.

```
Famous_Words "The program displayed ""Hello World"""
```

이름과 값은 대소문자를 구분합니다. 즉, 소문자는 대문자와 동일한 것으로 간주되지 않습니다. 예를 들어, FAMOUS\_WORDS 및 Famous\_Words는 두 개의 다른 이름입니다.

*NameValueString*의 길이(바이트)는 *StrucLength* 마이너스 MQRFH\_STRUC\_LENGTH\_FIXED와 같습니다. 일부 환경에서 사용자 데이터를 변환할 때 문제점을 피하려면 이 길이가 4의 배수여야 합니다.

*NameValueString*을 이 길이까지 공백으로 채우거나 문자열의 마지막 중요한 문자 뒤에 널 문자를 배치하여 일찍 종료하십시오. *NameValueString*의 지정된 길이까지 널 문자와 그 뒤의 바이트는 무시됩니다.

**참고:** 이 필드의 길이가 고정되어 있지 않기 때문에, 지원되는 프로그래밍 언어를 위해 제공되는 구조의 선언에서 필드는 생략됩니다.

## MQRFH2 - 규칙 및 형식화 헤더 2

MQRFH2 헤더는 MQRFH 헤더를 기반으로 하지만 유니코드 문자열이 변환 없이 전송되도록 허용하며 숫자 데이터 유형을 수행할 수 있습니다. MQRFH2 구조는 버전-2 규칙의 형식 및 형식화 헤더를 정의합니다. 이 헤더를 사용하여 XML과 유사한 구문을 사용하여 인코딩된 데이터를 전송합니다. 메시지는 두 개 이상의 MQRFH2 구조가 연속으로 포함될 수 있으며, 선택적으로 일련의 구조 중 마지막 MQRFH2 구조 뒤에 사용자 데이터가 있습니다.

### 가용성

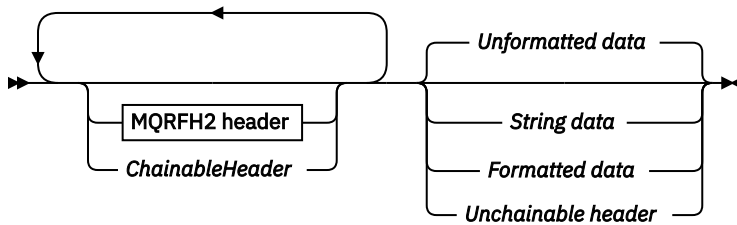
모든 IBM MQ 시스템 및 해당 시스템에 연결된 IBM MQ MQI clients

### 형식 이름

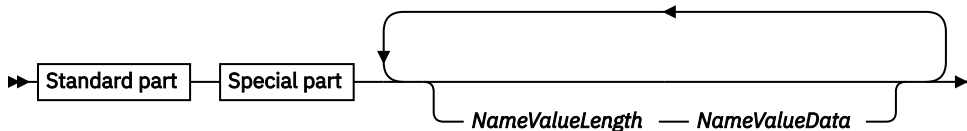
MQFMT\_RF\_HEADER\_2

### Syntax

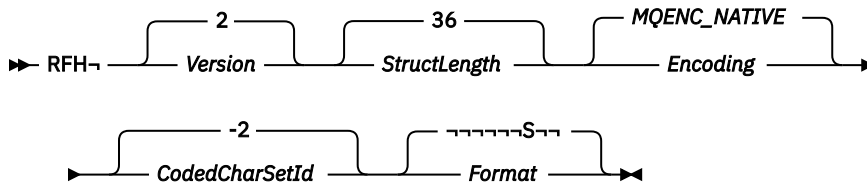
#### IBM MQ Message



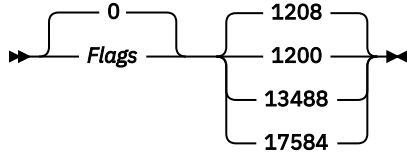
#### MQRFH2 header



#### Standard part



### Special part



## 문자 세트 및 인코딩

특수 규칙이 MQRFH2 구조에 사용된 문자 세트 및 인코딩에 적용됩니다.

- *NameValueData* 이외의 필드는 MQRFH2가 애플리케이션 메시지 데이터의 시작에 있는 경우 MQMD 구조의 해당 필드가 또는 MQRFH2에 선행하는 헤더 구조의 *CodedCharSetId* 및 *Encoding* 필드가 지정한 문자 세트 및 인코딩에 있습니다.

문자 세트는 큐 이름에 유효한 문자에 사용되는 단일 바이트 문자가 있어야 합니다.

MQGMO\_CONVERT가 MQGET 호출에서 지정된 경우 큐 관리자는 *NameValueData*가 아닌 MQRFH2 필드를 요청된 문자 세트 및 인코딩으로 변환합니다.

- *NameValueData*는 *NameValueCCSID* 필드가 지정한 문자 세트에 있습니다. 나열된 유니코드 문자 세트만 *NameValueCCSID*에 유효합니다. 세부사항은 *NameValueCCSID*에 대한 설명을 참조하십시오.

일부 문자 세트에는 인코딩에 의존하는 표현이 있습니다. *NameValueCCSID*가 해당 문자 세트 중 하나인 경우 *NameValueData*는 MQRFH2의 기타 필드와 동일한 인코딩에 있어야 합니다.

MQGMO\_CONVERT가 MQGET 호출에서 지정되면 큐 관리자는 *NameValueData*를 요청된 인코딩으로 변환하지만 해당 문자 세트는 변경하지 않습니다.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 514. MQRFH2에 대한 MQRFH2의 필드		
필드 이름	상수의 이름	상수의 값
StrucId (구조 ID)	MQRFH_STRUC_ID	'RFH'
버전 (구조 버전 번호)	MQRFH_VERSION_2	2
StrucLength (MQRFH2 구조의 길이 (바이트))	MQRFH_STRUC_LENGTH_FIXED_2	36
인코딩 (마지막 <i>NameValueData</i> 필드 뒤에 오는 데이터의 숫자 인코딩)	MQENC_NATIVE	환경에 따라 다름
CodedCharSetId (마지막 <i>NameValueData</i> 필드 뒤에 오는 데이터의 문자 세트 ID)	MQCCSI_INHERIT	-2

표 514. MQRFH2 에 대한 MQRFH2 의 필드 (계속)		
필드 이름	상수의 이름	상수의 값
형식 (마지막 <i>NameValueData</i> 필드 뒤에 오는 데이터의 형식 이름)	MQFMT_NONE	공백
플래그 (플래그)	MQRFH_NONE	0
NameValueCCSID ( <i>NameValueData</i> 필드에 있는 데이터의 코드화 문자 세트 ID)	없음	1208
NameValue길이 ( <i>NameValueData</i> 필드에 있는 데이터의 길이 (바이트))	없음	None
NameValue데이터 (메시지 특성의 이름-값 쌍)	없음	없음
<b>참고:</b> 1. ~ 기호는 단일 공백 문자를 나타냅니다. 2. C 프로그래밍 언어의 매크로 변수 MQRFH2_DEFAULT는 테이블에 나열되는 값을 포함합니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오. <pre style="background-color: #f0f0f0; padding: 5px;">MQRFH2 MyRFH2 = {MQRFH2_DEFAULT};</pre>		

## 언어 선언

### MQRFH2의 C 선언

```
typedef struct tagMQRFH2 MQRFH2;
struct tagMQRFH2 {
    MQCHAR4  StrucId;           /* Structure identifier */
    MQLONG   Version;          /* Structure version number */
    MQLONG   StrucLength;      /* Total length of MQRFH2 including all
                               NameValueLength and NameValueData
                               fields */
    MQLONG   Encoding;         /* Numeric encoding of data that follows
                               last NameValueData field */
    MQLONG   CodedCharSetId;   /* Character set identifier of data that
                               follows last NameValueData field */
    MQCHAR8  Format;           /* Format name of data that follows last
                               NameValueData field */
    MQLONG   Flags;            /* Flags */
    MQLONG   NameValueCCSID;   /* Character set identifier of
                               NameValueData */
};
```

### MQRFH2의 COBOL 선언

```
** MQRFH2 structure
10 MQRFH2.
** Structure identifier
15 MQRFH2-STRUCID PIC X(4).
** Structure version number
15 MQRFH2-VERSION PIC S9(9) BINARY.
** Total length of MQRFH2 including all NAMEVALUELENGTH and
** NAMEVALUEDATA fields
15 MQRFH2-STRUCLNGTH PIC S9(9) BINARY.
```

```

** Numeric encoding of data that follows last NAMEVALUEDATA field
15 MQRFH2-ENCODING PIC S9(9) BINARY.
** Character set identifier of data that follows last NAMEVALUEDATA
** field
15 MQRFH2-CODEDCHARSETID PIC S9(9) BINARY.
** Format name of data that follows last NAMEVALUEDATA field
15 MQRFH2-FORMAT PIC X(8).
** Flags
** MQRFH2-FLAGS PIC S9(9) BINARY.
** Character set identifier of NAMEVALUEDATA
15 MQRFH2-NAMEVALUECCSID PIC S9(9) BINARY.

```

## MQRFH2의 PL/I 선언

```

dcl
  1 MQRFH2 based,
  3 StrucId      char(4),          /* Structure identifier */
  3 Version      fixed bin(31), /* Structure version number */
  3 StrucLength  fixed bin(31), /* Total length of MQRFH2 including
                                all NameValueLength and
                                NameValueData fields */
  3 Encoding     fixed bin(31), /* Numeric encoding of data that
                                follows last NameValueData field */
  3 CodedCharSetId fixed bin(31), /* Character set identifier of data
                                that follows last NameValueData
                                field */
  3 Format        char(8),          /* Format name of data that follows
                                last NameValueData field */
  3 Flags        fixed bin(31), /* Flags */
  3 NameValueCCSID fixed bin(31); /* Character set identifier of
                                NameValueData */

```

## MQRFH2의 상위 레벨 어셈블러 선언

```

MQRFH          DSECT
MQRFH_STRUCID  DS CL4 Structure identifier
MQRFH_VERSION  DS F   Structure version number
MQRFH_STRUCLNGTH DS F   Total length of MQRFH2 including all
* NAMEVALUELENGTH and NAMEVALUEDATA fields
MQRFH_ENCODING DS F   Numeric encoding of data that follows
* last NAMEVALUEDATA field
MQRFH_CODEDCHARSETID DS F Character set identifier of data that
* follows last NAMEVALUEDATA field
MQRFH_FORMAT   DS CL8 Format name of data that follows last
* NAMEVALUEDATA field
MQRFH_FLAGS    DS F   Flags
MQRFH_NAMEVALUECCSID DS F Character set identifier of
* NAMEVALUEDATA
*
MQRFH_LENGTH   EQU *-MQRFH
                ORG MQRFH
MQRFH_AREA     DS CL(MQRFH_LENGTH)

```

## MQRFH2의 Visual Basic 선언

```

Type MQRFH2
  StrucId      As String*4 'Structure identifier'
  Version      As Long     'Structure version number'
  StrucLength  As Long     'Total length of MQRFH2 including all'
                                'NameValueLength and NameValueData fields'
  Encoding     As Long     'Numeric encoding of data that follows'
                                'last NameValueData field'
  CodedCharSetId As Long   'Character set identifier of data that'
                                'follows last NameValueData field'
  Format        As String*8 'Format name of data that follows last'
                                'NameValueData field'
  Flags        As Long     'Flags'
  NameValueCCSID As Long   'Character set identifier of NameValueData'
End Type

```

### ***MQRFH2*에 대한 *StrucId* (*MQCHAR4*)**

이는 규칙 및 형식화 헤더 2구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 *MQRFH2\_STRUC\_ID*입니다.

값은 다음과 같아야 합니다.

#### ***MQRFH2\_STRUC\_ID***

규칙 및 형식화 헤더 2구조의 ID입니다.

C 프로그래밍 언어의 경우, 상수 *MQRFH2\_STRUC\_ID\_ARRAY*도 정의됩니다. 이는 *MQRFH2\_STRUC\_ID*와 동일한 값을 갖지만 문자열이 아닌 문자의 배열입니다.

### ***MQRFH2*의 버전 (*MQLONG*)**

구조 버전 번호이며 값은 다음과 같아야 합니다.

#### ***MQRFH\_VERSION\_2***

버전-2 규칙 및 형식화 헤더 구조.

이 필드의 초기값은 *MQRFH\_VERSION\_2*입니다.

### ***MQRFH2*의 경우 *StrucLength* (*MQLONG*)**

이는 구조의 끝에 있는 *NameValueLength* 및 *NameValueData* 필드를 포함하여 *MQRFH2* 구조의 길이(바이트)입니다. 구조의 끝에 다음 순서로 다중 쌍의 *NameValueLength* 및 *NameValueData* 필드가 있는 것은 올바릅니다.

```
length1, data1, length2, data2, ...
```

*StrucLength*는 구조의 끝에서 마지막 *NameValueData* 필드 다음에 올 수 있는 사용자 데이터를 포함하지 않습니다.

일부 환경에서 사용자 데이터 변환 문제점을 방지하려면 *StrucLength*가 4의 배수여야 합니다.

다음 상수는 구조의 고정된 파트 길이(즉, *NameValueLength* 및 *NameValueData* 필드를 제외한 길이)를 제공합니다.

#### ***MQRFH\_STRUC\_LENGTH\_FIXED\_2***

*MQRFH2* 구조의 고정된 부분 길이.

이 필드의 초기값은 *MQRFH\_STRUC\_LENGTH\_FIXED\_2*입니다.

### ***MQRFH2*에 대한 인코딩 (*MQLONG*)**

이는 마지막 *NameValueData* 필드 다음에 오는 데이터의 숫자 인코딩을 지정합니다. *MQRFH2* 구조 자체의 숫자 데이터에는 적용되지 않습니다.

*MQPUT* 또는 *MQPUT1* 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다.

이 필드의 초기값은 *MQENC\_NATIVE*입니다.

### ***MQRFH2*의 *CodedCharSetId* (*MQLONG*)**

이는 마지막 *NameValueData* 필드 다음에 오는 데이터의 문자 세트 ID를 지정합니다. *MQRFH2* 구조 자체의 문자 데이터에는 적용되지 않습니다.

*MQPUT* 또는 *MQPUT1* 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 다음 특수 값을 사용할 수 있습니다.

#### ***MQCCSI\_INHERIT***

이 구조 다음에 오는 데이터의 문자 데이터는 이 구조와 동일한 문자 세트로 되어 있습니다.

큐 관리자가 구조의 실제 문자 세트 ID에 메시지로 송신한 구조에서 이 값을 변경합니다. 오류가 발생하지 않으면 *MQGET* 호출에서 *MQCCSI\_INHERIT* 값을 리턴하지 않습니다.

*MQMD*의 *PutApplType* 필드 값이 *MQAT\_BROKER*인 경우 *MQCCSI\_INHERIT*를 사용할 수 없습니다.

이 필드의 초기값은 MQCCSI\_INHERIT입니다.

### **MQRFH2의 형식 (MQCHAR8)**

마지막 *NameValueData* 필드 뒤에 오는 데이터의 형식 이름을 지정합니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 이 필드의 코딩 규칙은 MQMD의 *Format* 필드에 대한 규칙과 동일합니다.

이 필드의 초기값은 MQFMT\_NONE입니다.

### **MQRFH2에 대한 플래그 (MQLONG)**

이 필드의 초기값은 MQRFH\_NONE입니다. MQRFH\_NONE을 지정해야 합니다.

#### **MQRFH\_NONE**

플래그가 없습니다.

#### **MQRFH\_INTERNAL**

MQRFH2 헤더에 내부적으로 설정된 특성이 있습니다.

MQRFH\_INTERNAL은 큐 관리자가 사용합니다.

맨 위 16비트 MQRFH\_FLAGS\_RESTRICTED\_MASK는 큐 관리자가 설정하는 플래그를 위해 예약되어 있습니다. 사용자가 설정할 수 있는 플래그는 맨 아래 16비트에서 정의됩니다.

### **NameValue MQRFH2**

*NameValueData* 필드에 있는 데이터의 코드화된 문자 세트 ID를 지정합니다. 이는 MQRFH2 구조에서 기타 문자열의 문자 세트와 다르며 구조의 끝에서 마지막 *NameValueData* 필드 다음에 오는 데이터의 문자 세트(있는 경우)와 다를 수 있습니다.

*NameValueCCSID*에는 다음 값 중 하나가 있어야 합니다.

CCSID	의미
1200	UTF-16, 가장 최근 지원된 유니코드 버전
13488	UTF-16, 유니코드 버전 2.0 서브세트
17584	UTF-16, 유니코드 버전 3.0 서브세트(유로 기호 포함)
1208	UTF-8, 가장 최근 지원된 유니코드 버전

UTF-16 문자 세트의 경우 *NameValueData*의 인코딩(바이트 순서)이 MQRFH2 구조에서 기타 필드의 인코딩과 동일해야 합니다.

대리 코드 포인트('X'D800' - 'X'DFFF')에 의해 UTF-16로 또는 UTF-8의 4바이트로 표시되는 유니코드 기본 다국어 평면을 넘는 문자(U+FFFF 초과)는 지원되지 않습니다.

**참고:** *NameValueCCSID*에 위에 나열된 값 중 하나가 없고 MQRFH2 구조에 MQGET 호출의 변환이 필요한 경우 호출이 이유 코드 MQRC\_SOURCE\_CCsid\_ERROR와 함께 완료되며 메시지가 변환되지 않은 상태로 리턴됩니다.

이 필드의 초기값은 1208입니다.

### **NameValue MQRFH2**

해당 *NameValueData* 필드의 길이

이는 *NameValueData* 필드의 데이터 길이(바이트)를 지정합니다. *NameValueLength*는 4의 배수여야 합니다.



**참고:** *NameValueLength* 및 *NameValueData* 필드는 선택사항이지만, 존재하는 경우 쌍으로 발생하고 인접해야 합니다. 필드의 쌍은 필요한 만큼 반복될 수 있습니다. 예:

```
length1 data1 length2 data2 length3 data3
```

이 필드는 선택사항이기 때문에 지원되는 프로그래밍 언어를 위해 제공되는 구조의 선언에서 필드는 생략됩니다.

### **NameValue MQRFH2 의 데이터 (MQCHARn)**

*NameValueData* 는 메시지 특성의 이름-값 쌍을 포함하는 폴더를 포함하는 가변 길이 필드입니다. 폴더는 XML 형 구문을 사용하여 인코딩된 데이터를 포함하는 가변 길이 문자열입니다. 문자열의 길이(바이트)는 *NameValueData* 필드 앞에 있는 *NameValueLength* 필드에 의해 제공됩니다. 길이는 4의 배수여야 합니다.

*NameValueLength* 및 *NameValueData* 필드는 선택사항이지만, 존재하는 경우 쌍으로 발생하고 인접해야 합니다. 필드의 쌍은 필요한 만큼 반복될 수 있습니다. 예:

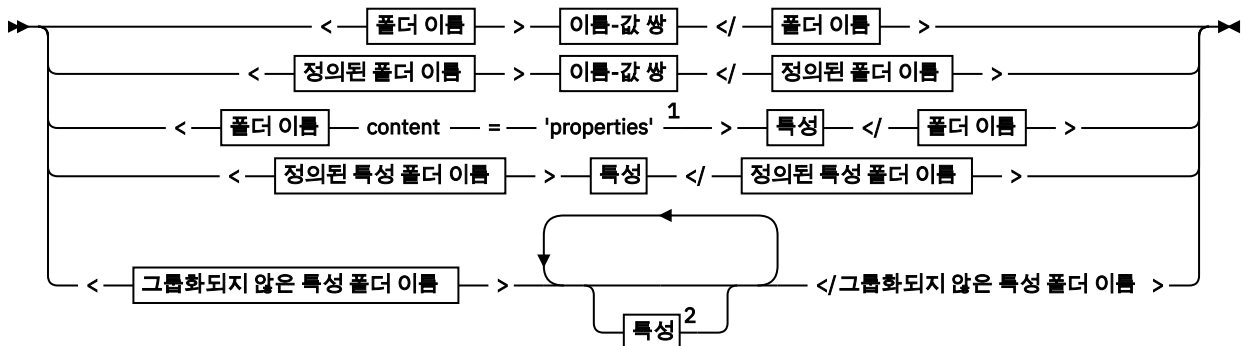
```
length1 data1 length2 data2 length3 data3
```

*NameValueData*는 MQGET 호출에서 지정된 문자 세트로 변환되지 않습니다. MQGMO\_CONVERT 옵션을 사용하여 메시지를 검색하는 경우에도 *NameValueData*은(는) 원래 문자 세트로 그대로 남아 있습니다. 그러나 *NameValueData*는 MQGET 호출에서 지정된 인코딩으로 변환됩니다.

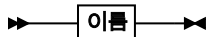
#### **참고:**

- 이 필드는 선택사항이기 때문에 지원되는 프로그래밍 언어를 위해 제공되는 구조의 선언에서 필드는 생략됩니다.
- "정의됨" 및 "예약됨"이라는 용어가 구문 다이어그램에서 사용됩니다. "정의됨"은 IBM MQ가 이름을 사용함을 의미합니다. "예약됨"은 IBM MQ가 이후에 사용하기 위해 이름을 예약함을 의미합니다.

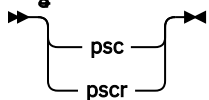
### **NameValueData 구문**



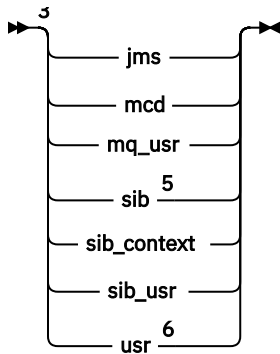
#### **폴더 이름**



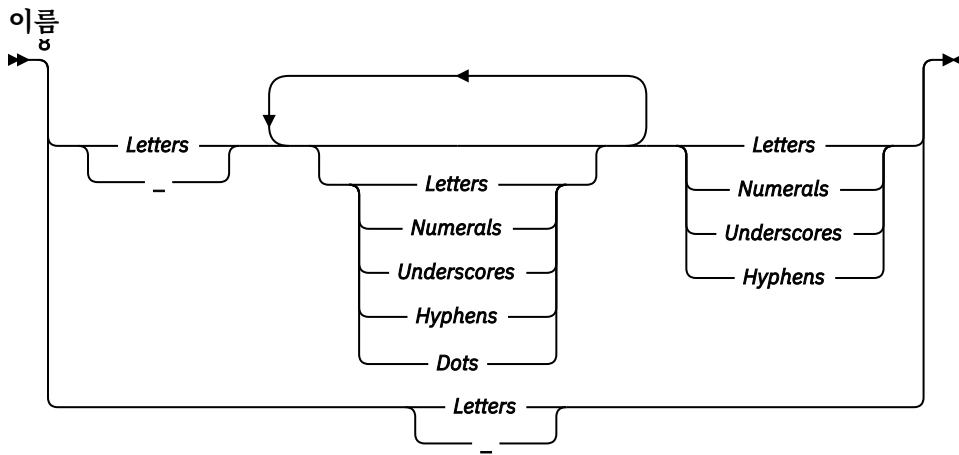
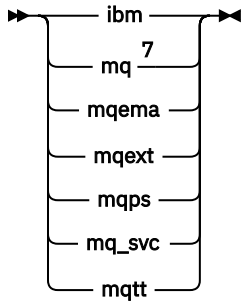
#### **정의된 폴더 이름**



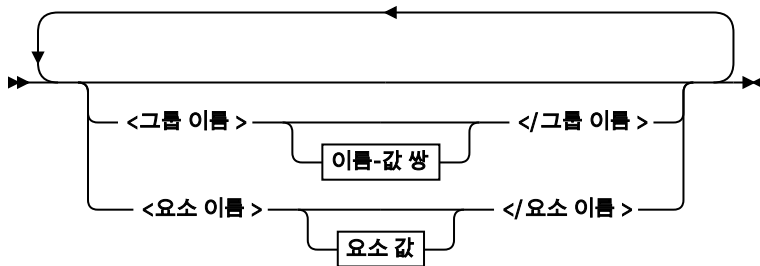
#### **정의된 특성 폴더 이름**



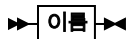
그룹화되지 않은 특성 폴더 이름



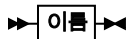
이름-값 쌍



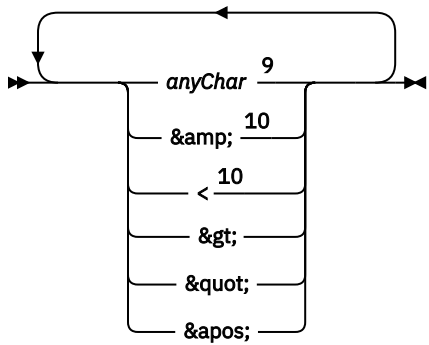
그룹 이름



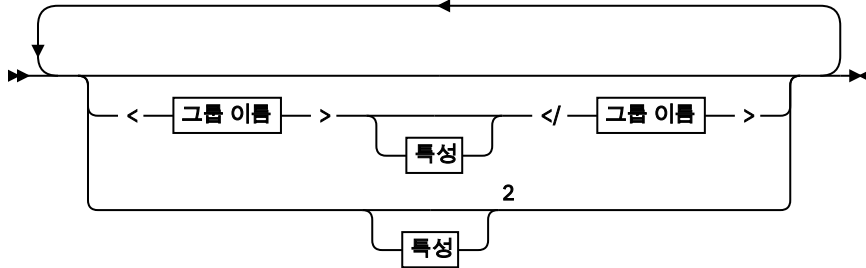
요소 이름



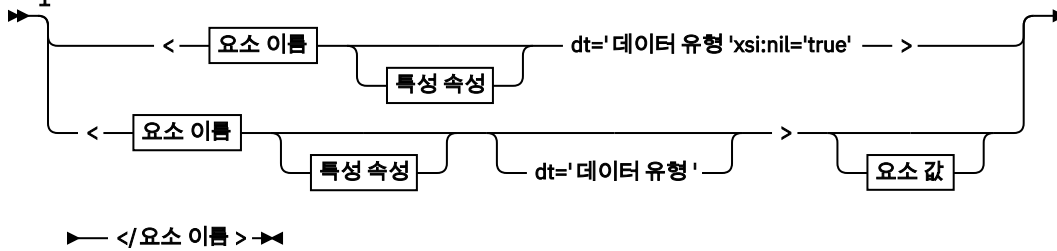
요소 값



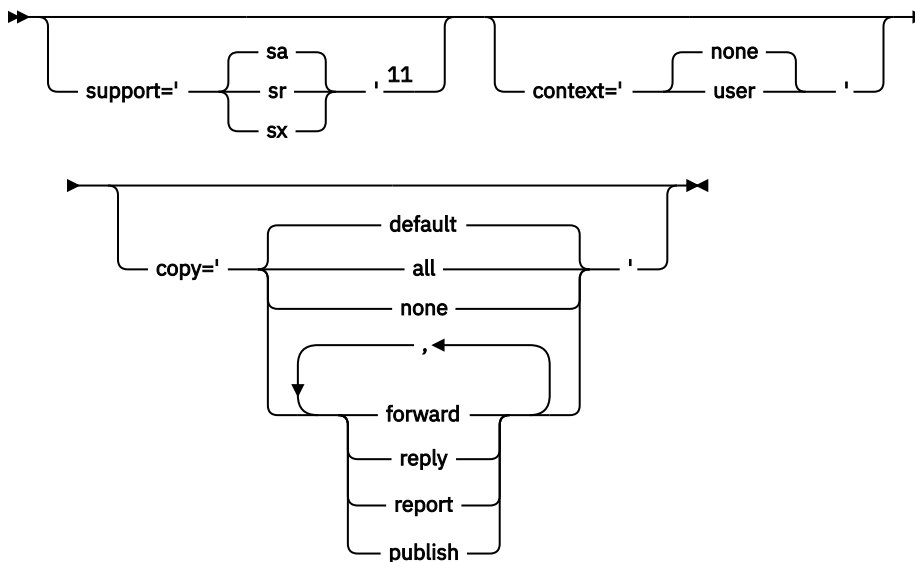
특성



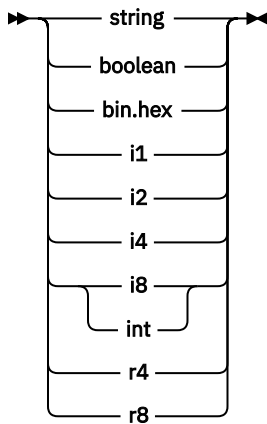
특성



특성 속성



데이터 유형



참고:

- 1 큰따옴표 또는 작은따옴표가 유효합니다.
- 2 올바르지 않은 특성 이름을 사용하지 마십시오. 531 페이지의 『올바르지 않은 특성 이름』의 내용을 참조하십시오. 예약된 특성 이름은 정의된 용도로만 사용하십시오. 530 페이지의 『정의된 특성 이름』의 내용을 참조하십시오.
- 3 이름은 소문자여야 합니다.
- 4 하나의 psc 및 pscr 폴더만 지원됩니다.
- 5 WebSphere Application Server Service Integration Bus는 후속 MQRFH2 헤더의 sib, sib\_context 및 sib\_usr 폴더를 무시하고 첫 번째 MQRFH2 헤더의 속성만 중요합니다.
- 6 하나의 usr 폴더가 MQRFH2에 있어야 합니다. usr 폴더의 특성은 한 번만 발생해야 합니다.
- 7 첫 번째 mq 폴더의 특성만 중요합니다. 폴더가 UTF-8이면, 단일 바이트 UTF-8 문자만 지원됩니다. 유일한 공백 문자는 유니코드 U+0020입니다.
- 8 유효한 문자는 W3C XML 스펙에서 정의되며, 기본적으로 유니코드 카테고리 L1, Lu, Lo, Lt, N1, Mc, Mn, Lm, 및 Nd(으)로 구성됩니다. 유니코드 문자 범주를 참조하십시오.
- 9 모든 문자가 중요합니다. 선두 문자 및 후미 문자 공백은 요소 값의 일부입니다.
- 10 올바르지 않은 문자를 사용하지 마십시오. 530 페이지의 『올바르지 않은 문자』의 내용을 참조하십시오. 이러한 올바르지 않은 문자 대신 이스케이프 순서를 사용하십시오.
- 11 지원 특성 속성은 mq 폴더에서만 올바릅니다.

## 폴더 이름

*NameValueData*에는 단일 폴더가 포함되어 있습니다. 다중 폴더를 작성하려면 다중 *NameValueData* 필드를 작성하십시오. 메시지 내에서 단일 MQRFH2 헤더에 다중 *NameValueData* 필드를 작성할 수 있습니다. 또는 여러 개의 *NameValueData* 필드를 포함하는 여러 개의 체인된 MQRFH2 헤더를 작성할 수 있습니다.

MQRFH2 헤더의 순서 및 *NameValueData* 필드의 순서는 폴더의 논리 콘텐츠에 차이가 없습니다. 하나의 메시지에 동일한 폴더가 둘 이상 있는 경우 폴더가 전체적으로 구문 분석됩니다. 동일한 폴더의 여러 인스턴스에서 동일한 특성이 발생하는 경우 목록으로 구문 분석됩니다.

MQRFH2의 올바른 구문 분석은 폴더가 실제로 메시지에 저장될 수 있는 대체 방법의 영향을 받지 않습니다.

네 개의 폴더는 이 규칙을 따르지 않습니다. mq, sib, sib\_context 및 sib\_usr 폴더의 첫 번째 인스턴스만 구문 분석됩니다.

체인된 MQRFH2 헤더의 결합된 콘텐츠에서 동일한 특성이 두 번 이상 발생하는 경우, 특성의 첫 번째 인스턴스만 구문 분석됩니다. 특성이 API 호출(예: MQSETMP)을 사용하여 설정되고 애플리케이션에 의해 MQRFH2에 직접 추가되는 경우 API 호출이 우선합니다.

폴더 이름은 이름-값 쌍 또는 그룹을 포함하는 폴더의 이름입니다. 그룹 및 이름-값 쌍은 폴더 트리의 동일한 레벨에서 혼합될 수 있습니다. 521 페이지의 그림 1의 내용을 참조하십시오. 그룹 이름 및 요소 이름을 결합하지 마십시오. 521 페이지의 그림 2의 내용을 참조하십시오.

---

```
<group1><nvp1>value</nvp1></group1><group2><nvp2>value</nvp2></group2>
<group3><nvp1>value</nvp1></group3><nvp3>value</nvp3>
```

그림 1. 그룹 및 이름-값 쌍의 올바른 사용

---

```
<group1><nvp1> value </nvp1> value </group1>
```

그림 2. 그룹 및 이름-값 쌍의 잘못된 사용

---

올바르지 않거나 예약된 폴더 이름을 사용하지 마십시오. 530 페이지의 『올바르지 않은 경로 이름』 및 530 페이지의 『예약된 폴더 또는 특성 폴더 이름』의 내용을 참조하십시오. 정의된 폴더 이름은 정의된 용도로만 사용하십시오. 522 페이지의 『정의된 폴더 이름』의 내용을 참조하십시오.

'content=properties' 속성을 폴더 이름 태그에 추가하면 폴더가 특성 폴더가 됩니다. 521 페이지의 그림 3의 내용을 참조하십시오.

---

```
<myFolder></myfolder>
<myPropertyFolder contents='properties'></myPropertyFolder>
```

그림 3. 폴더 및 특성 폴더의 예

---

폴더 이름은 대소문자를 구분합니다. 폴더 이름과 특성 폴더 이름은 동일한 네임스페이스를 공유합니다. 이름은 달라야 합니다. 521 페이지의 그림 4의 Folder1은(는) 521 페이지의 그림 5의 Folder2과(와) 다른 이름이어야 합니다.

---

```
< Folder1 ><NVP1> value </NVP1></ Folder1 >
```

그림 4. Folder1 네임스페이스

---

```
< Folder2 content='properties'>< Property1 > value </ Property1 ></ Folder2 >
```

그림 5. Folder2 네임스페이스

---

다른 폴더에 있는 그룹, 특성, 이름-값 쌍에는 다른 네임스페이스가 있습니다. 521 페이지의 그림 5의 Property1은(는) 521 페이지의 그림 6의 Property1과(와) 다른 특성입니다.

---

```
<Folder3 content='properties'>< Property1 > value </ Property1 ></Folder3>
```

그림 6. Folder3 네임스페이스

---

특성 폴더는 두 가지 중요한 면에서 비특성 폴더와 다릅니다.

1. 특성 폴더에는 특성이 있고 비특성 폴더에는 이름-값 쌍이 있습니다. 폴더는 구문론적으로 약간 다릅니다.

- 특성 MQI 또는 JMS 메시지 특성과 같은 정의된 인터페이스를 사용하여 메시지 특성에 액세스하십시오. 인터페이스는 MQRFH2의 특성 폴더가 잘 형성되었는지 확인합니다. 올바르게 생성된 특성 폴더는 다른 플랫폼 및 다른 릴리스의 큐 관리자 사이에 상호 운용 가능합니다.

메시지 특성 MQI는 MQRFH2을(를) 읽고 쓸 수 있는 강력한 방법이며 MQRFH2을(를) 올바르게 구문 분석하는 데 어려움이 없습니다.

## 정의된 폴더 이름

정의된 폴더 이름은 IBM MQ 또는 다른 제품에서 사용하도록 예약된 폴더의 이름입니다. 동일한 이름의 폴더를 작성하지 말고 폴더에 사용자의 이름-값 쌍을 추가하지 마십시오. 정의된 폴더는 psc 및 pscr입니다.

psc 및 pscr은 큐된 발행/구독에서 사용됩니다.

MQMF\_SEGMENT 또는 MQMF\_SEGMENTATION\_ALLOWED을(를) 사용하는 세그먼트화된 메시지는 정의된 폴더 이름의 MQRFH2을(를) 포함할 수 없습니다. MQPUT은 이유 코드 2443, MQRC\_SEGMENTATION\_NOT\_ALLOWED와 함께 실패합니다.

## 정의된 특성 폴더 이름

정의된 특성 폴더 이름은 IBM MQ 또는 다른 제품이 사용하는 특성 폴더의 이름입니다. 폴더의 이름 및 해당 콘텐츠를 특성 폴더를 참조하십시오. 정의된 특성 폴더 이름은 IBM MQ가 예약한 모든 폴더 이름의 서브세트입니다. 530 페이지의 『예약된 폴더 또는 특성 폴더 이름』의 내용을 참조하십시오.

정의된 특성 폴더에 저장된 모든 요소는 특성입니다. 정의된 특성 폴더에 저장된 요소에는 content='properties' 속성이 없어야 합니다.

정의된 특성 폴더 usr, mq\_usr 및 sib\_usr에만 특성을 추가할 수 있습니다. mq 및 sib와 같은 다른 특성 폴더에서 IBM MQ는 인식되지 않는 특성을 무시하거나 버립니다.

정의된 각 특성 폴더에 대한 설명은 애플리케이션 프로그램이 사용할 수 있는 IBM MQ가 정의된 특성을 나열합니다. JMS 특성을 설정하거나 가져와서 일부 특성에 간접적으로 액세스하고 MQSETMP 및 MQINQMP MQI 호출을 사용하여 일부에 직접 액세스할 수 있습니다.

정의된 특성 폴더는 또한 IBM MQ가 예약되었지만 애플리케이션이 액세스 권한이 없는 기타 특성을 포함합니다. 예약된 특성의 이름은 나열되지 않습니다. 예약된 특성이 usr, mq\_usr, sib\_usr 특성 폴더에 없습니다. 그러나 올바르게 지 않은 특성 이름으로 특성을 작성하지 마십시오. 531 페이지의 『올바르지 않은 특성 이름』의 내용을 참조하십시오.

## 특성 폴더

### jms

jms에는 JMS 헤더 필드와 MQMD에서 완전히 표현할 수 없는 JMSX 속성이 포함되어 있습니다. jms 폴더는 항상 JMS MQRFH2에 있습니다.

표 515. jms 특성 이름, 동의어, 데이터 유형, 폴더			
특성 동의어	특성 이름	데이터 유형	폴더
JMSDestination	jms.Dst	string	<jms><Dst> destination </Dst></jms>
JMSExpiration	jms.Exp	i8	<jms><Exp> expiration </Exp></jms>
JMSCorrelation	jms.Cid	string	<jms><Cid> correlationId </Cid></jms>
JMSDelivery	jms.Dlv	i4	<jms><Dlv> delivery </Dlv></jms>

표 515. <i>jms</i> 특성 이름, 동의어, 데이터 유형, 폴더 (계속)			
특성 동의어	특성 이름	데이터 유형	폴더
JMSPriority	<code>jms.Priority</code>	i4	<code>&lt;jms&gt;&lt;Pri&gt; priority &lt;/Pri&gt;&lt;/jms&gt;</code>
JMSReplyTo	<code>jms.Rto</code>	string	<code>&lt;jms&gt;&lt;Rto&gt; replyToURI &lt;/Rto&gt;&lt;/jms&gt;</code>
JMSTimestamp	<code>jms.Tms</code>	i8	<code>&lt;jms&gt;&lt;Tms&gt; timestamp &lt;/Tms&gt;&lt;/jms&gt;</code>
JMSXGroupID	<code>jms.Gid</code>	string	<code>&lt;jms&gt;&lt;Gid&gt; groupId &lt;/Gid&gt;&lt;/jms&gt;</code>
JMSXGroupSeq	<code>jms.Seq</code>	i4	<code>&lt;jms&gt;&lt;Seq&gt; messageSequenceNo &lt;/Seq&gt;&lt;/jms&gt;</code>

*jms* 폴더에 사용자 고유의 특성을 추가하지 마십시오.

### **mcd**

*mcd*에는 메시지의 형식을 설명하는 특성이 포함되어 있습니다. 예를 들어, 메시지 서비스 도메인 *Msd* 특성은 `JMSTextMessage`, `JMSBytesMessage`, `JMSStreamMessage`, `JMSMapMessage`, `JMSObjectMessage` 또는 널로 *JMS* 메시지를 식별합니다.

*mcd* 폴더는 항상 `MQRFH2`을(를) 포함하는 *JMS* 메시지에 있습니다.

이 메시지는 항상 *IBM Integration Bus*에서 전송된 `MQRFH2`을(를) 포함하는 메시지에 표시됩니다. 이 폴더는 메시지의 도메인, 형식, 유형 및 메시지 세트를 설명합니다.

표 516. <i>mcd</i> 특성 이름, 동의어, 데이터 유형 및 폴더			
특성 동의어	특성 이름	데이터 유형	폴더
	<code>mcd.Msd</code>	string	<code>&lt;mcd&gt;&lt;Msd&gt;messageDomain&lt;/Msd&gt;&lt;/mcd&gt;</code>
	<code>mcd.Set</code>	string	<code>&lt;mcd&gt;&lt;Set&gt;messageDomain&lt;/Set&gt;&lt;/mcd&gt;</code>
	<code>mcd.Type</code>	string	<code>&lt;mcd&gt;&lt;Type&gt;messageDomain&lt;/Type&gt;&lt;/mcd&gt;</code>
	<code>mcd.Fmt</code>	string	<code>&lt;mcd&gt;&lt;Fmt&gt;messageDomain&lt;/Fmt&gt;&lt;/mcd&gt;</code>

*mcd* 폴더에 사용자 고유의 특성을 추가하지 마십시오.

### **mq\_usr**

*mq\_usr*에는 *JMS* 사용자 정의 특성으로 표시되지 않는 애플리케이션 정의 특성이 포함되어 있습니다. *JMS* 요구사항을 충족하지 않는 특성은 이 폴더에 배치할 수 있습니다.

*mq\_usr* 폴더에서 특성을 작성할 수 있습니다. *mq\_usr*에서 작성하는 특성은 `content='properties'` 속성을 사용하여 새 폴더에서 작성하는 특성과 유사합니다.

### **sib**

*sib*에는 *WebSphere Application Server* 서비스 통합 버스(*WAS/SIB*) 시스템 메시지 특성이 포함되어 있습니다. *sib* 속성은 지원되는 유형이 아니기 때문에 *IBM MQ JMS* 애플리케이션에 *JMS* 속성으로 노출되지 않습니다. 예를 들어, 일부 *sib* 특성은 바이트 배열이므로 *JMS* 특성으로 표시할 수 없습니다. 일부 *sib* 특성은

WAS/SIB 애플리케이션에 JMS\_IBM\_\* 특성으로 표시됩니다. 이러한 특성에는 정방향 및 역방향 라우팅 경로 특성이 포함됩니다.

sib 폴더에 사용자 고유의 특성을 추가하지 마십시오.

### sib\_context

sib\_context에는 WAS/SIB 사용자 애플리케이션 또는 JMS 특성에 노출되지 않은 WAS/SIB 시스템 메시지 특성이 포함되어 있습니다. sib\_context에는 웹 서비스에 사용되는 보안 및 트랜잭션 특성이 포함되어 있습니다.

sib\_context 폴더에 사용자 고유의 특성을 추가하지 마십시오.

### sib\_usr

sib\_usr에는 지원되는 유형이 아니므로 JMS 사용자 특성으로 표시되지 않는 WAS/SIB 사용자 메시지 특성이 포함됩니다. sib\_usr은(는) SIMessage 인터페이스에서 WAS/SIB 애플리케이션에 노출됩니다. [서비스 통합 개발](#)을 참조하십시오.

sib\_usr 특성의 유형은 bin.hex이어야 하며 값은 올바른 형식이어야 합니다. IBM MQ 애플리케이션이 잘못된 형식으로 폴더에 bin.hex 유형의 형식을 쓰는 경우 애플리케이션이 IOException을 수신합니다. 특성의 데이터 유형이 bin.hex가 아닌 경우 애플리케이션은 ClassCastException을 수신합니다.

이 폴더를 사용하여 JMS 사용자 특성을 WAS/SIB에 사용 가능하게 하지 마십시오. 대신 usr 폴더를 사용하십시오.

sib\_usr 폴더에서 특성을 작성할 수 있습니다.

### usr

usr에는 메시지와 연관된 애플리케이션 정의 JMS 특성이 포함되어 있습니다. usr 폴더는 애플리케이션이 애플리케이션 정의 특성을 설정한 경우에만 표시됩니다.

usr은(는) 기본 특성 폴더입니다. 특성이 폴더 이름 없이 설정된 경우 usr 폴더에 배치됩니다.

표 517. usr 특성 이름, 동의어, 데이터 유형, 폴더			
특성 동의어	특성 이름	데이터 유형	폴더
	usr.contentType	string	<usr><contentType>text/xml; charset=utf-8</contentType></usr>
	usr.endpointURL	string	<usr><endpointURL> URI </endpointURL></usr>
	usr.targetService	string	<usr><targetService> serviceName </targetService></usr>
	usr.soapAction	string	<usr><soapAction> name </soapAction></usr>
	usr.transportVersion	string	<usr><transportVersion> version </transportVersion></usr>

usr 폴더에서 특성을 작성할 수 있습니다.

MQMF\_SEGMENT 또는 MQMF\_SEGMENTATION\_ALLOWED을(를) 사용하는 세그먼트화된 메시지는 정의된 특성 폴더 이름을 가진 MQRFH2을(를) 포함할 수 없습니다. MQPUT은 이유 코드 2443, MQRC\_SEGMENTATION\_NOT\_ALLOWED와 함께 실패합니다.



## 그룹화되지 않은 특성 폴더 이름

### ibm

ibm에는 IBM MQ만 사용하는 특성이 포함됩니다.

표 518. <i>ibm</i> 특성 이름, 동의어, 데이터 유형 및 폴더			
특성 동의어	특성 이름	데이터 유형	폴더
	ibm.rfp	string	<ibm><rfp>fingerprint</rfp></ibm>

ibm 폴더에 사용자 고유의 특성을 추가하지 마십시오.

### mq

mq에는 IBM MQ만 사용하는 특성이 포함됩니다.

다음 제한사항이 mq 폴더의 특성에 적용됩니다.

- 메시지에서 중요한 첫 번째 mq 폴더에 있는 특성에 대해서만 MQ가 수행합니다. 메시지의 다른 mq 폴더에 있는 특성은 무시됩니다.
- 폴더에서 단일 바이트 UTF-8 문자만 허용됩니다. 폴더에서 멀티바이트 문자를 사용하면 구문 분석에 실패할 수 있으므로 메시지가 거부될 수 있습니다.
- 폴더에 이스케이프 문자열을 사용하지 마십시오. 이스케이프 문자열은 요소의 실제 값으로 처리됩니다.
- 유니코드 문자 U+0020만 폴더 내의 공백으로 처리됩니다. 다른 모든 문자는 중요한 문자로 처리되며 폴더의 구문 분석이 실패할 수 있으므로 메시지가 거부됩니다.

mq 폴더의 구문 분석에 실패하거나 폴더가 이러한 제한사항을 준수하지 않는 경우 이유 코드 2527, MQRC\_RFH\_RESTRICTED\_FORMAT\_ERR과(와) 함께 메시지가 거부됩니다.

mq 폴더에 사용자 고유의 특성을 추가하지 마십시오.

### mqema

mqema에는 WebSphere Application Server만 사용하는 특성이 포함됩니다. 폴더가 mqext(으)로 대체되었습니다.

mqema 폴더에 사용자 고유의 특성을 추가하지 마십시오.

### mqext

mqext에는 다음 유형의 특성이 포함되어 있습니다.

- WebSphere Application Server에서만 사용되는 특성
- 메시지 지연 전달과 관련된 특성

IBM 정의 특성 또는 사용된 전달 지연 중 적어도 하나가 애플리케이션에 설정된 경우에만 이 폴더가 있습니다.

표 519. <i>mqext</i> 특성 이름, 동의어, 데이터 유형 및 폴더			
특성 동의어	특성 이름	데이터 유형	폴더
JMSArmCorrelator	mqext.Arm	string	<mqext><Arm>armCorrelator</Arm></mqext>
JMSRMCorrelator	mqext.Wrm	string	<mqext><Wrm>wrmCorrelator</Wrm></mqext>
JMSDeliveryTime	mqext.Dlt	i8	<mqext><Dlt>DeliveryTime</Dlt></mqext>

표 519. <i>mqext</i> 특성 이름, 동의어, 데이터 유형 및 폴더 (계속)			
특성 동의어	특성 이름	데이터 유형	폴더
JMSDeliveryDelay	mqext.Dly	i8	<mqext><Dly>DeliveryTime</Dly></mqext>

mqext 폴더에 사용자 고유의 특성을 추가하지 마십시오.

### mqps

mqps에는 IBM MQ 발행/구독에서만 사용되는 특성이 포함되어 있습니다. 통합 발행/구독 특성 중 적어도 하나가 애플리케이션에 설정된 경우에만 이 폴더가 있습니다.

표 520. <i>mqps</i> 특성 이름, 동의어, 데이터 유형 및 폴더			
특성 동의어	특성 이름	데이터 유형	폴더
MQTopicString	mqps.Top	string	<mqps><Top>topicString</Top></mqps>
MQSubUserData	mqps.Sud	string	<mqps><Sud>subscriberUserData...</Sud></mqps>
MQIsRetained	mqps.Ret	boolean	<mqps><Ret>isRetained</Ret></mqps>
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>
MQPubStrInpData	mqpse.Sid	string	<mqps><Sid>publicationData</Sid></mqps>
MQPubFormat	mqpse.Pfmt	i8	<mqps><Pfmt>messageFormat</Pfmt></mqps>

mqps 폴더에 사용자 고유의 특성을 추가하지 마십시오.

### mq\_svc

mq\_svc에는 SupportPac MA93이(가) 사용하는 특성이 있습니다.

mq\_svc 폴더에 사용자 고유의 특성을 추가하지 마십시오.

### mqtt

mqtt에는 MQ Telemetry에서 사용하는 특성이 포함됩니다.

표 521. <i>mqtt</i> 특성 이름, 동의어, 데이터 유형, 폴더			
특성 동의어	특성 이름	데이터 유형	폴더
	mqtt.clientId	string	<mqtt><clientId> topicString </clientId></mqtt>
	mqtt.qos	i4	<mqtt><qos> qualityOfService </qos></mqtt>

표 521. mqtt 특성 이름, 동의어, 데이터 유형, 폴더 (계속)			
특성 동의어	특성 이름	데이터 유형	폴더
	mqtt.msgid	string	<mqtt><msgid> <i>messageIdentifier</i> </msgid></mqtt>

mqtt 폴더에 사용자 고유의 특성을 추가하지 마십시오.

MQMF\_SEGMENT 또는 MQMF\_SEGMENTATION\_ALLOWED을(를) 사용하는 세그먼트화된 메시지는 그룹화되지 않은 특성 폴더 이름을 가진 MQRFH2을(를) 포함할 수 없습니다. MQPUT은 이유 코드 2443, MQRC\_SEGMENTATION\_NOT\_ALLOWED와 함께 실패합니다.

## 이름-값 쌍

구문 다이어그램에서 "이름-값 쌍"은 일반 폴더의 콘텐츠를 설명합니다. 일반 폴더에는 그룹 및 요소가 포함됩니다. 요소는 이름-값 쌍입니다. 그룹에는 요소 및 다른 그룹이 포함됩니다.

트리의 용어에서 요소는 Leaf 노드이고 그룹은 내부 노드입니다. 내부 노드 및 폴더(루트 노드)에는 내부 노드와 Leaf 노드의 혼합이 포함될 수 있습니다. 하나의 노드가 동시에 내부 노드와 Leaf 노드 둘 다 될 수는 없습니다. [521 페이지의 그림 2](#)의 내용을 참조하십시오.

## 특성

구문 다이어그램에서 "특성"은 특성 폴더의 콘텐츠를 설명합니다. 특성 폴더에는 그룹 및 특성이 포함됩니다. 특성은 선택적 데이터 유형 속성이 있는 이름-값 쌍입니다. 그룹에는 특성 및 다른 그룹이 포함됩니다.

트리의 용어에서 특성은 Leaf 노드이고 그룹은 내부 노드입니다. 내부 노드 및 특성 폴더(루트 노드)에는 내부 노드와 Leaf 노드의 혼합이 포함될 수 있습니다. 하나의 노드가 동시에 내부 노드와 Leaf 노드 둘 다 될 수는 없습니다. [521 페이지의 그림 2](#)의 내용을 참조하십시오.

## 특성

메시지 특성은 특성 폴더에 있는 이름-값 쌍입니다. 선택적으로 데이터 유형 속성 및 특성 속성을 포함할 수 있습니다. 예를 들어, 다음 코드를 참조하십시오. 데이터 유형 속성이 생략된 경우 특성 유형은 string입니다.

```
<pf><p1 dt='i8' > value </p1></pf>
```

메시지 특성의 이름은 XML과 같은 <> 구문이 점으로 대체된 전체 경로 이름입니다. 예를 들어, myPropertyFolder1.myGroup1.myGroup2.myProperty1 는 다음과 같이 NameValueData 문자열에 맵핑됩니다. 문자열은 더 읽기 쉽도록 형식화됩니다.

```
<myPropertyFolder1>
  <myGroup1>
    <myGroup2>
      <myProperty1>value</myProperty1>
    </myGroup2>
  </myGroup1>
</myPropertyFolder1>
```

특성 폴더에는 여러 특성이 포함될 수 있습니다. 예를 들어, [527 페이지의 그림 7](#)의 특성은 새 클러스터에서 판매 큐를 분리하고 게이트웨이 클러스터 전송 큐를 분리하기 위한 변경사항의 특성 폴더에 맵핑됩니다.

```
myPropertyFolder1.myProperty4
myPropertyFolder1.myGroup1.myGroup2.myProperty1
myPropertyFolder1.myGroup1.myGroup2.myProperty2
myPropertyFolder1.myGroup1.myProperty3
```

그림 7. 루트 이름이 동일한 여러 특성

```

<myPropertyFolder1>
  <myProperty4>value</myProperty4>
  <myGroup1>
    <myGroup2>
      <myProperty1>value</myProperty1>
      <myProperty2>value</myProperty2>
    </myGroup2>
    <myProperty3>value</myProperty3>
  </myGroup1>
</myPropertyFolder1>

```

그림 8. 다중 특성 이름 맵핑

## 이름

이름은 문자 또는 밑줄로 시작해야 합니다. 콜론을 포함하지 않아야 하고 마침표로 끝나지 않아야 하며 문자, 숫자, 밑줄, 하이픈 및 점만 포함해야 합니다. 유효한 문자는 W3C XML 스펙에서 정의되며, 기본적으로 유니코드 카테고리 Ll, Lu, Lo, Lt, Nl, Mc, Mn, Lm, 및 Nd(으)로 구성됩니다. [유니코드 문자 범주를 참조하십시오](#).

특성 또는 이름-값 쌍의 전체 경로는 530 페이지의 『올바르지 않은 경로 이름』에서 설명하는 규칙을 위반하지 않아야 합니다. 경로는 4095바이트로 제한되고 유니코드 호환성 문자를 포함하지 않아야 하며 XML 문자열로 시작하지 않아야 합니다.

## 그룹 이름

그룹 이름의 구문은 이름과 동일합니다. 그룹 이름은 선택사항입니다. 특성 및 이름-값 쌍은 폴더의 루트에 배치될 수 있습니다. 특성 및 이름-값 쌍을 구성하는 데 도움이 되는 경우 그룹을 사용하십시오.

## 요소 이름

요소 이름의 구문은 이름과 동일합니다.

## 요소 값

요소 값에는 < *Element name* > 태그와 < /*Element name* > 사이의 모든 공백이 포함됩니다. 값에 < 및 & 문자를 사용하지 마십시오. 그러면 < 및 &(으)로 바꾸십시오.

## 특성 속성

특성 속성은 특성 디스크립터 필드를 맵핑합니다. 맵핑은 다음과 같습니다.

### 지원

#### sa (기본값)

MQPD\_SUPPORT\_OPTIONAL

#### sr

MQPD\_SUPPORT\_REQUIRED

#### sx

MQPD\_SUPPORT\_REQUIRED\_IF\_LOCAL

### 컨텍스트

#### none (기본값)

MQPD\_NO\_CONTEXT

#### 사용자

MQPD\_USER\_CONTEXT

## CopyOptions

### forward

MQPD\_COPY\_FORWARD

### 응답

MQPD\_COPY\_REPLY

### 보고

MQPD\_COPY\_REPORT

### 발행(Publish)

MQPD\_COPY\_PUBLISH

### 모두

MQPD\_COPY\_ALL

all을 다른 옵션과 결합하여 사용하지 마십시오.

### 기본값

MQPD\_COPY\_DEFAULT

default를 다른 옵션과 결합하여 사용하지 마십시오. default는 forward + report + publish와 동일합니다.

### 없음

MQPD\_COPY\_NONE

none을 다른 옵션과 결합하여 사용하지 마십시오.

Support 특성 속성은 mq 폴더의 특성에만 적용 가능합니다.

Context 및 CopyOptions 특성 속성은 모든 특성 폴더에 적용 가능합니다.

## 데이터 유형

MQRFH2 데이터 유형은 다음과 같이 메시지 특성 유형에 매핑됩니다.

표 522. 데이터 유형 매핑	
MQRFH2 데이터 유형	메시지 특성 유형
bin.hex	MQBYTE[]
boolean	MQBOOL
i1	MQINT8
i2	MQINT16
i4	MQINT32
i8	MQINT64
r4	MQFLOAT32
r8	MQFLOAT64
string	MQCHAR[]

데이터 유형이 없는 요소는 string 유형으로 가정합니다.

널 값은 요소 속성 `xsi:nil='true'` (으)로 표시됩니다. 널이 아닌 값에는 `xsi:nil='false'` 속성을 사용하지 마십시오. 예를 들어, 다음 특성에는 널값이 있습니다.

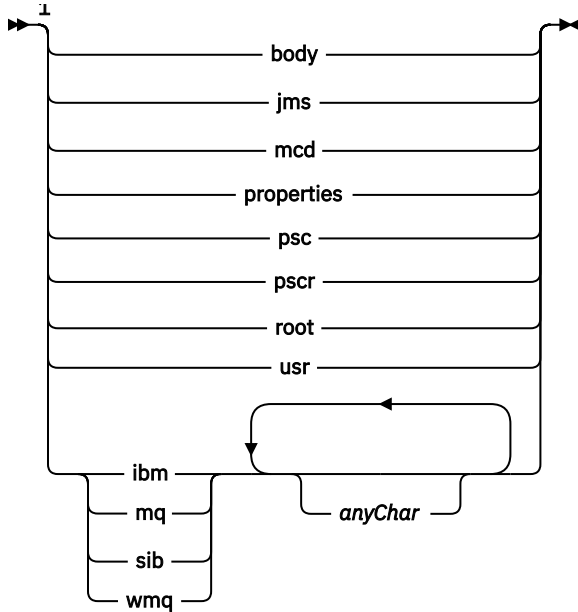
```
<NullProperty  
xsi:nil='true'></NullProperty>
```

바이트 또는 문자열 특성에는 비어 있는 값이 있을 수 있습니다. 비어 있는 값은 길이가 요소 값이 0인 MQRFH2 요소로 표시됩니다. 예를 들어, 다음 특성에는 비어 있는 값이 있습니다.

```
<EmptyProperty></EmptyProperty>
```

## 예약된 폴더 또는 특성 폴더 이름

다음 문자열 중 하나로 시작하지 않는 폴더의 이름 또는 특성 폴더를 제한하십시오. 접두부는 IBM에서 작성한 폴더 또는 특성 이름을 위해 예약됩니다.

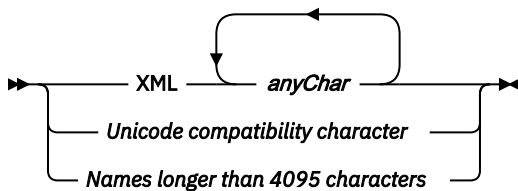


참고:

- 1 예약된 폴더 또는 특성 이름에는 소문자와 대문자의 혼합이 포함됩니다.

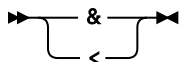
## 올바르지 않은 경로 이름

다음 문자열을 포함하지 않는 이름-값 쌍의 전체 경로 또는 특성을 제한하십시오.



## 올바르지 않은 문자

리터럴 "&" 및 "<" 대신에 이스케이프 시퀀스 &amp; 및 <을(를) 항상 사용하십시오.

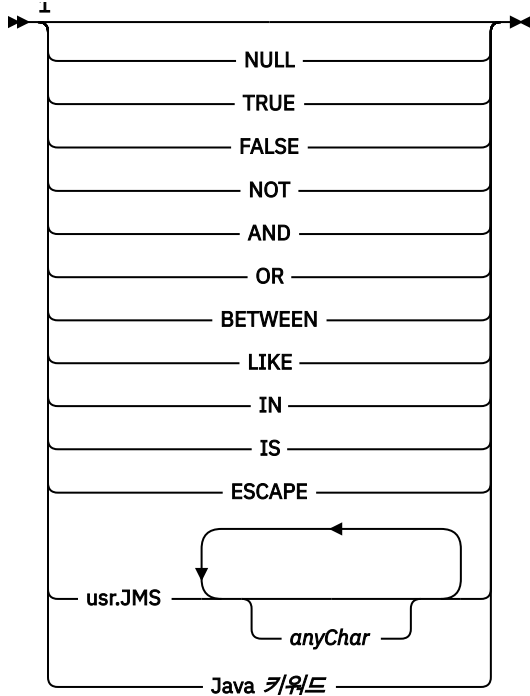


## 정의된 특성 이름

정의된 특성 이름은 IBM MQ 또는 기타 제품에서 정의되고 IBM MQ 및 사용자 애플리케이션에서 사용되는 특성의 이름입니다. 정의된 특성은 정의된 특성 폴더에만 있습니다. 정의된 특성 이름은 특성 폴더의 설명에서 설명합니다. 특성 폴더를 참조하십시오.

## 올바르지 않은 특성 이름

다음 규칙과 일치하는 특성 이름을 구성하지 마십시오. 규칙은 특성 요소 이름뿐만 아니라 특성에 이름을 지정하는 전체 특성 경로에 적용됩니다.



참고:

- 1 올바른지 않은 특성 이름에는 대문자와 소문자의 결합이 포함될 수 있습니다.

## 올바르지 않은 속성

특성 폴더 및 특성은 지원되는 [528 페이지의 『특성 속성』](#) 및 [529 페이지의 『데이터 유형』](#) 만 포함할 수 있습니다.

지원되지 않는 XML형 속성(예: 특성 폴더 또는 특성에 포함된 인용된 문자열 값이 있는 이름)을 제거할 수 있습니다.

MQRFH2 헤더에 남아 있는 비특성 요소 또는 비특성 폴더에 포함된 XML형 속성입니다.

## MQRMH - 참조 메시지 헤더

MQRMH 구조는 참조 메시지 헤더의 형식을 정의합니다. 이 헤더는 하나의 큐 관리자에서 다른 큐 관리자로 벌크 데이터라고 하는 매우 큰 양의 데이터를 전송하기 위해 사용자 작성 메시지 채널 엑시트와 함께 사용됩니다. 일반 메시징과의 차이점은 벌크 데이터가 큐에 저장되지 않는다는 것입니다. 대신 벌크 데이터에 대한 참조만 큐에 저장됩니다. 이렇게 하면 소수의 매우 큰 메시지로 인해 IBM MQ 자원이 고갈될 가능성이 줄어듭니다.

## 가용성

MQRMH 구조는 다음 플랫폼에서 사용 가능합니다.

- ▶ AIX AIX
- ▶ IBM i IBM i
- ▶ Linux Linux
- ▶ Windows Windows

및 해당 시스템에 연결된 IBM MQ 클라이언트

## 형식 이름

MQFMT\_REF\_MSG\_HEADER

## 문자 세트 및 인코딩

MQRMH의 문자 데이터 및 오프셋 필드에 의해 주소 지정된 문자열은 로컬 큐 관리자의 문자 세트에 있어야 합니다. 이는 **CodedCharSetId** 큐 관리자 속성에 의해 지정됩니다. MQRMH의 숫자 데이터는 고유 시스템 인코딩에 있어야 합니다. 이는 C 프로그래밍 언어에 대해 MQENC\_NATIVE의 값으로 지정됩니다.

MQRMH의 문자 세트 및 인코딩을 다음을 사용하여 *CodedCharSetId* 및 *Encoding* 필드로 설정하십시오.

- MQMD(MQRMH 구조가 메시지 데이터의 시작에 있는 경우) 또는
- MQRMH 구조에 선행하는 헤더 구조(다른 모든 경우).

## 사용법

애플리케이션은 MQRMH로 구성된 메시지를 넣지만 벌크 데이터는 생략합니다. 메시지 채널 에이전트(MCA)가 전송 큐에서 메시지를 읽을 때 사용자 제공 메시지 엑시트가 참조 메시지 헤더를 처리하기 위해 호출됩니다. MCA에서 채널을 통해 메시지를 다음 큐 관리자에 송신하기 전에, 엑시트는 MQRMH 구조에서 식별한 벌크 데이터를 참조 메시지에 추가할 수 있습니다.

수신 종료 시 참조 메시지를 기다리는 메시지 엑시트가 있어야 합니다. 참조 메시지가 수신되면 엑시트가 메시지 에서 MQRMH 다음에 오는 벌크 데이터에서 오브젝트를 작성한 후 벌크 데이터 없이 참조 메시지에서 전달해야 합니다. 나중에 참조 메시지를 읽는 애플리케이션에서 벌크 데이터 없이 큐에서 참조 메시지를 검색할 수 있습니다.

일반적으로, MQRMH 구조는 메시지에 있는 전부입니다. 그러나 메시지가 전송 큐에 있는 경우 하나 이상의 추가 헤더가 MQRMH 구조 앞에 옵니다.

참조 메시지도 분배 목록에 송신할 수 있습니다. 이런 경우 메시지가 전송 큐에 있을 때 MQDH 구조 및 관련 레코드는 MQRMH 구조에 우선합니다.

**참고:** 메시지 엑시트가 올바르게 처리할 수 없으므로 참조 메시지를 세그먼트화된 메시지로 전송하지 마십시오.

## 데이터 변환

데이터 변환을 위해 MQRMH 구조 변환에는 소스 환경 데이터, 소스 오브젝트 이름, 목적지 환경 데이터 및 목적지 오브젝트 이름의 변환이 포함됩니다. 구조 시작에서 *StrucLength* 바이트 내의 기타 바이트는 데이터 변환 후에 제거되거나 정의되지 않은 값을 가집니다. 다음 명령문이 모두 true인 경우, 벌크 데이터가 변환됩니다.

- 데이터 변환이 수행될 때 벌크 데이터는 메시지에 존재합니다.
- MQRMH의 *Format* 필드에는 MQFMT\_NONE이 아닌 값이 있습니다.
- 사용자 작성 데이터 변환 엑시트는 지정된 형식 이름으로 존재합니다.

그러나 메시지가 큐에 있는 경우 일반적으로 벌크 데이터가 메시지에 있지 않고 결과적으로 MQGMO\_CONVERT 옵션이 벌크 데이터를 변환함에 주의하십시오.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 523. MQRMH에 대한 MQRMH의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQRMH_STRUC_ID	'RMH~'
버전 (구조 버전 번호)	MQRMH_VERSION_1	1



표 523. MQRMH에 대한 MQRMH의 필드 (계속)		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucLength (고정 필드의 끝에 있는 문자열을 포함하지만 벌크 데이터는 포함하지 않는 MQRMH의 총 길이)	없음	0
인코딩 (벌크 데이터의 숫자 인코딩)	MQENC_NATIVE	환경에 따라 다름
CodedCharSetId (대량 데이터의 문자 세트 ID)	MQCCSI_UNDEFINED	0
형식 (벌크 데이터의 형식 이름)	MQFMT_NONE	공백
플래그 (참조 메시지 플래그)	MQRMHF_NOT_LAST	0
ObjectType (오브젝트 유형)	없음	공백
ObjectInstanceId (오브젝트 인스턴스 ID)	MQOII_NONE	널
SrcEnv길이 (소스 환경 데이터의 길이)	없음	0
SrcEnvOffset (소스 환경 데이터의 오프셋)	없음	0
SrcName길이 (소스 오브젝트 이름의 길이)	없음	0
SrcNameOffset (소스 오브젝트 이름의 오프셋)	없음	0
DestEnv길이 (대상 환경 데이터의 길이)	없음	0
DestEnvOffset (대상 환경 데이터의 오프셋)	없음	0
DestNameLength (대상 오브젝트 이름의 길이)	없음	0
DestNameOffset (대상 오브젝트 이름의 오프셋)	없음	0
DataLogicalLength (벌크 데이터의 길이)	없음	0
DataLogicalOffset (벌크 데이터의 낮은 오프셋)	없음	0
DataLogicalOffset2 (벌크 데이터의 높은 오프셋)	없음	0

**참고:**

- ↵ 기호는 단일 공백 문자를 나타냅니다.
- C 프로그래밍 언어의 매크로 변수MQRMH\_DEFAULT는 테이블에 나열되는 값을 포함합니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오.

```
MQRMH MyRMH = {MQRMH_DEFAULT};
```

## 언어 선언

### MQRMH의 C 선언

```
typedef struct tagMQRMH MQRMH;
struct tagMQRMH {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    StrucLength;      /* Total length of MQRMH, including
                               strings at end of fixed fields, but
                               not the bulk data */
    MQLONG    Encoding;         /* Numeric encoding of bulk data */
    MQLONG    CodedCharSetId;   /* Character set identifier of bulk
                               data */
    MQCHAR8   Format;           /* Format name of bulk data */
    MQLONG    Flags;            /* Reference message flags */
    MQCHAR8   ObjectType;       /* Object type */
};
```

```

MQBYTE24  ObjectInstanceId; /* Object instance identifier */
MQLONG    SrcEnvLength; /* Length of source environment data */
MQLONG    SrcEnvOffset; /* Offset of source environment data */
MQLONG    SrcNameLength; /* Length of source object name */
MQLONG    SrcNameOffset; /* Offset of source object name */
MQLONG    DestEnvLength; /* Length of destination environment
                          data */
MQLONG    DestEnvOffset; /* Offset of destination environment
                          data */
MQLONG    DestNameLength; /* Length of destination object name */
MQLONG    DestNameOffset; /* Offset of destination object name */
MQLONG    DataLogicalLength; /* Length of bulk data */
MQLONG    DataLogicalOffset; /* Low offset of bulk data */
MQLONG    DataLogicalOffset2; /* High offset of bulk data */
};

```

## MQRMH의 COBOL 선언

```

** MQRMH structure
10 MQRMH.
** Structure identifier
15 MQRMH-STRUCID PIC X(4).
** Structure version number
15 MQRMH-VERSION PIC S9(9) BINARY.
** Total length of MQRMH, including strings at end of fixed fields,
** but not the bulk data
15 MQRMH-STRUCLength PIC S9(9) BINARY.
** Numeric encoding of bulk data
15 MQRMH-ENCODING PIC S9(9) BINARY.
** Character set identifier of bulk data
15 MQRMH-CODEDCHARSETID PIC S9(9) BINARY.
** Format name of bulk data
15 MQRMH-FORMAT PIC X(8).
** Reference message flags
15 MQRMH-FLAGS PIC S9(9) BINARY.
** Object type
15 MQRMH-OBJECTTYPE PIC X(8).
** Object instance identifier
15 MQRMH-OBJECTINSTANCEID PIC X(24).
** Length of source environment data
15 MQRMH-SRCENVLENGTH PIC S9(9) BINARY.
** Offset of source environment data
15 MQRMH-SRCENVOFFSET PIC S9(9) BINARY.
** Length of source object name
15 MQRMH-SRCNAMELENGTH PIC S9(9) BINARY.
** Offset of source object name
15 MQRMH-SRCNAMEOFFSET PIC S9(9) BINARY.
** Length of destination environment data
15 MQRMH-DESTENVLENGTH PIC S9(9) BINARY.
** Offset of destination environment data
15 MQRMH-DESTENVOFFSET PIC S9(9) BINARY.
** Length of destination object name
15 MQRMH-DESTNAMELENGTH PIC S9(9) BINARY.
** Offset of destination object name
15 MQRMH-DESTNAMEOFFSET PIC S9(9) BINARY.
** Length of bulk data
15 MQRMH-DATALOGICALENGTH PIC S9(9) BINARY.
** Low offset of bulk data
15 MQRMH-DATALOGICALOFFSET PIC S9(9) BINARY.
** High offset of bulk data
15 MQRMH-DATALOGICALOFFSET2 PIC S9(9) BINARY.

```

## MQRMH의 PL/I 선언

```

dcl
1 MQRMH based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 StrucLength fixed bin(31), /* Total length of MQRMH,
                              including strings at end of
                              fixed fields, but not the bulk
                              data */
3 Encoding fixed bin(31), /* Numeric encoding of bulk
                              data */
3 CodedCharSetId fixed bin(31), /* Character set identifier of
                              bulk data */
3 Format char(8), /* Format name of bulk data */

```

```

3 Flags          fixed bin(31), /* Reference message flags */
3 ObjectType     char(8),      /* Object type */
3 ObjectInstanceId char(24),    /* Object instance identifier */
3 SrcEnvLength   fixed bin(31), /* Length of source environment
data */
3 SrcEnvOffset   fixed bin(31), /* Offset of source environment
data */
3 SrcNameLength  fixed bin(31), /* Length of source object name */
3 SrcNameOffset  fixed bin(31), /* Offset of source object name */
3 DestEnvLength  fixed bin(31), /* Length of destination
environment data */
3 DestEnvOffset  fixed bin(31), /* Offset of destination
environment data */
3 DestNameLength fixed bin(31), /* Length of destination object
name */
3 DestNameOffset fixed bin(31), /* Offset of destination object
name */
3 DataLogicalLength fixed bin(31), /* Length of bulk data */
3 DataLogicalOffset fixed bin(31), /* Low offset of bulk data */
3 DataLogicalOffset2 fixed bin(31); /* High offset of bulk data */

```

### MQRMH의 상위 레벨 어셈블러 선언

```

MQRMH          DSECT
MQRMH_STRUCID  DS   CL4   Structure identifier
MQRMH_VERSION  DS   F     Structure version number
MQRMH_STRUCLNGTH DS   F     Total length of MQRMH, including
*                strings at end of fixed fields, but
*                not the bulk data
MQRMH_ENCODING DS   F     Numeric encoding of bulk data
MQRMH_CODEDCCHARSETID DS   F     Character set identifier of bulk
*                data
MQRMH_FORMAT   DS   CL8   Format name of bulk data
MQRMH_FLAGS    DS   F     Reference message flags
MQRMH_OBJECTTYPE DS   CL8   Object type
MQRMH_OBJECTINSTANCEID DS   XL24 Object instance identifier
MQRMH_SRCENVLENGTH DS   F     Length of source environment data
MQRMH_SRCENVOFFSET DS   F     Offset of source environment data
MQRMH_SRCNAMELENGTH DS   F     Length of source object name
MQRMH_SRCNAMEOFFSET DS   F     Offset of source object name
MQRMH_DESTENVLENGTH DS   F     Length of destination environment
*                data
MQRMH_DESTENVOFFSET DS   F     Offset of destination environment
*                data
MQRMH_DESTNAMELENGTH DS   F     Length of destination object name
MQRMH_DESTNAMEOFFSET DS   F     Offset of destination object name
MQRMH_DATALOGICALLLENGTH DS   F     Length of bulk data
MQRMH_DATALOGICALOFFSET DS   F     Low offset of bulk data
MQRMH_DATALOGICALOFFSET2 DS   F     High offset of bulk data
*
MQRMH_LENGTH   EQU   *-MQRMH
                ORG   MQRMH
MQRMH_AREA     DS   CL(MQRMH_LENGTH)

```

### MQRMH의 Visual Basic 선언

```

Type MQRMH
  StrucId      As String*4 'Structure identifier'
  Version      As Long     'Structure version number'
  StrucLength  As Long     'Total length of MQRMH, including
  'strings at end of fixed fields, but'
  'not the bulk data'
  Encoding     As Long     'Numeric encoding of bulk data'
  CodedCharSetId As Long   'Character set identifier of bulk data'
  Format       As String*8 'Format name of bulk data'
  Flags       As Long     'Reference message flags'
  ObjectType   As String*8 'Object type'
  ObjectInstanceId As MBYTE24 'Object instance identifier'
  SrcEnvLength As Long     'Length of source environment data'
  SrcEnvOffset As Long     'Offset of source environment data'
  SrcNameLength As Long    'Length of source object name'
  SrcNameOffset As Long    'Offset of source object name'
  DestEnvLength As Long    'Length of destination environment'
  'data'
  DestEnvOffset As Long    'Offset of destination environment'
  'data'
  DestNameLength As Long   'Length of destination object name'

```

DestNameOffset	As Long	'Offset of destination object name'
DataLogicalLength	As Long	'Length of bulk data'
DataLogicalOffset	As Long	'Low offset of bulk data'
DataLogicalOffset2	As Long	'High offset of bulk data'
End	Type	

### ***MQRMH*의 경우 *StrucId* (*MQCHAR4*)**

참조 메시지 헤더 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQRMH\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

#### **MQRMH\_STRUC\_ID**

참조 메시지 헤더 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQRMH\_STRUC\_ID\_ARRAY도 정의됩니다. 이 값은 MQRMH\_STRUC\_ID와 동일하지만 문자열이 아닌 문자의 배열입니다.

### ***MQRMH*의 버전 (*MQLONG*)**

구조 버전 번호. 값은 다음과 같아야 합니다.

#### **MQRMH\_VERSION\_1**

버전1 참조 메시지 헤더 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### **MQRMH\_CURRENT\_VERSION**

참조 메시지 헤더 구조의 현재 버전.

이 필드의 초기값은 MQRMH\_VERSION\_1입니다.

### ***MQRMH*용 *StrucLength* (*MQLONG*)**

고정 필드의 끝에 있는 문자열을 포함하지만 벌크 데이터는 포함하지 않는 MQRMH의 총 길이.

이 필드의 초기값은 0입니다.

### ***MQRMH*용 인코딩 (*MQLONG*)**

이는 벌크 데이터의 숫자 인코딩을 지정합니다. MQRMH 구조 자체의 숫자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다.

이 필드의 초기값은 MQENC\_NATIVE입니다.

### ***MQRMH*용 *CodedCharSetId* (*MQLONG*)**

이는 벌크 데이터의 문자 세트 ID를 지정합니다. MQRMH 구조 자체의 문자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 다음 특수 값을 사용할 수 있습니다.

#### **MQCCSI\_INHERIT**

이 구조 다음에 오는 데이터의 문자 데이터는 이 구조와 동일한 문자 세트로 되어 있습니다.

큐 관리자가 구조의 실제 문자 세트 ID에 메시지로 송신한 구조에서 이 값을 변경합니다. 오류가 발생하지 않으면 MQGET 호출에서 MQCCSI\_INHERIT 값을 리턴하지 않습니다.

MQMD의 PutApp1Type 필드 값이 MQAT\_BROKER인 경우 MQCCSI\_INHERIT를 사용하지 마십시오.

이 값은 다음 환경에서 지원됩니다.

-  AIX
-  IBM i
-  Linux
-  Windows

및 해당 시스템에 연결된 IBM MQ 클라이언트  
이 필드의 초기값은 MQCCSI\_UNDEFINED입니다.

### **MQRMH의 형식 (MQCHAR8)**

벌크 데이터의 형식 이름을 지정합니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 이 필드의 코딩 규칙은 MQMD의 *Format* 필드에 대한 규칙과 동일합니다.

이 필드의 초기값은 MQFMT\_NONE입니다.

### **MQRMH의 플래그 (MQLONG)**

이는 참조 메시지 플래그입니다. 다음의 플래그가 정의됩니다.

#### **MQRMHF\_LAST**

이 플래그는 참조 메시지가 참조된 오브젝트의 마지막 부분을 나타내거나 포함한다는 것을 표시합니다.

#### **MQRMHF\_NOT\_LAST**

참조 메시지는 오브젝트의 마지막 부분을 나타내거나 포함하지 않습니다. MQRMHF\_NOT\_LAST는 프로그램 문서화를 지원합니다. 이 옵션은 기타와 함께 사용하기 위한 용도가 아니지만, 해당 값이 0이므로 해당 사용을 감지할 수 없습니다.

이 필드의 초기값은 MQRMHF\_NOT\_LAST입니다.

### **MQRMH의 경우 ObjectType (MQCHAR8)**

이는 메시지 엑시트가 지원하는 참조 메시지의 유형을 인식하는 데 사용할 수 있는 이름입니다. 이름은 *Format* 필드와 동일한 규칙을 준수해야 합니다. 537 페이지의 『MQRMH의 형식 (MQCHAR8)』의 내용을 참조하십시오.

이 필드의 초기값은 8개의 공백입니다.

### **MQRMH의 ObjectInstanceID (MQBYTE24)**

이 필드를 사용하여 오브젝트의 특정 인스턴스를 식별하십시오. 필요하지 않은 경우 다음 값으로 설정하십시오.

#### **MQOII\_NONE**

지정된 오브젝트 인스턴스 ID가 없습니다. 이 값은 필드 길이만큼의 2진 0입니다.

C 프로그래밍 언어의 경우 상수 MQOII\_NONE\_ARRAY도 정의됩니다. 이는 MQOII\_NONE과 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

이 필드의 길이는 MQ\_OBJECT\_INSTANCE\_ID\_LENGTH로 지정됩니다. 이 필드의 초기값은 MQOII\_NONE입니다.

### **SrcEnvMQRMH의 길이 (MQLONG)**

소스 환경 데이터의 길이입니다. 이 필드가 0이면 소스 환경 데이터가 없으며 *SrcEnvOffset*(이)가 무시됩니다.

이 필드의 초기값은 0입니다.

### **SrcEnvOffset(MQLONG)**

이 필드는 MQRMH 구조의 시작에서 소스 환경 데이터의 오프셋을 지정합니다. 해당 데이터가 작성자로 알려진 경우 참조 메시지의 작성자가 소스 환경 데이터를 지정할 수 있습니다. 예를 들어, Windows에서 소스 환경 데이터는 벌크 데이터가 포함된 오브젝트의 디렉토리 경로일 수 있습니다. 그러나 작성자가 소스 환경 데이터를 알지 못하는 경우 사용자 제공 메시지 엑시트가 필요한 환경 정보를 판별해야 합니다.

소스 환경 데이터의 길이는 *SrcEnvLength*로 지정됩니다. 이 길이가 0인 경우 소스 환경 데이터가 없고 *SrcEnvOffset*이 무시됩니다. 존재하는 경우 소스 환경 데이터는 구조의 시작부터 *StrucLength* 바이트 내에 완전히 상주해야 합니다.

애플리케이션은 환경 데이터가 구조에서 마지막으로 고정된 필드 바로 뒤에 시작되었거나 *SrcNameOffset*, *DestEnvOffset*, *DestNameOffset* 필드로 주소 지정된 데이터와 인접하다고 가정할 수 없습니다.

이 필드의 초기값은 0입니다.

### **SrcNameMQRMH의 길이 (MQLONG)**

소스 오브젝트 이름의 길이입니다. 이 필드가 0이면 소스 오브젝트 이름이 없으며 *SrcNameOffset*이(가) 무시됩니다.

이 필드의 초기값은 0입니다.

### **SrcNameMQRMH에 대한 오프셋 (MQLONG)**

이 필드는 MQRMH 구조의 시작에서 소스 오브젝트 이름의 오프셋을 지정합니다. 해당 데이터가 작성자로 알려진 경우 참조 메시지의 작성자가 소스 오브젝트 이름을 지정할 수 있습니다. 그러나 작성자가 소스 오브젝트 이름을 알지 못하는 경우 사용자 제공 메시지 엑시트가 액세스할 오브젝트를 지정해야 합니다.

소스 오브젝트 이름의 길이는 *SrcNameLength*(으)로 제공됩니다. 이 길이가 0이면 소스 오브젝트 이름이 없으며 *SrcNameOffset*이(가) 무시됩니다. 존재하는 경우, 소스 오브젝트 이름은 구조의 시작으로부터 *StrucLength* 바이트 내에 완전히 상주해야 합니다.

애플리케이션은 소스 오브젝트 이름이 *SrcEnvOffset*, *DestEnvOffset* 및 *DestNameOffset* 필드에 의해 주소 지정된 데이터와 연속되는 것으로 가정하지 않아야 합니다.

이 필드의 초기값은 0입니다.

### **DestEnvMQRMH의 길이 (MQLONG)**

이는 목적지 환경 데이터의 길이입니다. 이 필드가 0이면 목적지 환경 데이터가 없으며 *DestEnvOffset*이(가) 무시됩니다.

### **DestEnvMQRMH에 대한 오프셋 (MQLONG)**

이 필드는 MQRMH 구조의 시작에서 목적지 환경 데이터의 오프셋을 지정합니다. 해당 데이터가 작성자로 알려진 경우 참조 메시지의 작성자가 목적지 환경 데이터를 지정할 수 있습니다. 예를 들어, Windows에서 목적지 환경 데이터는 벌크 데이터가 저장되는 오브젝트의 디렉토리 경로일 수 있습니다. 그러나 작성자가 목적지 환경 데이터를 알지 못하는 경우 사용자 제공 메시지 엑시트에서 필요한 환경 정보가 있는지 판별해야 합니다.

목적지 환경 데이터의 길이는 *DestEnvLength*로 지정됩니다. 이 길이가 0인 경우 목적지 환경 데이터가 없고 *DestEnvOffset*이 무시됩니다. 존재하는 경우 목적지 환경 데이터는 구조의 시작부터 *StrucLength* 바이트 내에 완전히 상주해야 합니다.

애플리케이션은 목적지 환경 데이터가 *SrcEnvOffset*, *SrcNameOffset*, *DestNameOffset* 필드로 주소 지정된 데이터와 인접하다고 가정할 수 없습니다.

이 필드의 초기값은 0입니다.

### **DestNameMQRMH의 길이 (MQLONG)**

목적지 오브젝트 이름의 길이입니다. 이 필드가 0이면 목적지 오브젝트 이름이 없으며 *DestNameOffset*이(가) 무시됩니다.

### **DestNameMQRMH에 대한 오프셋 (MQLONG)**

이 필드는 MQRMH 구조의 시작에서 목적지 오브젝트 이름의 오프셋을 지정합니다. 해당 데이터가 작성자로 알려진 경우 참조 메시지의 작성자가 목적지 오브젝트 이름을 지정할 수 있습니다. 그러나 작성자가 목적지 오브젝트 이름을 알지 못하는 경우 사용자 제공 메시지 엑시트가 작성되거나 수정될 오브젝트를 식별해야 합니다.

목적지 오브젝트 이름의 길이는 *DestNameLength*에 의해 제공됩니다. 이 길이가 0이면 목적지 오브젝트 이름이 없으며 *DestNameOffset*이(가) 무시됩니다. 존재하는 경우, 목적지 오브젝트 이름은 구조의 시작으로부터 *StrucLength* 바이트 내에 완전히 상주해야 합니다.

애플리케이션은 목적지 오브젝트 이름이 *SrcEnvOffset*, *SrcNameOffset* 및 *DestEnvOffset* 필드에 의해 주소 지정된 데이터와 연속되는 것으로 가정하지 않아야 합니다.

이 필드의 초기값은 0입니다.

### **MQRMH의 DataLogical길이 (MQLONG)**

*DataLogicalLength* 필드는 MQRMH 구조가 참조하는 벌크 데이터의 길이를 지정합니다.

벌크 데이터가 메시지에 실제로 존재하는 경우, 데이터는 MQRMH 구조의 시작으로부터 *StrucLength* 바이트의 오프셋에서 시작됩니다. 전체 메시지의 길이에서 *StrucLength* (를) 뺀 값이 존재하는 벌크 데이터의 길이를 제공합니다.

데이터가 메시지에 있는 경우 *DataLogicalLength* 은(는) 관련된 데이터의 양을 지정합니다. 일반적인 경우는 *DataLogicalLength* 이(가) 메시지에 있는 데이터 길이와 동일한 값을 갖는 경우입니다.

MQRMH 구조가 오브젝트의 나머지 데이터를 표시하는 경우(지정된 논리적 오프셋에서 시작), *DataLogicalLength* 에 대해 0 값을 사용하여 벌크 데이터가 실제로 메시지에 표시되지 않는다고 가정할 수 있습니다.

데이터가 존재하지 않는 경우 MQRMH의 끝은 메시지 끝과 일치합니다.

이 필드의 초기값은 0입니다.

### **MQRMH에 대한 DataLogical오프셋 (MQLONG)**

이 필드는 벌크 데이터가 일부를 구성하는 오브젝트의 시작에서 벌크 데이터의 낮은 오프셋을 지정합니다. 오브젝트의 시작에서 벌크 데이터의 오프셋은 논리 오프셋이라고 합니다. 이는 MQRMH 구조의 시작에서 벌크 데이터의 실제 오프셋이 아닙니다. 해당 오프셋은 *StrucLength* 로 지정됩니다.

대형 오브젝트가 참조 메시지를 사용하여 송신되도록 허용하기 위해, 논리 오프셋이 두 개 필드로 나뉘며 이러한 두 필드의 합으로 실제 논리 오프셋이 지정됩니다.

- *DataLogicalOffset* 은 논리 오프셋을 1,000,000,000으로 나눌 때 얻은 나머지를 나타냅니다. 따라서 값의 범위는 0 - 999,999,999입니다.
- *DataLogicalOffset2* 는 논리 오프셋을 1,000,000,000으로 나눌 때 얻은 결과를 나타냅니다. 따라서 논리 오프셋에 있는 1 000 000 000의 완전한 배수의 수입니다. 배수의 수는 0 - 999 999 999 범위에 있습니다.

이 필드의 초기값은 0입니다.

### **MQRMH용 DataLogicalOffset2 (MQLONG)**

이 필드는 벌크 데이터가 일부를 구성하는 오브젝트의 시작에서 벌크 데이터의 높은 오프셋을 지정합니다. 값은 0 - 999,999,999의 범위에 있습니다. 세부사항은 *DataLogicalOffset* 을(를) 참조하십시오.

이 필드의 초기값은 0입니다.

## **MQRR - 응답 레코드**

MQRR 구조를 사용하여 목적지가 분배 목록일 때 단일 목적지 큐에 대한 열기 또는 넣기 조작의 결과인 완료 코드 및 이유 코드를 수신하십시오. MQRR은 MQOPEN, MQPUT, MQPUT1 호출의 출력 구조입니다.

### **가용성**

MQRR 구조는 다음 플랫폼에서 사용 가능합니다.

-  AIX
-  IBM i
-  Linux
-  Windows

및 해당 시스템에 연결된 IBM MQ 클라이언트

## 문자 세트 및 인코딩

MQRR의 데이터는 MQENC\_NATIVE로 지정된 로컬 큐 관리자의 인코딩 및 **CodedCharSetId** 큐 관리자 속성으로 지정된 문자 세트에 있어야 합니다. 그러나 애플리케이션이 MQ MQI 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트와 인코딩으로 작성되어야 합니다.

## 사용법

MQOPEN 및 MQPUT 호출 또는 MQPUT1 호출에서 이러한 구조의 배열을 제공하여, 호출의 결과가 혼합될 때 (즉, 호출이 목록의 일부 큐에 대해 성공하지만 다른 큐에 대해 실패할 때) 분배 목록의 모든 큐에 대한 완료 코드 및 이유 코드를 판별할 수 있습니다. 호출에서 이유 코드 MQRC\_MULTIPLE\_REASONS는 큐 관리자가 응답 레코드(애플리케이션이 제공한 경우)를 설정함을 표시합니다.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 524. MQRR의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
CompCode (큐의 완료 코드)	MQCC_OK	0
이유 (큐에 대한 이유 코드)	MQRC_NONE	0

**참고:**

- C 프로그래밍 언어의 매크로 변수 MQRR\_DEFAULT는 테이블에 나열되는 값을 포함합니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오.

```
MQRR MyRR = {MQRR_DEFAULT};
```

## 언어 선언

MQRR에 대한 C 선언

```
typedef struct tagMQRR MQRR;  
struct tagMQRR {  
    MQLONG  CompCode; /* Completion code for queue */  
    MQLONG  Reason; /* Reason code for queue */  
};
```

MQRR에 대한 COBOL 선언

```
** MQRR structure  
10 MQRR.  
** Completion code for queue  
15 MQRR-COMPCODE PIC S9(9) BINARY.  
** Reason code for queue  
15 MQRR-REASON PIC S9(9) BINARY.
```

MQRR에 대한 PL/I 선언

```
dcl  
1 MQRR based,  
3 CompCode fixed bin(31), /* Completion code for queue */  
3 Reason fixed bin(31); /* Reason code for queue */
```

MQRR에 대한 Visual Basic 선언

```
Type MQRR
```



```
CompCode As Long 'Completion code for queue'
Reason As Long 'Reason code for queue'
End Type
```

### MQRR의 경우 *CompCode* (MQLONG)

MQOPEN 또는 MQPUT1 호출에 제공된 MQOR 구조 배열에서 해당하는 요소가 지정한 이름의 큐에 대한 열기 또는 넣기 조작의 결과인 완료 코드입니다.

이 필드는 항상 출력 필드입니다. 이 필드의 초기값은 MQCC\_OK입니다.

### MQRR에 대한 이유 (MQLONG)

MQOPEN 또는 MQPUT1 호출에 제공된 MQOR 구조 배열에서 해당하는 요소가 지정한 이름의 큐에 대한 열기 또는 넣기 조작의 결과인 이유 코드입니다.

이 필드는 항상 출력 필드입니다. 이 필드의 초기값은 MQRC\_NONE입니다.

### MQSCO - SSL/TLS 구성 옵션

MQCD 구조의 TLS 필드와 함께 MQSCO 구조를 사용하면 IBM MQ MQI client 로 실행 중인 애플리케이션이 채널 프로토콜이 TCP/IP일 때 클라이언트 연결에 대한 TLS 사용을 제어하는 구성 옵션을 지정할 수 있습니다. 이 구조는 MQCONNX 호출의 입력 매개변수입니다.

### 가용성

MQSCO 구조는 다음 클라이언트에서 사용 가능합니다.

-  AIX
-  IBM i
-  Linux
-  Windows

클라이언트 채널에 대한 채널 프로토콜이 TCP/IP가 아니면 MQSCO 구조가 무시됩니다.

### 문자 세트 및 인코딩

MQSCO의 데이터는 **CodedCharSetId** 큐 관리자 속성에 의해 지정된 문자 세트 및 MQENC\_NATIVE에 의해 지정된 로컬 큐 관리자의 인코딩에 있어야 합니다.

### 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 525. MQSCO의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
<u>StrucId</u> (구조 ID)	MQSCO_STRUC_ID	'SCO~'
<u>버전</u> (구조 버전 번호)	MQSCO_CURRENT_VERSION	1
<u>KeyRepository</u> (키 저장소의 위치)	없음	널 문자열 또는 공백
<u>CryptoHardware</u> (암호화 하드웨어의 세부사항)	없음	널 문자열 또는 공백
<u>AuthInfoRecCount</u> (존재하는 MQAIR 레코드 수)	없음	0
<u>AuthInfoRecOffset</u> (MQSCO의 시작에서 첫 번째 MQAIR 레코드의 오프셋)	없음	0

표 525. MQSCO의 필드 (계속)		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
AuthInfoRecPtr (첫 번째 MQAIR 레코드의 주소)	없음	널 포인터 또는 널 바이트
참고: <i>Version</i> 이 MQSCO_VERSION_2 미만인 경우 다음 두 필드는 무시됩니다.		
KeyReset수 (TLS 비밀 키 재설정 수)	MQSCO_RESET_COUNT_DEFAULT	0
547 페이지의 『MQSCO의 경우 FipsRequired (MQLONG)』 (IBM MQ에서 FIPS 인증 암호화 알고리즘 사용)	MQSSL_FIPS_NO	0
참고: <i>Version</i> 이 MQSCO_VERSION_3 미만인 경우 다음 두 필드는 무시됩니다.		
EncryptionPolicySuiteB (스위트 B 암호화 알고리즘만 사용)	MQ_SUITE_B_NONE, MQ_SUITE_B_NOT_AVAILABLE, MQ_SUITE_B_NOT_AVAILABLE, MQ_SUITE_B_NOT_AVAILABLE	1, 0, 0, 0
참고: <i>Version</i> 이 MQSCO_VERSION_4 미만인 경우 다음 두 필드는 무시됩니다.		
CertificateVal정책 (인증서 유효성 검증 정책)	MQ_CERT_VAL_POLICY_DEFAULT	0
참고: <i>Version</i> 이 MQSCO_VERSION_5 미만인 경우 다음 두 필드는 무시됩니다.		
CertificateLabel (사용 중인 인증서 레이블에 대해 자세히 설명함)	없음	널 문자열 또는 공백
참고: <i>Version</i> 이 MQSCO_VERSION_6 미만인 경우 나머지 필드는 무시됩니다.		
KeyRepoPasswordPtr (TLS키 저장소 비밀번호 주소)	없음	널 포인터 또는 널 바이트
KeyRepoPasswordOffset (TLS키 저장소 비밀번호 오프셋)	없음	0
KeyRepoPasswordLength (TLS키 저장소 비밀번호 길이)	없음	0

**참고:**

1. ~ 기호는 단일 공백 문자를 나타냅니다.
2. C 프로그래밍 언어의 매크로 변수MQSCO\_DEFAULT는 테이블에 나열된 값을 포함합니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오.

```
MQSCO MySCO = {MQSCO_DEFAULT};
```

**언어 선언**

MQSCO의 C 선언

```
typedef struct tagMQSCO MQSCO;
```

```

struct tagMQSCO {
    MQCHAR4    StrucId;                /* Structure identifier */
    MQLONG     Version;                /* Structure version number */
    MQCHAR256  KeyRepository;         /* Location of TLS key */
                                        /* repository */
    MQCHAR256  CryptoHardware;        /* Cryptographic hardware */
                                        /* configuration string */
    MQLONG     AuthInfoRecCount;      /* Number of MQAIR records */
                                        /* present */
    MQLONG     AuthInfoRecOffset;     /* Offset of first MQAIR */
                                        /* record from start of */
                                        /* MQSCO structure */
    PMQAIR     AuthInfoRecPtr;        /* Address of first MQAIR */
                                        /* record */
/* Ver:1 */
    MQLONG     KeyResetCount;         /* Number of unencrypted */
                                        /* bytes sent/received */
                                        /* before secret key is */
                                        /* reset */
    MQLONG     FipsRequired;          /* Using FIPS-certified */
                                        /* algorithms */
/* Ver:2 */
    MQLONG     EncryptionPolicySuiteB[4]; /* Use only Suite B */
                                        /* cryptographic algorithms */
/* Ver:3 */
    MQLONG     CertificateValPolicy;  /* Certificate validation */
                                        /* policy */
/* Ver:4 */
    MQCHAR64   CertificateLabel;      /* Certificate label */
/* Ver:5 */
    MQPTR      KeyRepoPasswordPtr;    /* Address of key */
                                        /* repository password */
    MQLONG     KeyRepoPasswordOffset; /* Offset of key repository */
                                        /* password */
    MQLONG     KeyRepoPasswordLength; /* Length of key repository */
                                        /* password */
/* Ver:6 */
};

```

## MQSCO의 COBOL 선언

```

** MQSCO structure
10 MQSCO.
** Structure identifier
15 MQSCO-STRUCID PIC X(4).
** Structure version number
15 MQSCO-VERSION PIC S9(9) BINARY.
** Location of TLS key repository
15 MQSCO-KEYREPOSITORY PIC X(256).
** Cryptographic hardware configuration string
15 MQSCO-CRYPTOHARDWARE PIC X(256).
** Number of MQAIR records present
15 MQSCO-AUTHINFORECCOUNT PIC S9(9) BINARY.
** Offset of first MQAIR record from start of MQSCO structure
15 MQSCO-AUTHINFORECOFFSET PIC S9(9) BINARY.
** Address of first MQAIR record
15 MQSCO-AUTHINFORECPTR POINTER.
** Version 1 **
** Number of unencrypted bytes sent/received before secret key is
** reset
15 MQSCO-KEYRESETCOUNT PIC S9(9) BINARY.
** Using FIPS-certified algorithms
15 MQSCO-FIPSREQUIRED PIC S9(9) BINARY.
** Version 2 **
** Use only Suite B cryptographic algorithms
15 MQSCO-ENCRYPTIONPOLICYSUITEB PIC S9(9) BINARY OCCURS 4.
** Version 3 **
** Certificate validation policy setting
15 MQSCO-CERTIFICATEVALPOLICY PIC S9(9) BINARY.
** Version 4 **
** SSL/TLS certificate label
15 MQSCO-CERTIFICATELABEL PIC X(64).
** Version 5 **
** Add padding to ensure that pointers start on correct
** boundaries
15 FILLER PIC S9(9) BINARY VALUE 0.
** Address of key repository password
15 MQSCO-KEYREPOPASSWORDPTR POINTER.
** Offset of key repository password

```

```

15 MQSCO-KEYREPOPASSWORDOFFSET PIC S9(9) BINARY.
** Length of key repository password
15 MQSCO-KEYREPOPASSWORDLENGTH PIC S9(9) BINARY.
** Version 6 **

```

## MQSCO의 PL/I 선언

```

dcl
  1 MQSCO based,
    3 StrucId          char(4),          /* Structure identifier */
    3 Version          fixed bin(31),    /* Structure version number */
    3 KeyRepository    char(256),        /* Location of TLS key
                                         repository */
    3 CryptoHardware   char(256),        /* Cryptographic hardware
                                         configuration string */
    3 AuthInfoRecCount fixed bin(31),    /* Number of MQAIR records
                                         present */
    3 AuthInfoRecOffset fixed bin(31),   /* Offset of first MQAIR record
                                         from start of MQSCO structure */
    3 AuthInfoRecPtr   pointer,          /* Address of first MQAIR record */
    3 KeyResetCount    fixed bin(31),    /* Key reset count */
/* Version 1 */
    3 FipsRequired     fixed bin(31),    /* FIPS required */
/* Version 2 */
    3 EncryptionPolicySuiteB (4) fixed bin(31), /* Suite B encryption policy */
/* Version 3 */
    3 CertificateValPolicy fixed bin(31), /* Certificate validation policy */
/* Version 4 */
    3 CertificateLabel  char(64),        /* SSL/TLS certificate label */
/* Version 5 */
    3 KeyRepoPasswordPtr pointer,        /* Address of key repository
                                         password */
    3 KeyRepoPasswordOffset fixed bin(31), /* Offset of key repository
                                         password */
    3 KeyRepoPasswordLength fixed bin(31); /* Length of key repository
                                         password */
/* Version 6 */

```

## MQSCO의 Visual Basic 선언

```

Type MQSCO
  StrucId          As String*4  'Structure identifier'
  Version          As Long      'Structure version number'
  KeyRepository    As String*256 'Location of TLS key repository'
  CryptoHardware   As String*256 'Cryptographic hardware configuration'
  AuthInfoRecCount As Long      'Number of MQAIR records present'
  AuthInfoRecOffset As Long     'Offset of first MQAIR record from'
  AuthInfoRecPtr   As MQPTR     'Address of first MQAIR record'
  KeyResetCount    As Long      'Number of unencrypted bytes sent/received before secret key
is reset'
  'Version 1'
  FipsRequired     As Long      'Mandatory FIPS CipherSpecs?'
  'Version 2'
End Type

```

### 관련 참조

312 페이지의 『MQCNO - 연결 옵션』

MQCNO 구조를 사용하면 애플리케이션이 큐 관리자에 대한 연결과 관련된 옵션을 지정할 수 있습니다. 구조는 MQCONNX 호출의 입/출력 매개변수입니다.

### MQSCO의 경우 StrucId (MQCHAR4)

이는 SSL/TLS 구성 옵션 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQSCO\_STRUC\_ID입니다. 값은 다음과 같아야 합니다.

#### MQSCO\_STRUC\_ID

SSL/TLS 구성 옵션 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQSCO\_STRUC\_ID\_ARRAY도 정의됩니다. 이 값은 MQSCO\_STRUC\_ID와 동일하지만 문자열이 아닌 문자의 배열입니다.

## MQSCO의 버전 (MQLONG)

구조 버전 번호이며 값은 다음과 같아야 합니다.

### MQSCO\_VERSION\_1

버전-1 TLS 구성 옵션 구조 ID.

### MQSCO\_VERSION\_2

버전-2 TLS 구성 옵션 구조 ID.

### MQSCO\_VERSION\_3

버전-3 TLS 구성 옵션 구조입니다.

### MQSCO\_VERSION\_4

버전-4 TLS 구성 옵션 구조입니다.

### MQSCO\_VERSION\_5

버전-5 TLS 구성 옵션 구조입니다.

### MQSCO\_VERSION\_6

Version-6 TLS 구성 옵션 구조입니다.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

### MQSCO\_CURRENT\_VERSION

TLS 구성 옵션 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQSCO\_VERSION\_1입니다.

Multi

## MQSCO의 경우 KeyRepository (MQCHAR256)

이 필드는 IBM i, AIX, Linux, and Windows 시스템에서 실행 중인 IBM MQ MQI clients에만 관련됩니다. 키와 인증서가 저장되는 키 데이터베이스 파일의 위치를 지정합니다. 파일 접미부를 지정하지 않으면 .kdb 접미부가 자동으로 추가됩니다.

각 키 데이터베이스 파일에는 연관된 암호 숨김 파일이 있을 수 있습니다. 스테쉬 파일은 키 데이터베이스에 대한 프로그램 액세스를 허용하는 데 사용되는 인코딩된 비밀번호를 보유합니다. 비밀번호 스테쉬 파일은 동일한 디렉토리에 상주해야 하고, 키 데이터베이스와 동일한 파일 스템이 있어야 하며 접미부 .sth로 끝나야 합니다.

예를 들어, 키 데이터베이스 파일이 /xxx/yyy/key.kdb인 경우 비밀번호 스테쉬 파일은 /xxx/yyy/key.sth여야 합니다. 여기서 xxx 및 yyy 는 디렉토리 이름을 나타냅니다.

키 데이터베이스 비밀번호는 MQSCO 구조의 KeyRepoPasswordPtr 또는 KeyRepoPasswordOffset 필드를 사용하여 지정할 수도 있습니다.

값이 필드의 길이보다 짧으면 널 문자로 값을 종료하거나 필드 길이에 맞게 공백으로 채웁니다. 값은 확인되지 않습니다. 키 저장소에 액세스하는 중에 오류가 발생하는 경우 이유 코드 MQRC\_KEY\_REPOSITORY\_ERROR와 함께 호출이 실패합니다.

IBM MQ MQI client에서 TLS 연결을 실행하려면 KeyRepository를 올바른 키 데이터베이스 파일 이름으로 설정하십시오.

입력 필드입니다. 이 필드의 길이는 MQ\_SSL\_KEY\_REPOSITORY\_LENGTH로 지정됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 공백 문자입니다.

## MQSCO의 경우 CryptoHardware (MQCHAR256)

이 필드는 클라이언트 시스템에 연결된 암호화 하드웨어의 구성 세부사항을 제공합니다.

필드를 다음 형식의 문자열로 설정하거나 공백 또는 널로 두십시오.

```
GSK_PKCS11=the PKCS #11 driver path and file name;the PKCS #11 token label;the PKCS #11 token password;symmetric cipher setting;
```

PKCS #11 인터페이스를 준수하는 암호화 하드웨어를 사용하려면(예: IBM 4960 또는 IBM 4764) 각각 세미콜론으로 종료되는 PKCS #11 드라이버 경로, PKCS #11 토큰 레벨, PKCS #11 토큰 비밀번호 문자열을 지정해야 합니다.

PKCS #11 드라이버 경로는 PKCS #11 카드 지원을 위해 제공되는 공유 라이브러리의 절대 경로입니다. PKCS #11 driver file name은 공유 라이브러리의 이름입니다. PKCS #11 경로 및 파일 이름에 필요한 값의 예는 다음과 같습니다.

```
/usr/lib/pkcs11/PKCS11_API.so
```

PKCS #11 토큰 레이블은 하드웨어를 구성할 때 사용한 레이블과 일치해야 합니다.

암호화 하드웨어 구성이 필요하지 않은 경우 필드를 공백 또는 널로 설정하십시오.

값이 필드의 길이보다 짧으면 널 문자로 값을 종료하거나 필드 길이에 맞게 공백으로 채웁니다. 암호화 하드웨어를 구성하기 위해 사용할 때 값이 올바르지 않거나 실패로 이어지는 경우 이유 코드 MQRC\_CRYPTOHARDWARE\_ERROR와 함께 호출이 실패합니다.

입력 필드입니다. 이 필드의 길이는 MQ\_SSL\_CRYPTOHARDWARE\_LENGTH로 지정됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 공백 문자입니다.

### **MQSCO의 AuthInfoRecCount (MQLONG)**

*AuthInfoRecPtr* 또는 *AuthInfoRecOffset* 필드에서 처리하는 인증 정보(MQAIR) 레코드의 수입니다. 자세한 정보는 268 페이지의 『MQAIR - 인증 정보 레코드』의 내용을 참조하십시오. 값은 0 이상이어야 합니다. 값이 올바르지 않은 경우 이유 코드 MQRC\_AUTH\_INFO\_REC\_COUNT\_ERROR와 함께 호출이 실패합니다.

입력 필드입니다. 이 필드의 초기값은 0입니다.

### **MQSCO의 AuthInfoRecOffset (MQLONG)**

MQSCO 구조의 시작에서 첫 번째 인증 정보 레코드의 오프셋(바이트)입니다. 오프셋은 양수 또는 음수일 수 있습니다. *AuthInfoRecCount*이(가) 0인 경우 필드는 무시됩니다.

*AuthInfoRecOffset* 또는 *AuthInfoRecPtr*을(를) 사용하여 MQAIR 레코드를 지정할 수 있지만 둘 다 지정할 수는 없습니다. 자세한 내용은 *AuthInfoRecPtr* 필드의 설명을 참조하십시오.

입력 필드입니다. 이 필드의 초기값은 0입니다.

### **MQSCO의 AuthInfoRecPtr (PMQAIR)**

첫 번째 인증 정보 레코드의 주소입니다. *AuthInfoRecCount*이(가) 0인 경우 필드는 무시됩니다.

다음 두 가지 방법 중 하나로 MQAIR 레코드의 배열을 제공할 수 있습니다.

- 포인터 필드 *AuthInfoRecPtr* 사용.

이 경우 애플리케이션은 MQSCO 구조에서 분리된 MQAIR 레코드의 배열을 선언할 수 있으며 *AuthInfoRecPtr*을 배열의 주소로 설정할 수 있습니다.

다른 환경에 이식 가능한 방법으로 포인터 데이터 유형을 지원하는 프로그래밍 언어(예: C 프로그래밍 언어)에 *AuthInfoRecPtr*을 사용하는 것을 고려하십시오.

- 오프셋 필드 *AuthInfoRecOffset* 사용.

이 경우 애플리케이션은 MQAIR 레코드의 배열 앞에 오는 MQSCO가 포함된 복합 구조를 선언하고 MQSCO 구조의 시작에서부터 배열의 첫 번째 레코드의 오프셋으로 *AuthInfoRecOffset*을 설정해야 합니다. 이 값이 정확하고 MQLONG 내에 수용할 수 있는 값이 있는지 확인하십시오(가장 제한적 프로그래밍 언어가 COBOL이며 올바른 범위는 -999 999 999 - +999 999 999임).

포인터 데이터 유형을 지원하지 않거나 다른 환경으로 이식 불가능한 방식으로 포인터 데이터 유형을 구현하는 프로그래밍 언어(예: COBOL 프로그래밍 언어)에 *AuthInfoRecOffset*을 사용하는 것을 고려하십시오.

사용자가 선택하는 기술에 상관없이 *AuthInfoRecPtr* 및 *AuthInfoRecOffset* 중 하나만 사용할 수 있습니다. 둘 모두 0이 아닌 경우 이유 코드 MQRC\_AUTH\_INFO\_REC\_ERROR와 함께 호출이 실패합니다.

입력 필드입니다. 이 필드의 초기값은 포인터를 지원하는 프로그래밍 언어로 된 널 포인터이며, 그렇지 않은 경우 모두 널 바이트 문자열입니다.

**참고:** 프로그래밍 언어가 포인터 데이터 유형을 지원하지 않는 플랫폼에서 이 필드는 적절한 길이의 바이트 문자 열로 선언됩니다.

### **MQSCO의 KeyReset수 (MQLONG)**

이는 비밀 키를 재조정하기 전에 TLS 대화 내에서 전송 및 수신된 복호화된 총 바이트 수를 나타냅니다.

바이트 수에는 MCA에서 전송된 제어 정보가 포함됩니다.

TLS 비밀 키 재설정 계수를 1B - 32KB 범위로 지정하는 경우 TLS 채널은 32KB의 비밀 키 재설정 계수를 사용합니다. 이것은 TLS 비밀 키 재설정 값이 작은 경우에 발생하는 초과 키 재설정의 처리 비용을 피하기 위함입니다.

입력 필드입니다. 값은 0 - 999 999 999 범위의 숫자이며, 기본값은 0입니다. 비밀 키가 재조정되지 않음을 표시하기 위해 0의 값을 사용하십시오.

### **MQSCO의 경우 FipsRequired (MQLONG)**

사용되는 암호화 모듈이 하드웨어 제품에서 제공한 모듈이 되도록 IBM MQ를 암호화 하드웨어로 구성할 수 있습니다. 사용 중인 암호화 하드웨어 제품에 따라 특정 레벨로 FIPS 인증될 수 있습니다. 이 필드를 사용하여 암호화가 IBM MQ 제공 소프트웨어에 제공된 경우에는 FIPS 인증 알고리즘만 사용하도록 지정하십시오.

**참고:** AIX, Linux, and Windows에서 IBM MQ는 IBM Crypto for C (ICC) 암호화 모듈을 통해 FIPS 140-2 준수를 제공합니다. 이 모듈의 인증서가 히스토리 상태로 이동되었습니다. 고객은 IBM Crypto for C (ICC) 인증서를 보고 NIST에서 제공하는 조언을 알고 있어야 합니다. 대체 FIPS 140-2 모듈이 현재 진행 중이며 해당 상태는 [프로세스 목록의 NIST CMVP 모듈에서](#) 검색하여 볼 수 있습니다.

IBM MQ Operator 3.2.0 및 큐 관리자 컨테이너 이미지 9.4.0.0 이상은 UBI 9를 기반으로 합니다. FIPS 140-2 준수가 현재 보류 중이며 해당 상태는 [프로세스 목록의 NIST CMVP 모듈에서](#) "Red Hat Enterprise Linux 9-OpenSSL FIPS 제공자"를 검색하여 볼 수 있습니다.

IBM MQ가 설치되는 경우 일부 FIPS 인증 모듈을 제공하는 TLS 암호화의 구현도 설치됩니다.

가능한 값은 다음과 같습니다.

#### **MQSSL\_FIPS\_NO**

이는 기본값입니다. 이 값으로 설정하는 경우:

- 특정 플랫폼에서 지원되는 CipherSpec을 사용할 수 있습니다.
- 암호화 하드웨어를 사용하지 않고 실행하는 경우 FIPS 140-2 인증 암호화를 사용하여 IBM MQ 플랫폼에서 CipherSpec을 실행합니다.

FIPS 인증 CipherSpec 목록은 [CipherSpec 사용에서](#) 설명된 표를 참조하십시오.

#### **MQSSL\_FIPS\_YES**

이 값으로 설정하는 경우, 암호화를 수행하기 위한 암호화 하드웨어를 사용하지 않으면 다음 사항을 확실히 준수해야 합니다.

- 이 클라이언트 연결에 적용되는 CipherSpec에서는 FIPS 인증 암호화 알고리즘만 사용할 수 있습니다.
- 특정 Cipher Spec이 사용되는 경우 인바운드 및 아웃바운드 TLS 채널 연결만 성공합니다.

자세한 정보는 [CipherSpec 사용](#)을 참조하십시오.

**참고:** 가능한 경우, FIPS 전용 CipherSpecs가 구성되면 MQI 클라이언트는

MQRC\_SSL\_INITIALIZATION\_ERROR와 함께 비FIPS CipherSpec을 지정하는 연결을 거부합니다. IBM MQ는 이러한 모든 연결을 거부한다고 보장하지 않으며 IBM MQ 구성이 FIPS 준수인지 판별하는 것은 사용자의 책임입니다.

### **MQSCO의 EncryptionPolicySuiteB (MQLONG)**

이 필드는 스위트 B 준수 암호화가 사용되는지 여부와 이용되는 강도의 레벨을 지정합니다. 값은 다음 중 하나 이상입니다.

- MQ\_SUITE\_B\_NONE

스위트 B 준수 암호화가 사용되지 않습니다.

- MQ\_SUITE\_B\_128\_BIT



스위트 B 128비트 강도 보안이 사용됩니다.

- MQ\_SUITE\_B\_192\_BIT

스위트 B 192비트 강도 보안이 사용됩니다.

참고: MQ\_SUITE\_B\_NONE을 이 필드의 다른 값과 함께 사용하는 것은 올바르지 않습니다.

### MQSCO에 대한 *CertificateVal* 정책 (MQLONG)

이 필드는 사용되는 인증서 유효성 검증 정책의 유형을 지정합니다.

다음 값 중 하나로 필드를 설정할 수 있습니다.

#### MQ\_CERT\_VAL\_POLICY\_ANY

소스 소켓 라이브러리에서 지원하는 각 인증서 유효성 검증 정책을 적용합니다. 정책이 인증서 체인을 유효한 것으로 간주하는 경우 인증서 체인을 승인합니다.

#### MQ\_CERT\_VAL\_POLICY\_RFC5280

RFC5280 준수 인증서 유효성 검증 정책만 적용합니다. 이 설정은 임의(ANY) 설정보다 엄격한 유효성 검증을 제공하지만 일부 오래된 디지털 인증서는 거부합니다.

#### MQ\_CERT\_VAL\_POLICY\_NONE

인증서 유효성 검증 정책을 적용하지 않습니다. 이 설정은 클라이언트 애플리케이션 전용이며 신뢰 체인의 유효성을 검증하지 않고 TLS 서버 인증서를 승인합니다.

이 필드의 초기값은 MQ\_CERT\_VAL\_POLICY\_ANY입니다.

### MQSCO의 경우 *CertificateLabel* (MQCHAR64)

이 필드는 사용 중인 인증서 레이블의 세부사항을 제공합니다.

IBM MQ에서 *CertificateLabel* 필드의 기본값을 공백으로 초기화합니다.

이 속성은 런타임에 기본값으로 해석되고, 역호환 가능합니다.

예를 들어, *CertificateLabel* 필드에 기본값인 공백을 사용하거나 5.0 이전 버전의 MQSCO를 지정하는 경우 이미 있던 기본값 *ibmwebspheremquser\_id*가 사용됩니다.

#### MQSCO의 *KeyRepoPasswordPtr* (MQPTR)

TLS키 저장소 비밀번호 문구의 주소 (바이트)입니다.

입력 필드입니다. 이 필드의 초기값은 포인터를 지원하는 프로그래밍 언어로 된 널 포인터이며, 그렇지 않은 경우 모두 널 바이트 문자열입니다. *Version* 이 MQSCO\_VERSION\_6미만인 경우 이 필드는 무시됩니다.

이 필드는 IBM i, AIX, Linux, and Windows 시스템에서 실행 중인 IBM MQ MQI clients 에만 관련됩니다.

키 저장소 비밀번호 문구는 일반 텍스트 문자열 또는 **runmqicred** 유틸리티를 사용하여 암호화된 비밀번호 문구로 지정할 수 있습니다.

암호화된 비밀번호 문구를 제공하는 경우 동일한 클라이언트 애플리케이션에서 제공하는 MQCSP 구조에서 비밀번호 문구를 암호화하는 데 사용된 초기 키를 지정하십시오.

이 필드를 사용하여 지정된 키 저장소 비밀번호 문구는 *MQKEYRPWD* 환경 변수 또는 클라이언트 구성 파일의 SSL 스탠자에 있는 *SSLKeyRepositoryPassword* 특성을 사용하여 지정된 키 저장소 비밀번호 문구를 대체합니다.

*KeyRepoPasswordOffset* 또는 *KeyRepoPasswordPtr* 를 사용하여 키 저장소 비밀번호 문구를 지정할 수 있지만 둘 다 지정할 수는 없습니다.

#### 관련 태스크

[AIX, Linux 및 Windows에서 IBM MQ MQI 클라이언트에 대한 초기 키 제공](#)

[IBM MQ 컴포넌트 구성 파일에서 비밀번호 보호](#)

#### 관련 참조

[runmqicred \(IBM MQ 클라이언트 비밀번호 보호\)](#)

[336 페이지의 『MQCSP의 경우 InitialKeyPtr \(MQPTR\)』](#)

[비밀번호 보호 시스템에 대한 초기 키의 주소입니다.](#)



### Multi MQSCO의 KeyRepoPasswordOffset (MQLONG)

이는 MQSCO 구조의 시작에서 TLS키 저장소 비밀번호 문구의 오프셋 (바이트) 입니다. 오프셋은 양수 또는 음수 일 수 있습니다.

*KeyRepoPasswordOffset* 또는 *KeyRepoPasswordPtr* 를 사용하여 키 저장소 비밀번호 문구를 지정할 수 있지만 둘 다 지정할 수는 없습니다. 자세한 정보는 *KeyRepoPasswordPtr* 필드의 설명을 참조하십시오.

입력 필드입니다. 이 필드의 초기값은 0입니다. *Version* 이 MQSCO\_VERSION\_6미만인 경우 이 필드는 무시됩니다.

### Multi MQSCO의 KeyRepoPasswordLength (MQLONG)

TLS키 저장소 비밀번호 문구의 길이입니다.

IBM i에서 키 저장소 비밀번호 문구의 최대 길이는 128자입니다. 키 저장소 비밀번호 문구가 허용되는 최대 길이보다 큰 경우 MQRC\_KEY\_REPOSITORY\_ERROR와 함께 연결에 실패합니다.

입력 필드입니다. 이 필드의 초기값은 0입니다. *Version* 이 MQSCO\_VERSION\_6미만인 경우 이 필드는 무시됩니다.

## MQSD - 구독 디스크립터

MQSD 구조는 작성된 구독에 대한 세부사항을 지정하는 데 사용됩니다. 구조는 MQSUB 호출의 입/출력 매개변수입니다. 자세한 정보는 [MQSUB 사용법 참고](#)를 참조하십시오.

### 가용성

MQSD 구조는 다음 플랫폼에서 사용 가능합니다.

- ▶ AIX AIX
- ▶ IBM i IBM i
- ▶ Linux Linux
- ▶ Windows Windows
- ▶ z/OS z/OS

및 해당 시스템에 연결된 IBM MQ MQI clients

### 버전

MQSD의 현재 버전은 MQSD\_VERSION\_1입니다.

### 문자 세트 및 인코딩

MQSD의 데이터는 MQENC\_NATIVE로 지정된 로컬 큐 관리자의 인코딩 및 **CodedCharSetId** 큐 관리자 속성으로 지정된 문자 세트에 있어야 합니다. 그러나 애플리케이션이 MQ MQI 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트와 인코딩으로 작성되어야 합니다.

### 관리되는 구독

애플리케이션이 해당 구독과 일치하는 발행에 대한 목적지로 특별한 큐를 사용할 필요가 없고 관리되는 구독 기능을 사용할 수 있습니다. 애플리케이션이 관리되는 구독을 사용하도록 선택하는 경우, 큐 관리자는 MQSUB 호출의 출력으로 오브젝트 핸들을 제공하여, 발행된 메시지가 송신되는 목적지를 구독자에게 알려줍니다. 자세한 정보는 [Hobj \(MQHOBJ\) - 입출력\(I/O\)](#)을 참조하십시오.

구독이 제거되면 다음 상황의 경우 큐 관리자 또한 관리되는 목적지에서 검색되지 않은 메시지 정리를 착수합니다.

- 구독이 제거되고(MQCO\_REMOVE\_SUB와 함께 MQCLOSE의 사용으로) 및 관리되는 Hobj가 처리완료된 경우.

- 지속 불가능 구독(MQSO\_NON\_DURABLE)을 사용하는 애플리케이션에 대한 연결이 끊어진 경우 내재적 수단으로.
- 구독이 만료되었기 때문에 만기에 의해 제거되고 관리되는 Hobj가 처리완료됩니다.

이 정리가 발행하여 닫힌 지속 불가능 구독의 메시지가 큐 관리자의 공간을 차지하지 않도록 지속 불가능 구독과 함께 관리되는 구독을 사용해야 합니다. 지속 가능 구독은 또한 관리되는 목적지를 사용할 수 있습니다.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQSD_STRUC_ID	'SD~'~'
버전 (구조 버전 번호)	MQSD_VERSION_1	1
옵션 (옵션)	MQSO_NON_DURABLE	0
ObjectName (오브젝트 이름)	없음	널 문자열 또는 공백
AlternateUserID (대체 사용자 ID)	없음	널 문자열 또는 공백
AlternateSecurityID (대체 보안 ID)	MQSID_NONE	널
SubExpiry (등록 만기)	MQEI_UNLIMITED	-1
ObjectString (오브젝트 문자열)	없음	MQCHARV에 정의된 이름과 값
SubName (등록 이름)	없음	MQCHARV에 정의된 이름과 값
SubUserData (구독 사용자 데이터)	없음	MQCHARV에 정의된 이름과 값
SubCorrelId (구독 상관 ID)	MQCI_NONE	널
PubPriority (발행 우선순위)	MQPRI_PRIORITY_AS_Q_DEF	-3
PubAccounting토큰 (게시 계정 토큰)	MQACT_NONE	널
PubAppIdentityData (발행 애플리케이션 ID 데이터)	없음	널 문자열 또는 공백
SelectionString (선택 기준을 제공하는 문자열)	없음	MQCHARV에 정의된 이름과 값
SubLevel (구독 레벨)	없음	1
ResObjectString (긴 오브젝트 이름)	없음	MQCHARV에 정의된 이름과 값

### 참고:

1. ~ 기호는 단일 공백 문자를 나타냅니다.
2. 널 문자열 값 또는 공백은 C의 널 문자열과 기타 프로그래밍 언어의 공백 문자를 나타냅니다.
3. C 프로그래밍 언어의 매크로 변수MQSD\_DEFAULT는 테이블에 나열되는 값을 포함합니다. 즉, 구조 내의 필드에 대한 초기값을 제공하기 위해 다음과 같은 방법으로 사용될 수 있습니다.

```
MQSD MySD = {MQSD_DEFAULT};
```

## 언어 선언

### MQSD의 C 선언

```
typedef struct tagMQSD MQSD;
struct tagMQSD {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;          /* Structure version number */
    MQLONG     Options;          /* Options associated with subscribing */
    MQCHAR48   ObjectName;       /* Object name */
    MQCHAR12   AlternateUserId;  /* Alternate user identifier */
    MQBYTE40   AlternateSecurityId; /* Alternate security identifier */
    MQLONG     SubExpiry;        /* Expiry of Subscription */
    MQCHARV    ObjectString;     /* Object Long name */
    MQCHARV    SubName;          /* Subscription name */
    MQCHARV    SubUserData;      /* Subscription User data */
    MQBYTE24   SubCorrelId;      /* Correlation Id related to this subscription */
    MQLONG     PubPriority;       /* Priority set in publications */
    MQBYTE32   PubAccountingToken; /* Accounting Token set in publications */
    MQCHAR32   PubApplIdentityData; /* Appl Identity Data set in publications */
    MQCHARV    SelectionString;  /* Message selector structure */
    MQLONG     SubLevel;         /* Subscription level */
    MQCHARV    ResObjectString;  /* Resolved Long object name*/
    /* Ver:1 */
};
```

### MQSD의 COBOL 선언

```
** Address of variable length string
20 MQSD-OBJECTSTRING-VSPTR          POINTER.
** Offset of variable length string
20 MQSD-OBJECTSTRING-VSOFFSET       PIC S9(9) BINARY.
** size of buffer
20 MQSD-OBJECTSTRING-VSBUFSIZE      PIC S9(9) BINARY.
** Length of variable length string
20 MQSD-OBJECTSTRING-VSLENGTH       PIC S9(9) BINARY.
** CCSID of variable length string
20 MQSD-OBJECTSTRING-VSCCSID        PIC S9(9) BINARY.
** Subscription name
15 MQSD-SUBNAME.
** Address of variable length string
20 MQSD-SUBNAME-VSPTR               POINTER.
** Offset of variable length string
20 MQSD-SUBNAME-VSOFFSET            PIC S9(9) BINARY.
** size of buffer
20 MQSD-SUBNAME-VSBUFSIZE           PIC S9(9) BINARY.
** Length of variable length string
20 MQSD-SUBNAME-VSLENGTH            PIC S9(9) BINARY.
** CCSID of variable length string
20 MQSD-SUBNAME-VSCCSID             PIC S9(9) BINARY.
** Subscription User data
15 MQSD-SUBUSERDATA.
** Address of variable length string
20 MQSD-SUBUSERDATA-VSPTR           POINTER.
** Offset of variable length string
20 MQSD-SUBUSERDATA-VSOFFSET        PIC S9(9) BINARY.
** size of buffer
20 MQSD-SUBUSERDATA-VSBUFSIZE       PIC S9(9) BINARY.
** Length of variable length string
20 MQSD-SUBUSERDATA-VSLENGTH        PIC S9(9) BINARY.
** CCSID of variable length string
20 MQSD-SUBUSERDATA-VSCCSID         PIC S9(9) BINARY.
** Correlation Id related to this subscription
15 MQSD-SUBCORRELID                 PIC X(24).
** Priority set in publications
15 MQSD-PUBPRIORITY                 PIC S9(9) BINARY.
** Accounting Token set in publications
15 MQSD-PUBACCOUNTINGTOKEN          PIC X(32).
** Appl Identity Data set in publications
15 MQSD-PUBAPPLIDENTITYDATA         PIC X(32).
** Message Selector
15 MQSD-SELECTIONSTRING.
** Address of variable length string
20 MQSD-SELECTIONSTRING-VSPTR       POINTER.
** Offset of variable length string
20 MQSD-SELECTIONSTRING-VSOFFSET     PIC S9(9) BINARY.
** size of buffer
```

```

20 MQSD-SELECTIONSTRING-VSBUFSIZE      PIC S9(9) BINARY.
** Length of variable length string
20 MQSD-SELECTIONSTRING-VSLENGTH      PIC S9(9) BINARY.
** CCSID of variable length string
20 MQSD-SELECTIONSTRING-VSCCSID      PIC S9(9) BINARY.
** Selection criteria
20 MQSD-SELECTIONSTRING-SUBLEVEL      PIC S9(9) BINARY.
** Long object name
20 MQSD-SELECTIONSTRING-RESOBJSTRING  PIC S9(9) BINARY.

```

## MQSD의 PL/I 선언

```

dcl
1 MQSD based,
3 StructId      char(4),      /* Structure identifier */
3 Version       fixed bin(31), /* Structure version number */
3 Options       fixed bin(31), /* Options associated with subscribing */
3 ObjectName    char(48),     /* Object name */
3 AlternateUserId char(12),   /* Alternate user identifier */
3 AlternateSecurityId char(40), /* Alternate security identifier */
3 SubExpiry     fixed bin(31), /* Expiry of Subscription */
3 ObjectString, /* Object Long name */
5 VSPtr        pointer,      /* Address of variable length string */
5 VSOffset     fixed bin(31), /* Offset of variable length string */
5 VSBufSize    fixed bin(31), /* size of buffer */
5 VSLength     fixed bin(31), /* Length of variable length string */
5 VSCCSID      fixed bin(31); /* CCSID of variable length string */
3 SubName,     /* Subscription name */
5 VSPtr        pointer,      /* Address of variable length string */
5 VSOffset     fixed bin(31), /* Offset of variable length string */
5 VSBufSize    fixed bin(31), /* size of buffer */
5 VSLength     fixed bin(31), /* Length of variable length string */
5 VSCCSID      fixed bin(31); /* CCSID of variable length string */
3 SubUserData, /* Subscription User data */
5 VSPtr        pointer,      /* Address of variable length string */
5 VSOffset     fixed bin(31), /* Offset of variable length string */
5 VSBufSize    fixed bin(31), /* size of buffer */
5 VSLength     fixed bin(31), /* Length of variable length string */
5 VSCCSID      fixed bin(31), /* CCSID of variable length string */
3 SubCorrelId  char(24),     /* Correlation Id related to this subscription */
3 PubPriority   fixed bin(31), /* Priority set in publications */
3 PubAccountingToken char(32), /* Accounting Token set in publications */
3 PubApplIdentityData char(32), /* Appl Identity Data set in publications */
3 SelectionString, /* Message Selection */
5 VSPtr        pointer,      /* Address of variable length string */
5 VSOffset     fixed bin(31), /* Offset of variable length string */
5 VSBufSize    fixed bin(31), /* size of buffer */
5 VSLength     fixed bin(31), /* Length of variable length string */
5 VSCCSID      fixed bin(31), /* CCSID of variable length string */
3 SubLevel     fixed bin(31), /* Subscription level */
3 ResObjectString, /* Resolved Long object name */
5 VSPtr        pointer,      /* Address of variable length string */
5 VSOffset     fixed bin(31), /* Offset of variable length string */
5 VSBufSize    fixed bin(31), /* size of buffer */
5 VSLength     fixed bin(31), /* Length of variable length string */
5 VSCCSID      fixed bin(31); /* CCSID of variable length string */

```

## MQSD의 상위 레벨 어셈블러 선언

```

MQSD          DSECT
MQSD_STRUCID  DS CL4 Structure identifier
MQSD_VERSION  DS F Structure version number
MQSD-OPTIONS  DS F Options associated with subscribing
MQSD-OBJECTNAME DS CL48 Object name
MQSD_ALTERNATEUSERID DS CL12 Alternate user identifier
MQSD_ALTERNATESECURITYID DS CL40 Alternate security identifier
MQSD_SUBEXPIRY DS F Expiry of Subscription
MQSD-OBJECTSTRING DS 0F Object Long name
MQSD-OBJECTSTRING_VSPTR DS F Address of variable length string
MQSD-OBJECTSTRING_VSOFFSET DS F Offset of variable length string
MQSD-OBJECTSTRING_VSBUFSIZE DS F size of buffer
MQSD-OBJECTSTRING_VSLENGTH DS F Length of variable length string
MQSD-OBJECTSTRING_VSCCSID DS F CCSID of variable length string
MQSD-OBJECTSTRING_LENGTH EQU *-MQSD-OBJECTSTRING
ORG MQSD-OBJECTSTRING
MQSD-OBJECTSTRING_AREA DS CL(MQSD-OBJECTSTRING_LENGTH)
*
MQSD-SUBNAME DS 0F Subscription name
MQSD-SUBNAME_VSPTR DS F Address of variable length string

```

```

MQSD_SUBNAME_VSOFFSET DS F Offset of variable length string
MQSD_SUBNAME_VSBUFSIZE DS F size of buffer
MQSD_SUBNAME_VSLENGTH DS F Length of variable length string
MQSD_SUBNAME_VSCCSID DS F CCSID of variable length string
MQSD_SUBNAME_LENGTH EQU *-MQSD_SUBNAME
ORG MQSD_SUBNAME
MQSD_SUBNAME_AREA DS CL(MQSD_SUBNAME_LENGTH)
*
MQSD_SUBUSERDATA DS 0F Subscription User data
MQSD_SUBUSERDATA_VSPTR DS F Address of variable length string
MQSD_SUBUSERDATA_VSOFFSET DS F Offset of variable length string
MQSD_SUBUSERDATA_VSBUFSIZE DS F size of buffer
MQSD_SUBUSERDATA_VSLENGTH DS F Length of variable length string
MQSD_SUBUSERDATA_VSCCSID DS F CCSID of variable length string
MQSD_SUBUSERDATA_LENGTH EQU *-MQSD_SUBUSERDATA
ORG MQSD_SUBUSERDATA
MQSD_SUBUSERDATA_AREA DS CL(MQSD_SUBUSERDATA_LENGTH)
*
MQSD_SUBCORRELID DS CL24 Correlation Id related to this subscription
MQSD_PUBPRIORITY DS F Priority set in publications
MQSD_PUBACCOUNTINGTOKEN DS CL32 Accounting Token set in publications
MQSD_PUBAPPLIDENTITYDATA DS CL32 Appl Identity Data set in publications
*
MQSD_SELECTIONSTRING DS F Message Selector
MQSD_SELECTIONSTRING_VSPTR DS F Address of variable length string
MQSD_SELECTIONSTRING_VSOFFSET DS F Offset of variable length string
MQSD_SELECTIONSTRING_VSBUFSIZE DS F size of buffer
MQSD_SELECTIONSTRING_VSLENGTH DS F Length of variable length string
MQSD_SELECTIONSTRING_VSCCSID DS F CCSID of variable length string
MQSD_SELECTIONSTRING_LENGTH EQU *-MQSD_SELECTIONSTRING
ORG MQSD_SELECTIONSTRING
MQSD_SELECTIONSTRING_AREA DS CL(MQSD_SELECTIONSTRING_LENGTH)
*
MQSD-SUBLEVEL DS F Subscription level
*
MQSD_RESOBJECTSTRING DS F Resolved Long object name
MQSD_RESOBJECTSTRING_VSPTR DS F Address of variable length string
MQSD_RESOBJECTSTRING_VSOFFSET DS F Offset of variable length string
MQSD_RESOBJECTSTRING_VSBUFSIZE DS F size of buffer
MQSD_RESOBJECTSTRING_VSLENGTH DS F Length of variable length string
MQSD_RESOBJECTSTRING_VSCCSID DS F CCSID of variable length string
MQSD_RESOBJECTSTRING_LENGTH EQU *-MQSD_RESOBJECTSTRING
ORG MQSD_RESOBJECTSTRING
MQSD_RESOBJECTSTRING_AREA DS CL(MQSD_RESOBJECTSTRING_LENGTH)
*
MQSD_LENGTH EQU *-MQSD
ORG MQSD
MQSD_AREA DS CL(MQSD_LENGTH)

```

## **MQSD의 경우 StrucId (MQCHAR4)**

구독 디스크립터 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQSD\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

### **MQSD\_STRUC\_ID**

구독 디스크립터 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQSD\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQSD\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

## **MQSD의 버전 (MQLONG)**

구조 버전 번호이며 값은 다음과 같아야 합니다.

### **MQSD\_VERSION\_1**

버전-1 구독 디스크립터 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

### **MQSD\_CURRENT\_VERSION**

구독 디스크립터 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQSD\_VERSION\_1입니다.

## MQSD에 대한 옵션 (MQLONG)

이는 MQSUB 호출의 조치를 제어하는 옵션을 제공합니다.

다음 옵션 중 최소 하나를 지정해야 합니다.

- MQSO\_ALTER
- MQSO\_RESUME
- MQSO\_CREATE

둘 이상의 옵션을 지정하려면 값을 함께 추가하거나(동일한 상수를 두 번 이상 추가하지 않음), 비트 단위 OR 연산을 사용하여 값을 결합하십시오(프로그래밍 언어가 비트 연산을 지원하는 경우).

이 토픽에 올바르지 않은 조합이 언급되어 있습니다. 기타 모든 조합은 유효합니다.

**액세스 또는 작성 옵션:** 액세스 및 작성 옵션은 구독 작성 여부 또는 기존 구독이 리턴되거나 대체되는지 여부를 제어합니다. 해당 옵션 중 하나 이상 지정해야 합니다.

표 526. 액세스 및 작성 옵션의 올바른 조합	
옵션 조합	참고
MQSO_CREATE	구독이 없는 경우 구독을 작성합니다. 구독이 있으면 이 조합이 실패합니다.
MQSO_RESUME	기존 구독을 재개합니다. 이 조합은 구독이 없으면 실패합니다.
MQSO_CREATE + MQSO_RESUME	구독이 없는 경우 구독을 작성하고 구독이 있는 경우 일치하는 구독을 재개합니다. 이 조합은 여러 번 실행되는 애플리케이션을 사용할 때 유용합니다.
MQSO_ALTER(참고 참조)	모든 필드를 MQSD에서 지정된 사항과 일치하도록 대체하여 기존 구독을 재개합니다. 이 조합은 구독이 없으면 실패합니다.
MQSO_CREATE + MQSO_ALTER(참고 참조)	구독이 없는 경우 구독을 작성하고 구독이 있는 경우 일치하는 구독을 재개하여 MQSD에 지정된 사항과 일치하도록 필드를 대체합니다. 이 조합은 진행하기 전에 구독이 특정 상태에 속하는지 확인하려는 애플리케이션에서 사용하면 유용합니다.

### 참고:

MQSO\_ALTER를 지정하는 옵션이 MQSO\_RESUME도 지정할 수 있으나 이 조합에는 MQSO\_ALTER만 지정하는 것 외에 추가 효과는 없습니다. 구독을 대체하기 위해 MQSUB를 호출하는 것은 구독도 계속될 것임을 뜻하기 때문에 MQSO\_ALTER는 MQSO\_RESUME을 의미합니다. 그러나 그 반대는 해당되지 않습니다. 구독을 재개한다고 해서 반드시 대체되는 것을 의미하지는 않습니다.

### MQSO\_CREATE

지정된 토픽에 대해 새 구독을 작성합니다. 동일한 *SubName*을 사용하는 구독이 있으면 MQRC\_SUB\_ALREADY\_EXISTS와 함께 호출이 실패합니다. MQSO\_CREATE 옵션을 MQSO\_RESUME과 결합하면 이 실패를 방지할 수 있습니다. *SubName*이 항상 필요한 것은 아닙니다. 자세한 정보는 해당 필드의 설명을 참조하십시오.

MQSO\_CREATE를 MQSO\_RESUME과 조합하면 구독이 있는 경우 지정된 *SubName*에 대한 기존 구독에 핸들을 리턴합니다. 기존 구독이 없으면 MQSD에서 제공되는 모든 필드를 사용하여 새 구독이 작성됩니다.

MQSO\_CREATE를 MQSO\_ALTER와 결합해도 비슷한 효과를 얻을 수 있습니다.

### MQSO\_RESUME

*SubName*에 의해 지정된 일치하는 기존 구독에 핸들을 리턴합니다. 일치하는 구독 속성에 대한 변경이 수행되지 않고 MQSD 구조의 출력에서 리턴됩니다. 다음 MQSD 필드만 사용됩니다: *StrucId*, *Version*, *Options*, *AlternateUserId* and *AlternateSecurityId*, *SubName*.

전체 구독 이름과 일치하는 구독이 존재하지 않는 경우 이유 코드 MQRC\_NO\_SUBSCRIPTION과 함께 호출이 실패합니다. MQSO\_CREATE 옵션을 MQSO\_RESUME와 결합하면 이 실패를 방지할 수 있습니다.

구독의 사용자 ID는 구독을 작성한 사용자 ID입니다. 또는 나중에 다른 사용자 ID로 대체된 경우에는 가장 최근에 성공적으로 대체된 사용자 ID입니다. AlternateUserId가 사용되고 해당 사용자에 대해 다른 사용자 ID를 사용할 수 있는 경우 다른 사용자 ID는 구독이 작성된 사용자 ID 대신 구독을 작성한 사용자 ID로 기록됩니다.

MQSO\_ANY\_USERID 옵션을 사용하지 않고 작성한 일치하는 구독이 있으며 구독의 사용자 ID가 애플리케이션이 구독에 대한 핸들을 요청하는 ID와 다른 경우 이유 코드 MQRC\_IDENTITY\_MISMATCH와 함께 호출이 실패합니다.

일치하는 구독이 있으며 현재 사용 중인 경우 MQRC\_SUBSCRIPTION\_IN\_USE와 함께 호출이 실패합니다.

SubName에 이름 지정된 구독이 애플리케이션에서 재개 또는 대체하는 데 유효한 구독이 아닌 경우, MQRC\_INVALID\_SUBSCRIPTION과 함께 호출이 실패합니다.

MQSO\_RESUME은 MQSO\_ALTER의 의미가 있으므로 이를 해당 옵션과 결합할 필요가 없습니다. 하지만 두 옵션을 결합해도 오류가 발생하지는 않습니다.

## MQSO\_ALTER

SubName의 이름으로 지정된 이름과 일치하는 전체 구독 이름을 가진 기존 구독으로 핸들을 리턴합니다. 해당 속성에 대해 대체가 허용되지 않는 경우 외에는 MQSD에서 지정된 이름과 다른 구독의 속성이 구독에서 대체됩니다. 세부사항은 각 속성의 설명에 기록되며 다음 표에 요약되어 있습니다. 변경할 수 없는 속성을 대체하거나 MQSO\_IMMUTABLE 옵션을 설정한 구독을 대체하려고 시도하는 경우 다음 표에 표시된 이유 코드와 함께 호출이 실패합니다.

전체 구독 이름과 일치하는 구독이 존재하지 않는 경우 이유 코드 MQRC\_NO\_SUBSCRIPTION과 함께 호출이 실패합니다. MQSO\_CREATE 옵션을 MQSO\_ALTER와 결합하여 이 실패를 방지할 수 있습니다.

MQSO\_CREATE를 MQSO\_ALTER와 조합하면 구독이 있는 경우 지정된 SubName에 대한 기존 구독에 핸들을 리턴합니다. 기존 구독이 없으면 MQSD에서 제공되는 모든 필드를 사용하여 새 구독이 작성됩니다.

구독의 사용자 ID는 구독을 작성한 사용자 ID입니다. 또는 나중에 다른 사용자 ID로 대체되는 경우 가장 최근에 성공한 대체의 사용자 ID입니다. AlternateUserId가 사용되고 해당 사용자에 대해 다른 사용자 ID를 사용할 수 있는 경우 다른 사용자 ID는 구독이 작성된 사용자 ID 대신 구독을 작성한 사용자 ID로 기록됩니다.

MQSO\_ANY\_USERID 옵션을 사용하지 않고 작성한 일치하는 구독이 있으며 구독의 사용자 ID가 애플리케이션이 구독에 대한 핸들을 요청하는 ID와 다른 경우 이유 코드 MQRC\_IDENTITY\_MISMATCH와 함께 호출이 실패합니다.

일치하는 구독이 있으며 현재 사용 중인 경우 MQRC\_SUBSCRIPTION\_IN\_USE와 함께 호출이 실패합니다.

SubName에 이름 지정된 구독이 애플리케이션에서 재개 또는 대체하는 데 유효한 구독이 아닌 경우, MQRC\_INVALID\_SUBSCRIPTION과 함께 호출이 실패합니다.

다음 표에는 MQSO\_ALTER가 MQSD 및 MQSUB의 속성 값을 대체하는 기능이 표시되어 있습니다.

표 527. 대체할 수 있는 MQSD 및 MQSUB의 속성			
데이터 유형 디스크립터 또는 함수 호출	필드 이름	MQSO_ALTER를 사용하여 이 속성을 대체할 수 있음	이유 코드
MQSD	지속 가능성 옵션	아니오	MQRC_DURABILITY_NOT_ALTERABLE
MQSD	목적지 옵션	예	없음
MQSD	등록 옵션	예(참고 556 페이지의 『1』 참조)	MQSO_GROUP_SUB를 대체하려고 시도하는 경우 MQRC_GROUPING_NOT_ALTERABLE
MQSD	발행물 옵션	예(참고 556 페이지의 『2』 참조)	없음
MQSD	와일드카드 옵션	아니오	MQRC_TOPIC_NOT_ALTERABLE
MQSD	기타 옵션	아니오(참고 556 페이지의 『3』 참조)	없음
MQSD	ObjectName	아니오	MQRC_TOPIC_NOT_ALTERABLE
MQSD	AlternateUserId	아니오(참고 556 페이지의 『4』 참조)	없음

표 527. 대체할 수 있는 MQSD 및 MQSUB의 속성 (계속)			
데이터 유형 디스크립터 또는 함수 호출	필드 이름	MQSO ALTER를 사용하여 이 속성을 대체할 수 있음	이유 코드
MQSD	AlternateSecurityId	아니오(참고 556 페이지의 『4』 참조)	없음
MQSD	SubExpiry	예	없음
MQSD	ObjectString	아니오	MQRC_TOPIC_NOT_ALTERABLE
MQSD	SubName	아니오(참고 556 페이지의 『5』 참조)	없음
MQSD	SubUserData	예	없음
MQSD	SubCorrelId	예(참고 556 페이지의 『6』 참조)	그룹화된 구독인 경우 MQRC_GROUPING_NOT_ALTERABLE
MQSD	PubPriority	예	없음
MQSD	PubAccountingToken	예	없음
MQSD	PubApplIdentityData	예	없음
MQSD	SubLevel	아니오	MQRC_SUBLEVEL_NOT_ALTERABLE
MQSUB	Hobj	예(참고 556 페이지의 『6』 참조)	그룹화된 구독인 경우 MQRC_GROUPING_NOT_ALTERABLE

**참고:**

1. MQSO\_GROUP\_SUB는 대체할 수 없습니다.
2. MQSO\_NEW\_PUBLICATIONS\_ONLY는 구독의 일부가 아니므로 대체할 수 없습니다.
3. 해당 옵션은 구독의 일부가 아님
4. 이 속성은 구독의 일부가 아님
5. 이 속성은 대체되는 구독의 ID임
6. 그룹화된 하위(MQSO\_GROUP\_SUB)의 일부인 경우를 제외하고 대체 가능

**지속성 옵션:** 다음 옵션은 구독의 지속성을 제어합니다. 이러한 옵션 중 하나만 지정할 수 있습니다. MQSO ALTER 옵션을 사용하여 기존 구독을 대체 중인 경우 구독의 지속성을 변경할 수 없습니다. MQSO\_RESUME를 사용하여 MQSUB 호출에서 리턴 시 적절한 지속성 옵션이 설정됩니다.

**MQSO DURABLE**

MQCO\_REMOVE\_SUB 옵션과 함께 MQCLOSE를 사용하여 명시적으로 제거될 때까지 이 토픽에 대한 구독이 남아있도록 요청합니다. 이 구독이 명시적으로 제거되지 않으면 이 애플리케이션이 큐 관리자에 대한 연결을 끊은 후에도 구독이 남아 있습니다.

지속적 구독을 허용하지 않는 것으로 정의된 토픽에 대해 지속적 구독이 요청되는 경우 MQRC\_DURABILITY\_NOT\_ALLOWED와 함께 호출이 실패합니다.

**MQSO NON DURABLE**

큐 관리자에 대한 애플리케이션 연결이 끊길 때 구독이 아직 명시적으로 제거되지 않은 경우 이 토픽에 대한 구독이 제거되도록 요청합니다. MQSO\_NON\_DURABLE은 MQSO\_DURABLE 옵션과 반대이며 프로그램 문서를 지원하기 위해 정의됩니다. 지정된 것이 없으면 이 값이 기본값입니다.

**목적지 옵션:** 다음 옵션은 구독된 토픽에 대한 발행물이 전송되는 목적지를 제어합니다. MQSO ALTER 옵션을 사용하여 기존 구독을 대체하는 경우 구독에 대한 발행물에 사용되는 목적지가 변경될 수 있습니다. MQSO\_RESUME를 사용하여 MQSUB 호출에서 리턴 시 적절한 경우 이 옵션이 설정됩니다.

**MQSO MANAGED**

발행물이 전송되는 목적지가 큐 관리자에 의해 관리되도록 요청합니다.

Hobj에서 리턴되는 오브젝트 핸들은 큐 관리자가 관리하는 큐를 나타내며 후속 MQGET, MQCB, MQINQ 또는 MQCLOSE 호출과 함께 사용됩니다.

MQSO\_MANAGED가 지정되어 있지 않으면 이전 MQSUB 호출에서 리턴되는 오브젝트 핸들이 Hobj 매개변수에서 제공될 수 없습니다.



## **MQSO\_NO\_MULTICAST**

발행물이 전송되는 목적지가 멀티캐스트 그룹 주소가 되지 않도록 요청합니다. 이 옵션은 MQSO\_MANAGED 옵션과 결합될 때만 유효합니다. **Hobj** 매개변수에서 큐에 대한 핸들이 제공되는 경우 이 구독에 멀티캐스트를 사용할 수 없고 이 옵션이 올바르지 않습니다.

토픽이 MCAST (ONLY) 설정을 사용하여 멀티캐스트 구독만 허용하도록 정의된 경우 이유 코드 MQRC\_MULTICAST\_REQUIRED와 함께 호출이 실패합니다.

**범위 옵션:** 다음 옵션은 생성되는 구독의 범위를 제어합니다. MQSO\_ALTER 옵션을 사용하여 기존 구독을 대체하는 경우 이 구독 범위 옵션을 변경할 수 없습니다. MQSO-RESUME를 사용하여 MQSUB 호출에서 리턴 시 적절한 범위 옵션이 설정됩니다.

## **MQSO\_SCOPE\_QMGR**

이 구독은 로컬 큐 관리자에서만 작성됩니다. 프록시 구독은 네트워크의 기타 큐 관리자에 분배되지 않습니다. 이 큐 관리자에서 발행되는 발행물만 이 구독자에게 전송됩니다. 이는 SUBSCOPE 토픽 속성을 사용하여 설정된 모든 작동을 대체합니다.

**참고:** 설정되지 않은 경우 구독 범위는 SUBSCOPE 토픽 속성으로 판별됩니다.

**등록 옵션:** 다음 옵션은 이 구독에 대해 큐 관리자에 작성되는 등록의 세부사항을 제어합니다. MQSO\_ALTER 옵션을 사용하여 기존 구독을 대체하는 경우 이러한 등록 옵션을 변경할 수 있습니다. MQSO-RESUME를 사용하여 MQSUB 호출에서 리턴 시 적절한 등록 옵션이 설정됩니다.

## **MQSO\_GROUP\_SUB**

이 구독은 같은 큐를 사용하고 같은 상관 ID를 지정하는 같은 SubLevel의 다른 구독과 함께 그룹화되므로, 겹치는 토픽 문자열 세트 사용으로 인해 둘 이상의 발행물 메시지가 구독 그룹으로 제공되도록 야기하는 토픽에 대한 발행으로 인해 한 메시지만 큐로 전달됩니다. 이 옵션을 사용하지 않으면 일치하는 각 고유 구독 (SubName에 의해 식별됨)이 발행물의 사본과 함께 제공되며 이로 인해 다수의 구독이 공유하는 큐에 발행물의 사본이 둘 이상 배치될 수 있습니다.

그룹에서 가장 중요한 구독만 발행물 사본과 함께 제공됩니다. 가장 중요한 구독은 와일드카드가 있는 지점까지의 전체 토픽 이름을 바탕으로 합니다. 그룹 내에 와일드카드 설계 혼합이 사용된 경우 와일드카드 지점까지만 중요합니다. 동일한 큐를 공유하는 구독 그룹 내에 다른 와일드카드 설계를 결합하지 않도록 권장합니다.

그룹화된 구독을 새로 작성할 때도 여전히 고유 SubName이 있어야 하지만 그룹에서 기존 구독의 전체 토픽 이름과 일치하는 경우 MQRC\_DUPLICATE\_GROUP\_SUB와 함께 호출이 실패합니다.

그룹에서 가장 중요한 구독이 MQSO\_NOT\_OWN\_PUBS를 지정하고 이 발행물이 동일한 애플리케이션의 발행물인 경우 큐로 발행물이 전달되지 않습니다.

이 옵션을 사용하여 작성된 구독을 대체하면 그룹화를 암시하는 필드, MQSUB 호출의 Hobj(큐 및 큐 관리자 이름을 나타냄), SubCorrelId를 변경할 수 없습니다. 이를 대체하려고 시도하면 MQRC\_GROUPING\_NOT\_ALTERABLE과 함께 호출이 실패합니다.

이 옵션은 MQCI\_NONE으로 설정되지 않은 SubCorrelId가 있는 MQSO\_SET\_CORREL\_ID와 결합해야 하며 MQSO\_MANAGED와 결합할 수는 없습니다.

## **MQSO\_ANY\_USERID**

MQSO\_ANY\_USERID가 지정된 경우 구독자의 ID는 단일 사용자 ID로 제한되지 않습니다. 이로 인해 적절한 권한이 있을 때 사용자가 구독을 대체하거나 재개할 수 있습니다. 단일 사용자에 한해서 언제든지 구독 가능합니다. 다른 애플리케이션에서 현재 사용 중인 구독 사용을 계속하려고 시도하면 MQRC\_SUBSCRIPTION\_IN\_USE와 함께 호출이 실패합니다.

이 옵션을 기존 구독에 추가하려면 MQSUB 호출(MQSO\_ALTER 사용)이 원래 구독 자체와 동일한 사용자 ID에서 시작되어야 합니다.

MQSUB 호출이 MQSO\_ANY\_USERID가 설정된 기존 구독을 참조하고 사용자 ID가 원래 구독과 다른 경우 새 사용자 ID에 토픽을 구독할 수 있는 권한이 있을 때에만 호출에 성공합니다. 성공적으로 완료되면 이 구독자에 대한 추가 발행물을 발행물 메시지에 새 사용자 ID가 설정된 구독자 큐에 넣습니다.

MQSO\_ANY\_USERID 및 MQSO\_FIXED\_USERID를 모두 지정하지 마십시오. 둘 다 지정하지 않은 경우 기본 값은 MQSO\_FIXED\_USERID입니다.

## **MQSO\_FIXED\_USERID**

MQSO\_FIXED\_USERID가 지정된 경우, 구독을 마지막으로 대체하는 사용자 ID로만 구독을 대체하거나 계속할 수 있습니다. 구독이 변경되지 않은 경우 구독을 작성한 사용자 ID입니다.

MQSUB 동사가 MQSO\_ANY\_USERID가 설정된 기존 구독을 참조하고 MQSO\_ALTER를 사용하여 구독을 대체하고 MQSO\_FIXED\_USERID 옵션을 사용하는 경우 구독의 사용자 ID는 이 새 사용자 ID와 혼합됩니다. 새 사용자 ID에 토픽을 구독할 수 있는 권한이 있는 경우에만 호출에 성공합니다.

구독을 소유할 때 기록된 사용자 ID와 다른 사용자 ID가 MQSO\_FIXED\_USERID 구독을 계속하거나 대체하려고 시도하는 경우 MQRC\_IDENTITY\_MISMATCH와 함께 호출이 실패합니다. DISPLAY SBSTATUS 명령을 사용하여 구독에 대한 소유 사용자 ID를 볼 수 있습니다.

MQSO\_ANY\_USERID 및 MQSO\_FIXED\_USERID를 모두 지정하지 마십시오. 둘 다 지정하지 않은 경우 기본값은 MQSO\_FIXED\_USERID입니다.

**발행 옵션:** 다음 옵션은 이 구독자에게 발행물이 전송되는 방식을 제어합니다. MQSO\_ALTER 옵션을 사용하여 기존 구독을 대체하는 경우 이러한 발행 옵션을 변경할 수 있습니다.

## **MQSO\_NOT\_OWN\_PUBS**

애플리케이션에서 자체 발행물을 확인하지 않음을 브로커에게 알립니다. 연결 핸들이 동일한 경우 발행은 동일한 애플리케이션에서 비롯된 것으로 간주됩니다. MQSO\_RESUME를 사용하여 MQSUB 호출에서 리턴 시 적절한 경우 이 옵션이 설정됩니다.

## **MQSO\_NEW\_PUBLICATIONS\_ONLY**

이 구독이 작성될 때 현재 보유된 발행물이 전송되지 않으며 새 발행물만 전송됩니다. 이 옵션은 MQSO\_CREATE가 지정된 경우에만 적용됩니다. 구독에 대한 후속 변경으로 구독 플로우가 대체되지는 않으므로 토픽에 보유되는 발행물은 이미 새 발행물로 해당 구독자에게 전송되었을 것입니다.

MQSO\_CREATE 없이 이 옵션을 지정하면 MQRC\_OPTIONS\_ERROR와 함께 호출이 실패합니다.

MQSO\_RESUME를 사용하여 MQSUB 호출에서 리턴 시 이 옵션을 사용하여 구독을 작성한 경우에도 이 옵션이 설정되지 않습니다.

이 옵션이 사용되지 않는 경우 이전에 보유된 메시지는 제공된 목적지 큐로 전송됩니다.

MQRC\_RETAINED\_MSG\_Q\_ERROR 또는 MQRC\_RETAINED\_NOT\_DELIVERED 오류로 인해 이 조치가 실패하는 경우 구독 작성이 실패합니다.

## **MQSO\_PUBLICATIONS\_ON\_REQUEST**

이 옵션을 설정하면 구독자가 특히 필요할 때 정보를 요청하도록 표시합니다. 큐 관리자는 구독자에게 요청하지 않은 메시지는 보내지 않습니다. 이전 MQSUB 호출의 Hsub 핸들을 사용하여 MQSUBRQ 호출이 작성될 때마다 보유된 발행물(또는 토픽에서 와일드카드가 지정된 경우 가능한 다중 발행물)이 구독자에게 전송됩니다. 발행물이 이 옵션을 사용하여 MQSUB 호출의 결과로 송신됩니다. MQSO\_RESUME를 사용하여 MQSUB 호출에서 리턴 시 적절한 경우 이 옵션이 설정됩니다.

이 옵션은 1보다 높은 SubLevel과 결합될 수 없습니다.

**미리 읽기 옵션:** 다음 옵션은 애플리케이션이 요청하기 전에 비지속 메시지가 애플리케이션으로 전송되는지 여부를 제어합니다.

## **MQSO\_READ\_AHEAD\_AS\_Q\_DEF**

MQSUB 호출이 관리되는 핸들을 사용하면 구독된 토픽과 연관된 모델 큐의 기본 미리 읽기 속성에 따라 애플리케이션이 요청하기 전에 애플리케이션으로 메시지를 발송할지 여부를 판별합니다.

이는 기본값입니다.

## **MQSO\_NO\_READ\_AHEAD**

MQSUB 호출이 관리되는 핸들을 사용하면 애플리케이션이 요청하기 전에 애플리케이션으로 메시지가 전송되지 않습니다.

## **MQSO\_READ\_AHEAD**

MQSUB 호출이 관리되는 핸들을 사용하면 애플리케이션이 요청하기 전에 애플리케이션으로 메시지가 전송될 수 있습니다.

## **참고:**

다음은 미리 읽기 옵션에 적용되는 참고입니다.

1. 이러한 옵션 중 하나만 지정될 수 있습니다. MQOO\_READ\_AHEAD 및 MQOO\_NO\_READ\_AHEAD가 둘 다 지정되면 이유 코드 MQRC\_OPTIONS\_ERROR가 리턴됩니다. 이러한 옵션은 MQSO\_MANAGED가 지정된 경우에만 적용됩니다.
2. 이전에 열린 큐가 전달되는 경우 MQSUB에 대해 적용할 수 없습니다. 요청 시 미리 읽기가 사용되지 않을 수 있습니다. 첫 번째 MQGET 호출에서 사용되는 MQGET 옵션이 미리 읽기를 금지할 수 있습니다. 또한, 미리 읽기가 지원되지 않는 큐 관리자에 클라이언트가 연결 중일 때도 미리 읽기를 사용할 수 없습니다. 애플리케이션이 IBM MQ 클라이언트로 실행 중이지 않은 경우 이러한 옵션은 무시됩니다.

**와일드카드 옵션:** 다음 옵션은 MQSD의 ObjectString 필드에서 제공되는 문자열에서 와일드카드가 해석되는 방법을 제어합니다. 이러한 옵션 중 하나만 지정할 수 있습니다. MQSO\_ALTER 옵션을 사용하여 기존 구독을 대체하는 경우 이러한 와일드카드 옵션을 변경할 수 없습니다. MQSO\_RESUME을 사용하여 MQSUB 호출에서 리턴 시 적절한 와일드카드 옵션이 설정됩니다.

### MQSO\_WILDCARD\_CHAR

와일드카드는 토픽 문자열 내의 문자에 대해서만 작동합니다.

MQSO\_WILDCARD\_CHAR에 의해 정의되는 작동이 다음 표에 표시됩니다.

표 528. 와일드카드를 해석하는 방법	
특수 문자	동작
정방향 슬래시(/)	중요하지 않음, 다만 다른 문자임
별표(*)	와일드카드, 0 또는 그 이상의 문자
물음표(?)	와일드카드, 1개의 문자
퍼센트 부호(%)	문자열에서 사용하고 특수 문자로 해석되지 않는 (*), (?) 또는 (%) 문자를 허용하는 이스케이프 문자입니다(예: (%*), (%?) 또는 (%%%)).

예를 들어, 다음 토픽에 대한 발행입니다.

```
/level0/level1/level2/level3/level4
```

다음 토픽을 사용하여 구독자와 일치합니다.

```
*
/*
/ level0/level1/level2/level3/*
/ level0/level1/*/level3/level4
/ level0/level1/le?el2/level3/level4
```

**참고:** 와일드 카드의 이 사용은 발행/구독에 대한 MQRFH1 형식 메시지를 사용할 때 IBM MQ V6 및 WebSphere MB V6에서 제공된 의미를 정확하게 공급합니다. 새로 작성된 애플리케이션에는 사용하지 말고 해당 버전에 대해 이전에 실행 중이었으며 MQSO\_WILDCARD\_TOPIC에서 설명된 대로 기본 와일드카드 작동을 사용하도록 변경되지 않은 애플리케이션에만 사용하는 것이 좋습니다.

### MQSO\_WILDCARD\_TOPIC

와일드카드는 토픽 문자열 내의 토픽 요소에 대해서만 작동합니다. 없음이 선택된 경우 이는 기본 작동입니다.

MQSO\_WILDCARD\_TOPIC에서 필요로 하는 작동이 다음 표에 표시되어 있습니다.

표 529. 와일드카드를 해석하는 방법	
특수 문자	동작
(/)	토픽 레벨 구분 기호
숫자 부호(#)	와일드카드: 다중 토픽 레벨

표 529. 와일드카드를 해석하는 방법 (계속)	
특수 문자	동작
더하기 부호(+)	와일드카드: 단일 토픽 레벨

**참고:**

(+) 및 (#)는 토픽 레벨에서 다른 문자(자신 포함)와 혼합되는 경우 와일드카드로 처리되지 않습니다. 다음 문자열에서 (#) 및 (+) 문자는 일반 문자로 처리됩니다.

```
level0/level1/#+/level3/level#
```

예를 들어, 다음 토픽에 대한 발행입니다.

```
/level0/level1/level2/level3/level4
```

다음 토픽을 사용하여 구독자와 일치합니다.

```
#
/#
/ level0/level1/level2/level3/#
/ level0/level1/+/level3/level4
```

**기타 옵션:** 다음 옵션은 구독 대신 API 호출이 발행되는 방식을 제어합니다. MQSO\_RESUME을 사용하여 MQSUB 호출에서 리턴 시 해당 옵션은 변경되지 않습니다. 자세한 내용은 561 페이지의 『MQSD에 대한 AlternateUserID (MQCHAR12)』의 내용을 참조하십시오.

**MQSO\_ALTERNATE\_USER\_AUTHORITY**

AlternateUserId 필드는 이 MQSUB 호출을 유효성 검증하는 데 사용할 사용자 ID를 포함합니다. 애플리케이션을 실행 중인 사용자 ID에 권한이 부여되었는지 여부에 관계없이 이 AlternateUserId에 지정된 액세스 옵션을 사용하여 오브젝트를 열 수 있는 권한이 부여된 경우에만 호출이 성공할 수 있습니다.

**MQSO\_SET\_CORREL\_ID**

구독은 SubCorrelId 필드에서 제공되는 상관 ID를 사용하는 것입니다. 이 옵션이 지정되지 않은 경우 큐 관리자가 구독 시간에 상관 ID를 자동으로 작성하고 이 ID는 SubCorrelId 필드의 애플리케이션으로 리턴됩니다. 자세한 정보는 563 페이지의 『MQSD의 SubCorrelID (MQBYTE24)』의 내용을 참조하십시오.

이 옵션은 MQSO\_MANAGED와 결합할 수 없습니다.

**MQSO\_SET\_IDENTITY\_CONTEXT**

구독은 PubAccountingToken 및 PubApplIdentityData 필드에서 제공되는 계정 토큰 및 애플리케이션 ID 데이터를 사용하는 것입니다.

이 옵션을 지정하면 MQOO\_SET\_IDENTITY\_CONTEXT와 함께 MQOPEN 호출을 사용하여 목적지 큐에 액세스한 경우와 동일한 권한 검사가 수행됩니다(목적지 큐에서 권한 검사를 수행하지 않는 경우 MQSO\_MANAGED 옵션을 사용하는 경우 제외).

이 옵션을 지정하지 않으면 다음과 같이 이 구독자에게 전송된 발행에 연관된 기본 컨텍스트 정보가 있습니다.

표 530. 이 구독자에게 송신된 발행물에 대한 기본 컨텍스트 정보	
<b>MQMD의 필드</b>	<b>사용된 값</b>
UserIdentifier	구독이 작성될 때 구독에 연관되는 사용자 ID입니다.
AccountingToken	가능한 경우 환경에서 판별됩니다. 그렇지 않으면 MQACT_NONE으로 설정됩니다.
ApplIdentityData	공백으로 설정

이 옵션은 MQSO\_CREATE 및 MQSO\_ALTER에서만 유효합니다. MQSO\_RESUME와 함께 사용하면 *PubAccountingToken* 및 *PubApplIdentityData* 필드가 무시되므로 이 옵션은 적용되지 않습니다.

이전에 구독이 ID 컨텍스트 정보를 제공한 경우 이 옵션을 사용하지 않고 구독을 대체하면 대체된 구독에 대해 기본 컨텍스트 정보가 생성됩니다.

여러 사용자 ID가 MQSO\_ANY\_USERID 옵션과 함께 이 옵션을 사용할 수 있게 하는 구독이 다른 사용자 ID에 의해 재개되는 경우 구독을 소유한 새 사용자 ID에 대한 기본 ID 컨텍스트가 생성되고 새 ID 컨텍스트가 포함된 후속 발행물이 전달됩니다.

## MQSO\_FAIL\_IF QUIESCING

큐 관리자가 정지 중인 경우 MQSUB 호출에 실패합니다. z/OS의 CICS 또는 IMS 애플리케이션의 경우 이 옵션은 연결이 정지 중 상태일 때 MQSUB 호출이 실패하게 합니다.

## MQSD의 경우 ObjectName (MQCHAR48)

로컬 큐 관리자에서 정의된 토픽 오브젝트의 이름입니다.

이름에는 다음 문자가 포함될 수 있습니다.

- 대문자 알파벳(A - Z)
- 소문자 알파벳(a - z)
- 숫자(0 - 9)
- 마침표(.), 슬래시(/), 밑줄(\_), 퍼센트(%)

이름에 선두 문자 또는 임베드된 공백을 사용할 수 없으나 후미 문자 공백은 포함할 수 있습니다. 이름에서 중요한 데이터의 끝을 표시하기 위해 널 문자를 사용하십시오. 널 및 그 뒤에 오는 모든 문자는 공백으로 처리됩니다. 다음 제한사항이 표시된 환경에서 적용됩니다.

- EBCDIC 가타카나를 사용하는 시스템에서는 소문자를 사용할 수 없습니다.
- z/OS의 경우:
  - 밑줄로 시작하거나 끝나는 이름을 피하십시오. 조작 및 제어판에서 이를 처리할 수 없습니다.
  - 퍼센트 문자에는 RACF에 대한 특별한 의미가 있습니다. RACF가 외부 보안 관리자로 사용되는 경우 이름은 퍼센트를 포함할 수 없습니다. 퍼센트를 포함하는 경우 RACF 일반 프로파일이 사용될 때 보안 검사에 해당 이름이 포함되지 않습니다.
- IBM i에서 소문자, 슬래시 또는 퍼센트를 포함하는 이름은 명령에 지정할 때 따옴표로 묶어야 합니다. 이 인용 부호는 구조의 필드로서 또는 호출의 매개변수로서 발생하는 이름에는 지정될 수 없습니다.

*ObjectName*은 전체 토픽 이름을 구성하는 데 사용됩니다.

전체 토픽 이름은 두 개의 서로 다른 필드 *ObjectName* 및 *ObjectString*에서 빌드될 수 있습니다. 이 두 필드를 사용하는 방법에 대한 자세한 내용은 [토픽 문자열 결합](#)을 참조하십시오.

*ObjectName* 필드로 식별된 오브젝트를 찾을 수 없는 경우 *ObjectString*에서 지정된 문자열이 있어도 이유 코드 MQRC\_UNKNOWN\_OBJECT\_NAME과 함께 호출이 실패합니다.

MQSO\_RESUME 옵션을 사용하여 MQSUB 호출에서 리턴 시 이 필드는 변경되지 않습니다.

이 필드의 길이는 MQ\_TOPIC\_NAME\_LENGTH로 지정됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백입니다.

MQSO\_ALTER 옵션을 사용하여 기존 구독을 대체하는 경우 구독된 토픽 오브젝트의 이름은 변경할 수 없습니다. 이 필드 및 *ObjectString* 필드는 생략할 수 있습니다. 제공되면 동일한 전체 토픽 이름으로 해석해야 합니다. 그렇지 않은 경우 MQRC\_TOPIC\_NOT\_ALTERABLE과 함께 호출이 실패합니다.

## MQSD에 대한 AlternateUserID (MQCHAR12)

MQSO\_ALTERNATE\_USER\_AUTHORITY를 지정하는 경우 이 필드는 애플리케이션이 현재 실행 중인 사용자 ID 대신 MQSUB 호출의 **Hobj** 매개변수에서 지정된 목적지 큐에 대한 출력 및 구독의 권한을 검사하는 데 사용되는 대체 사용자 ID를 포함합니다.

성공적이면 이 필드에 지정된 사용자 ID는 애플리케이션이 현재 실행되고 있는 사용자 ID 대신 사용자 ID를 소유하는 구독으로 기록됩니다.

MQSO\_ALTERNATE\_USER\_AUTHORITY가 지정되고 이 필드가 필드의 끝 또는 첫 번째 널 문자까지 완전히 공백인 경우 사용자 권한이 출력을 위한 목적지 큐 또는 지정된 옵션이 있는 이 토픽에 대해 구독해야 하는 경우에만 구독이 성공할 수 있습니다.

MQSO\_ALTERNATE\_USER\_AUTHORITY가 지정되지 않으면 이 필드가 무시됩니다.

표시된 환경에서 다음 차이점이 존재합니다.

- z/OS에서 AlternateUserId 중 처음 8자만 구독의 권한을 검사하는 데 사용됩니다. 그러나 현재 사용자 ID에 이 특정 대체 사용자 ID를 지정할 권한이 부여되어야 합니다. 대체 사용자 ID 중 12자 모두 이 검사에 사용됩니다. 사용자 ID는 외부 보안 관리자가 허용하는 문자만 포함해야 합니다.

MQSO\_RESUME을 사용하여 MQSUB 호출에서 리턴 시 이 필드는 변경되지 않습니다.

입력 필드입니다. 이 필드의 길이는 MQ\_USER\_ID\_LENGTH에 지정되어 있습니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 12자의 공백 문자입니다.

### **MQSD의 AlternateSecurityID (MQBYTE40)**

적절한 권한 검사를 수행할 수 있도록 AlternateUserId와 함께 권한 서비스에 전달되는 보안 ID입니다.

AlternateSecurityId는 MQSO\_ALTERNATE\_USER\_AUTHORITY가 지정된 경우에만 사용되며, AlternateUserId 필드는 첫 번째 널 문자 또는 필드의 끝까지 전체적으로 비어있지 않습니다.

MQSO\_RESUME을 사용하여 MQSUB 호출에서 리턴 시 이 필드는 변경되지 않습니다.

자세한 정보는 MQOD 데이터 유형에서 [477 페이지의 『MQOD에 대한 AlternateSecurityID \(MQBYTE40\)』](#)에 대한 설명을 참조하십시오.

### **MQSD의 경우 SubExpiry (MQLONG)**

10분의 1초로 표시되는 시간이며 이 시간 이후에 구독이 만료됩니다. 이 간격이 지나면 더 이상의 발행물은 이 구독과 일치하지 않습니다. 구독이 만기되는 즉시 발행이 큐에 더 이상 전송되지 않습니다. 하지만 이미 있는 발행은 영향을 받지 않습니다. *SubExpiry*에는 발행 만기에 대한 영향이 없습니다.

다음 특수 값이 인식됩니다.

#### **MQEI\_UNLIMITED**

구독에는 만기 시간이 없습니다.

MQSO\_ALTER 옵션을 사용하여 기존 구독을 대체하는 경우 구독의 만기를 변경할 수 있습니다.

MQSO\_RESUME 옵션을 사용하여 MQSUB 호출에서 리턴 시 이 필드는 남아 있는 만기 시간이 아니라 구독의 원래 만기 시간으로 설정됩니다.

### **MQSD의 경우 ObjectString (MQCHARV)**

이는 사용할 긴 오브젝트 이름입니다.

*ObjectString*은 전체 토픽 이름을 구성하는 데 사용됩니다.

전체 토픽 이름은 두 개의 서로 다른 필드 *ObjectName* 및 *ObjectString*에서 빌드될 수 있습니다. 이 두 필드를 사용하는 방법에 대한 자세한 내용은 [토픽 문자열 결합](#)을 참조하십시오.

*ObjectString*의 최대 길이는 10240입니다.

*ObjectString*이 *MQCHARV* 구조를 사용하는 방법에 대한 설명에 따라 올바르게 지정되지 않거나 최대 길이를 초과하는 경우 이유 코드 *MQRC\_OBJECT\_STRING\_ERROR*와 함께 호출이 실패합니다.

입력 필드입니다. 이 구조에 있는 필드의 초기값은 *MQCHARV* 구조의 값과 동일합니다.

*ObjectString*에서 와일드카드가 있는 경우 *MQSD*의 출력 필드에서 지정된 와일드카드 옵션을 사용하여 해당 와일드카드에 대한 해석을 제어할 수 있습니다.

*MQSO\_RESUME* 옵션을 사용하여 *MQSUB* 호출에서 리턴 시 이 필드는 변경되지 않습니다. 버퍼가 제공되면 사용된 전체 토픽 이름은 *ResObjectString* 필드에서 리턴됩니다.

*MQSO\_ALTER* 옵션을 사용하여 기존 구독을 대체하는 경우 구독된 토픽 오브젝트의 긴 이름은 변경할 수 없습니다. 이 필드 및 *ObjectName* 필드는 생략할 수 있습니다. 제공되면 동일한 전체 토픽 이름으로 해석해야 하거나 호출이 *MQRC\_TOPIC\_NOT\_ALTERABLE*과 함께 실패합니다.

### ***MQSD*의 경우 *SubName* (*MQCHARV*)**

이는 구독 이름을 지정합니다. *Options*가 *MQSO\_DURABLE* 옵션을 지정하는 경우에만 이 필드가 필요하지만 제공된 경우 *MQSO\_NON\_DURABLE*에 대해서도 큐 관리자가 사용합니다.

지정된 경우 *SubName*은 구독을 식별하는 데 사용된 메소드이므로 큐 관리자 내에서 고유해야 합니다.

*SubName*의 최대 길이는 10240입니다.

이 필드는 두 가지 목적을 서비스합니다. *MQSO\_DURABLE* 구독의 경우 이 필드를 사용하여 구독을 식별하므로 *MQCO\_KEEP\_SUB* 옵션을 사용하여 구독에 대한 핸들을 단거나 큐 관리자에서 연결을 끊은 경우 작성된 후 재개할 수 있습니다. 이는 *MQSO\_RESUME* 옵션과 함께 *MQSUB* 호출을 사용하여 수행됩니다. 또한 *DISPLAY SBSTATUS*의 *SUBID* 필드에 있는 *subscription*의 관리 뷰에도 표시됩니다.

*SubName*이 잘못 지정되면 *MQCHARV* 구조 사용 방법에 대한 설명에 따라 필요할 때 생략됩니다(즉, *SubName.VSLength*가 0임) 또는 최대 길이를 초과하는 경우 이유 코드 *MQRC\_SUB\_NAME\_ERROR*와 함께 호출이 실패합니다.

입력 필드입니다. 이 구조에 있는 필드의 초기값은 *MQCHARV* 구조의 값과 동일합니다.

*MQSO\_ALTER* 옵션을 사용하여 기존 구독을 대체하는 경우 참조된 구독을 찾는 데 사용된 식별 필드이므로 구독 이름을 변경할 수 없습니다. 이는 *MQSO\_RESUME* 옵션으로 *MQSUB* 호출에서 출력 시 변경되지 않습니다.

### ***MQSD*의 *SubUserData* (*MQCHARV*)**

이는 구독 사용자 데이터를 지정합니다. 이 필드에 제공된 구독에 대한 데이터는 이 구독에 송신된 모든 발행물의 *MQSubUserData* 메시지 특성으로 포함됩니다.

*SubUserData*의 최대 길이는 10240입니다.

*MQCHARV* 구조를 사용하는 방법에 대한 설명에 따라 *SubUserData*가 올바르게 지정되는 경우 또는 최대 길이를 초과하는 경우 이유 코드 *MQRC\_SUB\_USER\_DATA\_ERROR*와 함께 호출이 실패합니다.

입력 필드입니다. 이 구조에 있는 필드의 초기값은 *MQCHARV* 구조의 값과 동일합니다.

*MQSO\_ALTER* 옵션을 사용하여 기존 구독을 대체하는 경우 구독 사용자 데이터를 변경할 수 있습니다.

버퍼가 제공되고 *VSBufLen*에서 양의 버퍼 길이가 있는 경우 이 변수 길이 필드는 *MQSO\_RESUME* 옵션을 사용하여 *MQSUB* 호출에서 출력 시 리턴됩니다. 버퍼가 호출에서 제공되지 않는 경우 구독 사용자 데이터의 길이만 *MQCHARV*의 *VSLength* 필드에서 리턴됩니다. 제공된 버퍼가 필드를 리턴하는 데 필요한 공간보다 작은 경우 *VSBufLen*바이트만 제공된 버퍼에 리턴됩니다.

### ***MQSD*의 *SubCorrelID* (*MQBYTE24*)**

이 필드는 이 구독과 일치하는 모든 발행에 공통인 상관 ID를 포함합니다.



**주의:** 상관 ID는 계층이 아닌 발행/구독 클러스터의 큐 관리자 사이에만 전달될 수 있습니다.



이 구독과 일치하기 위해 송신된 모든 발행물은 메시지 디스크립터에서 이 상관 ID를 포함합니다. 여러 구독이 동일한 큐에서 발행을 가져오는 경우 상관 ID별로 MQGET을 사용하면 특정 구독의 발행만 확보할 수 있습니다. 이 상관 ID는 큐 관리자나 사용자가 생성할 수 있습니다.

옵션 MQSO\_SET\_CORREL\_ID가 지정되지 않으면 상관 ID는 큐 관리자에 의해 생성되고 이 필드는 이 구독에 발행된 각 메시지에서 설정되는 상관 ID가 포함된 출력 필드입니다. 생성된 상관 ID는 4바이트 제품 ID(ASCII 또는 EBCDIC 형식의 AMQX 또는 CSQM)로 구성되며, 그 뒤에 고유 문자열의 제품 특정 구현이 옵니다.

옵션 MQSO\_SET\_CORREL\_ID가 지정되면 상관 ID는 사용자에게 의해 생성되고 이 필드는 이 구독의 각 발행에서 설정되는 상관 ID가 포함된 입력 필드입니다. 이 경우 필드가 MQCI\_NONE을 포함하는 경우 이 구독에 발행된 각 메시지에서 설정된 상관 ID는 메시지의 원래 넣기로 작성된 상관 ID입니다.

MQSO\_GROUP\_SUB 옵션이 지정되고 지정된 상관 ID가 동일한 큐 및 겹치는 토픽 문자열을 사용하여 그룹화된 기존 구독과 동일한 경우 그룹에서 가장 중요한 구독에만 발행물 사본이 제공됩니다.

이 필드의 길이는 MQ\_CORREL\_ID\_LENGTH로 지정됩니다. 이 필드의 초기값은 MQCI\_NONE입니다.

MQSO\_ALTER 옵션을 사용하여 기존 구독을 대체하는 중이고 이 필드가 입력 필드인 경우, 구독이 그룹화된 구독이 아니면(즉 MQSO\_GROUP\_SUB 옵션을 사용하여 작성되었고 구독 ID를 변경할 수 없는 상황이 아니면) 구독 상관 ID가 변경될 수 있습니다.

MQSO\_RESUME을 사용하여 MQSUB 호출에서 리턴 시 이 필드는 구독의 현재 상관 ID로 설정됩니다.

### **MQSD의 경우 PubPriority (MQLONG)**

이는 이 구독과 일치하는 모든 발행 메시지에 대한 메시지 디스크립터(MQMD)의 *Priority* 필드에 있는 값입니다. MQMD에서 *Priority* 필드에 대한 자세한 정보는 439 페이지의 『MQMD의 우선순위 (MQLONG)』의 내용을 참조하십시오.

값은 0보다 크거나 같아야 하며, 0은 가장 낮은 우선순위입니다. 다음 특수 값을 사용할 수도 있습니다.

#### **MQPRI\_PRIORITY\_AS\_Q\_DEF**

구독 큐가 MQSUB 호출의 *Hobj* 필드에서 제공되고 관리 핸들이 아닌 경우 메시지의 우선순위는 이 큐의 **DefPriority** 속성에서 가져옵니다. 큐가 클러스터 큐이거나 큐 이름 해석 경로에 둘 이상의 정의가 있는 경우 439 페이지의 『MQMD의 우선순위 (MQLONG)』에 대해 설명된 대로 큐에 발행 메시지를 넣을 때 우선순위가 판별됩니다.

MQSUB 호출에서 관리되는 핸들을 사용하는 경우 메시지에 대한 우선순위는 구독하는 주제와 연관된 모델 큐의 **DefPriority** 속성에서 가져옵니다.

#### **MQPRI\_PRIORITY\_AS\_PUBLISHED**

메시지에 대한 우선순위는 원래 발행물의 우선순위입니다. 이는 필드의 초기값입니다.

MQSO\_ALTER 옵션을 사용하여 기존 구독을 대체하는 경우 이후 발행 메시지의 *Priority*를 변경할 수 있습니다.

MQSO\_RESUME을 사용하여 MQSUB 호출에서 리턴 시 이 필드는 구독에 사용되는 현재 우선순위로 설정됩니다.

### **MQSD의 PubAccounting토큰 (MQBYTE32)**

이는 이 구독과 일치하는 모든 발행 메시지에 대한 메시지 디스크립터(MQMD)의 *AccountingToken* 필드에 있는 값입니다. *AccountingToken*은 메시지 ID 컨텍스트의 부분입니다. 메시지 컨텍스트에 대한 자세한 정보는 메시지 컨텍스트를 참조하십시오. MQMD에서 *AccountingToken* 필드에 대한 자세한 정보는 446 페이지의 『MQMD의 경우 AccountingToken (MQBYTE32)』의 내용을 참조하십시오.

*PubAccountingToken* 필드에 대해 다음 특수 값을 사용할 수 있습니다.

#### **MQACT\_NONE**

계정 토큰이 지정되지 않습니다.



이 값은 필드 길이 만큼의 2진 0입니다.

C 프로그래밍 언어의 경우 상수 MQACT\_NONE\_ARRAY도 정의됩니다. 이는 MQACT\_NONE과 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

옵션 MQSO\_SET\_IDENTITY\_CONTEXT가 지정되지 않으면 큐 관리자가 기본 컨텍스트 정보로 회계 토큰을 생성하고 이 필드는 이 구독에 발행된 각 메시지에서 설정되는 *AccountingToken*이 포함된 출력 필드입니다.

옵션 MQSO\_SET\_IDENTITY\_CONTEXT가 지정되면 사용자가 회계 토큰을 생성하고 이 필드는 이 구독의 각 발행에서 설정되는 *AccountingToken*이 포함된 입력 필드입니다.

이 필드의 길이는 MQ\_ACCOUNTING\_TOKEN\_LENGTH로 지정됩니다. 이 필드의 초기값은 MQACT\_NONE입니다.

MQSO\_ALTER 옵션을 사용하여 기존 구독을 대체하는 경우 이후 발행 메시지의 *AccountingToken* 값을 변경할 수 있습니다.

MQSO\_RESUME을 사용하여 MQSUB 호출에서 리턴 시 이 필드는 구독에 사용되는 현재 *AccountingToken*으로 설정됩니다.

### **MQSD의 PubApplIdentityData (MQCHAR32)**

이는 이 구독과 일치하는 모든 발행 메시지에 대한 메시지 디스크립터(MQMD)의 *ApplIdentityData* 필드에 있는 값입니다. *ApplIdentityData*는 메시지 ID 컨텍스트의 부분입니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트를 참조하십시오](#). MQMD에서 *ApplIdentityData* 필드에 대한 자세한 정보는 [448 페이지의 『MQMD의 ApplIdentity데이터 \(MQCHAR32\)』의 내용을 참조하십시오](#).

옵션 MQSO\_SET\_IDENTITY\_CONTEXT가 지정되지 않은 경우 기본 컨텍스트 정보로, 이 구독에 대해 발행된 각 메시지에서 설정된 *ApplIdentityData*가 공백입니다.

옵션 MQSO\_SET\_IDENTITY\_CONTEXT가 지정된 경우 사용자가 *PubApplIdentityData*를 생성하고 이 필드는 이 구독에 대한 각 발행에서 설정할 *ApplIdentityData*를 포함하는 입력 필드입니다.

이 필드의 길이는 MQ\_APPL\_IDENTITY\_DATA\_LENGTH로 지정됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 32자의 공백 문자입니다.

MQSO\_ALTER 옵션을 사용하여 기존 구독을 대체하는 경우 이후 발행 메시지의 *ApplIdentityData*를 변경할 수 있습니다.

MQSO\_RESUME을 사용하여 MQSUB 호출에서 리턴 시 이 필드는 구독에 사용되는 현재 *ApplIdentityData*로 설정됩니다.

### **MQSD의 경우 SelectionString (MQCHARV)**

토픽에서 메시지를 구독할 때 사용되는 선택 기준을 제공하는 데 사용되는 문자열입니다.

버퍼가 제공되고 VSBufSize에서 양의 버퍼 길이가 있는 경우 이 변수 길이 필드는 MQSO\_RESUME 옵션을 사용하여 MQSUB 호출에서 출력 시 리턴됩니다. 버퍼가 호출에서 제공되지 않는 경우 선택 문자열의 길이만 MQCHARV의 VSLength 필드에서 리턴됩니다. 제공된 버퍼가 필드를 리턴하는 데 필요한 공간 보다 작은 경우, 제공된 버퍼에는 VSBufSize 바이트만 리턴됩니다.

[291 페이지의 『MQCHARV - 가변 길이 문자열』 구조의 사용 방법 설명에 의하면 SelectionString이 올바르게 제공되었거나 최대 길이를 초과하는 경우 호출은 MQRC\\_SELECTION\\_STRING\\_ERROR 이유 코드로 실패합니다.](#)

SelectionString 사용법은 [선택자](#)에서 설명됩니다.

### **MQSD의 경우 SubLevel (MQLONG)**

이는 구독과 연관된 레벨입니다. 발행 시 사용된 PubLevel보다 낮거나 동일한 가장 높은 SubLevel 값을 가진 구독 세트에 구독이 있는 경우에만 이 구독으로 발행이 전달됩니다. 그러나 발행이 유지된 경우 PubLevel 1에서 다 시 발행되기 때문에 상위 수준의 구독자는 더 이상 사용할 수 없습니다.

값의 범위는 0 - 9여야 합니다. 0이 가장 낮은 레벨입니다.

이 필드의 초기값은 1입니다.

자세한 정보는 [발행 인터셉트](#)를 참조하십시오.

MQSO ALTER 옵션을 사용하여 기존 구독을 대체하는 경우, SubLevel을 변경할 수 없습니다.

MQSO PUBLICATIONS\_ON\_REQUEST 옵션을 사용하여 SubLevel을 1보다 큰 값과 결합하는 것은 허용되지 않습니다.

MQSO RESUME을 사용하여 MQSUB 호출에서 리턴 시 이 필드는 구독에 사용되는 현재 레벨로 설정됩니다.

### MQSD에 대한 ResObject문자열 (MQCHARV)

이는 큐 관리자가 *ObjectName*에 제공된 이름을 해석한 후의 긴 오브젝트 이름입니다.

긴 오브젝트 이름을 *ObjectString*에 제공하고 *ObjectName*에는 아무 것도 입력하지 않은 경우, 이 필드에 리턴되는 값은 *ObjectString*에 제공한 값과 동일합니다.

이 필드가 생략되면(즉, ResObjectString.VSBufSize가 0임) ResObjectString이 리턴되지 않지만 길이가 ResObjectString.VSLength에서 리턴됩니다. 길이가 전체 ResObjectString보다 짧으면 잘리게 되며 여러 자의 가장 오른쪽 문자를 제공된 길이에 맞게 리턴합니다.

MQCHARV 구조의 사용 방법 설명에 의하면 ResObjectString이 올바르지 않게 지정되었거나 최대 길이를 초과하는 경우 호출은 MQRC\_RES\_OBJECT\_STRING\_ERROR 이유 코드로 실패합니다.

### MQSMPO - 메시지 특성 설정 옵션

MQSMPO 구조를 사용하여 애플리케이션이 메시지 특성이 설정되는 방법을 제어하는 옵션을 지정할 수 있습니다. 구조는 MQSETMP 호출의 입력 매개변수입니다.

### 가용성

모든 IBM MQ 시스템 및 IBM MQ 클라이언트.

### 문자 세트 및 인코딩

MQSMPO의 데이터는 애플리케이션의 문자 세트 및 애플리케이션의 인코딩 (MQENC\_NATIVE)에 있어야 합니다.

### 필드

참고: 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 531. MQSMPO의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQSMPO_STRUC_ID	'SMPO'
버전 (구조 버전 번호)	MQSMPO_VERSION_1	1
옵션 (옵션)	MQSMPO_NONE	0
ValueEncoding (등록 정보 값 인코딩)	MQENC_NATIVE	환경에 따라 다름
ValueCCSID (등록 정보 값 문자 세트)	MQCCSI_APPL	-3

표 531. MQSMP0의 필드 (계속)

필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
<p><b>참고:</b></p> <ol style="list-style-type: none"> <li>널 문자열 값 또는 공백은 C의 널 문자열과 기타 프로그래밍 언어의 공백 문자를 나타냅니다.</li> <li>C 프로그래밍 언어의 매크로 변수MQSMP0_DEFAULT는 테이블에 나열되는 값을 포함합니다. 즉, 구조 내의 필드에 대한 초기값을 제공하기 위해 다음과 같은 방법으로 사용될 수 있습니다.</li> </ol> <pre style="background-color: #f0f0f0; padding: 10px;">MQSMP0 MySMP0 = {MQSMP0_DEFAULT};</pre>		

## 언어 선언

### MQSMP0의 C 선언

```
typedef struct tagMQSMP0 MQSMP0;
struct tagMQSMP0 {
    MQCHAR4    StrucId;          /* Structure identifier */
    MQLONG     Version;         /* Structure version number */
    MQLONG     Options;        /* Options that control the action of MQSETMP */
    MQLONG     ValueEncoding;   /* Encoding of Value */
    MQLONG     ValueCCSID;     /* Character set identifier of Value */
};
```

### MQSMP0의 COBOL 선언

```
** MQSMP0 structure
 10 MQSMP0.
**   Structure identifier
 15 MQSMP0-STRUCID      PIC X(4).
**   Structure version number
 15 MQSMP0-VERSION     PIC S9(9) BINARY.
**   Options that control the action of MQSETMP
 15 MQSMP0-OPTIONS    PIC S9(9) BINARY.
**   Encoding of VALUE
 15 MQSMP0-VALUEENCODING PIC S9(9) BINARY.
**   Character set identifier of VALUE
 15 MQSMP0-VALUECCSID  PIC S9(9) BINARY.
```

### MQSMP0의 PL/I 선언

```
dcl
 1 MQSMP0 based,
 3 StrucId      char(4),          /* Structure identifier */
 3 Version      fixed bin(31),   /* Structure version number */
 3 Options      fixed bin(31),   /* Options that control the action of MQSETMP */
 3 ValueEncoding fixed bin(31), /* Encoding of Value */
 3 ValueCCSID   fixed bin(31), /* Character set identifier of Value */
```

### MQSMP0의 상위 레벨 어셈블러 선언

```
MQSMP0          DSECT
MQSMP0_STRUCID  DS CL4   Structure identifier
MQSMP0_VERSION  DS F     Structure version number
MQSMP0_OPTIONS  DS F     Options that control the action of
*                MQSETMP
MQSMP0_VALUEENCODING DS F Encoding of VALUE
MQSMP0_VALUECCSID DS F   Character set identifier of VALUE
MQSMP0_LENGTH   EQU *-MQSMP0
MQSMP0_AREA     DS CL(MQSMP0_LENGTH)
```

## MQSMP0의 경우 StrucId (MQCHAR4)

메시지 특성 설정 옵션 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQSMP0\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

### **MQSMPO\_STRUC\_ID**

메시지 특성 설정 옵션 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 **MQSMPO\_STRUC\_ID\_ARRAY**도 정의됩니다. 이는 **MQSMPO\_STRUC\_ID**와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### **MQSMPO의 버전 (MQLONG)**

구조 버전 번호이며 값은 다음과 같아야 합니다.

#### **MQSMPO\_VERSION\_1**

버전-1 세트 메시지 특성 옵션 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### **MQSMPO\_CURRENT\_VERSION**

세트 메시지 특성 옵션 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 **MQSMPO\_VERSION\_1**입니다.

### **MQSMPO의 옵션 (MQLONG)**

#### **위치 옵션**

다음 옵션은 특성 커서와 비교하여 특성의 상대 위치와 관련이 있습니다.

#### **MQSMPO\_SET\_FIRST**

지정된 이름과 일치하는 첫 번째 특성의 값을 설정하거나 존재하지 않는 경우 일치 계층이 있는 다른 모든 특성 이후에 새 특성을 추가합니다.

#### **MQSMPO\_SET\_PROP\_UNDER\_CURSOR**

특성 커서에 의해 지정된 특성의 값을 설정합니다. 특성 커서로 지정된 특성은 **MQIMPO\_INQ\_FIRST** 또는 **MQIMPO\_INQ\_NEXT** 옵션을 사용하여 조회된 특성입니다.

메시지 핸들이 **MQGET** 호출 시 재사용되거나 메시지 핸들이 **MQPUT** 호출 시 **MQGMO** 또는 **MQPMO** 구조의 *MsgHandle* 필드에서 지정되는 경우 특성 커서가 재설정됩니다.

이 옵션이 특성 커서가 아직 설정되지 않았을 때 사용되거나 특성 커서에 대한 특성 포인터가 삭제된 경우 완료 코드 **MQCC\_FAILED** 및 이유 코드 **MQRC\_PROPERTY\_NOT\_AVAILABLE**과 함께 호출이 실패합니다.

#### **MQSMPO\_SET\_PROP\_BEFORE\_CURSOR**

특성 커서로 지정된 특성 전에 새 특성을 설정합니다. 특성 커서로 지정된 특성은 **MQIMPO\_INQ\_FIRST** 또는 **MQIMPO\_INQ\_NEXT** 옵션을 사용하여 조회된 특성입니다.

메시지 핸들이 **MQGET** 호출 시 재사용되거나 메시지 핸들이 **MQPUT** 호출 시 **MQGMO** 또는 **MQPMO** 구조의 *MsgHandle* 필드에서 지정되는 경우 특성 커서가 재설정됩니다.

이 옵션이 특성 커서가 아직 설정되지 않았을 때 사용되거나 특성 커서에 대한 특성 포인터가 삭제된 경우 완료 코드 **MQCC\_FAILED** 및 이유 코드 **MQRC\_PROPERTY\_NOT\_AVAILABLE**과 함께 호출이 실패합니다.

#### **MQSMPO\_SET\_PROP\_AFTER\_CURSOR**

특성 커서에 의해 지정된 특성 이후에 새 특성을 설정합니다. 특성 커서로 지정된 특성은 **MQIMPO\_INQ\_FIRST** 또는 **MQIMPO\_INQ\_NEXT** 옵션을 사용하여 조회된 특성입니다.

메시지 핸들이 **MQGET** 호출 시 재사용되거나 메시지 핸들이 **MQPUT** 호출 시 **MQGMO** 또는 **MQPMO** 구조의 *MsgHandle* 필드에서 지정되는 경우 특성 커서가 재설정됩니다.

이 옵션이 특성 커서가 아직 설정되지 않았을 때 사용되거나 특성 커서에 대한 특성 포인터가 삭제된 경우 완료 코드 **MQCC\_FAILED** 및 이유 코드 **MQRC\_PROPERTY\_NOT\_AVAILABLE**과 함께 호출이 실패합니다.

## MQSMPO\_APPEND\_PROPERTY

계층 구조가 일치하는 다른 모든 특성 뒤에 새 특성이 추가됩니다. 지정된 이름과 일치하는 특성이 하나 이상 있는 경우 새 특성은 특성 목록의 끝에 추가됩니다.

이 옵션을 사용하면 동일한 이름을 가진 특성 목록을 작성할 수 있습니다.

설명된 옵션 중 필요한 옵션이 없는 경우 다음 옵션을 사용할 수 있습니다.

## MQSMPO\_NONE

옵션이 지정되지 않았습니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQSMPO\_SET\_FIRST입니다.

## MQSMPO의 경우 ValueEncoding (MQLONG)

값이 숫자인 경우 설정될 특성 값의 인코딩.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQENC\_NATIVE입니다.

## MQSMPO의 경우 ValueCCSID (MQLONG)

값이 문자열인 경우 설정될 특성 값의 문자 세트.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQCCSI\_APPL입니다.

## MQSRO - 구독 요청 옵션

MQSRO 구조로 애플리케이션이 구독 요청이 작성되는 방법을 제어하는 옵션을 지정할 수 있습니다. 구조는 MQSUBRQ 호출의 입/출력 매개변수입니다.

## 가용성

MQSRO 구조는 다음 플랫폼에서 사용 가능합니다.

- ▶ AIX AIX
- ▶ IBM i IBM i
- ▶ Linux Linux
- ▶ Windows Windows
- ▶ z/OS z/OS

및 해당 시스템에 연결된 IBM MQ MQI clients

## 버전

MQSRO의 현재 버전은 MQSRO\_VERSION\_1입니다.

## 문자 세트 및 인코딩

MQSRO의 데이터는 MQENC\_NATIVE로 지정된 로컬 큐 관리자의 인코딩 및 **CodedCharSetId** 큐 관리자 속성으로 지정된 문자 세트에 있어야 합니다. 그러나 애플리케이션이 MQ MQI 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트와 인코딩으로 작성되어야 합니다.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQSRO_STRUC_ID	'SRO~'
버전 (구조 버전 번호)	MQSRO_VERSION_1	1
옵션 (옵션)	MQSRO_NONE	0
NumPubs (서적 수)	없음	0

**참고:**

- ~ 기호는 단일 공백 문자를 나타냅니다.
- C 프로그래밍 언어의 매크로 변수 MQSRO\_DEFAULT는 테이블에 나열되는 값을 포함합니다. 즉, 구조 내의 필드에 대한 초기값을 제공하기 위해 다음과 같은 방법으로 사용될 수 있습니다.

```
MQSRO MySRO = {MQSRO_DEFAULT};
```

## 언어 선언

### MQSRO의 C 선언

```
typedef struct tagMQSRO MQSRO;
struct tagMQSRO {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;          /* Structure version number */
    MQLONG     Options;          /* Options that control the action of MQSUBRQ */
    MQLONG     NumPubs;          /* Number of publications sent */
    /* Ver:1 */
};
```

### MQSRO의 COBOL 선언

```
** MQSRO structure
10  MQSRO.
** Structure identifier
15  MQSRO-STRUCID          PIC X(4).
** Structure version number
15  MQSRO-VERSION         PIC S9(9) BINARY.
** Options that control the action of MQSUBRQ
15  MQSRO-OPTIONS        PIC S9(9) BINARY.
** Number of publications sent
15  MQSRO-NUMPUBS        PIC S9(9) BINARY.
```

### MQSRO의 PL/I 선언

```
dcl
1  MQSRO based,
3  StrucId          char(4),          /* Structure identifier */
3  Version          fixed bin(31),    /* Structure version number */
3  Options          fixed bin(31),    /* Options that control the action of MQSUBRQ */
3  NumPubs          fixed bin(31);    /* Number of publications sent */
```

### MQSRO의 상위 레벨 어셈블러 선언

```
MQSRO          DSECT
MQSRO_STRUCID  DS    CL4  Structure identifier
MQSRO_VERSION  DS    F    Structure version number
MQSRO_OPTIONS  DS    F    Options that control the action of MQSUBRQ
MQSRO_NUMPUBS  DS    F    Number of publications sent
*
MQSRO_LENGTH   EQU    *-MQSRO
MQSRO_AREA     DS    CL(MQSRO_LENGTH)
```

## **MQSRO의 경우 StrucId (MQCHAR4)**

구독 요청 옵션 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQSRO\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

### **MQSRO\_STRUC\_ID**

구독 요청 옵션 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQSRO\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQSRO\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

## **MQSRO의 버전 (MQLONG)**

구조 버전 번호이며 값은 다음과 같아야 합니다.

### **MQSRO\_VERSION\_1**

버전-1 구독 요청 옵션 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

### **MQSRO\_CURRENT\_VERSION**

구독 요청 옵션 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MQSRO\_VERSION\_1입니다.

## **MQSRO에 대한 옵션 (MQLONG)**

다음 옵션 중 하나를 지정해야 합니다. 하나의 옵션만 지정할 수 있습니다.

### **MQSRO\_FAIL\_IF QUIESCING**

큐 관리자가 정지 중인 경우 MQSUBRQ 호출이 실패합니다. z/OS에서 CICS 또는 IMS 애플리케이션의 경우 이 옵션은 연결이 정지 상태인 경우 MQSUBRQ 호출이 실패하도록 강제 실행합니다.

**기본 옵션:** 이전에 설명된 옵션이 필요하지 않은 경우 다음 옵션을 사용해야 합니다.

### **MQSRO\_NONE**

기타 옵션이 지정되지 않았음을 표시하려면 이 값을 사용하십시오. 모든 옵션은 기본 값을 가정합니다.

MQSRO\_NONE은 프로그램 문서화를 돕습니다. 이 옵션이 다른 옵션과 함께 사용하도록 작성된 것은 아니지만 해당 값이 0이기 때문에 이 사용을 감지할 수 없습니다.

## **MQSRO의 경우 NumPubs (MQLONG)**

이 호출의 결과로 구독 큐에 송신된 발행물의 수를 표시하기 위해 애플리케이션에 리턴된 출력 필드입니다. 이 수의 발행물이 이 호출의 결과로 송신되었다더라도 이 많은 메시지를 특히, 비지속 메시지만인 경우 애플리케이션에서 가져올 수 있다고 보장하지 않습니다.

구독된 토픽이 와일드카드를 포함하는 경우 둘 이상의 발행이 있을 수 있습니다. *Hsub*가 표시하는 구독이 작성될 때 토픽 문자열에 와일드카드가 없는 경우 이 호출의 결과로 최대 하나의 발행이 전송됩니다.

## **MQSTS - 상태 보고 구조**

MQSTS 구조는 MQSTAT 명령의 출력 매개변수입니다. MQSTAT 명령은 상태 정보를 검색하는 데 사용됩니다. 이 정보는 MQSTS 구조로 리턴됩니다.

## **문자 세트 및 인코딩**

MQSTS의 문자 데이터는 로컬 큐 관리자의 문자 세트에 있습니다. 이는 *CodedCharSetId* 큐 관리자 속성에 의해 제공됩니다. MQSTS의 숫자 데이터는 고유 시스템 인코딩에 있습니다. 이는 *Encoding*으로 지정됩니다.

## **필드**

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 532. MQSTS의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQSTS_STRUC_ID	'STAT~'
버전 (구조 버전 번호)	MQSTS_VERSION_1	1
CompCode (첫 번째 오류의 완료 코드)	MQCC_OK	0
이유 (첫 번째 오류의 이유 코드)	MQRC_NONE	0
PutSuccessCount (성공한 비동기 넣기 호출 수)	없음	0
PutWarning수 (경고가 있는 비동기 넣기 호출 수)	없음	0
PutFailureCount (실패한 비동기 넣기 호출 수)	없음	0
ObjectType (실패한 오브젝트의 유형)	MQOT_Q	1
ObjectName (실패한 오브젝트의 이름)	없음	널 문자열 또는 공백
ObjectQMgr이름 (실패한 오브젝트를 소유하는 큐 관리자의 이름)	없음	널 문자열 또는 공백
ResolvedObject이름 (목적지 큐의 해석된 이름)	없음	널 문자열 또는 공백
ResolvedQMgr이름 (목적지 큐 관리자의 해석된 이름)	없음	널 문자열 또는 공백
<b>참고:</b> 버전이 MQSTS_VERSION_2 미만인 경우 나머지 필드는 무시됩니다.		
ObjectString (실패한 오브젝트의 긴 오브젝트 이름)	MQCHARV_DEFAULT	{NULL,0,0,0,-3}
SubName (실패한 등록의 등록 이름)	MQCHARV_DEFAULT	{NULL,0,0,0,-3}
OpenOptions (실패와 연관된 열기 옵션)	없음	0
SubOptions(실패와 연관된 구독 옵션)	없음	0
<b>참고:</b> 1. ~ 기호는 단일 공백 문자를 나타냅니다. 2. 널 문자열 값 또는 공백은 C의 널 문자열과 기타 프로그래밍 언어의 공백 문자를 나타냅니다. 3. C 프로그래밍 언어에서 매크로 변수 MQSTS_DEFAULT는 테이블에 나열되는 값을 포함합니다. 즉, 구조 내의 필드에 대한 초기값을 제공하기 위해 다음과 같은 방법으로 사용될 수 있습니다. <pre style="background-color: #f0f0f0; padding: 5px; margin-top: 10px;">MQSTS MySTS = {MQSTS_DEFAULT};</pre>		

## 언어 선언

### MQSTS의 C 선언

```
typedef struct tagMQSTS MQSTS;
struct tagMQSTS {
    MQCHAR4   StructId;           /* Structure identifier */
    MQLONG    Version;           /* Structure version number */
    MQLONG    CompCode;          /* Completion Code of first error */
    MQLONG    Reason;            /* Reason Code of first error */
    MQLONG    PutSuccessCount;    /* Number of Async calls succeeded */
    MQLONG    PutWarningCount;    /* Number of Async calls had warnings */
    MQLONG    PutFailureCount;    /* Number of Async calls had failures */
    MQLONG    ObjectType;        /* Failing object type */
    MQCHAR48  ObjectName;        /* Failing object name */
    MQCHAR48  ObjectQMgrName;     /* Failing object queue manager name */
    MQCHAR48  ResolvedObjectName; /* Resolved name of destination queue */
    MQCHAR48  ResolvedQMgrName;   /* Resolved name of destination qmgr */
} /* Ver:1 */
```



```

MQCHARV  ObjectString;      /* Failing object long name */
MQCHARV  SubName;          /* Failing subscription name */
MQLONG   OpenOptions;     /* Failing open options */
MQLONG   SubOptions;      /* Failing subscription options */
/* Ver:2 */
};

```

## MQSTS의 COBOL 선언

```

** MQSTS structure
10 MQSTS.
** Structure identifier
15 MQSTS-STRUCID PIC X(4).
** Structure version number
15 MQSTS-VERSION PIC S9(9) BINARY.
** Completion Code of first error
15 MQSTS-COMPCODE PIC S9(9) BINARY.
** Reason Code of first error
15 MQSTS-REASON PIC S9(9) BINARY.
** Number of Async put calls succeeded
15 MQSTS-PUTSUCCESSCOUNT PIC S9(9) BINARY.
** Number of Async put calls had warnings
15 MQSTS-PUTWARNINGCOUNT PIC S9(9) BINARY.
** Number of Async put calls had failures
15 MQSTS-PUTFAILURECOUNT PIC S9(9) BINARY.
** Failing object type
15 MQSTS-OBJECTTYPE PIC S9(9) BINARY.
** Failing object name
15 MQSTS-OBJECTNAME PIC X(48).
** Failing object queue manager
15 MQSTS-OBJECTQMGRNAME PIC X(48).
** Resolved name of destination queue
15 MQSTS-RESOLVEDOBJECTNAME PIC X(48).
** Resolved name of destination qmgr
15 MQSTS-RESOLVEDQMGRNAME PIC X(48).
** Ver:1 **
** Failing object long name
15 MQSTS-OBJECTSTRING.
** Address of variable length string
20 MQSTS-OBJECTSTRING-VSPTR POINTER.
** Offset of variable length string
20 MQSTS-OBJECTSTRING-VSOFFSET PIC S9(9) BINARY.
** Size of buffer
20 MQSTS-OBJECTSTRING-VSBUFSIZE PIC S9(9) BINARY.
** Length of variable length string
20 MQSTS-OBJECTSTRING-VSLENGTH PIC S9(9) BINARY.
** CCSID of variable length string
20 MQSTS-OBJECTSTRING-VSCCSID PIC S9(9) BINARY.
** Failing subscription name
15 MQSTS-SUBNAME.
** Address of variable length string
20 MQSTS-SUBNAME-VSPTR POINTER.
** Offset of variable length string
20 MQSTS-SUBNAME-VSOFFSET PIC S9(9) BINARY.
** Size of buffer
20 MQSTS-SUBNAME-VSBUFSIZE PIC S9(9) BINARY.
** Length of variable length string
20 MQSTS-SUBNAME-VSLENGTH PIC S9(9) BINARY.
** CCSID of variable length string
20 MQSTS-SUBNAME-VSCCSID PIC S9(9) BINARY.
** Failing open options
15 MQSTS-OPENOPTIONS PIC S9(9) BINARY.
** Failing subscription options
15 MQSTS-SUBOPTIONS PIC S9(9) BINARY.
** Ver:2 **

```

## MQSTS의 PL/I 선언

```

dcl
1 MQSTS based,
3 StructId          char(4),          /* Structure identifier */
3 Version           fixed bin(31),   /* Structure version number */
3 CompCode         fixed bin(31),   /* Completion code */
3 Reason           fixed bin(31),   /* Reason code */
3 PutSuccessCount  fixed bin(31),   /* Put success count */
3 PutWarningCount  fixed bin(31),   /* Put warning count */
3 PutFailureCount  fixed bin(31),   /* Put failure count */

```

```

3 ObjectType          fixed bin(31), /* Object type */
3 ObjectName         char(48), /* Object name */
3 ObjectQmgrName     char(48), /* Object queue manager */
3 ResolvedObjectName char(48), /* Resolved Object name */
3 ResolvedQmgrName   char(48); /* Resolved Object queue manager */
/* Ver:1 */
3 ObjectString,      /* Failing object long name */
5 VSPtr pointer,    /* Address of variable length string */
5 VSOFFSET fixed bin(31), /* Offset of variable length string */
5 VSBUFSize fixed bin(31), /* Size of buffer */
5 VSLength fixed bin(31), /* Length of variable length string */
5 VSCCSID fixed bin(31); /* CCSID of variable length string */
3 SubName,          /* Failing subscription name */
5 VSPtr pointer,    /* Address of variable length string */
5 VSOFFSET fixed bin(31), /* Offset of variable length string */
5 VSBUFSize fixed bin(31), /* Size of buffer */
5 VSLength fixed bin(31), /* Length of variable length string */
5 VSCCSID fixed bin(31); /* CCSID of variable length string */
3 OpenOptions fixed bin(31), /* Failing open options */
3 SubOptions fixed bin(31); /* Failing subscription options */
/* Ver:2 */

```

## MQSTS의 상위 레벨 어셈블러 선언

MQSTS	DSECT		
MQSTS_STRUCID	DS	CL4	Structure identifier
MQSTS_VERSION	DS	F	Structure version number
MQSTS_COMPCODE	DS	F	Completion code
MQSTS_REASON	DS	F	Reason code
MQSTS_PUTSUCCESSCOUNT	DS	F	Success count
MQSTS_PUTWARNINGCOUNT	DS	F	Warning count
MQSTS_PUTFAILURECOUNT	DS	F	Failure count
MQSTS_OBJTYPE	DS	F	Object type
MQSTS_OBJNAME	DS	CL48	Object name
MQSTS_OBJQMGR	DS	CL48	Object queue manager
MQSTS_ROBJNAME	DS	CL48	Resolved object name
MQSTS_ROBJQMGR	DS	CL48	Resolved object queue manager
MQSTS_OBJECTSTRING	DS	0F	Force fullword alignment
MQSTS_OBJECTSTRING_VSPTR	DS	A	Address of variable length string
MQSTS_OBJECTSTRING_VSOFFSET	DS	F	Offset of variable length string
MQSTS_OBJECTSTRING_VSBUFSize	DS	F	Size of buffer
MQSTS_OBJECTSTRING_VSLength	DS	F	Length of variable length string
MQSTS_OBJECTSTRING_VSCCSID	DS	F	CCSID of variable length string
MQSTS_OBJECTSTRING_LENGTH	EQU	*	MQSTS_OBJECTSTRING
MQSTS_OBJECTSTRING_AREA	DS	ORG	MQSTS_OBJECTSTRING
			CL(MQSTS_OBJECTSTRING_LENGTH)
*			
MQSTS_SUBNAME	DS	0F	Force fullword alignment
MQSTS_SUBNAME_VSPTR	DS	A	Address of variable length string
MQSTS_SUBNAME_VSOFFSET	DS	F	Offset of variable length string
MQSTS_SUBNAME_VSBUFSize	DS	F	Size of buffer
MQSTS_SUBNAME_VSLength	DS	F	Length of variable length string
MQSTS_SUBNAME_VSCCSID	DS	F	CCSID of variable length string
MQSTS_SUBNAME_LENGTH	EQ	*	MQSTS_SUBNAME
			ORG MQSTS_SUBNAME
MQSTS_SUBNAME_AREA	DS		CL(MQSTS_SUBNAME_LENGTH)
*			
MQSTS_OPENOPTIONS	DS	F	Failing open options
MQSTS_SUBOPTIONS	DS	F	Failing subscription option
MQSTS_LENGTH	EQU	*	MQSTS
			ORG MQSTS
MQSTS_AREA	DS		CL(MQSTS_LENGTH)

## 관련 참조

755 페이지의 『MQSTAT - 상태 정보 검색』

상태 정보를 검색하려면 MQSTAT 호출을 사용하십시오. 리턴된 상태 정보 유형은 호출에서 지정된 유형 값으로 판별됩니다.

## MQSTS의 경우 StrucId (MQCHAR4)

상태 보고 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQSTS\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

### MQSTS\_STRUC\_ID

상태 보고 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQSTS\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQSTS\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### **MQSTS의 버전 (MQLONG)**

구조 버전 번호.

값은 다음과 같아야 합니다.

#### **MQSTS\_VERSION\_1**

버전 1 상태 보고 구조입니다.

#### **MQSTS\_VERSION\_2**

버전 2 상태 보고 구조입니다.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### **MQSTS\_CURRENT\_VERSION**

상태 보고 구조의 현재 버전입니다. 현재 버전은 MQSTS\_VERSION\_2입니다.

Version은 항상 입력 필드입니다. 해당 초기값은 MQSTS\_VERSION\_1입니다.

### **MQSTS의 경우 CompCode (MQLONG)**

보고되는 조작의 완료 코드입니다.

CompCode에 대한 해석은 MQSTAT **Type** 매개변수의 값에 따라 다릅니다.

#### **MQSTAT\_TYPE\_ASYNC\_ERROR**

이는 ObjectName에서 지정된 오브젝트의 이전 비동기 넣기 조작에서 발생하는 완료 코드입니다.

#### **MQSTAT\_TYPE\_RECONNECTION**

연결이 다시 연결 중이거나 다시 연결에 실패한 경우 이는 연결이 다시 연결을 시작하게 한 완료 코드입니다.

현재 연결되면 값은 MQCC\_OK입니다.

#### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

연결이 다시 연결에 실패하는 경우 이는 다시 연결을 실패하게 한 완료 코드입니다.

현재 연결되거나 다시 연결 중인 경우 값은 MQCC\_OK입니다.

CompCode는 항상 출력 필드입니다. 해당 초기값은 MQCC\_OK입니다.

### **MQSTS에 대한 이유 (MQLONG)**

보고되는 조작의 이유 코드입니다.

Reason에 대한 해석은 MQSTAT **Type** 매개변수의 값에 따라 다릅니다.

#### **MQSTAT\_TYPE\_ASYNC\_ERROR**

이는 ObjectName에서 지정된 오브젝트의 이전 비동기 넣기 조작에서 발생하는 이유 코드입니다.

#### **MQSTAT\_TYPE\_RECONNECTION**

연결이 다시 연결 중이거나 다시 연결에 실패한 경우 이는 다시 연결이 다시 연결을 시작하게 한 이유 코드입니다.

현재 연결되면 값은 MQRC\_NONE입니다.

#### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

연결이 다시 연결에 실패하는 경우 이는 다시 연결을 실패하게 한 이유 코드입니다.

현재 연결되거나 다시 연결 중인 경우 값은 MQRC\_NONE입니다.

Reason은 출력 필드입니다. 초기값은 MQRC\_NONE입니다.

### **MQSTS에 대한 PutSuccess수 (MQLONG)**

성공한 비동기 넣기 조작의 수입입니다.

PutSuccessCount의 값은 MQSTAT Type 매개변수의 값에 따라 다릅니다.

#### **MQSTAT\_TYPE\_ASYNC\_ERROR**

MQCC\_OK와 함께 완료된 MQSTS 구조에서 이름 지정된 오브젝트에 대한 비동기 넣기 조작의 수입입니다.

#### **MQSTAT\_TYPE\_RECONNECTION**

0입니다.

#### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

0입니다.

PutSuccessCount는 출력 필드입니다. 초기값은 0입니다.

### **MQSTS에 대한 PutWarning수 (MQLONG)**

경고로 끝난 비동기 넣기 조작의 수입입니다.

PutWarningCount의 값은 MQSTAT Type 매개변수의 값에 따라 다릅니다.

#### **MQSTAT\_TYPE\_ASYNC\_ERROR**

MQCC\_WARNING과 함께 완료된 MQSTS 구조에서 이름 지정된 오브젝트에 대한 비동기 넣기 조작의 수입입니다.

#### **MQSTAT\_TYPE\_RECONNECTION**

0입니다.

#### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

0입니다.

PutWarningCount는 출력 필드입니다. 초기값은 0입니다.

### **MQSTS에 대한 PutFailure수 (MQLONG)**

실패한 비동기 넣기 조작의 수입입니다.

PutFailureCount의 값은 MQSTAT Type 매개변수의 값에 따라 다릅니다.

#### **MQSTAT\_TYPE\_ASYNC\_ERROR**

MQCC\_FAILED와 함께 완료된 MQSTS 구조에서 이름 지정된 오브젝트에 대한 비동기 넣기 조작의 수입입니다.

#### **MQSTAT\_TYPE\_RECONNECTION**

0입니다.

#### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

0입니다.

PutFailureCount는 출력 필드입니다. 초기값은 0입니다.

### **MQSTS의 경우 ObjectType (MQLONG)**

보고되는 ObjectName에서 이름 지정된 오브젝트의 유형입니다.

ObjectType의 가능한 값은 163 페이지의 『MQOT \*(오브젝트 유형 및 확장 오브젝트 유형)』에 나열됩니다.

ObjectType은 출력 필드입니다. 해당 초기값은 MQOT\_Q입니다.

### **MQSTS의 경우 ObjectName (MQCHAR48)**

보고되는 오브젝트의 이름입니다.

ObjectName에 대한 해석은 MQSTAT Type 매개변수의 값에 따라 다릅니다.

## MQSTAT\_TYPE\_ASYNC\_ERROR

이는 넣기 조작에서 사용된 큐 또는 토픽의 이름이며 MQSTS 구조의 *CompCode* 및 *Reason* 필드에서 보고되는 실패입니다.

## MQSTAT\_TYPE\_RECONNECTION

연결이 다시 연결 중인 경우 이는 연결과 연관된 큐 관리자의 이름입니다.

## MQSTAT\_TYPE\_RECONNECTION\_ERROR

연결을 재연결하는 데 실패한 경우 이는 재연결을 실패하게 만든 오브젝트의 이름입니다. 실패에 대한 이유는 MQSTS 구조의 *CompCode* 및 *Reason* 필드에서 보고됩니다.

ObjectName은 출력 필드입니다. 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백 문자입니다.

## MQSTS의 ObjectQMgr이름 (MQCHAR48)

보고되는 큐 관리자의 이름입니다.

ObjectQMgrName에 대한 해석은 MQSTAT **Type** 매개변수의 값에 따라 다릅니다.

## MQSTAT\_TYPE\_ASYNC\_ERROR

다음은 *ObjectName* 오브젝트가 정의되어 있는 큐 관리자의 이름입니다. 첫 번째 널 문자 또는 필드의 마지막까지 완전히 비어 있는 이름은 애플리케이션이 연결된 큐 관리자를 나타냅니다(로컬 큐 관리자).

## MQSTAT\_TYPE\_RECONNECTION



**ObjectQMgrName** 필드는 다시 연결을 요청할 큐 관리자의 이름을 포함하거나, 큐 관리자가 지정되지 않은 경우 공백으로 둡니다. 가능한 경우 클라이언트는 해당 이름의 큐 관리자에 다시 연결하려고 시도합니다.



공백입니다.

## MQSTAT\_TYPE\_RECONNECTION\_ERROR

연결을 재연결하는 데 실패한 경우 이는 재연결을 실패하게 만든 오브젝트의 이름입니다. 실패에 대한 이유는 MQSTS 구조의 *CompCode* 및 *Reason* 필드에서 보고됩니다.

ObjectQMgrName은 출력 필드입니다. 해당 값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백 문자입니다.

## MQSTS에 대한 ResolvedObject이름 (MQCHAR48)

로컬 큐 관리자가 이름을 해석한 후 *ObjectName*으로 이름 지정된 오브젝트의 이름입니다.

ResolvedObjectName에 대한 해석은 MQSTAT **Type** 매개변수의 값에 따라 다릅니다.

## MQSTAT\_TYPE\_ASYNC\_ERROR

ResolvedObjectName은 로컬 큐 관리자가 이름을 해석한 후 *ObjectName*으로 이름 지정된 오브젝트의 이름입니다. 리턴된 이름은 *ResolvedQMgrName*이 식별한 큐 관리자에 있는 오브젝트의 이름입니다.

## MQSTAT\_TYPE\_RECONNECTION

공백입니다.

## MQSTAT\_TYPE\_RECONNECTION\_ERROR

공백입니다.

ResolvedObjectName은 출력 필드입니다. 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백 문자입니다.

## MQSTS의 ResolvedQMgr이름 (MQCHAR48)

로컬 큐 관리자가 이름을 해석한 후 목적지 큐 관리자의 이름입니다.

ResolvedQMGrName에 대한 해석은 MQSTAT **Type** 매개변수의 값에 따라 다릅니다.

#### **MQSTAT\_TYPE\_ASYNC\_ERROR**

ResolvedQMGrName은 로컬 큐 관리자가 이름을 해석한 후 목적지 큐 관리자의 이름입니다. 리턴된 이름은 *ResolvedObjectName*이 식별하는 오브젝트를 소유하는 큐 관리자의 이름입니다. *ResolvedQMGrName*은 로컬 큐 관리자의 이름일 수 있습니다.

#### **MQSTAT\_TYPE\_RECONNECTION**

공백입니다.

#### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

공백입니다.

ResolvedQMGrName은 항상 출력 필드입니다. 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백 문자입니다.

#### **MQSTS의 경우 ObjectString (MQCHARV)**

보고되고 있는 실패 오브젝트의 긴 오브젝트 이름. MQSTS의 버전 2 이상에만 제공됩니다.

ObjectString에 대한 해석은 MQSTAT **Type** 매개변수의 값에 따라 다릅니다.

#### **MQSTAT\_TYPE\_ASYNC\_ERROR**

이는 실패한 MQPUT 조작에 사용된 큐 또는 주제의 긴 오브젝트 이름입니다.

#### **MQSTAT\_TYPE\_RECONNECTION**

0 길이 문자열

#### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

재연결을 실패하게 만든 오브젝트의 긴 오브젝트 이름입니다.

ObjectString은 출력 필드입니다. 해당 초기값은 길이가 0인 문자열입니다.

#### **MQSTS의 경우 SubName (MQCHARV)**

실패한 구독의 이름. MQSTS의 버전 2 이상에만 제공됩니다.

SubName에 대한 해석은 MQSTAT **Type** 매개변수의 값에 따라 다릅니다.

#### **MQSTAT\_TYPE\_ASYNC\_ERROR**

0 길이 문자열.

#### **MQSTAT\_TYPE\_RECONNECTION**

0 길이 문자열.

#### **MQSTAT\_TYPE\_RECONNECTION\_ERROR**

재연결을 실패하게 만든 구독의 이름입니다. 사용 가능한 구독 이름이 없거나 실패가 구독과 관련이 없는 경우에는 길이가 0인 문자열 때문입니다.

SubName은 출력 필드입니다. 해당 초기값은 길이가 0인 문자열입니다.

#### **MQSTS의 경우 OpenOptions (MQLONG)**

OpenOptions는 보고되는 오브젝트를 여는 데 사용됩니다. MQSTS의 버전 2 이상에만 제공됩니다.

OpenOptions의 값은 MQSTAT **Type** 매개변수의 값에 따라 다릅니다.

#### **MQSTAT\_TYPE\_ASYNC\_ERROR**

0입니다.

#### **MQSTAT\_TYPE\_RECONNECTION**

0입니다.

## MQSTAT\_TYPE\_RECONNECTION\_ERROR

OpenOptions는 실패가 발생한 경우 사용됩니다. 실패에 대한 이유는 MQSTS 구조의 *CompCode* 및 *Reason* 필드에서 보고됩니다.

OpenOptions는 출력 필드입니다. 초기값은 0입니다.

## MQSTS의 경우 SubOptions (MQLONG)

SubOptions는 실패한 구독을 여는 데 사용됩니다. MQSTS의 버전 2 이상에만 제공됩니다.

SubOptions에 대한 해석은 MQSTAT **Type** 매개변수의 값에 따라 다릅니다.

## MQSTAT\_TYPE\_ASYNC\_ERROR

0입니다.

## MQSTAT\_TYPE\_RECONNECTION

0입니다.

## MQSTAT\_TYPE\_RECONNECTION\_ERROR

SubOptions는 실패가 발생한 경우 사용됩니다. 실패가 주제 구독과 관련이 없는 경우 리턴되는 값은 0입니다.

SubOptions는 출력 필드입니다. 초기값은 0입니다.

## MQTM - 트리거 메시지

MQTM 구조는 트리거 이벤트가 큐에 발생할 때 트리거 모니터 애플리케이션에 큐 관리자가 전송한 트리거 메시지의 데이터를 설명합니다. 이 구조는 TMI(IBM MQ Trigger Monitor Interface)의 일부이며 IBM MQ 프레임워크 인터페이스 중 하나입니다.

## 형식 이름

MQFMT\_TRIGGER

## 문자 세트 및 인코딩



MQTM의 문자 데이터는 MQTM을 생성하는 큐 관리자의 문자 세트에 있습니다. MQTM의 숫자 데이터는 MQTM을 생성하는 큐 관리자의 시스템 인코딩 데이터입니다.

MQTM의 문자 세트 및 인코딩은 *CodedCharSetId* 및 *Encoding* 필드에 의해 지정됩니다.

- MQMD(MQTM 구조가 메시지 데이터의 시작에 있는 경우) 또는
- MQTM 구조에 선행하는 헤더 구조(다른 모든 경우).

## 사용법

트리거 모니터 애플리케이션은 트리거 메시지의 일부 또는 모든 정보를 트리거 모니터 애플리케이션이 시작하는 애플리케이션으로 전달해야 할 수 있습니다. 시작된 애플리케이션이 필요로 할 수 있는 정보는 *QName*, *TriggerData*, *UserData*를 포함합니다. 환경에서 허용되고 시작된 애플리케이션에 편리한 항목에 따라, 트리거 모니터 애플리케이션은 MQTM 구조를 시작된 애플리케이션에 직접 전달하거나 MQTMC2 구조를 대신 전달할 수 있습니다. MQTMC2에 대한 정보는 585 페이지의 『MQTMC2 - 트리거 메시지 2(문자 형식)』의 내용을 참조하십시오.

-  z/OS에서 CKTI 트랜잭션을 사용하여 시작되는 MQAT\_CICS 애플리케이션의 경우, 전체 트리거 메시지 구조 MQTM은 시작된 트랜잭션에서 사용할 수 있도록 작성됩니다. 정보는 EXEC CICS RETRIEVE 명령을 사용하여 검색할 수 있습니다.
-  IBM i에서 IBM MQ가 있는 트리거 모니터 애플리케이션은 MQTMC2 구조를 시작된 애플리케이션에 전달합니다.

트리거 사용에 대한 정보는 트리거를 사용하여 IBM MQ 애플리케이션 시작을 참조하십시오.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 533. MQTM에 대한 MQTM의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQTM_STRUC_ID	'TM~~'
버전 (구조 버전 번호)	MQTM_VERSION_1	1
QName (트리거된 큐의 이름)	없음	널 문자열 또는 공백
ProcessName (프로세스 오브젝트의 이름)	없음	널 문자열 또는 공백
TriggerData (트리거 데이터)	없음	널 문자열 또는 공백
ApplType (애플리케이션 유형)	없음	0
AppId (애플리케이션 ID)	없음	널 문자열 또는 공백
EnvData (환경 데이터)	없음	널 문자열 또는 공백
UserData (사용자 데이터)	없음	널 문자열 또는 공백

**참고:**

- ~ 기호는 단일 공백 문자를 나타냅니다.
- 널 문자열 값 또는 공백은 C의 널 문자열과 기타 프로그래밍 언어의 공백 문자를 나타냅니다.
- C 프로그래밍 언어의 매크로 변수MQTM\_DEFAULT는 테이블에 나열되는 값을 포함합니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오.

```
MQTM MyTM = {MQTM_DEFAULT};
```

## 언어 선언

### MQTM의 C 선언

```
typedef struct tagMQTM MQTM;
struct tagMQTM {
    MQCHAR4    StrucId;        /* Structure identifier */
    MQLONG     Version;       /* Structure version number */
    MQCHAR48   QName;         /* Name of triggered queue */
    MQCHAR48   ProcessName;   /* Name of process object */
    MQCHAR64   TriggerData;   /* Trigger data */
    MQLONG     ApplType;      /* Application type */
    MQCHAR256  AppId;         /* Application identifier */
    MQCHAR128  EnvData;       /* Environment data */
    MQCHAR128  UserData;      /* User data */
};
```

### MQTM의 COBOL 선언

```
**      MQTM structure
10  MQTM.
**      Structure identifier
15  MQTM-STRUCID    PIC X(4).
**      Structure version number
15  MQTM-VERSION   PIC S9(9) BINARY.
**      Name of triggered queue
15  MQTM-QNAME     PIC X(48).
**      Name of process object
15  MQTM-PROCESSNAME PIC X(48).
**      Trigger data
15  MQTM-TRIGGERDATA PIC X(64).
**      Application type
```



```

15 MQTM-APPLTYPE PIC S9(9) BINARY.
** Application identifier
15 MQTM-APPLID PIC X(256).
** Environment data
15 MQTM-ENVDATA PIC X(128).
** User data
15 MQTM-USERDATA PIC X(128).

```

## MQTM의 PL/I 선언

```

dcl
1 MQTM based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 QName char(48), /* Name of triggered queue */
3 ProcessName char(48), /* Name of process object */
3 TriggerData char(64), /* Trigger data */
3 ApplType fixed bin(31), /* Application type */
3 ApplId char(256), /* Application identifier */
3 EnvData char(128), /* Environment data */
3 UserData char(128); /* User data */

```

## MQTM의 상위 레벨 어셈블러 선언

```

MQTM          DSECT
MQTM_STRUCID  DS CL4   Structure identifier
MQTM_VERSION  DS F     Structure version number
MQTM_QNAME    DS CL48  Name of triggered queue
MQTM_PROCESSNAME DS CL48 Name of process object
MQTM_TRIGGERDATA DS CL64 Trigger data
MQTM_APPLTYPE DS F     Application type
MQTM_APPLID   DS CL256 Application identifier
MQTM_ENVDATA  DS CL128 Environment data
MQTM_USERDATA DS CL128 User data
*
MQTM_LENGTH   EQU *-MQTM
              ORG MQTM
MQTM_AREA     DS CL(MQTM_LENGTH)

```

## MQTM의 Visual Basic 선언

```

Type MQTM
  StrucId As String*4 'Structure identifier'
  Version As Long 'Structure version number'
  QName As String*48 'Name of triggered queue'
  ProcessName As String*48 'Name of process object'
  TriggerData As String*64 'Trigger data'
  ApplType As Long 'Application type'
  ApplId As String*256 'Application identifier'
  EnvData As String*128 'Environment data'
  UserData As String*128 'User data'
End Type

```

## 트리거 메시지에 대한 MQMD

표 534. 큐 관리자가 생성한 트리거 메시지의 MQMD에 있는 필드에 대한 설정

MQMD의 필드	사용된 값
<i>StrucId</i>	MQMD_STRUC_ID
<i>Version</i>	MQMD_VERSION_1
<i>Report</i>	MQRO_NONE
<i>MsgType</i>	MQMT_DATAGRAM
<i>Expiry</i>	MQEI_UNLIMITED
<i>Feedback</i>	MQFB_NONE

표 534. 큐 관리자가 생성한 트리거 메시지의 MQMD에 있는 필드에 대한 설정 (계속)

MQMD의 필드	사용된 값
<i>Encoding</i>	MQENC_NATIVE
<i>CodedCharSetId</i>	큐 관리자의 <b>CodedCharSetId</b> 속성
<i>Format</i>	MQFMT_TRIGGER
<i>Priority</i>	이니시에이션 큐의 <b>DefPriority</b> 속성
<i>Persistence</i>	MQPER_NOT_PERSISTENT
<i>MsgId</i>	고유 값
<i>CorrelId</i>	MQCI_NONE
<i>BackoutCount</i>	0
<i>ReplyToQ</i>	공백
<i>ReplyToQMgr</i>	큐 관리자의 이름
<i>UserIdentifier</i>	공백
<i>AccountingToken</i>	MQACT_NONE
<i>ApplIdentityData</i>	공백
<i>PutApplType</i>	MQAT_QMGR 또는 메시지 채널 에이전트에 대해 적절하게 사용됨
<i>PutApplName</i>	큐 관리자 이름의 첫 번째 28바이트
<i>PutDate</i>	트리거 메시지를 송신한 날짜
<i>PutTime</i>	트리거 메시지를 송신한 시간
<i>ApplOriginData</i>	공백

트리거 메시지를 생성하는 애플리케이션은 다음을 제외하고 유사한 값을 설정하는 것이 좋습니다.

- *Priority* 필드를 MQPRI\_PRIORITY\_AS\_Q\_DEF로 설정할 수 있습니다(메시지를 넣을 때 큐 관리자가 이를 이니시에이션 큐의 기본 우선순위로 변경함).
- *ReplyToQMgr* 필드를 공백으로 설정할 수 있습니다(메시지를 넣을 때 큐 관리자가 이를 로컬 큐 관리자의 이름으로 변경함).
- 컨텍스트 필드를 애플리케이션에 적절하게 설정하십시오.

### **MQTM의 경우 StrucId (MQCHAR4)**

이는 트리거 메시지 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQTM\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

#### **MQTM\_STRUC\_ID**

트리거 메시지 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQTM\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQTM\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### **MQTM의 버전 (MQLONG)**

구조 버전 번호입니다. 값은 다음과 같아야 합니다.

#### **MQTM\_VERSION\_1**

트리거 메시지 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### **MQTM\_CURRENT\_VERSION**

트리거 메시지 구조의 현재 버전.

이 필드의 초기값은 MQTM\_VERSION\_1입니다.

### **MQTM의 QName (MQCHAR48)**

이는 트리거 이벤트가 발생한 큐의 이름이며, 트리거 모니터 애플리케이션에 의해 시작된 애플리케이션에서 사용됩니다. 큐 관리자는 이 필드를 트리거된 큐의 **QName** 속성 값으로 초기화합니다. 이 속성의 세부사항은 [804 페이지의 『큐의 속성』](#)의 내용을 참조하십시오.

정의된 필드 길이보다 짧은 이름은 오른쪽에 공백으로 채워집니다. 이름은 널 문자로 중간에 끝나지 않습니다.

이 필드의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백입니다.

### **MQTM의 경우 ProcessName (MQCHAR48)**

이는 트리거된 큐에 지정된 큐 관리자 프로세스 오브젝트의 이름이며 트리거 메시지를 수신하는 트리거 모니터 애플리케이션에 의해 사용될 수 있습니다. 큐 관리자는 **QName** 필드에 의해 식별되는 큐의 **ProcessName** 속성 값으로 이 필드를 초기화합니다. 이 속성의 세부사항은 [804 페이지의 『큐의 속성』](#)의 내용을 참조하십시오.

정의된 필드 길이보다 짧은 이름은 항상 오른쪽에 공백으로 채워집니다. 이름은 널 문자로 중간에 끝나지 않습니다.

이 필드의 길이는 MQ\_PROCESS\_NAME\_LENGTH로 지정됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백입니다.

### **MQTM의 경우 TriggerData (MQCHAR64)**

트리거 메시지를 수신하는 트리거 모니터 애플리케이션에서 사용할 자유 형식 데이터입니다. 큐 관리자는 **QName** 필드로 식별되는 큐의 **TriggerData** 속성 값을 사용하여 이 필드를 초기화합니다. 이 속성에 대한 세부사항은 [804 페이지의 『큐의 속성』](#)의 내용을 참조하십시오. 이 데이터의 콘텐츠는 큐 관리자에 중요하지 않습니다.

z/OS에서 CKTI 트랜잭션을 사용하여 시작하는 CICS 애플리케이션의 경우 이 정보는 사용되지 않습니다.

이 필드의 길이는 MQ\_TRIGGER\_DATA\_LENGTH로 지정됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 64자의 공백 문자입니다.

### **MQTM의 경우 ApplType (MQLONG)**

이는 시작될 프로그램의 네이처를 식별하고, 트리거 메시지를 수신하는 트리거 모니터 애플리케이션에서 사용됩니다. 큐 관리자는 **ProcessName** 필드로 식별되는 프로세스 오브젝트의 **ApplType** 속성 값을 사용하여 이 필드를 초기화합니다. 이 속성에 대한 세부사항은 [836 페이지의 『프로세스 정의에 대한 속성』](#)의 내용을 참조하십시오. 이 데이터의 콘텐츠는 큐 관리자에 중요하지 않습니다.

**ApplType**은 다음 표준 값 중 하나를 가질 수 있습니다. 사용자 정의된 유형을 사용할 수 있지만 MQAT\_USER\_FIRST - MQAT\_USER\_LAST 범위의 값으로 제한되어야 합니다.

#### **MQAT\_AIX**

AIX 애플리케이션(MQAT\_UNIX와 동일한 값).

#### **MQAT\_BATCH**

배치 애플리케이션입니다.

#### **MQAT\_BROKER**

브로커 애플리케이션입니다.

#### **MQAT\_CICS**

CICS 트랜잭션.

#### **MQAT\_CICS\_BRIDGE**

CICS bridge 애플리케이션입니다.

#### **MQAT\_CICS\_VSE**

CICS/VSE 트랜잭션.

**MQAT\_DOS**

PC DOS용 IBM MQ MQI client 애플리케이션입니다.

**MQAT\_IMS**

IMS 애플리케이션입니다.

**MQAT\_IMS\_BRIDGE**

IMS 브릿지 애플리케이션입니다.

**MQAT\_JAVA**

Java 애플리케이션입니다.

**MQAT\_MVS**

MVS 또는 TSO 애플리케이션(MQAT\_ZOS와 동일한 값)입니다.

**MQAT\_NOTES\_AGENT**

Lotus Notes Agent 애플리케이션입니다.

**MQAT\_OS390**

OS/390 애플리케이션(MQAT\_ZOS와 동일한 값)입니다.

**MQAT\_OS400**

IBM i 애플리케이션입니다.

**MQAT\_RRS\_BATCH**

RRS 배치 애플리케이션입니다.

**MQAT\_UNIX**

UNIX 애플리케이션입니다.

**MQAT\_UNKNOWN**

알 수 없는 유형의 애플리케이션입니다.

**MQAT\_USER**

사용자 정의된 애플리케이션 유형입니다.

**MQAT\_VOS**

Stratus VOS 애플리케이션입니다.

**MQAT\_WINDOWS**

16비트 Windows 애플리케이션입니다.

**MQAT\_WINDOWS\_NT**

32비트 Windows 애플리케이션입니다.

**MQAT\_WLM**

z/OS 워크로드 관리자 애플리케이션입니다.

**MQAT\_XCF**

XCF.

**MQAT\_ZOS**

z/OS 애플리케이션입니다.

**MQAT\_USER\_FIRST**

사용자 정의된 애플리케이션 유형의 최저값입니다.

**MQAT\_USER\_LAST**

사용자 정의된 애플리케이션 유형의 최고값입니다.

이 필드의 초기값은 0입니다.

**MQTM의 경우 ApplId (MQCHAR256)**

이는 시작될 애플리케이션을 식별하는 문자열이며, 트리거 메시지를 수신하는 트리거 모니터 애플리케이션에서 사용됩니다. 큐 관리자는 *ProcessName* 필드에서 식별한 프로세스 오브젝트의 **AppId** 속성 값이 있는 이 필드를 초기화합니다. 이 속성의 세부사항은 836 페이지의 『프로세스 정의에 대한 속성』의 내용을 참조하십시오. 이 데이터의 콘텐츠는 큐 관리자에 중요하지 않습니다.

*AppId*의 의미는 트리거 모니터 애플리케이션에 의해 판별됩니다. IBM MQ에서 제공되는 트리거 모니터는 *AppId*를 실행 가능 프로그램의 이름으로 설정해야 합니다. 다음 참고사항은 표시된 환경에서 적용됩니다.

- z/OS, *ApplId*는 다음과 같습니다.
  - CICS 트랜잭션 ID, CICS 트리거 모니터 트랜잭션 CKTI를 사용하여 시작된 애플리케이션의 경우
  - IMS 트랜잭션 ID, IMS 트리거 모니터 CSQQTRMN을 사용하여 시작된 애플리케이션의 경우
- Windows 시스템에서 프로그램 이름에 드라이브 및 디렉토리 경로를 접두부로 지정할 수 있습니다.
- IBM i에서 프로그램 이름에 라이브러리 이름 및 / 문자를 접두부로 지정할 수 있습니다.
- AIX and Linux에서 프로그램 이름에 디렉토리 경로를 접두부로 지정할 수 있습니다.

이 필드의 길이는 MQ\_PROCESS\_APPL\_ID\_LENGTH로 지정됩니다. 이 필드의 초기값은 C에서는 널 문자열이고 기타 프로그래밍 언어에서는 256자의 공백 문자입니다.

### **MQTM의 경우 EnvData (MQCHAR128)**

이는 시작될 애플리케이션과 관계가 있는 환경 관련 정보를 포함하는 문자열이며, 트리거 메시지를 수신하는 트리거 모니터 애플리케이션에서 사용합니다. 큐 관리자는 *ProcessName* 필드로 식별되는 프로세스 오브젝트의 **EnvData** 속성 값을 사용하여 이 필드를 초기화합니다. 이 속성에 대한 세부사항은 836 페이지의 『프로세스 정의에 대한 속성』의 내용을 참조하십시오. 이 데이터의 콘텐츠는 큐 관리자에 중요하지 않습니다.

z/OS에서 CKTI 트랜잭션을 사용하여 시작하는 CICS 애플리케이션 또는 CSQQTRMN 트랜잭션을 사용하여 시작하는 IMS 애플리케이션의 경우 이 정보는 사용되지 않습니다.

이 필드의 길이는 MQ\_PROCESS\_ENV\_DATA\_LENGTH로 지정됩니다. 이 필드의 초기값은 C로 작성된 널 문자열이며, 다른 프로그래밍 언어의 경우 128개 공백 문자입니다.

### **MQTM의 경우 UserData (MQCHAR128)**

이는 시작될 애플리케이션과 관계가 있는 사용자 정보를 포함하는 문자열이며, 트리거 메시지를 수신하는 트리거 모니터 애플리케이션에서 사용합니다. 큐 관리자는 *ProcessName* 필드로 식별되는 프로세스 오브젝트의 **UserData** 속성 값을 사용하여 이 필드를 초기화합니다. 이 속성에 대한 세부사항은 836 페이지의 『프로세스 정의에 대한 속성』의 내용을 참조하십시오. 이 데이터의 콘텐츠는 큐 관리자에 중요하지 않습니다.

Microsoft Windows에 대해 프로세스 정의가 *runmqtrm*에 전달되는 경우 문자 문자열은 큰따옴표 표시를 포함할 수 없습니다.

이 필드의 길이는 MQ\_PROCESS\_USER\_DATA\_LENGTH로 지정됩니다. 이 필드의 초기값은 C로 작성된 널 문자열이며, 다른 프로그래밍 언어의 경우 128개 공백 문자입니다.

## **MQTMC2 - 트리거 메시지 2(문자 형식)**

트리거 모니터 애플리케이션이 이니시에이션 큐에서 트리거 메시지 (MQTM) 를 검색할 때, 트리거 모니터는 트리거 모니터가 시작하는 애플리케이션에 트리거 메시지의 일부 또는 모든 정보를 전달해야 할 수 있습니다.

시작된 애플리케이션이 필요로 할 수 있는 정보는 *QName*, *TriggerData*, *UserData*를 포함합니다. 환경에서 허용되고 시작된 애플리케이션에 편리한 항목에 따라, 트리거 모니터 애플리케이션은 MQTM 구조를 시작된 애플리케이션에 직접 전달하거나 MQTMC2 구조를 대신 전달할 수 있습니다.


이 구조는 TMI(IBM MQ Trigger Monitor Interface)의 일부이며 IBM MQ 프레임워크 인터페이스 중 하나입니다.

### **문자 세트 및 인코딩**

MQTMC2의 문자 데이터는 로컬 큐 관리자의 문자 세트에 있습니다. 이는 **CodedCharSetId** 큐 관리자 속성에 의해 제공됩니다.

### **사용법**

MQTMC2 구조는 MQTM 구조의 형식과 매우 유사합니다. 차이점은 MQTM의 비문자 필드가 MQTMC2의 동일한 길이의 문자 필드로 변경되었으며 큐 관리자 이름이 구조 끝에 추가되었습니다.

-  z/OS에서 CSQQTRMN 애플리케이션을 사용하여 시작되는 MQAT\_IMS 애플리케이션의 경우 MQTMC2 구조를 시작된 애플리케이션에서 사용할 수 있도록 작성되었습니다.

- **IBM i** IBM i에서 IBM MQ가 있는 트리거 모니터 애플리케이션은 MQTMC2 구조를 시작된 애플리케이션에 전달합니다.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 535. MQTMC2의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
StrucId (구조 ID)	MQTMC_STRUC_ID	'TMC~'
버전 (구조 버전 번호)	MQTMC_VERSION_2	'~~~2'
QName (트리거된 큐의 이름)	없음	널 문자열 또는 공백
ProcessName (프로세스 오브젝트의 이름)	없음	널 문자열 또는 공백
TriggerData (트리거 데이터)	없음	널 문자열 또는 공백
ApplType (애플리케이션 유형)	없음	공백
ApplId (애플리케이션 ID)	없음	널 문자열 또는 공백
EnvData (환경 데이터)	없음	널 문자열 또는 공백
UserData (사용자 데이터)	없음	널 문자열 또는 공백
QMgrName (큐 관리자 이름)	없음	널 문자열 또는 공백

**참고:**

- ~ 기호는 단일 공백 문자를 나타냅니다.
- 널 문자열 값 또는 공백은 C의 널 문자열과 기타 프로그래밍 언어의 공백 문자를 나타냅니다.
- C 프로그래밍 언어의 매크로 변수MQTMC2\_DEFAULT는 위에 나열된 값을 포함합니다. 구조의 필드에 초기 값을 제공하기 위해 다음 방법으로 사용하십시오.

```
MQTMC2 MyTMC = {MQTMC2_DEFAULT};
```

## 언어 선언

### MQTMC2의 C 선언

```
typedef struct tagMQTMC2 MQTMC2;
struct tagMQTMC2 {
    MQCHAR4    StrucId;        /* Structure identifier */
    MQCHAR4    Version;       /* Structure version number */
    MQCHAR48   QName;         /* Name of triggered queue */
    MQCHAR48   ProcessName;   /* Name of process object */
    MQCHAR64   TriggerData;   /* Trigger data */
    MQCHAR4    ApplType;      /* Application type */
    MQCHAR256  ApplId;        /* Application identifier */
    MQCHAR128  EnvData;       /* Environment data */
    MQCHAR128  UserData;      /* User data */
    MQCHAR48   QMgrName;     /* Queue manager name */
};
```

### MQTMC2의 COBOL 선언

```
** MQTMC2 structure
   10 MQTMC2.
**   Structure identifier
   15 MQTMC2-STRUCID    PIC X(4).
```

```

**      Structure version number
15 MQTMC2-VERSION PIC X(4).
**      Name of triggered queue
15 MQTMC2-QNAME PIC X(48).
**      Name of process object
15 MQTMC2-PROCESSNAME PIC X(48).
**      Trigger data
15 MQTMC2-TRIGGERDATA PIC X(64).
**      Application type
15 MQTMC2-APPLTYPE PIC X(4).
**      Application identifier
15 MQTMC2-APPLID PIC X(256).
**      Environment data
15 MQTMC2-ENVDATA PIC X(128).
**      User data
15 MQTMC2-USERDATA PIC X(128).
**      Queue manager name
15 MQTMC2-QMGRNAME PIC X(48).

```

## MQTMC2의 PL/I 선언

```

dcl
  1 MQTMC2 based,
  3 StrucId char(4), /* Structure identifier */
  3 Version char(4), /* Structure version number */
  3 QName char(48), /* Name of triggered queue */
  3 ProcessName char(48), /* Name of process object */
  3 TriggerData char(64), /* Trigger data */
  3 ApplType char(4), /* Application type */
  3 ApplId char(256), /* Application identifier */
  3 EnvData char(128), /* Environment data */
  3 UserData char(128), /* User data */
  3 QMgrName char(48); /* Queue manager name */

```

## MQTMC2의 상위 레벨 어셈블러 선언

```

MQTMC2          DSECT
MQTMC2_STRUCID DS CL4      Structure identifier
MQTMC2_VERSION DS CL4      Structure version number
MQTMC2_QNAME    DS CL48     Name of triggered queue
MQTMC2_PROCESSNAME DS CL48  Name of process object
MQTMC2_TRIGGERDATA DS CL64  Trigger data
MQTMC2_APPLTYPE DS CL4      Application type
MQTMC2_APPLID  DS CL256    Application identifier
MQTMC2_ENVDATA DS CL128    Environment data
MQTMC2_USERDATA DS CL128   User data
MQTMC2_QMGRNAME DS CL48    Queue manager name
*
MQTMC2_LENGTH EQU *-MQTMC2
                ORG MQTMC2
MQTMC2_AREA DS CL(MQTMC2_LENGTH)

```

## MQTMC2의 Visual Basic 선언

```

Type MQTMC2
  StrucId As String*4 'Structure identifier'
  Version As String*4 'Structure version number'
  QName As String*48 'Name of triggered queue'
  ProcessName As String*48 'Name of process object'
  TriggerData As String*64 'Trigger data'
  ApplType As String*4 'Application type'
  ApplId As String*256 'Application identifier'
  EnvData As String*128 'Environment data'
  UserData As String*128 'User data'
  QMgrName As String*48 'Queue manager name'
End Type

```

## MQTMC2의 경우 StrucId (MQCHAR4)

이는 트리거 메시지 2 (문자 형식) 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQTMC2\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

## **MQTMC2\_STRUC\_ID**

트리거 메시지 (문자 형식) 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQTMC2\_STRUC\_ID\_ARRAY 도 정의됩니다. 이는 MQTMC2\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

## **MQTMC2 용 버전 (MQCHAR4)**

구조 버전 번호.

값은 다음과 같아야 합니다.

## **MQTMC\_VERSION\_2**

버전 2 트리거 메시지(문자 형식) 구조입니다.

C 프로그래밍 언어의 경우 상수 MQTMC\_VERSION\_2\_ARRAY도 정의됩니다. 이는 MQTMC\_VERSION\_2와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

## **MQTMC\_CURRENT\_VERSION**

트리거 메시지(문자 형식화) 구조의 현재 버전.

## **MQTMC2 용 QName (MQCHAR48)**

트리거된 큐의 이름.

MQTM 구조의 *QName* 필드를 참조하십시오.

## **MQTMC2 의 경우 ProcessName (MQCHAR48)**

프로세스 오브젝트의 이름입니다.

MQTM 구조의 *ProcessName* 필드를 참조하십시오.

## **MQTMC2 의 경우 TriggerData (MQCHAR64)**

트리거 데이터.

MQTM 구조의 *TriggerData* 필드를 참조하십시오.

## **MQTMC2 의 경우 ApplType (MQCHAR4)**

애플리케이션 유형.

원래 트리거 메시지의 MQTM 구조에 있는 *ApplType* 필드의 값에 상관 없이 이 필드에는 항상 공백이 포함됩니다.

## **MQTMC2 의 경우 ApplId (MQCHAR256)**

애플리케이션 ID입니다.

MQTM 구조의 *ApplId* 필드를 참조하십시오.

## **MQTMC2 의 경우 EnvData (MQCHAR128)**

환경 데이터.

MQTM 구조의 *EnvData* 필드를 참조하십시오.

## **MQTMC2 의 경우 UserData (MQCHAR128)**

사용자 데이터.

MQTM 구조의 *UserData* 필드를 참조하십시오.



## MQTMC2의 경우 QMgrName (MQCHAR48)

큐 관리자 이름.

트리거 이벤트가 발생한 큐 관리자의 이름입니다.

## MQWIH - 작업 정보 헤더

메시지가 z/OS 워크로드 관리자 (WLM)에 의해 처리되는 경우 메시지는 MQWIH 구조로 시작해야 합니다. 이 구조는 WLM이 처리할 메시지의 시작 부분에 있어야 하는 정보를 설명합니다.

## 가용성

모든 IBM MQ 시스템 및 이러한 시스템에 연결된 IBM MQ 클라이언트.

## 형식 이름

MQFMT\_WORK\_INFO\_HEADER

## 문자 세트 및 인코딩

MQWIH 구조의 필드는 MQWIH앞에 오는 헤더 구조의 *CodedCharSetId* 및 *Encoding* 필드 또는 MQWIH가 애플리케이션 메시지 데이터의 시작에 있는 경우 MQMD 구조의 해당 필드에 의해 제공되는 문자 세트 및 인코딩에 있습니다.

문자 세트는 큐 이름에 유효한 문자에 사용되는 단일 바이트 문자가 있어야 합니다.

## 사용법

IBM MQ 지원 플랫폼의 경우 MQWIH 구조를 포함하는 메시지를 작성하고 전송할 수 있지만 IBM MQ for z/OS 큐 관리자만 WLM과 상호작용할 수 있습니다. 그러므로 biz/OS 큐 관리자에서 WLM에 메시지를 가져오는 경우 큐 관리자 네트워크가 메시지가 라우트할 수 있는 하나 이상의 z/OS 큐 관리자를 포함해야 합니다.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 536. MQWIH의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
<u>StrucId</u> (구조 ID)	MQWIH_STRUC_ID	'WIH~'
<u>버전</u> (구조 버전 번호)	MQWIH_VERSION_1	1
<u>StrucLength</u> (MQWIH 구조의 길이)	MQWIH_LENGTH_1	120
<u>인코딩</u> (MQWIH 다음에 오는 데이터의 숫자 인코딩)	없음	0
<u>CodedCharSetId</u> (MQWIH 다음에 오는 데이터의 문자 세트 ID)	MQCCSI_UNDEFINED	0
<u>형식</u> (MQWIH 다음에 오는 데이터의 형식 이름)	MQFMT_NONE	공백
<u>플래그</u> (플래그)	MQWIH_NONE	0
<u>ServiceName</u> (서비스 이름)	없음	공백
<u>ServiceStep</u> (서비스 단계 이름)	없음	공백
<u>MsgToken</u> (메시지 토큰)	MQMTOK_NONE	널
<u>예약됨</u> (예약됨)	없음	공백

표 536. MQWIH의 필드 (계속)

필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
<p><b>참고:</b></p> <ol style="list-style-type: none"> <li>~ 기호는 단일 공백 문자를 나타냅니다.</li> <li>C 프로그래밍 언어의 매크로 변수MQWIH_DEFAULT는 테이블에 나열되는 값을 포함합니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오.</li> </ol> <pre>MQWIH MyWIH = {MQWIH_DEFAULT};</pre>		

## 언어 선언

### MQWIH의 C 선언

```
typedef struct tagMQWIH MQWIH;
struct tagMQWIH {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    StrucLength;      /* Length of MQWIH structure */
    MQLONG    Encoding;        /* Numeric encoding of data that follows
                               MQWIH */
    MQLONG    CodedCharSetId;   /* Character-set identifier of data that
                               follows MQWIH */
    MQCHAR8   Format;          /* Format name of data that follows
                               MQWIH */
    MQLONG    Flags;           /* Flags */
    MQCHAR32  ServiceName;     /* Service name */
    MQCHAR8   ServiceStep;     /* Service step name */
    MQBYTE16  MsgToken;        /* Message token */
    MQCHAR32  Reserved;        /* Reserved */
};
```

### MQWIH의 COBOL 선언

```
** MQWIH structure
10 MQWIH.
** Structure identifier
15 MQWIH-STRUCID PIC X(4).
** Structure version number
15 MQWIH-VERSION PIC S9(9) BINARY.
** Length of MQWIH structure
15 MQWIH-STRUCLNGTH PIC S9(9) BINARY.
** Numeric encoding of data that follows MQWIH
15 MQWIH-ENCODING PIC S9(9) BINARY.
** Character-set identifier of data that follows MQWIH
15 MQWIH-CODEDCHARSETID PIC S9(9) BINARY.
** Format name of data that follows MQWIH
15 MQWIH-FORMAT PIC X(8).
** Flags
15 MQWIH-FLAGS PIC S9(9) BINARY.
** Service name
15 MQWIH-SERVICENAME PIC X(32).
** Service step name
15 MQWIH-SERVICESTEP PIC X(8).
** Message token
15 MQWIH-MSGTOKEN PIC X(16).
** Reserved
15 MQWIH-RESERVED PIC X(32).
```

### MQWIH의 PL/I 선언

```
dcl
1 MQWIH based,
3 StrucId char(4), /* Structure identifier */
3 Version fixed bin(31), /* Structure version number */
3 StrucLength fixed bin(31), /* Length of MQWIH structure */
```

```

3 Encoding      fixed bin(31), /* Numeric encoding of data that
                  follows MQWIH */
3 CodedCharSetId fixed bin(31), /* Character-set identifier of data
                  that follows MQWIH */
3 Format         char(8), /* Format name of data that follows
                  MQWIH */
3 Flags         fixed bin(31), /* Flags */
3 ServiceName   char(32), /* Service name */
3 ServiceStep   char(8), /* Service step name */
3 MsgToken      char(16), /* Message token */
3 Reserved      char(32); /* Reserved */

```

### MQWIH의 상위 레벨 어셈블러 선언

```

MQWIH          DSECT
MQWIH_STRUCID  DS    CL4  Structure identifier
MQWIH_VERSION  DS    F    Structure version number
MQWIH_STRUCLNGTH DS  F    Length of MQWIH structure
MQWIH_ENCODING DS  F    Numeric encoding of data that follows
*              MQWIH
MQWIH_CODEDCHARSETID DS  F  Character-set identifier of data that
*              follows MQWIH
MQWIH_FORMAT   DS    CL8  Format name of data that follows MQWIH
MQWIH_FLAGS    DS    F    Flags
MQWIH_SERVICENAME DS  CL32 Service name
MQWIH_SERVICESTEP DS  CL8  Service step name
MQWIH_MSGTOKEN DS  XL16  Message token
MQWIH_RESERVED DS  CL32  Reserved
*
MQWIH_LENGTH   EQU  *-MQWIH
               ORG  MQWIH
MQWIH_AREA     DS    CL(MQWIH_LENGTH)

```

### MQWIH의 Visual Basic 선언

```

Type MQWIH
  StrucId      As String*4 'Structure identifier'
  Version      As Long     'Structure version number'
  Struclength  As Long     'Length of MQWIH structure'
  Encoding     As Long     'Numeric encoding of data that follows'
                  'MQWIH'
  CodedCharSetId As Long   'Character-set identifier of data that'
                  'follows MQWIH'
  Format       As String*8  'Format name of data that follows MQWIH'
  Flags       As Long      'Flags'
  ServiceName As String*32  'Service name'
  ServiceStep As String*8   'Service step name'
  MsgToken    As MQBYTE16  'Message token'
  Reserved    As String*32  'Reserved'
End Type

```

### **MQWIH의 StrucId (MQCHAR4)**

작업 정보 헤더 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQWIH\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

#### **MQWIH\_STRUC\_ID**

작업 정보 헤더 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQWIH\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQWIH\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### **MQWIH의 버전 (MQLONG)**

구조 버전 번호입니다. 값은 다음과 같아야 합니다.

#### **MQWIH\_VERSION\_1**

버전-1 작업 정보 헤더 구조 ID.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

### **MQWIH\_CURRENT\_VERSION**

작업 정보 헤더 구조의 현재 버전.

이 필드의 초기값은 MQWIH\_VERSION\_1입니다.

### **MQWIH의 StrucLength (MQLONG)**

이는 MQWIH 구조의 길이입니다. 값은 다음과 같아야 합니다.

### **MQWIH\_LENGTH\_1**

버전-1 작업 정보 헤더 구조의 길이.

다음 상수는 현재 버전의 길이를 지정합니다.

### **MQWIH\_CURRENT\_LENGTH**

작업 정보 헤더 구조의 현재 버전 길이입니다.

이 필드의 초기값은 MQWIH\_LENGTH\_1입니다.

### **MQWIH의 인코딩 (MQLONG)**

MQWIH 구조 뒤에 오는 데이터의 숫자 인코딩을 지정합니다. MQWIH 구조 자체의 숫자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다.

이 필드의 초기값은 0입니다.

### **MQWIH의 CodedCharSetId (MQLONG)**

MQWIH 구조 뒤에 오는 데이터의 문자 세트 ID를 지정합니다. MQWIH 구조 자체의 문자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 다음 특수 값을 사용할 수 있습니다.

### **MQCCSI\_INHERIT**

이 구조 다음에 오는 데이터의 문자 데이터는 이 구조와 동일한 문자 세트로 되어 있습니다.

큐 관리자가 구조의 실제 문자 세트 ID에 메시지로 송신한 구조에서 이 값을 변경합니다. 오류가 발생하지 않으면 MQGET 호출에서 MQCCSI\_INHERIT 값을 리턴하지 않습니다.

MQMD의 *PutApplType* 필드 값이 MQAT\_BROKER인 경우 MQCCSI\_INHERIT를 사용할 수 없습니다.

이 필드의 초기값은 MQCCSI\_UNDEFINED입니다.

### **MQWIH의 형식 (MQCHAR8)**

MQWIH 구조 뒤에 오는 데이터의 형식 이름을 지정합니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 이 필드의 코딩 규칙은 MQMD의 *Format* 필드에 대한 규칙과 동일합니다.

이 필드의 길이는 MQ\_FORMAT\_LENGTH에 지정되어 있습니다. 이 필드의 초기값은 MQFMT\_NONE입니다.

### **MQWIH의 플래그 (MQLONG)**

값은 다음과 같아야 합니다.

### **MQWIH\_NONE**

플래그가 없습니다.

이 필드의 초기값은 MQWIH\_NONE입니다.

### **MQWIH의 경우 ServiceName (MQCHAR32)**

메시지를 처리하는 서비스의 이름입니다.

이 필드의 길이는 MQ\_SERVICE\_NAME\_LENGTH로 지정됩니다. 이 필드의 초기값은 32개의 공백 문자입니다.

## MQWIH의 경우 ServiceStep (MQCHAR8)

이것은 메시지와 관련된 ServiceName 단계의 이름입니다.

이 필드의 길이는 MQ\_SERVICE\_STEP\_LENGTH로 지정됩니다. 이 필드의 초기값은 8개의 빈 문자입니다.

## MQWIH의 MsgToken (MQBYTE16)

고유하게 메시지를 식별하는 메시지 토큰입니다.

MQPUT 및 MQPUT1 호출의 경우 이 필드는 무시됩니다. 이 필드의 길이는 MQ\_MSG\_TOKEN\_LENGTH로 지정됩니다. 이 필드의 초기값은 MQMTOK\_NONE입니다.

## MQWIH용으로 예약됨 (MQCHAR32)

이는 예약된 필드이며 비어 있어야 합니다.

## MQXP - 엑시트 매개변수 블록

MQXP 구조는 API 교차 엑시트에 대한 입출력 (I/O) 매개변수로 사용됩니다. 이 엑시트에 대한 자세한 정보는 [API 교차 엑시트를 참조하십시오](#).

## 문자 세트 및 인코딩

MQXP의 문자 데이터는 로컬 큐 관리자의 문자 세트에 있습니다. 이는 **CodedCharSetId** 큐 관리자 속성에 의해 지정됩니다. MQXP 내의 숫자 데이터는 고유 시스템 인코딩입니다. 이는 MQENC\_NATIVE에서 지정합니다.

## 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 537. MQXP의 필드	
필드 이름 및 설명	상수의 이름
StrucId (구조 ID)	MQXP_STRUC_ID
버전 (구조 버전 번호)	MQXP_VERSION_1
ExitId (엑시트 ID)	MQXT_API_CROSSING_EXIT
ExitReason (엑시트 호출 이유)	없음
ExitResponse (엑시트로부터의 응답)	없음
ExitCommand (API 호출 코드)	없음
ExitParmCount (매개변수 개수)	없음
예약됨 (예약됨)	없음
ExitUser영역 (사용자 영역)	없음

## 언어 선언

MQXP의 C 선언

```
typedef struct tagMQXP MQXP;
struct tagMQXP {
    MQCHAR4    StructId;           /* Structure identifier */
    MQLONG     Version;           /* Structure version number */
    MQLONG     ExitId;            /* Exit identifier */
    MQLONG     ExitReason;        /* Reason for invocation of exit */
    MQLONG     ExitResponse;      /* Response from exit */
    MQLONG     ExitCommand;       /* API call code */
    MQLONG     ExitParmCount;     /* Parameter count */
};
```

```

MQLONG    Reserved;        /* Reserved */
MQBYTE16  ExitUserArea;    /* User area */
};

```

## MQXP의 COBOL 선언

```

** MQXP structure
10 MQXP.
** Structure identifier
15 MQXP-STRUCID PIC X(4).
** Structure version number
15 MQXP-VERSION PIC S9(9) BINARY.
** Exit identifier
15 MQXP-EXITID PIC S9(9) BINARY.
** Reason for invocation of exit
15 MQXP-EXITREASON PIC S9(9) BINARY.
** Response from exit
15 MQXP-EXITRESPONSE PIC S9(9) BINARY.
** API call code
15 MQXP-EXITCOMMAND PIC S9(9) BINARY.
** Parameter count
15 MQXP-EXITPARMCOUNT PIC S9(9) BINARY.
** Reserved
15 MQXP-RESERVED PIC S9(9) BINARY.
** User area
15 MQXP-EXITUSERAREA PIC X(16).

```

## MQXP의 PL/I 선언

```

dcl
1 MQXP based,
3 StrucId      char(4),          /* Structure identifier */
3 Version      fixed bin(31),    /* Structure version number */
3 ExitId       fixed bin(31),    /* Exit identifier */
3 ExitReason   fixed bin(31),    /* Reason for invocation of exit */
3 ExitResponse fixed bin(31),    /* Response from exit */
3 ExitCommand  fixed bin(31),    /* API call code */
3 ExitParmCount fixed bin(31),  /* Parameter count */
3 Reserved     fixed bin(31),    /* Reserved */
3 ExitUserArea char(16);        /* User area */

```

## MQXP의 상위 레벨 어셈블러 선언

```

MQXP          DSECT
MQXP_STRUCID  DS CL4 Structure identifier
MQXP_VERSION  DS F   Structure version number
MQXP_EXITID   DS F   Exit identifier
MQXP_EXITREASON DS F Reason for invocation of exit
MQXP_EXITRESPONSE DS F Response from exit
MQXP_EXITCOMMAND DS F API call code
MQXP_EXITPARMCOUNT DS F Parameter count
MQXP_RESERVED DS F Reserved
MQXP_EXITUSERAREA DS XL16 User area
*
MQXP_LENGTH   EQU *-MQXP
               ORG MQXP
MQXP_AREA     DS CL(MQXP_LENGTH)

```

### MQXP의 경우 StrucId (MQCHAR4)

엑시트 매개변수 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQXP\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

#### MQXP\_STRUC\_ID

엑시트 매개변수 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQXP\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQXP\_STRUC\_ID와 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

### MQXP용 버전 (MQLONG)

구조 버전 번호입니다. 값은 다음과 같아야 합니다.

#### **MQXP\_VERSION\_1**

엑시트 매개변수 블록 구조의 버전 번호입니다.

**참고:** 이 구조의 새 버전을 도입해도 기존 부분의 레이아웃은 변경되지 않습니다. 따라서 엑시트는 버전 번호가 엑시트가 사용해야 하는 필드를 포함하는 최저 버전 이상인지 확인해야 합니다.

엑시트의 입력 필드입니다.

#### **MQXP의 경우 ExitId (MQLONG)**

이는 입력 항목에서 엑시트 루틴으로 설정되고 엑시트의 유형을 표시합니다.

#### **MQXT\_API\_CROSSING\_EXIT**

CICS의 API 교차 엑시트입니다.

엑시트의 입력 필드입니다.

#### **MQXP의 경우 ExitReason (MQLONG)**

이는 엑시트 루틴에 대한 입력 항목에서 설정됩니다. API 교차 엑시트의 경우 이는 API 호출의 실행 전 또는 후에 루틴이 호출되는지 여부를 표시합니다.

#### **MQXR\_BEFORE**

API 실행 전입니다.

#### **MQXR\_AFTER**

API 실행 후입니다.

엑시트의 입력 필드입니다.

#### **MQXP의 경우 ExitResponse (MQLONG)**

호출자와 통신하기 위해 엑시트가 값을 설정합니다. 다음 값이 정의됩니다.

#### **MQXCC\_OK**

엑시트가 성공적으로 완료되었습니다.

#### **MQXCC\_SUPPRESS\_FUNCTION**

함수 차단.

API 호출 전에 호출된 API 교차 엑시트에서 이 값을 설정하는 경우, API 호출이 수행되지 않습니다. 호출의 *CompCode*가 MQCC\_FAILED로 설정되고 *Reason*이 MQRC\_SUPPRESSED\_BY\_EXIT로 설정되고 엑시트 종료 시 다른 모든 매개변수는 남아 있습니다.

API 호출 후에 호출된 API 교차 엑시트에서 이 값을 설정하는 경우 큐 관리자가 이를 무시합니다.

#### **MQXCC\_SKIP\_FUNCTION**

함수를 건너뜁니다.

API 호출 이전에 호출된 API 교차 엑시트에 의해 이 값이 설정된 경우 API 호출이 수행되지 않습니다. 엑시트 종료 시 *CompCode* 및 *Reason* 및 기타 모든 매개변수는 남아 있습니다.

API 호출 후에 호출된 API 교차 엑시트에서 이 값을 설정하는 경우 큐 관리자가 이를 무시합니다.

이는 엑시트의 출력 필드입니다.

#### **MQXP의 경우 ExitCommand (MQLONG)**

이 필드는 엑시트 루틴에 대한 입력 항목에서 설정됩니다. 이는 엑시트가 호출되도록 한 API 호출을 식별합니다.

#### **MQXC\_CALLBACK**

CALLBACK 호출.

#### **MQXC\_MQBACK**

MQBACK 호출.

#### **MQXC\_MQCB**

MQCB 호출.

**MQXC\_MQCLOSE**  
MQCLOSE 호출.

**MQXC\_MQCMIT**  
MQCMIT 호출.

**MQXC\_MQCTL**  
MQCTL 호출.

**MQXC\_MQGET**  
MQGET 호출.

**MQXC\_MQINQ**  
MQINQ 호출.

**MQXC\_MQOPEN**  
MQOPEN 호출.

**MQXC\_MQPUT**  
MQPUT 호출.

**MQXC\_MQPUT1**  
MQPUT1 호출.

**MQXC\_MQSET**  
MQSET 호출.

**MQXC\_MQSTAT**  
MQSTAT 호출.

**MQXC\_MQSUB**  
MQSUB 호출.

**MQXC\_MQSUBRQ**  
MQSUBRQ 호출.

엑시트의 입력 필드입니다.

### **MQXP의 ExitParm수 (MQLONG)**

이 필드는 엑시트 루틴에 대한 입력 항목에서 설정됩니다. 여기에는 MQ 호출이 가져오는 매개변수의 수가 포함됩니다.

표 538. 각 MQ 호출의 매개변수 수

---

호출 이름	매개변수의 수
MQBACK	3
MQCLOSE	5
MQCMIT	3
MQGET	9
MQINQ	10
MQOPEN	6
MQPUT	8
MQPUT1	8
MQSET	10

엑시트의 입력 필드입니다.

### **MQXP용으로 예약됨 (MQLONG)**

이 필드는 예약된 필드입니다. 해당 값은 엑시트에 중요하지 않습니다.



## MQXP의 ExitUser영역 (MQBYTE16)

이는 엑시트가 사용할 수 있는 필드입니다. 이는 태스크에 대한 엑시트의 첫 번째 호출 전에 필드의 길이에 대해 2진 0으로 초기화되고 그 후 엑시트에 의해 작성된 이 필드의 변경사항은 엑시트 호출 간에 보존됩니다. 다음 값이 정의됩니다.

### MQXUA\_NONE

사용자 정보가 없습니다.

이 값은 필드 길이 만큼의 2진 0입니다.

C 프로그래밍 언어의 경우 상수 MQXUA\_NONE\_ARRAY도 정의됩니다. 이는 MQXUA\_NONE과 동일한 값을 가지지만 문자열이 아닌 문자의 배열입니다.

이 필드의 길이는 MQ\_EXIT\_USER\_AREA\_LENGTH에서 제공합니다. 이는 엑시트의 입출력(I/O) 필드입니다.

## MQXQH - 전송 큐 헤더

MQXQH 구조는 전송 큐에 있을 때 메시지의 애플리케이션 메시지 데이터의 앞에 붙여지는 정보를 설명합니다. 전송 큐는 리모트 큐에 목적지 지정되는(즉, 로컬 큐 관리자에 속하지 않은 큐에 목적지 지정되는) 메시지를 임시로 보유하는 로컬 큐의 특수 유형입니다. 전송 큐는 MQUS\_TRANSMISSION 값을 가진 **Usage** 큐 속성으로 표시됩니다.

### 형식 이름

MQFMT\_XMIT\_Q\_HEADER

### 문자 세트 및 인코딩

MQXQH의 데이터는 MQENC\_NATIVE로 지정된 로컬 큐 관리자의 인코딩 및 **CodedCharSetId** 큐 관리자 속성으로 지정된 문자 세트에 있어야 합니다.

MQXQH의 문자 세트 및 인코딩을 다음을 사용하여 *CodedCharSetId* 및 *Encoding* 필드로 설정하십시오.

- 별도의 MQMD(MQXQH 구조가 메시지 데이터의 시작에 있는 경우) 또는
- MQXQH 구조에 선행하는 헤더 구조(다른 모든 경우).

### 필드

**참고:** 다음 표에서 필드는 알파벳순이 아닌 사용법별로 그룹화됩니다. 하위 주제는 동일한 순서를 따릅니다.

표 539. MQXQH에 대한 MQXQH의 필드		
필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
<u>StrucId</u> (구조 ID)	MQXQH_STRUC_ID	'XQH~'
<u>버전</u> (구조 버전 번호)	MQXQH_VERSION_1	1
<u>RemoteQName</u> (목적지 큐의 이름)	없음	널 문자열 또는 공백
<u>RemoteQMgr</u> 이름 (목적지 큐 관리자의 이름)	없음	널 문자열 또는 공백
<u>MsgDesc</u> (원래 메시지 디스크립터)	MQMD와 동일한 이름 및 값. 413 페이지의 표 500의 내용을 참조하십시오.	-

표 539. MQXQH에 대한 MQXQH의 필드 (계속)

필드 이름 및 설명	상수의 이름	상수의 초기값 (있는 경우)
<p><b>참고:</b></p> <ol style="list-style-type: none"> <li>~ 기호는 단일 공백 문자를 나타냅니다.</li> <li>널 문자열 값 또는 공백은 C의 널 문자열과 기타 프로그래밍 언어의 공백 문자를 나타냅니다.</li> <li>C 프로그래밍 언어의 매크로 변수MQXQH_DEFAULT는 테이블에 나열되는 값을 포함합니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오.</li> </ol> <pre>MQXQH MyXQH = {MQXQH_DEFAULT};</pre>		

## 언어 선언

### MQXQH의 C 선언

```
typedef struct tagMQXQH MQXQH;
struct tagMQXQH {
    MQCHAR4    StructId;           /* Structure identifier */
    MQLONG     Version;           /* Structure version number */
    MQCHAR48   RemoteQName;       /* Name of destination queue */
    MQCHAR48   RemoteQMgrName;    /* Name of destination queue manager */
    MQMD1      MsgDesc;          /* Original message descriptor */
};
```

### MQXQH의 COBOL 선언

```
** MQXQH structure
   10 MQXQH.
**   Structure identifier
   15 MQXQH-STRUCID           PIC X(4).
**   Structure version number
   15 MQXQH-VERSION         PIC S9(9) BINARY.
**   Name of destination queue
   15 MQXQH-REMOTEQNAME     PIC X(48).
**   Name of destination queue manager
   15 MQXQH-REMOTEQMGRNAME  PIC X(48).
**   Original message descriptor
   15 MQXQH-MSGDESC.
**   Structure identifier
   20 MQXQH-MSGDESC-STRUCID  PIC X(4).
**   Structure version number
   20 MQXQH-MSGDESC-VERSION  PIC S9(9) BINARY.
**   Report options
   20 MQXQH-MSGDESC-REPORT   PIC S9(9) BINARY.
**   Message type
   20 MQXQH-MSGDESC-MSGTYPE  PIC S9(9) BINARY.
**   Expiry time
   20 MQXQH-MSGDESC-EXPIRY   PIC S9(9) BINARY.
**   Feedback or reason code
   20 MQXQH-MSGDESC-FEEDBACK PIC S9(9) BINARY.
**   Numeric encoding of message data
   20 MQXQH-MSGDESC-ENCODING PIC S9(9) BINARY.
**   Character set identifier of message data
   20 MQXQH-MSGDESC-CODEDCHARSETID PIC S9(9) BINARY.
**   Format name of message data
   20 MQXQH-MSGDESC-FORMAT   PIC X(8).
**   Message priority
   20 MQXQH-MSGDESC-PRIORITY PIC S9(9) BINARY.
**   Message persistence
   20 MQXQH-MSGDESC-PERSISTENCE PIC S9(9) BINARY.
**   Message identifier
   20 MQXQH-MSGDESC-MSGID    PIC X(24).
**   Correlation identifier
   20 MQXQH-MSGDESC-CORRELID PIC X(24).
**   Backout counter
   20 MQXQH-MSGDESC-BACKOUTCOUNT PIC S9(9) BINARY.
**   Name of reply-to queue
```

```

20 MQXQH-MSGDESC-REPLYTOQ      PIC X(48).
** Name of reply queue manager
20 MQXQH-MSGDESC-REPLYTOQMGR   PIC X(48).
** User identifier
20 MQXQH-MSGDESC-USERIDENTIFIER PIC X(12).
** Accounting token
20 MQXQH-MSGDESC-ACCOUNTINGTOKEN PIC X(32).
** Application data relating to identity
20 MQXQH-MSGDESC-APPLIDENTITYDATA PIC X(32).
** Type of application that put the message
20 MQXQH-MSGDESC-PUTAPPLTYPE    PIC S9(9) BINARY.
** Name of application that put the message
20 MQXQH-MSGDESC-PUTAPPLNAME    PIC X(28).
** Date when message was put
20 MQXQH-MSGDESC-PUTDATE        PIC X(8).
** Time when message was put
20 MQXQH-MSGDESC-PUTTIME        PIC X(8).
** Application data relating to origin
20 MQXQH-MSGDESC-APPLORIGINDATA PIC X(4).

```

## MQXQH의 PL/I 선언

```

dcl
  1 MQXQH based,
  3 StrucId          char(4),          /* Structure identifier */
  3 Version          fixed bin(31),    /* Structure version number */
  3 RemoteQName      char(48),        /* Name of destination queue */
  3 RemoteQMgrName   char(48),        /* Name of destination queue
manager */
  3 MsgDesc,
  5 StrucId          char(4),          /* Original message descriptor */
  5 Version          fixed bin(31),    /* Structure version number */
  5 Report           fixed bin(31),    /* Report options */
  5 MsgType          fixed bin(31),    /* Message type */
  5 Expiry           fixed bin(31),    /* Expiry time */
  5 Feedback         fixed bin(31),    /* Feedback or reason code */
  5 Encoding         fixed bin(31),    /* Numeric encoding of message
data */
  5 CodedCharSetId   fixed bin(31),    /* Character set identifier of
message data */
  5 Format            char(8),          /* Format name of message data */
  5 Priority          fixed bin(31),    /* Message priority */
  5 Persistence      fixed bin(31),    /* Message persistence */
  5 MsgId            char(24),        /* Message identifier */
  5 CorrelId         char(24),        /* Correlation identifier */
  5 BackoutCount     fixed bin(31),    /* Backout counter */
  5 ReplyToQ         char(48),        /* Name of reply-to queue */
  5 ReplyToQMgr      char(48),        /* Name of reply queue manager */
  5 UserIdentifier    char(12),        /* User identifier */
  5 AccountingToken  char(32),        /* Accounting token */
  5 ApplIdentityData char(32),        /* Application data relating to
identity */
  5 PutApplType      fixed bin(31),    /* Type of application that put the
message */
  5 PutApplName      char(28),        /* Name of application that put the
message */
  5 PutDate          char(8),          /* Date when message was put */
  5 PutTime          char(8),          /* Time when message was put */
  5 ApplOriginData   char(4);        /* Application data relating to
origin */

```

## MQXQH의 상위 레벨 어셈블러 선언

MQXQH	DSECT		
MQXQH_STRUCID	DS	CL4	Structure identifier
MQXQH_VERSION	DS	F	Structure version number
MQXQH_REMOTENAME	DS	CL48	Name of destination queue
MQXQH_REMOTEMGRNAME	DS	CL48	Name of destination queue manager
*			
MQXQH_MSGDESC	DS	0F	Force fullword alignment
MQXQH_MSGDESC_STRUCID	DS	CL4	Structure identifier
MQXQH_MSGDESC_VERSION	DS	F	Structure version number
MQXQH_MSGDESC_REPORT	DS	F	Report options
MQXQH_MSGDESC_MSGTYPE	DS	F	Message type
MQXQH_MSGDESC_EXPIRY	DS	F	Expiry time
MQXQH_MSGDESC_FEEDBACK	DS	F	Feedback or reason code
MQXQH_MSGDESC_ENCODING	DS	F	Numeric encoding of message

* MQXQH_MSGDESC_CODEDCHARSETID	DS	F	data	Character set identifier of message data
* MQXQH_MSGDESC_FORMAT	DS	CL8	Format name of message data	Format name of message data
MQXQH_MSGDESC_PRIORITY	DS	F	Message priority	Message priority
MQXQH_MSGDESC_PERSISTENCE	DS	F	Message persistence	Message persistence
MQXQH_MSGDESC_MSGID	DS	XL24	Message identifier	Message identifier
MQXQH_MSGDESC_CORRELID	DS	XL24	Correlation identifier	Correlation identifier
MQXQH_MSGDESC_BACKOUTCOUNT	DS	F	Backout counter	Backout counter
MQXQH_MSGDESC_REPLYTOQ	DS	CL48	Name of reply-to queue	Name of reply-to queue
MQXQH_MSGDESC_REPLYTOQMGR	DS	CL48	Name of reply queue manager	Name of reply queue manager
MQXQH_MSGDESC_USERIDENTIFIER	DS	CL12	User identifier	User identifier
MQXQH_MSGDESC_ACCOUNTINGTOKEN	DS	XL32	Accounting token	Accounting token
MQXQH_MSGDESC_APPLIDENTITYDATA	DS	CL32	Application data relating to identity	Application data relating to identity
* MQXQH_MSGDESC_PUTAPPLTYPE	DS	F	Type of application that put the message	Type of application that put the message
* MQXQH_MSGDESC_PUTAPPLNAME	DS	CL28	Name of application that put the message	Name of application that put the message
* MQXQH_MSGDESC_PUTDATE	DS	CL8	Date when message was put	Date when message was put
MQXQH_MSGDESC_PUTTIME	DS	CL8	Time when message was put	Time when message was put
MQXQH_MSGDESC_APPLORIGINDATA	DS	CL4	Application data relating to origin	Application data relating to origin
* MQXQH_MSGDESC_LENGTH	EQU	*-MQXQH_MSGDESC		
	ORG	MQXQH_MSGDESC		
MQXQH_MSGDESC_AREA	DS	CL(MQXQH_MSGDESC_LENGTH)		
* MQXQH_LENGTH	EQU	*-MQXQH		
	ORG	MQXQH		
MQXQH_AREA	DS	CL(MQXQH_LENGTH)		

## MQXQH의 Visual Basic 선언

```

Type MQXQH
  StructId      As String*4 'Structure identifier'
  Version       As Long      'Structure version number'
  RemoteQName   As String*48 'Name of destination queue'
  RemoteQMName As String*48 'Name of destination queue manager'
  MsgDesc       As MQMD1    'Original message descriptor'
End Type

```

## 별도의 메시지 디스크립터의 필드

전송 큐에 있는 메시지에는 두 개의 메시지 디스크립터가 있습니다.

- 하나의 메시지 디스크립터는 메시지 데이터와 별도로 저장됩니다. 개별 메시지 디스크립터라고 하며 메시지가 전송 큐에 우치되는 경우 큐 관리자에 의해 생성됩니다. 개별 메시지 디스크립터의 일부 필드는 MQPUT 또는 MQPUT1 호출에서 애플리케이션에 의해 제공된 메시지 디스크립터에서 복사됩니다.

별도의 메시지 디스크립터는 전송 큐에서 메시지가 제거될 때 MQGET 호출의 **MsgDesc** 매개변수에서 애플리케이션에 리턴되는 디스크립터입니다.

- 두 번째 메시지 디스크립터는 메시지 데이터의 일부로 MQXQH 구조 내에 저장됩니다. 임베드된 메시지 디스크립터라고 하며 MQPUT 또는 MQPUT1 호출에서 (최소 변형으로) 애플리케이션에 의해 제공되는 메시지 디스크립터의 사본입니다.

임베드된 메시지 디스크립터는 항상 버전 1 MQMD입니다. 애플리케이션에서 넣은 메시지가 MQMD에 있는 하나 이상의 버전 2 필드에 대해 기본값이 아닌 값을 가지고 있는 경우 MQMDE 구조는 MQXQH 뒤에 오고, 차례대로 애플리케이션 메시지 데이터가 뒤에 옵니다(있는 경우). MQMDE는 다음 중 하나입니다.

- 큐 관리자에 의해 생성됨(애플리케이션에서 메시지를 넣는 데 버전 2 MQMD를 사용하는 경우) 또는
- 이미 애플리케이션 메시지 데이터의 시작에 존재함(애플리케이션에서 메시지를 넣는 데 버전 1 MQMD를 사용하는 경우).

임베드된 메시지 디스크립터는 마지막 목적지 큐에서 메시지가 제거될 때 MQGET 호출의 **MsgDesc** 매개변수에서 애플리케이션에 리턴되는 디스크립터입니다.

개별 메시지 디스크립터의 필드는 표시된 대로 큐 관리자에 의해 설정됩니다. 큐 관리자가 버전 2 MQMD를 지원하지 않는 경우 버전 1 MQMD가 기능의 손실 없이 사용됩니다.

표 540. 별도의 MQMD의 필드에 사용된 값

별도의 MQMD의 필드	사용된 값
<i>StrucId</i>	MQMD_STRUC_ID
<i>Version</i>	MQMD_VERSION_2
<i>Report</i>	임베드된 메시지 디스크립터로부터 복사되었지만 0으로 설정된 MQRO_ACCEPT_UNSUP_IF_XMIT_MASK로 식별되는 비트가 포함됩니다. (메시지를 전송 큐에 넣거나 전송 큐에서 메시지가 제거될 때 COA 또는 COD 보고서 메시지가 생성되는 것을 방지합니다.)
<i>MsgType</i>	임베드된 메시지 디스크립터에서 복사합니다.
<i>Expiry</i>	임베드된 메시지 디스크립터에서 복사합니다.
<i>Feedback</i>	임베드된 메시지 디스크립터에서 복사합니다.
<i>Encoding</i>	MQENC_NATIVE(참고 참조)
<i>CodedCharSetId</i>	큐 관리자의 <b>CodedCharSetId</b> 속성.
<i>Format</i>	MQFMT_XMIT_Q_HEADER
<i>Priority</i>	임베드된 메시지 디스크립터에서 복사합니다.
<i>Persistence</i>	임베드된 메시지 디스크립터에서 복사합니다.
<i>MsgId</i>	새 값이 큐 관리자에 의해 생성됩니다. 이 메시지 ID는 큐 관리자가 이전에 설명된 임베드된 메시지 디스크립터에 대해 생성했을 수 있는 <i>MsgId</i> 와 다릅니다.
<i>CorrelId</i>	임베드된 메시지 디스크립터의 <i>MsgId</i> 입니다. SYSTEM.CLUSTER.TRANSMIT.QUEUE에 넣을 메시지의 경우 내부 사용을 위해 <i>CorrelId</i> 가 예약됩니다.
<i>BackoutCount</i>	0
<i>ReplyToQ</i>	임베드된 메시지 디스크립터에서 복사합니다.
<i>ReplyToQMgr</i>	임베드된 메시지 디스크립터에서 복사합니다.
<i>UserIdentifier</i>	임베드된 메시지 디스크립터에서 복사합니다.
<i>AccountingToken</i>	임베드된 메시지 디스크립터에서 복사합니다. SYSTEM.CLUSTER.TRANSMIT.QUEUE에 넣을 메시지의 경우 내부 사용을 위해 <i>AccountingToken</i> 이 예약됩니다.
<i>ApplIdentityData</i>	임베드된 메시지 디스크립터에서 복사합니다.
<i>PutApplType</i>	MQAT_QMGR
<i>PutApplName</i>	큐 관리자 이름의 첫 번째 28바이트.
<i>PutDate</i>	메시지를 전송 큐에 넣은 날짜.
<i>PutTime</i>	메시지를 전송 큐에 넣은 시간.
<i>ApplOriginData</i>	공백
<i>GroupId</i>	MQGI_NONE
<i>MsgSeqNumber</i>	1
<i>Offset</i>	0
<i>MsgFlags</i>	MQMF_NONE
<i>OriginalLength</i>	MQOL_UNDEFINED

- Windows에서 Micro Focus COBOL에 대한 MQENC\_NATIVE의 값은 C의 값과 다릅니다. 별도의 메시지 디스크립터에 있는 *Encoding* 필드의 값은 항상 이러한 환경에서 C의 값입니다. 이 값은 10진수 546입니다. 또한 MQXQH 구조의 정수 필드는 이 값에 해당하는 인코딩에 있습니다(고유 Intel 인코딩).

## 임베드된 메시지 디스크립터의 필드

임베드된 메시지 디스크립터의 필드에는 다음을 제외하고 MQPUT 또는 MQPUT1 호출의 **MsgDesc** 매개변수에 있는 값과 동일한 값이 있습니다.

- *Version* 필드에는 항상 값 MQMD\_VERSION\_1이 있습니다.
- *Priority* 필드에 값 MQPRI\_PRIORITY\_AS\_Q\_DEF가 있는 경우 이는 큐의 **DefPriority** 속성 값으로 대체됩니다.
- *Persistence* 필드에 값 MQPER\_PERSISTENCE\_AS\_Q\_DEF가 있는 경우 이는 큐의 **DefPersistence** 속성 값으로 대체됩니다.
- *MsgId* 필드에 값 MQMI\_NONE이 있거나 MQPMO\_NEW\_MSG\_ID 옵션이 지정되었거나 메시지가 분배 목록 메시지인 경우 *MsgId*는 큐 관리자가 생성한 새 메시지 ID로 대체됩니다.

분배 목록 메시지를 다른 전송 큐에 위치한 더 작은 분배 목록 메시지로 분할하는 경우 각 새 임베드된 메시지 디스크립터의 *MsgId* 필드는 원래 분배 목록 메시지에서의 동일합니다.

- MQPMO\_NEW\_CORREL\_ID 옵션이 지정되는 경우 *CorrelId*는 큐 관리자가 생성한 새 상관 ID로 대체됩니다.
- 컨텍스트 필드는 **PutMsgOpts** 매개변수에서 지정된 MQPMO\_\*\_CONTEXT 옵션으로 표시되는 대로 설정됩니다. 컨텍스트 필드는 다음과 같습니다.

- *AccountingToken*
- *ApplIdentityData*
- *ApplOriginData*
- *PutApplName*
- *PutApplType*
- *PutDate*
- *PutTime*
- *UserIdentifier*

- 하나 이상의 버전 2 필드에 기본값이 아닌 값이 있는 경우 MQMD에서 버전 2 필드(존재하는 경우)가 제거되고 MQMDE 구조로 이동됩니다.

## 리모트 큐에 메시지 넣기

애플리케이션이 리모트 큐에 메시지를 넣을 때 (리모트 큐의 이름을 직접 지정하거나 리모트 큐의 로컬 정의를 사용하여) 로컬 큐 관리자는 다음을 수행합니다.

- 임베드된 메시지 디스크립터를 포함하는 MQXQH 구조 작성
- 하나가 필요하고 이미 존재하지 않는 경우 MQMDE 추가
- 애플리케이션 메시지 데이터 추가
- 메시지를 적절한 전송 큐에 넣기

## 전송 큐에 직접 메시지 넣기

애플리케이션은 또한 메시지를 전송 큐에 직접 넣을 수 있습니다. 이 경우 애플리케이션은 애플리케이션 메시지 데이터에 접두부로 MQXQH 구조를 지정해야 하며 적절한 값으로 필드를 초기화해야 합니다. 또한 MQPUT 또는 MQPUT1 호출의 **MsgDesc** 매개변수에 있는 *Format* 필드에는 MQFMT\_XMIT\_Q\_HEADER값이 있어야 합니다.

애플리케이션에서 작성한 MQXQH 구조의 문자 데이터는 로컬 큐 관리자의 문자 세트에 있어야 하고 (**CodedCharSetId** 큐 관리자 속성에 의해 정의됨) 정수 데이터는 고유 시스템 인코딩에 있어야 합니다. 뿐만

아니라, MQXQH 구조의 문자 데이터는 정의된 필드 길이까지 공백으로 채워져야 합니다. 큐 관리자가 널 및 후속 문자를 MQXQH 구조의 공백으로 변환하지 않기 때문에, 데이터는 널 문자로 중간에 끝나면 안됩니다.

그러나 큐 관리자는 MQXQH 구조가 있는지 또는 올바른 값이 필드에 지정되었는지 확인하지 않습니다.

애플리케이션은 SYSTEM.CLUSTER.TRANSMIT.QUEUE에 직접 해당 메시지를 넣을 수 없습니다.

## 전송 큐에서 메시지 가져오기

전송 큐에서 메시지를 가져오는 애플리케이션은 고유 방식으로 MQXQH 구조에서 정보를 처리해야 합니다. 애플리케이션 메시지 데이터의 시작 부분에 MQXQH 구조가 있으면 MQGET 호출의 **MsgDesc** 매개변수에서 *Format* 필드에 리턴되는 MQFMT\_XMIT\_Q\_HEADER값으로 표시됩니다. **MsgDesc** 매개변수의 *CodedCharSetId* 및 *Encoding* 필드에서 리턴된 값은 MQXQH 구조에서 문자 및 정수 데이터의 문자 세트 및 인코딩을 표시합니다. 애플리케이션 메시지 데이터의 문자 세트 및 인코딩은 임베드된 메시지 디스크립터에서 *CodedCharSetId* 및 *Encoding* 필드로 표시됩니다.

## MQXQH의 경우 *StrucId* (MQCHAR4)

전송 큐 헤더 구조의 구조 ID입니다. 항상 입력 필드입니다. 해당 값은 MQXQH\_STRUC\_ID입니다.

값은 다음과 같아야 합니다.

### MQXQH\_STRUC\_ID

전송 큐 헤더 구조의 ID입니다.

C 프로그래밍 언어의 경우 상수 MQXQH\_STRUC\_ID\_ARRAY도 정의됩니다. 이 값은 MQXQH\_STRUC\_ID와 동일하지만 문자열이 아닌 문자의 배열입니다.

## MQXQH의 버전 (MQLONG)

구조 버전 번호입니다. 값은 다음과 같아야 합니다.

### MQXQH\_VERSION\_1

전송 큐 헤더 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

### MQXQH\_CURRENT\_VERSION

전송 큐 헤더 구조의 현재 버전.

이 필드의 초기값은 MQXQH\_VERSION\_1입니다.

## MQXQH의 경우 *RemoteQName* (MQCHAR48)

이것은 메시지의 명백한 최종 목적지인 메시지 큐의 이름입니다. 예를 들어, 이 큐가 *RemoteQMgrName*에 다른 리모트 큐의 로컬 정의가 되도록 정의된 경우 이 큐가 최종 목적지가 아닐 수 있습니다.

메시지가 분배 목록 메시지인 경우(즉, 임베드된 메시지 설명자의 *Format* 필드가 MQFMT\_DIST\_HEADER), *RemoteQName*은(는) 공백입니다.

이 필드의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백입니다.

## MQXQH에 대한 *RemoteQMgr* 이름 (MQCHAR48)

메시지의 명확한 최종 목적지인 큐를 소유하는 큐 공유 그룹 또는 큐 관리자의 이름입니다.

메시지가 분배 목록 메시지인 경우, *RemoteQMgrName*은(는) 공백입니다.

이 필드의 길이는 MQ\_Q\_MGR\_NAME\_LENGTH로 제공됩니다. 이 필드의 초기값은 C에서는 널 문자열이며 기타 프로그래밍 언어에서는 48자의 공백입니다.

## MQXQH용 *MsgDesc* (MQMD1)

이는 임베드된 메시지 디스크립터이며, 메시지를 원래 리모트 큐에 넣을 때 MQPUT 또는 MQPUT1 호출에서 **MsgDesc** 매개변수로 지정되는 메시지 디스크립터 MQMD의 달기 사본입니다.


**참고:** 이는 버전-1 MQMD입니다.

이 구조에서 필드의 초기값은 MQMD 구조의 초기값과 동일합니다.

## 함수 호출

이 절은 가능한 모든 MQI 호출에 대한 정보를 제공합니다. 가능한 각 언어에 대한 설명, 구문, 매개변수 정보, 사용법 참고, 언어 호출이 각 다른 호출에 제공됩니다.

### 관련 참조

 [MQI 호출의 CEDF 출력 예](#)

## 호출 설명

이 절은 MQI 호출을 설명합니다.

- [606 페이지의 『MQBACK - 백아웃 변경』](#)
- [610 페이지의 『MQBEGIN - 작업 단위 시작』](#)
- [613 페이지의 『MQBUFMH - 버퍼를 메시지 핸들로 변환』](#)
- [616 페이지의 『MQCB - 콜백 관리』](#)
- [625 페이지의 『MQCB FUNCTION - 콜백 함수』](#)
- [627 페이지의 『MQCLOSE - 오브젝트 닫기』](#)
- [634 페이지의 『MQCMIT - 변경사항 커미트』](#)
- [637 페이지의 『MQCONN - 큐 관리자 연결』](#)
- [644 페이지의 『MQCONNX - 연결 큐 관리자\(확장\)』](#)
- [650 페이지의 『MQCRTMH - 메시지 핸들 작성』](#)
- [653 페이지의 『MQCTL - 콜백 제어』](#)
- [659 페이지의 『MQDISC - 큐 관리자 연결 끊기』](#)
- [663 페이지의 『MQDLTMH - 메시지 핸들 삭제』](#)
- [665 페이지의 『MQDLTMP - 메시지 특성 삭제』](#)
- [668 페이지의 『MQGET - 메시지 가져오기』](#)
- [680 페이지의 『MQINQ - 오브젝트 속성 조회』](#)
- [696 페이지의 『MQINQMP - 메시지 특성 조회』](#)
- [701 페이지의 『MQMHBUF - 버퍼로 메시지 핸들 변환』](#)
- [705 페이지의 『MQOPEN - 오브젝트 열기』](#)
- [722 페이지의 『MQPUT - 넣기 메시지』](#)
- [735 페이지의 『MQPUT1 - 하나의 메시지 넣기』](#)
- [745 페이지의 『MQSET - 오브젝트 속성 설정』](#)
- [751 페이지의 『MQSETMP - 메시지 특성 설정』](#)
- [755 페이지의 『MQSTAT - 상태 정보 검색』](#)
- [701 페이지의 『MQMHBUF - 버퍼로 메시지 핸들 변환』](#)
- [759 페이지의 『MQSUB - 구독 등록』](#)
- [765 페이지의 『MQSUBRQ - 구독 요청』](#)

UNIX 플랫폼에서 *man* 페이지 양식의 온라인 도움말을 이러한 호출에 사용할 수 있습니다.

**참고:** MQXCNVC 및 MQ\_DATA\_CONV\_EXIT 데이터 변환과 연관된 호출은 [869 페이지의 『데이터 변환 엑시트』](#)에 설명되어 있습니다.



## 호출 설명에서 사용된 규칙

각 호출의 경우 이 토픽 콜렉션은 프로그래밍 언어와 독립적인 형식으로 호출의 매개변수 및 사용법에 대한 설명을 제공합니다. 지원되는 각 프로그래밍 언어로, 매개변수의 일반 선언 및 일반적인 호출의 호출이 다음에 나타납니다.

**중요사항:** IBM MQ API 호출을 코딩할 때 다음 절에서 설명된 대로 모든 관련 매개변수가 제공되는지 확인해야 합니다. 이를 수행하지 못하면 예측 불가능한 결과가 생성될 수 있습니다.

각 호출의 설명에는 다음 섹션이 포함되어 있습니다.

### 호출 이름

호출 이름, 뒤에 호출 목적에 대한 간단한 설명이 나옵니다.

### 매개변수

각 매개변수의 경우, 이름 뒤에 데이터 유형이 괄호로 표시되며 다음 중 하나입니다.

#### 입력

호출 시 매개변수에 정보를 제공합니다.

#### 출력

호출이 완료되거나 실패할 때 큐 관리자가 매개변수의 정보를 리턴합니다.

#### 입출력(I/O)

호출 시 매개변수에 정보를 제공하며 호출이 완료되거나 실패할 때 큐 관리자가 정보를 변경합니다.

예를 들면, 다음과 같습니다.

*Compcode (MQLONG) - 출력*

일부 경우 데이터 유형은 구조입니다. 모든 경우 [234 페이지의 『기본 데이터 유형』](#)에 데이터 유형 또는 구조에 대한 자세한 정보가 있습니다.

각 호출의 마지막 두 개 매개변수는 완료 코드와 이유 코드입니다. 완료 코드는 호출이 성공, 일부 성공 또는 실패되었는지를 표시합니다. 호출의 일부 성공 또는 실패에 대한 추가 정보는 이유 코드에서 제공됩니다. 각각의 완료 코드 및 이유 코드에 대한 자세한 정보는 [840 페이지의 『리턴 코드』](#)의 내용을 참조하십시오.

### 사용법 참고

호출에 대한 추가 정보이며, 사용 방법 및 사용에 대한 제한사항을 설명합니다.

### 어셈블러 언어 호출

일반적인 호출의 호출 및 어셈블러 언어에서 해당 매개변수의 선언입니다.

### C 호출

일반적인 호출의 호출 및 C에서 해당 매개변수의 선언입니다.

### COBOL 호출

일반적인 호출의 호출 및 COBOL에서 해당 매개변수의 선언입니다.

### PL/I 호출

일반적인 호출의 호출 및 PL/I에서 해당 매개변수의 선언입니다.

모든 매개변수는 참조로 전달됩니다.

### Visual Basic 호출

일반적인 호출의 호출 및 Visual Basic에서 해당 매개변수의 선언입니다.

기타 표기법 규칙은 다음과 같습니다.

### Constants

상수 이름은 대문자로 표시됩니다(예: MQOO\_OUTPUT). 접두부가 같은 상수 세트는 MQIA\_\*와 같이 표시됩니다. 상수 값은 [61 페이지의 『Constants』](#)의 내용을 참조하십시오.

### 배열

일부 호출의 경우 매개변수는 고정된 크기가 없는 문자열의 배열입니다. 해당 매개변수의 설명에서 소문자 n은 숫자 상수를 표시합니다. 해당 매개변수에 대한 선언을 코딩하는 경우 n을 필요한 숫자 값으로 바꾸십시오.

## C 언어에서 호출 사용

입력 전용이고 MQHCONN, MQHOBJ, MQHMSG 또는 MQLONG 유형인 매개변수는 값으로 전달됩니다. 다른 모든 매개변수의 경우 매개변수의 주소가 값으로 전달됩니다.

함수를 호출할 때마다 주소가 전달하는 모든 매개변수를 지정할 필요가 없습니다. 특정 매개변수가 필요하지 않은 경우 매개변수 데이터의 주소 대신 함수 호출에서 널 포인터를 매개변수로 지정하십시오. 이러한 매개변수는 호출 설명에서 식별됩니다.

매개변수가 호출의 값으로 리턴되지 않습니다. C 용어에서 이는 모든 호출이 void를 리턴함을 의미합니다.

버퍼 매개변수 선언

**MQGET, MQPUT, MQPUT1** 호출에는 각각 정의되지 않은 데이터 유형이 있는 하나의 매개변수가 있습니다(*Buffer* 매개변수). 이 매개변수를 사용하여 애플리케이션의 메시지 데이터를 전송하고 수신하십시오.

이러한 종류의 매개변수는 MQBYTE의 배열로 C 예제에 표시됩니다. 이 방식으로 매개변수를 선언할 수 있지만 일반적으로 메시지의 데이터 레이아웃을 설명하는 특정 구조로 선언하는 것이 보다 편리합니다. 데이터 정렬의 주소를 호출 호출의 매개변수로 지정할 수 있도록 함수 프로토타입이 매개변수를 void에 대한 포인터로 선언합니다.

void에 대한 포인터는 정의되지 않은 형식의 데이터에 대한 포인터입니다. 이는 다음으로 정의됩니다.

```
typedef void *PMQVOID;
```

## MQBACK - 백아웃 변경

MQBACK 호출은 마지막 동기점 이후에 발생한 모든 메시지 가져오기 및 넣기가 백아웃되도록 큐 관리자에 지시합니다.

작업 단위의 일부로 넣어진 메시지는 삭제되지만, 작업 단위의 일부로 검색된 메시지는 큐에 복구됩니다.

- z/OS에서 이 호출은 배치 프로그램에 의해서만 사용됩니다(IMS 배치 DL/I 프로그램 포함).

## 구문

MQBACK(*Hconn*, *Compcode*, *Reason*)

## 매개변수

### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *Hconn*의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

### Compcode

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_WARNING

경고(일부 완료).

#### MQCC\_FAILED

호출에 실패했습니다.

## 원인

유형: MQLONG - 출력

*CompCode*이(가) MQCC\_OK인 경우:

**MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode이(가) MQCC\_WARNING인 경우:

**MQRC\_OUTCOME\_PENDING**

(2124, X'84C') 백아웃 조작의 결과가 보류 중입니다.

CompCode이(가) MQCC\_FAILED인 경우:

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

**MQRC\_API\_EXIT\_ERROR**

(2374, X'946') API 엑시트가 실패했습니다.

**MQRC\_ASID\_MISMATCH**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**MQRC\_CF\_STRUC\_IN\_USE**

(2346, X'92A') 커플링 기능 구조가 사용 중입니다.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**MQRC\_ENVIRONMENT\_ERROR**

(2012, X'7DC') 호출이 환경 내에서 유효하지 않습니다.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**MQRC\_OBJECT\_DAMAGED**

(2101, X'835') 오브젝트가 손상되었습니다.

**MQRC\_OUTCOME\_MIXED**

(2123, X'84B') 커밋 또는 백아웃 조작의 결과가 혼합되어 있습니다.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') 큐 관리자가 종료되었습니다.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**MQRC\_STORAGE\_MEDIUM\_FULL**

(2192, X'890') 외부 스토리지 매체가 가득 찼습니다.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드를 참조하십시오](#).

## 사용법 참고

- 큐 관리자가 작업 단위를 조정하는 경우에만 이 호출을 사용할 수 있습니다. 다음과 같습니다.
  - 로컬 작업 단위. 여기서 변경사항은 MQ 자원에만 영향을 줍니다.
  - 글로벌 작업 단위. 여기서 변경사항은 MQ 자원에 영향을 줄 뿐만 아니라 기타 자원 관리자에 속한 자원에도 영향을 줄 수 있습니다.

로컬 및 글로벌 작업 단위에 대한 세부사항은 [610 페이지의 『MQBEGIN - 작업 단위 시작』](#)의 내용을 참조하십시오.
- 큐 관리자가 작업 단위를 조정하지 않는 환경에서는 MQBACK 대신 적절한 백아웃을 사용하십시오. 이 환경에서는 애플리케이션의 비정상적인 종료로 인해 발생한 암시적 백아웃을 지원할 수도 있습니다.

- z/OS에서는 다음 호출을 사용하십시오.
    - 배치 프로그램(IMS 배치 DL/I 프로그램 포함)은 작업 단위가 MQ 자원에만 영향을 미치는 경우 MQBACK 호출을 사용할 수 있습니다. 그러나 작업 단위가 MQ 자원 및 기타 자원 관리자에 속한 자원 모두에 영향을 미치는 경우(예: Db2®) z/OS RRS(Recoverable Resource Service)가 제공하는 SRRBACK 호출을 사용하십시오. SRRBACK 호출은 RRS 조정을 위해 설정한 자원 관리자에 속한 자원의 변경사항을 백아웃합니다.
    - CICS 애플리케이션은 EXEC CICS SYNCPOINT ROLLBACK 명령을 사용하여 작업 단위를 백아웃해야 합니다. CICS 애플리케이션에 대해 MQBACK 호출을 사용하지 마십시오.
    - IMS 애플리케이션(일괄 처리 DL/I 프로그램 제외)은 작업 단위를 백아웃하기 위해 ROLB과(와) 같은 IMS 호출을 사용해야 합니다. (배치 DL/I 프로그램 이외의) IMS 애플리케이션에 대해 MQBACK 호출을 사용하지 마십시오.
  - IBM i에서는 큐 관리자가 조정하는 로컬 작업 단위에 이 호출을 사용하십시오. 이는 커밋 정의가 작업 레벨에 존재하지 않아야 함을 의미합니다. 즉, **CMTSCOPE(\*JOB)** 매개변수가 있는 STRCMTCTL 명령이 작업에 대해 실행되지 않았어야 합니다.
3. 애플리케이션이 작업 단위에서 커밋되지 않은 변경사항으로 종료되는 경우, 해당 변경사항의 배치는 애플리케이션이 정상으로 또는 비정상적으로 종료되는지 여부에 달려 있습니다. 추가적인 세부사항은 [659 페이지의 『MQDISC - 큐 관리자 연결 끊기』](#)의 사용 시 참고사항을 참조하십시오.
  4. 애플리케이션이 논리 메시지의 세그먼트 또는 그룹에서 메시지를 넣거나 가져올 때 큐 관리자는 마지막 성공한 MQPUT 및 MQGET 호출에 대한 메시지 그룹 및 논리 메시지와 관련된 정보를 보유하고 있습니다. 이 정보는 큐 핸들과 연관되어 있으며 다음과 같은 내용을 포함합니다.
    - MQMD에 있는 *GroupId*, *MsgSeqNumber*, *Offset* 및 *MsgFlags* 필드의 값입니다.
    - 메시지가 작업 단위의 일부인지 여부.
    - MQPUT 호출의 경우: 메시지가 지속적인지 또는 비지속적인지 여부.

큐 관리자는 다음 각각에 대해 한 세트씩, 그룹 및 세그먼트 정보의 세 개 세트를 유지합니다.

    - 마지막 성공적인 MQPUT 호출(이는 작업 단위의 일부일 수 있음).
    - 큐에서 메시지를 제거한 마지막 성공적인 MQGET 호출(이는 작업 단위의 일부일 수 있음).
    - 큐에서 메시지를 찾아보는 마지막으로 성공한 MQGET 호출(작업 단위의 일부가 될 수 없음).
  5. MQGET 호출과 연관된 정보가 현재 작업 단위에서 해당 큐 핸들에 대해 처음으로 성공한 MQGET 호출 전에 가지고 있던 값으로 복원됩니다.
 

작업 단위가 시작된 이후 애플리케이션을 통해 업데이트되었지만 작업 단위 범위의 외부에 있는 큐에서는 작업 단위가 백업된 경우 그룹 및 세그먼트 정보가 복원되지 않습니다.

작업의 단위가 백아웃될 때 그룹과 세그먼트 정보를 해당 이전 값으로 복구하면 애플리케이션이 여러 작업 단위 전체에 걸쳐 많은 세그먼트로 구성되어 대량 메시지 그룹 또는 대량 논리 메시지를 펼칠 수 있습니다. 그리고 작업 단위 중 하나가 실패하면 메시지 그룹 또는 논리 메시지의 정확한 위치에서 재시작하도록 허용합니다.

로컬 큐 관리자에 제한된 큐 스토리지만 있는 경우에는 여러 작업 단위를 사용하면 유용할 수 있습니다. 시스템 장애가 발생한 경우 그러나 애플리케이션은 정확한 위치에 메시지를 넣거나 가져와서 재시작할 수 있기 위한 충분한 정보를 유지보수해야 합니다.

시스템 장애 후에 올바른 지점에서 다시 시작하는 방법에 대한 세부사항은 [485 페이지의 『MQPMO - 메시지 넣기 옵션』](#)에서 설명하는 MQPMO\_LOGICAL\_ORDER 옵션 및 [362 페이지의 『MQGMO - 메시지 가져오기 옵션』](#)에서 설명하는 MQGMO\_LOGICAL\_ORDER 옵션을 참조하십시오.

나머지 사용법 참고사항은 큐 관리자가 작업 단위를 통합할 때만 적용됩니다.
  6. 작업 단위는 연결 핸들과 동일한 범위를 갖습니다. 특정 작업 단위에 영향을 미치는 모든 MQ 호출은 동일한 연결 핸들을 사용하여 수행해야 합니다. 다른 연결 핸들을 사용하여 실행된 호출(예: 다른 애플리케이션이 발한 호출)은 다음 작업 단위에 영향을 줍니다. 연결 핸들의 범위에 대한 정보는 [637 페이지의 『MQCONN - 큐 관리자 연결』](#)에 설명된 **Hconn** 매개변수를 참조하십시오.
  7. 현재 작업 단위의 일부로서 넣거나 검색된 메시지만 이 호출의 영향을 받습니다.

8. 작업 단위 내에서 MQGET, MQPUT 또는 MQPUT1 호출을 발행했지만 커밋 또는 백아웃 호출을 발행하지 않고 장시간 실행 중인 애플리케이션이 다른 애플리케이션에서 사용할 수 없는 메시지로 큐를 채울 수 있습니다. 이러한 가능성에 대비하기 위해 관리자는 **MaxUncommittedMsgs** 큐 관리자 속성을 런어웨이 애플리케이션이 큐를 채우는 것을 방지할 만큼 충분히 낮지만 예상 메시징 애플리케이션이 올바르게 작동할 수 있을 만큼 충분히 높은 값으로 설정해야 합니다.

## C 호출

```
MQBACK (Hconn, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN  Hconn;      /* Connection handle */
MQLONG   CompCode;   /* Completion code */
MQLONG   Reason;     /* Reason code qualifying CompCode */
```

## COBOL 호출

```
CALL 'MQBACK' USING HCONN, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```
** Connection handle
01 HCONN    PIC S9(9) BINARY.
** Completion code
01 COMPCODE PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON   PIC S9(9) BINARY.
```

## PL/I 호출

```
call MQBACK (Hconn, CompCode, Reason);
```

매개변수를 다음과 같이 선언합니다.

```
dcl Hconn      fixed bin(31); /* Connection handle */
dcl CompCode   fixed bin(31); /* Completion code */
dcl Reason     fixed bin(31); /* Reason code qualifying CompCode */
```

## 상위 레벨 어셈블러 호출

```
CALL MQBACK,(HCONN,COMPCODE,REASON)
```

매개변수를 다음과 같이 선언합니다.

```
HCONN      DS F Connection handle
COMPCODE   DS F Completion code
REASON     DS F Reason code qualifying COMPCODE
```

## Visual Basic 호출

```
MQBACK Hconn, CompCode, Reason
```

매개변수를 다음과 같이 선언합니다.

```
Dim Hconn      As Long 'Connection handle'  
Dim CompCode  As Long 'Completion code'  
Dim Reason    As Long 'Reason code qualifying CompCode'
```

## MQBEGIN - 작업 단위 시작

MQBEGIN 호출이 큐 관리자가 조정하고 외부 자원 관리자를 포함할 수 있는 작업 단위를 시작합니다.

### 구문

MQBEGIN(*Hconn*, *BeginOptions*, *Compcode*, *Reason*)

### 매개변수

#### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *Hconn*의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

*Hconn*은 비공유 연결 핸들이어야 합니다. 공유 연결 핸들이 지정되면 이유 코드 MQRC\_HCONN\_ERROR와 함께 호출이 실패합니다. 공유 및 비공유 핸들에 대한 자세한 정보는 312 페이지의 『MQCNO - 연결 옵션』에서 MQCNO\_HANDLE\_SHARE\_\* 옵션에 대한 설명을 참조하십시오.

#### BeginOptions

유형: MQBO - 입출력(I/O)

277 페이지의 『MQBO - 시작 옵션』에서 설명된 대로 MQBEGIN의 조치를 제어하는 옵션입니다.

옵션이 필요하지 않은 경우 C 또는 S/390 어셈블러로 작성된 프로그램은 MQBO 구조의 주소를 지정하는 대신에 널 매개변수 주소를 지정할 수 있습니다.

#### CompCode

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

##### MQCC\_OK

정상적으로 완료되었습니다.

##### MQCC\_WARNING

경고(일부 완료).

##### MQCC\_FAILED

호출에 실패했습니다.

#### 원인

유형: MQLONG - 출력

*CompCode*이(가) MQCC\_OK인 경우:

##### MQRC\_NONE

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_WARNING인 경우:

##### MQRC\_NO\_EXTERNAL\_PARTICIPANTS

(2121, X'849') 참여하는 자원 관리자가 등록되어 있지 않습니다.

##### MQRC\_PARTICIPANT\_NOT\_AVAILABLE

(2122, X'84A') 참여하는 자원 관리자를 사용할 수 없습니다.

*CompCode*이(가) MQCC\_FAILED인 경우:

**MQRC\_API\_EXIT\_ERROR**

(2374, X'946') API 엑시트가 실패했습니다.

**MQRC\_BO\_ERROR**

(2134, X'856') 시작 옵션 구조가 올바르지 않습니다.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**MQRC\_ENVIRONMENT\_ERROR**

(2012, X'7DC') 호출이 환경 내에서 유효하지 않습니다.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**MQRC\_OPTIONS\_ERROR**

(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') 큐 관리자가 종료되었습니다.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

**MQRC\_UOW\_IN\_PROGRESS**

(2128, X'850') 작업 단위가 이미 시작되었습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

## 사용법 참고

1. MQBEGIN 호출을 사용하여 큐 관리자가 조정하고 기타 자원 관리자가 소유한 자원에 대한 변경사항을 포함할 수 있는 작업 단위를 시작하십시오. 큐 관리자는 작업 단위의 세 가지 유형을 지원합니다.
  - **큐 관리자 통합 로컬 작업 단위:** 큐 관리자가 참여하는 유일한 자원 관리자이므로 큐 관리자가 작업 단위 코디네이터로 수행하는 작업 단위입니다.
    - 이 작업 단위 유형을 시작하려면 작업 단위의 첫 번째 MQPUT, MQPUT1 또는 MQGET 호출에서 MQPMO\_SYNCPOINT 또는 MQGMO\_SYNCPOINT 옵션을 지정하십시오.
    - 이 작업 단위 유형을 커밋 또는 백아웃하려면 MQCMIT 또는 MQBACK 호출을 사용하십시오.
  - **큐 관리자 통합 글로벌 작업 단위:** 큐 관리자가 기타 자원 관리자에 속한 자원 및 MQ 자원 모두에 대해 작업 단위 코디네이터로 수행하는 작업 단위입니다. 해당 자원 관리자는 작업 단위의 자원의 모든 변경사항이 커밋되거나 함께 백아웃되는지 확인하기 위해 큐 관리자와 협력합니다.
    - 이 작업 단위 유형을 시작하려면 MQBEGIN 호출을 사용하십시오.
    - 이 작업 단위 유형을 커밋 또는 백아웃하려면 MQCMIT 및 MQBACK 호출을 사용하십시오.
  - **외부 조정된 글로벌 작업 단위:** 큐 관리자가 참여 중이지만 큐 관리자가 작업 단위 조정자로 수행하지 않는 작업 단위입니다. 대신 큐 관리자가 협업하는 외부 작업 단위 조정자가 있습니다.
    - 이 작업 단위 유형을 시작하려면 외부 작업 단위 조정자가 제공하는 관련 호출을 사용하십시오.  
MQBEGIN 호출이 작업 단위를 시작하려고 시도하는 데 사용되는 경우 이유 코드 MQRC\_ENVIRONMENT\_ERROR와 함께 호출이 실패합니다.
    - 이 작업 단위 유형을 커밋 또는 백아웃하려면 외부 작업 단위 조정자가 제공하는 커밋 및 백아웃 호출을 사용하십시오.



MQCMIT 또는 MQBACK 호출을 사용하여 작업 단위를 커밋 또는 백아웃하는 경우 이유 코드 MQRC\_ENVIRONMENT\_ERROR와 함께 호출이 실패합니다.

2. 애플리케이션이 작업 단위에서 커밋되지 않은 변경사항으로 종료되는 경우 해당 변경사항의 배치는 애플리케이션이 정상으로 또는 비정상적으로 종료되는지 여부에 달려 있습니다. 추가적인 세부사항은 [659 페이지의 『MQDISC - 큐 관리자 연결 끊기』](#)의 사용 시 참고사항을 참조하십시오.
3. 애플리케이션은 동시에 단지 하나의 작업 단위에만 참여할 수 있습니다. 작업 단위 유형에 상관없이 애플리케이션에 대해 존재하는 작업 단위가 이미 있는 경우 MQBEGIN 호출이 이유 코드 MQRC\_UOW\_IN\_PROGRESS와 함께 실패합니다.
4. MQBEGIN 호출은 MQ MQI 클라이언트 환경에서 올바르지 않습니다. 호출을 사용하려는 시도가 이유 코드 MQRC\_ENVIRONMENT\_ERROR와 함께 실패합니다.
5. 큐 관리자가 글로벌 작업 단위에 대해 작업 단위 조정자로 수행하는 경우 작업 단위에 참여할 수 있는 자원 관리자는 큐 관리자 구성 파일에서 정의됩니다.
6. IBM i에서 작업 단위의 세 가지 유형은 다음과 같이 지원됩니다.
  - 큐 관리자 조정 로컬 작업 단위 는 커밋 정의가 작업 레벨에 없는 경우에만 사용할 수 있습니다. 즉, **CMTSOPE(\*JOB)** 매개변수가 있는 STRCMTCTL 명령은 작업에 대해 실행되지 않아야 합니다.
  - 큐 관리자 통합 글로벌 작업 단위는 지원되지 않습니다.
  - 외부적으로 조정된 글로벌 작업 단위 는 커밋 정의가 작업 레벨에 있는 경우에만 사용할 수 있습니다. 즉, **CMTSOPE(\*JOB)** 매개변수와 함께 STRCMTCTL 명령을 작업에 대해 실행해야 합니다. 이를 완료한 경우 IBM i COMMIT 및 ROLLBACK 조작용 참여하는 기타 자원 관리자에 속한 자원 및 MQ 자원에 적용됩니다.

## C 호출

```
MQBEGIN (Hconn, &BeginOptions, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN Hconn;          /* Connection handle */
MQBO    BeginOptions;   /* Options that control the action of MQBEGIN */
MQLONG  CompCode;      /* Completion code */
MQLONG  Reason;        /* Reason code qualifying CompCode */
```

## COBOL 호출

```
CALL 'MQBEGIN' USING HCONN, BEGINOPTIONS, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```
** Connection handle
01 HCONN          PIC S9(9) BINARY.
** Options that control the action of MQBEGIN
01 BEGINOPTIONS.
   COPY CMQBOV.
** Completion code
01 COMPCODE      PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON        PIC S9(9) BINARY.
```

## PL/I 호출

```
call MQBEGIN (Hconn, BeginOptions, CompCode, Reason);
```

매개변수를 다음과 같이 선언합니다.



```

dcl Hconn          fixed bin(31); /* Connection handle */
dcl BeginOptions  like MQBO;     /* Options that control the action of
MQBEGIN */
dcl CompCode      fixed bin(31); /* Completion code */
dcl Reason        fixed bin(31); /* Reason code qualifying CompCode */

```

## Visual Basic 호출

```
MQBEGIN Hconn, BeginOptions, CompCode, Reason
```

매개변수를 다음과 같이 선언합니다.

```

Dim Hconn          As Long 'Connection handle'
Dim BeginOptions  As MQBO 'Options that control the action of MQBEGIN'
Dim CompCode      As Long 'Completion code'
Dim Reason        As Long 'Reason code qualifying CompCode'

```

## MQBUFMH - 버퍼를 메시지 핸들로 변환

MQBUFMH 함수 호출은 버퍼를 메시지 핸들로 변환시키며 MQMHBUF 호출의 역수입니다.

이 호출은 버퍼의 메시지 디스크립터 및 MQRFH2 특성을 가져오고 메시지 핸들을 통해 사용 가능하게 합니다. 메시지 데이터의 MQRFH2 특성이 선택적으로 제거되었습니다. 특성이 제거된 후 메시지 디스크립터의 *Encoding*, *CodedCharSetId* 및 *Format* 필드가 업데이트되고 필요한 경우 버퍼의 콘텐츠를 올바르게 설명합니다.

## 구문

```
MQBUFMH(Hconn, Hmsg, BufMsgHOpts, MsgDesc, BufferLength, Buffer, DataLength, Compcode, Reason)
```

## 매개변수

### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. **Hconn** 값이 **Hmsg** 매개변수에 지정된 메시지 핸들 작성에 사용된 연결 핸들과 일치해야 합니다.

메시지 핸들이 MQHC\_UNASSOCIATED\_HCONN을 사용하여 작성된 경우 메시지 핸들로 버퍼를 변환하는 스레드에서 올바른 연결을 설정해야 합니다. 올바른 연결이 설정되지 않는 경우 MQRC\_CONNECTION\_BROKEN과 함께 호출이 실패합니다.

### Hmsg

유형: MQHMQSG - 입력

이는 버퍼가 필요한 메시지 핸들입니다. 값은 이전 MQCRTMH 호출에서 리턴합니다.

### BufMsgHOpts

유형: MQBMHO - 입력

MQBMHO 구조를 사용하여 애플리케이션이 메시지 핸들이 버퍼에서 생성되는 방법을 제어하는 옵션을 지정할 수 있습니다.

자세한 내용은 272 페이지의 『MQBMHO - 버퍼 대 메시지 핸들 옵션』의 내용을 참조하십시오.

### MsgDesc

유형: MQMD - 입출력(I/O)

*MsgDesc* 구조는 메시지 디스크립터 특성을 포함하며 버퍼 영역의 콘텐츠를 설명합니다.

호출의 출력에서 특성은 선택적으로 버퍼 영역에서 제거되고 이 경우에 메시지 디스크립터가 올바르게 버퍼 영역을 설명하기 위해 업데이트됩니다.

이 구조의 데이터는 애플리케이션의 문자 세트 및 인코딩에 있어야 합니다.

### **BufferLength**

유형: MQLONG - 입력

*BufferLength*는 버퍼 영역의 길이(바이트)입니다.

0바이트의 *BufferLength*가 올바르며 버퍼 영역이 데이터를 포함하지 않음을 표시합니다.

### **버퍼**

유형: MQBYTEExBufferLength - 입출력(I/O)

610 페이지의 『MQBEGIN - 작업 단위 시작』에서 설명된 대로 MQBEGIN의 조치를 제어하는 옵션입니다.

**Buffer**는 메시지 버퍼가 포함된 영역을 정의합니다. 대부분의 데이터에 대해 4바이트 경계에서 버퍼를 맞추어야 합니다.

**Buffer**에 문자 또는 숫자 데이터가 포함된 경우 **MsgDesc** 매개변수의 *CodedCharSetId* 및 *Encoding* 필드를 데이터에 적합한 값으로 설정하십시오. 그러면 필요한 경우 데이터를 변환할 수 있습니다.

특성이 메시지 버퍼에 있으면 선택적으로 제거됩니다. 나중에 호출에서 리턴 시 메시지 핸들에서 사용할 수 있게 됩니다.

C 프로그래밍 언어에서 매개변수는 pointer-to-void로 선언됩니다. 이는 모든 유형의 데이터 주소를 매개변수로 지정할 수 있다는 의미입니다.

**BufferLength** 매개변수가 0인 경우 **Buffer**는 참조되지 않습니다. 이 경우 C 또는 시스템/390 어셈블러로 작성된 프로그램이 전달한 매개변수 주소가 널일 수 있습니다.

### **DataLength**

유형: MQLONG - 출력

제거된 특성을 가질 수 있는 버퍼의 길이(바이트)입니다.

### **CompCode**

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

#### **MQCC\_OK**

정상적으로 완료되었습니다.

#### **MQCC\_FAILED**

호출에 실패했습니다.

### **원인**

유형: MQLONG - 출력

*CompCode*이(가) MQCC\_OK인 경우:

#### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_FAILED인 경우:

#### **MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'089C') 어댑터를 사용할 수 없습니다.

#### **MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

#### **MQRC\_ASID\_MISMATCH**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

#### **MQRC\_BMHO\_ERROR**

(2489, X'09B9') 메시지 핸들 옵션 구조에 대한 버퍼가 올바르지 않습니다.

#### **MQRC\_BUFFER\_ERROR**

(2004, X'07D4') 버퍼 매개변수가 올바르지 않습니다.

**MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'07D5') 버퍼 길이 매개변수가 올바르지 않습니다.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'08AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'07D9') 큐 관리자에 대한 연결이 유실되었습니다.

**MQRC\_HMSG\_ERROR**

(2460, X'099C') 메시지 핸들이 올바르지 않습니다.

**MQRC\_MD\_ERROR**

(2026, X'07EA') 메시지 디스크립터가 올바르지 않습니다.

**MQRC\_MSG\_HANDLE\_IN\_USE**

(2499, X'09C3') 메시지 핸들이 이미 사용 중입니다.

**MQRC\_OPTIONS\_ERROR**

(2046, X'07FE') 옵션이 올바르지 않거나 일치하지 않습니다.

**MQRC\_RFH\_ERROR**

(2334, X'091E') MQRFH2 구조가 올바르지 않습니다.

**MQRC\_RFH\_FORMAT\_ERROR**

(2421, X'0975') 특성을 포함하는 MQRFH2 폴더를 구문 분석할 수 없습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

## 사용법 참고

MQBUFMH 호출은 API 엑시트로 인터셉트될 수 없으며 버퍼가 애플리케이션 공간에서 메시지 핸들로 변환됩니다. 호출은 큐 관리자에 도달하지 않습니다.

## C 호출

```
MQBUFMH (Hconn, Hmsg, &BufMsgHOpts, &MsgDesc, BufferLength, Buffer,
         &DataLength, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN Hconn;          /* Connection handle */
MQHMSG  Hmsg;           /* Message handle */
MQBMHO  BufMsgHOpts;   /* Options that control the action of MQBUFMH */
MQMD    MsgDesc;       /* Message descriptor */
MQLONG  BufferLength;   /* Length in bytes of the Buffer area */
MQBYTE  Buffer[n];      /* Area to contain the message buffer */
MQLONG  DataLength;    /* Length of the output buffer */
MQLONG  CompCode;      /* Completion code */
MQLONG  Reason;        /* Reason code qualifying CompCode */
```

## COBOL 호출

```
CALL 'MQBUFMH' USING HCONN, HMSG, BUFMSGHOPTS, MSGDESC, BUFFERLENGTH,
                   BUFFER, DATALENGTH, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```
** Connection handle
01 HCONN          PIC S9(9) BINARY.
** Message handle
01 HMSG           PIC S9(18) BINARY.
```

```

** Options that control the action of MQBUFMH
01 BUFMSGHOPTS.
   COPY CMQBMHOV.
** Message descriptor
01 MSGDESC.
   COPY CMQMD.
** Length in bytes of the Buffer area
01 BUFFERLENGTH PIC S9(9) BINARY.
** Area to contain the message buffer
01 BUFFER PIC X(n).
** Length of the output buffer
01 DATALENGTH PIC S9(9) BINARY.
** Completion code
01 COMPCODE PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON PIC S9(9) BINARY.

```

## PL/I 호출

```
call MQBUFMH (Hconn, Hmsg, BufMsgHOpts, MsgDesc, BufferLength, Buffer,
DataLength, CompCode, Reason);
```

매개변수를 다음과 같이 선언합니다.

```

dcl Hconn          fixed bin(31); /* Connection handle */
dcl Hmsg           fixed bin(63); /* Message handle */
dcl BufMsgHOpts   like MQBMHO;   /* Options that control the action of
MQBUFMH */
dcl MsgDesc       like MQMD;     /* Message descriptor */
dcl BufferLength   fixed bin(31); /* Length in bytes of the Buffer area */
dcl Buffer         char(n);       /* Area to contain the message buffer */
dcl DataLength    fixed bin(31); /* Length of the output buffer */
dcl CompCode      fixed bin(31); /* Completion code */
dcl Reason        fixed bin(31); /* Reason code qualifying CompCode */

```

## 상위 레벨 어셈블러 호출

```
CALL MQBUFMH, (HCONN,HMSG,BUFMSGHOPTS,MSGDESC,BUFFERLENGTH,BUFFER,
DATALENGTH,COMPCODE,REASON)
```

매개변수를 다음과 같이 선언합니다.

HCONN	DS	F	Connection handle
HMSG	DS	D	Message handle
BUFMSGHOPTS	CMQBMHOA	,	Options that control the action of MQBUFMH
MSGDESC	CMQMDA	,	Message descriptor
BUFFERLENGTH	DS	F	Length in bytes of the BUFFER area
BUFFER	DS	CL(n)	Area to contain the properties
DATALENGTH	DS	F	Length of the output buffer
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## MQCB - 콜백 관리

MQCB 호출은 지정된 오브젝트 핸들의 콜백을 등록하고 활성화를 제어하며 콜백으로 변경합니다.

콜백은 특정 이벤트가 발생할 때 IBM MQ에서 호출하는 함수 포인터로 또는 동적으로 링크할 수 있는 함수 이름으로 지정되는 코드의 조각입니다.

클라이언트에서 MQCB 및 MQCTL을 사용하려면, 채널의 협상된 **SHARECNV** 매개변수가 0이 아닌 값에 동의한 서버에 연결되어야 합니다.

정의될 수 있는 호출의 유형은 다음과 같습니다.

## 메시지 이용자

지정된 선택 기준을 충족하는 메시지를 오브젝트 핸들에서 사용할 수 있을 때 메시지 이용자 콜백 함수가 호출됩니다.

하나의 콜백 함수만 각 오브젝트 핸들에 등록할 수 있습니다. 단일 큐가 다중 선택 기준을 읽으면 큐는 여러 번 열려야하고 이용자 함수가 각 핸들에 등록됩니다.

## 이벤트 핸들러

이벤트 핸들러는 전체 콜백 환경에 영향을 주는 조건에 호출됩니다.

이벤트 조건이 발생하면 함수가 호출됩니다(예를 들어, 큐 관리자나 연결 중지 또는 정지).

단일 메시지 이용자에 특정한 조건에 대해 함수가 호출되지 않습니다(예: MQRC\_GET\_INHIBITED). 그러나 콜백 함수가 정상적으로 종료되지 않는 경우에는 호출됩니다.

## 구문

MQCB(*Hconn, Operation, CallbackDesc, Hobj, MsgDesc, GetMsgOpts, CompCode, Reason*)

## 매개변수

### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *Hconn*의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

CICS 용 z/OS 애플리케이션에서 MQHC\_DEF\_HCONN에 대해 다음 특수 값을 지정하여 이 실행 단위와 연관된 연결 핸들을 사용할 수 있습니다.

### Operation

유형: MQLONG - 입력

지정된 오브젝트 핸들에 대해 정의된 콜백에서 처리 중인 조작입니다. 다음 옵션 중 하나를 지정해야 합니다. 둘 이상의 옵션을 지정하려면 값을 함께 추가하거나(동일한 상수를 두 번 이상 추가하지 않음), 비트 단위 OR 연산을 사용하여 값을 결합하십시오(프로그래밍 언어가 비트 연산을 지원하는 경우).

### MQOP\_REGISTER

정의된 오브젝트 핸들에 대한 콜백 함수를 정의하십시오. 이 조작은 호출될 함수 및 사용될 선택 기준을 정의합니다.

콜백 함수가 오브젝트 핸들에 이미 정의된 경우 정의는 대체됩니다. 콜백을 대체하는 동안 오류가 감지되는 경우 함수가 등록 취소됩니다.

호출이 이전에 등록 취소된 동일한 콜백 함수에 등록되는 경우 이는 대체 작업으로 처리됩니다. 초기 또는 마지막 호출은 호출되지 않습니다.

MQOP\_SUSPEND 또는 MQOP\_RESUME와 함께 MQOP\_REGISTER를 사용할 수 있습니다.

### MQOP\_DEREGISTER

오브젝트 핸들의 메시지 이용을 중지하고 콜백에 적합한 핸들을 제거합니다.

연관된 핸들이 처리완료되는 경우 콜백이 자동으로 등록 취소됩니다.

MQOP\_DEREGISTER가 이용자 내에서 호출되고 콜백에 정의된 중지 호출이 있는 경우 이용자에서 리턴 시 호출됩니다.

등록된 이용자가 있는 *Hobj*에 대해 이 조작이 실행되면 호출이 MQRC\_CALLBACK\_NOT\_REGISTERED와 함께 리턴됩니다.

### MQOP\_SUSPEND

오브젝트 핸들에 대한 메시지의 이용 일시중단.

이 조작이 이벤트 핸들러에 적용되는 경우 일시중단되는 동안 이벤트 핸들러는 이벤트를 가져오지 않고 일시중단 상태인 동안에는 재개될 때 조작에 제공하지 않습니다.

일시중단되는 동안 이용자 함수는 제어 유형 콜백을 계속 가져옵니다.

## MQOP\_RESUME

오브젝트 핸들에 대한 메시지의 이용 재개.

이 조작이 이벤트 핸들러에 적용되는 경우 일시중단되는 동안 이벤트 핸들러는 이벤트를 가져오지 않고 일시중단 상태인 동안에는 재개될 때 조작에 제공하지 않습니다.

### CallbackDesc

유형: MQCBD - 입력

이는 등록할 때 사용된 애플리케이션 및 옵션에 의해 등록되고 있는 콜백 함수를 식별하는 구조입니다.

구조에 대한 세부사항은 [MQCBD](#)를 참조하십시오.

콜백 디스크립터는 MQOP\_REGISTER 옵션에만 필요합니다. 디스크립터가 필요하지 않은 경우 전달된 매개 변수 주소는 널일 수 있습니다.

### Hobj

유형: MQHOBJ - 입력

이 핸들은 이용될 메시지에서 오브젝트에 설정된 액세스를 나타냅니다. 이는 이전 [MQOPEN](#) 또는 [MQSUB](#) 호출에서 리턴된 핸들입니다(**Hobj** 매개변수에서).

*Hobj*는 이벤트 핸들러 루틴(MQCBT\_EVENT\_HANDLER)을 정의할 때 필요하지 않으며 MQHO\_NONE으로 지정해야 합니다.

*Hobj*가 MQOPEN 호출에서 리턴된 경우 다음 옵션 중 하나 이상으로 큐를 열어야 합니다.

- MQOO\_INPUT\_SHARED
- MQOO\_INPUT\_EXCLUSIVE
- MQOO\_INPUT\_AS\_Q\_DEF
- MQOO\_BROWSE

### MsgDesc

유형: MQMD - 입력

이 구조는 필요한 메시지의 속성 및 검색된 메시지의 속성을 나타냅니다.

**MsgDesc** 매개변수는 이용자에게 필요한 메시지의 속성 및 메시지 이용자에게 전달될 MQMD의 버전을 정의합니다.

MQMD의 *MsgId*, *CorrelId*, *GroupId*, *MsgSeqNumber* 및 *Offset*은 **GetMsgOpts** 매개변수에 지정된 옵션에 따라 메시지 선택에 사용됩니다.

*Encoding* 및 *CodedCharSetId*는 MQGMO\_CONVERT 옵션을 지정하는 경우 메시지 변환에 사용됩니다.

세부사항은 [MQMD](#)를 참조하십시오.

*MsgDesc*는 필드의 기본값이 아닌 값이 필요한 경우 및 MQOP\_REGISTER에 대해 사용됩니다. *MsgDesc*는 이벤트 핸들러에 사용되지 않습니다.

디스크립터가 필요하지 않은 경우 전달된 매개변수 주소는 널일 수 있습니다.

다수 이용자가 중첩된 선택자가 있는 동일 큐에 대해 등록되는 경우 각 메시지에 대해 선택된 이용자가 정의되지 않습니다.

### GetMsgOpts

유형: MQGMO - 입력

**GetMsgOpts** 매개변수는 메시지 이용자가 메시지를 가져오는 방법을 제어합니다. 이 매개변수의 모든 옵션에는 다음을 제외하고 MQGET 호출에서 사용할 때 [362 페이지의 『MQGMO - 메시지 가져오기 옵션』](#)에서 설명된 의미가 있습니다.

#### MQGMO\_SET\_SIGNAL

이 옵션은 허용되지 않습니다.

#### MQGMO\_BROWSE\_FIRST, MQGMO\_BROWSE\_NEXT, MQGMO\_MARK\_\*

찾아보기 이용자에 전달된 메시지 순서는 해당 옵션의 결합으로 결정됩니다. 중요한 결합은 다음과 같습니다.

**MQGMO\_BROWSE\_FIRST**

큐의 첫 번째 메시지는 이용자에게 반복해서 전달됩니다. 이는 이용자가 콜백에서 메시지를 파괴적으로 이용할 때 유용합니다. 이 옵션은 주의해서 사용하십시오.

**MQGMO\_BROWSE\_NEXT**

큐의 종료에 도달할 때까지 현재 커서 위치에서 이용자에게 큐의 각 메시지가 제공됩니다.

**MQGMO\_BROWSE\_FIRST + MQGMO\_BROWSE\_NEXT**

커서는 큐의 시작으로 재설정됩니다. 커서가 큐의 끝에 도달할 때까지 이용자에게 각 메시지가 제공됩니다.

**MQGMO\_BROWSE\_FIRST + MQGMO\_MARK\_\***

큐의 시작부터 이용자에게 큐에서 표시되지 않은 첫 번째 메시지가 제공된 후 이 이용자에 대해 표시됩니다. 이 결합은 이용자가 현재 커서 위치 뒤에 추가된 새 메시지를 수신할 수 있는지 확인합니다.

**MQGMO\_BROWSE\_NEXT + MQGMO\_MARK\_\***

커서 위치의 시작부터 이용자에게 큐에서 표시되지 않은 다음 메시지가 제공된 후 이 이용자에 대해 표시됩니다. 현재 커서 위치 뒤의 큐에 메시지를 추가할 수 있으므로 이 결합을 주의해서 사용하십시오.

**MQGMO\_BROWSE\_FIRST + MQGMO\_BROWSE\_NEXT + MQGMO\_MARK\_\***

이 결합은 허용되지 않습니다. 사용되면 호출이 MQRC\_OPTIONS\_ERROR를 리턴합니다.

**MQGMO\_NO\_WAIT, MQGMO\_WAIT, WaitInterval**

이러한 옵션은 이용자가 호출되는 방법을 제어합니다.

**MQGMO\_NO\_WAIT**

이용자는 MQRC\_NO\_MSG\_AVAILABLE로 호출되지 않습니다. 이용자는 메시지 및 이벤트에 대해서만 호출됩니다.

**0 WaitInterval이 있는 MQGMO\_WAIT**

사용 가능한 메시지가 없고 이용자가 시작되었거나 마지막 "메시지 없음" 이유 코드 이후로 이용자가 하나 이상의 메시지를 전달한 경우 MQRC\_NO\_MSG\_AVAILABLE 코드가 이용자에게 전달됩니다.

이는 0 대기 간격이 지정될 때 이용자가 사용 중인 루프에서 폴링하는 것을 방지합니다.

**MQGMO\_WAIT 및 양의 WaitInterval**

이용자는 이유 코드가 MQRC\_NO\_MSG\_AVAILABLE인 지정된 대기 간격 이후에 호출됩니다. 이 호출은 메시지가 이용자에게 전달되었는지 여부와 관계없이 작성됩니다. 이로 인해 사용자가 하트비트 또는 배치 유형 처리를 수행할 수 있습니다.

**MQGMO\_WAIT 및 MQWI\_UNLIMITED의 WaitInterval**

이는 MQRC\_NO\_MSG\_AVAILABLE을 리턴하기 전에 무한 대기를 지정합니다. 이용자는 MQRC\_NO\_MSG\_AVAILABLE로 호출되지 않습니다.

*GetMsgOpts*는 MQOP\_REGISTER에 대해서만 사용되며 필드에 대해 기본값 이외의 값이 필요한 경우에만 사용됩니다. *GetMsgOpts*는 이벤트 핸들러에 사용되지 않습니다.

*GetMsgOpts*가 필요하지 않은 경우 전달된 매개변수 주소는 널일 수 있습니다. 이 매개변수를 사용하는 것은 MQGMO\_FAIL\_IF\_QUIESCING과 함께 MQGMO\_DEFAULT를 지정하는 것과 같습니다.

메시지 특성 핸들이 MQGMO 구조에서 제공되는 경우 사본이 이용자 콜백에 전달된 MQGMO 구조에서 제공됩니다. MQCB 호출에서 리턴 시 애플리케이션은 메시지 특성 핸들을 삭제할 수 있습니다.

**CompCode**

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_WARNING**

경고(일부 완료).

**MQCC\_FAILED**

호출에 실패했습니다.

## 원인

유형: MQLONG - 출력

다음 목록의 이유 코드는 큐 관리자가 **Reason** 매개변수에 대해 리턴할 수 있는 코드입니다.

*CompCode*이(가) MQCC\_OK인 경우:

### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_FAILED인 경우:

### **MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') 어댑터를 사용할 수 없습니다.

### **MQRC\_ADAPTER\_CONV\_LOAD\_ERROR**

(2133, X'855') 데이터 변환 서비스 모듈을 로드할 수 없습니다.

### **MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

### **MQRC\_API\_EXIT\_ERROR**

(2374, X'946') API 엑시트가 실패했습니다.

### **MQRC\_API\_EXIT\_LOAD\_ERROR**

(2183, X'887') API 엑시트를 로드할 수 없습니다.

### **MQRC\_ASID\_MISMATCH**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

### **MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'7D5') 버퍼 길이 매개변수가 올바르지 않습니다.

### **MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

### **MQRC\_CALLBACK\_LINK\_ERROR**

(2487, X'9B7') 올바르지 않은 콜백 유형 필드.

### **MQRC\_CALLBACK\_NOT\_REGISTERED**

(2448, X'990') 등록된 콜백이 없어 등록 해제, 일시중단 또는 재개할 수 없습니다.

### **MQRC\_CALLBACK\_ROUTINE\_ERROR**

(2486, X'9B6') *CallbackFunction* 또는 *CallbackName*은 지정해야 하지만 둘 다 지정해서는 안됩니다.

### **MQRC\_CALLBACK\_TYPE\_ERROR**

(2483, X'9B3') 올바르지 않은 콜백 유형 필드.

### **MQRC\_CBD\_OPTIONS\_ERROR**

(2484, X'9B4') 올바르지 않은 MQCBD 옵션 필드입니다.

### **MQRC\_CICS\_WAIT\_FAILED**

(2140, X'85C') CICS에서 대기 요청이 거부되었습니다.

### **MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

### **MQRC\_CONNECTION\_NOT\_AUTHORIZED**

(2217, X'8A9') 연결할 권한이 부여되지 않았습니다.

### **MQRC\_CONNECTION\_QUIESCING**

(2202, X'89A') 연결이 정지됩니다.

### **MQRC\_CONNECTION\_STOPPING**

(2203, X'89B') 연결이 종료됩니다.

### **MQRC\_CORREL\_ID\_ERROR**

(2207, X'89F') 상관 ID 오류입니다.

### **MQRC\_DATA\_LENGTH\_ERROR**

(2010, X'7DA') 데이터 길이 매개변수가 올바르지 않습니다.



**MQRC\_ENVIRONMENT\_ERROR**

(2012, X'7DC') 호출이 환경 내에서 유효하지 않습니다.

**MQRC\_FUNCTION\_NOT\_SUPPORTED**

(2298, X'8FA') 요청된 함수는 현재 환경에서 사용할 수 없습니다.

**MQRC\_GET\_INHIBITED**

(2016, X'7E0') 큐에 대해 가져오기가 금지되었습니다.

**MQRC\_GLOBAL\_UOW\_CONFLICT**

(2351, X'92F') 글로벌 작업 단위 충돌입니다.

**MQRC\_GMO\_ERROR**

(2186, X'88A') 메시지 가져오기 옵션 구조가 올바르지 않습니다.

**MQRC\_HANDLE\_IN\_USE\_FOR\_UOW**

(2353, X'931') 글로벌 작업 단위에 대해 사용 중인 핸들입니다.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**MQRC\_HOBJ\_ERROR**

(2019, X'7E3') 오브젝트 핸들이 올바르지 않습니다.

**MQRC\_INCONSISTENT\_BROWSE**

(2259, X'8D3') 찾아보기 지정에 일관성이 없습니다.

**MQRC\_INCONSISTENT\_UOW**

(2245, X'8C5') 작업 단위 지정에 일관성이 없습니다.

**MQRC\_INVALID\_MSG\_UNDER\_CURSOR**

(2246, X'8C6') 커서에 있는 메시지가 검색에 대해 유효하지 않습니다.

**MQRC\_LOCAL\_UOW\_CONFLICT**

(2352, X'930') 글로벌 작업 단위가 로컬 작업 단위와 충돌합니다.

**MQRC\_MATCH\_OPTIONS\_ERROR**

(2247, X'8C7') 일치 옵션이 올바르지 않습니다.

**MQRC\_MAX\_MSG\_LENGTH\_ERROR**

(2485, X'9B4') Incorrect *MaxMsgLength* 필드.

**MQRC\_MD\_ERROR**

(2026, X'7EA') 메시지 디스크립터가 올바르지 않습니다.

**MQRC\_MODULE\_ENTRY\_NOT\_FOUND**

(2497, X'9C1') 지정된 함수 시작점을 모듈에서 찾을 수 없습니다.

**MQRC\_MODULE\_INVALID**

(2496, X'9C0') 모듈을 발견했지만 틀린 유형이 있습니다. 32비트 또는 64비트 또는 유효한 동적 링크 라이브러리가 아닙니다.

**MQRC\_MODULE\_NOT\_FOUND**

(2495, X'9BF') 모듈을 검색 경로에서 찾을 수 없거나 로드 권한이 부여되지 않았습니다.

**MQRC\_MSG\_SEQ\_NUMBER\_ERROR**

(2250, X'8CA') 메시지 순서 번호가 올바르지 않습니다.

**MQRC\_MSG\_TOKEN\_ERROR**

(2331, X'91B') 메시지 토큰 사용이 올바르지 않습니다.

**MQRC\_NO\_MSG\_AVAILABLE**

(2033, X'7F1') 사용 가능한 메시지가 없습니다.

**MQRC\_NO\_MSG\_UNDER\_CURSOR**

(2034, X'7F2') 찾아보기 커서가 메시지에 위치해 있지 않습니다.

**MQRC\_NOT\_OPEN\_FOR\_BROWSE**

(2036, X'7F4') 큐가 읽기 전용으로 열려있지 않습니다.

**MQRC\_NOT\_OPEN\_FOR\_INPUT**

(2037, X'7F5') 큐가 입력을 위해 열려있지 않습니다.

- MQRC\_OBJECT\_CHANGED**  
(2041, X'7F9') 오브젝트가 열린 후에 정의가 변경되었습니다.
- MQRC\_OBJECT\_DAMAGED**  
(2101, X'835') 오브젝트가 손상되었습니다.
- MQRC\_OPERATION\_ERROR**  
(2206, X'89E') API 호출에 대해 올바르지 않은 조작 코드입니다.
- MQRC\_OPTIONS\_ERROR**  
(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.
- MQRC\_PAGESET\_ERROR**  
(2193, X'891') 페이지 세트 데이터 세트에 액세스하는 중에 오류가 발생했습니다.
- MQRC\_Q\_DELETED**  
(2052, X'804') 큐가 삭제되었습니다.
- MQRC\_Q\_INDEX\_TYPE\_ERROR**  
(2394, X'95A') 큐의 색인 유형이 올바르지 않습니다.
- MQRC\_Q\_MGR\_NAME\_ERROR**  
(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.
- MQRC\_Q\_MGR\_NOT\_AVAILABLE**  
(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.
- MQRC\_Q\_MGR QUIESCING**  
(2161, X'871') 큐 관리자가 정지됩니다.
- MQRC\_Q\_MGR\_STOPPING**  
(2162, X'872') 큐 관리자가 종료되었습니다.
- MQRC\_RESOURCE\_PROBLEM**  
(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.
- MQRC\_SIGNAL\_OUTSTANDING**  
(2069, X'815') 이 핸들의 미해결 신호.
- MQRC\_STORAGE\_NOT\_AVAILABLE**  
(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.
- MQRC\_SUPPRESSED\_BY\_EXIT**  
(2109, X'83D') 엑시트 프로그램에서 호출이 억제되었습니다.
- MQRC\_SYNCPOINT\_LIMIT\_REACHED**  
(2024, X'7E8') 현재 작업 단위 내에서 더 이상 메시지를 핸들링할 수 없습니다.
- MQRC\_SYNCPOINT\_NOT\_AVAILABLE**  
(2072, X'818') 동기점 지원을 사용할 수 없습니다.
- MQRC\_UNEXPECTED\_ERROR**  
(2195, X'893') 예상치 못한 오류가 발생했습니다.
- MQRC\_UOW\_ENLISTMENT\_ERROR**  
(2354, X'932') 글로벌 작업 단위에서의 등록이 실패했습니다.
- MQRC\_UOW\_MIX\_NOT\_SUPPORTED**  
(2355, X'933') 작업 단위 호출의 혼합은 지원되지 않습니다.
- MQRC\_UOW\_NOT\_AVAILABLE**  
(2255, X'8CF') 사용할 큐 관리자에 대해 작업 단위를 사용할 수 없습니다.
- MQRC\_WAIT\_INTERVAL\_ERROR**  
(2090, X'82A') MQGMO의 대기 간격이 올바르지 않습니다.
- MQRC\_WRONG\_GMO\_VERSION**  
(2256, X'8D0') 올바르지 않은 버전의 MQGMO가 제공되었습니다.
- MQRC\_WRONG\_MD\_VERSION**  
(2257, X'8D1') 올바르지 않은 버전의 MQMD가 제공되었습니다.
- 이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

## 사용법 참고

1. MQCB는 큐에서 사용 가능하고 지정된 기준과 일치하는, 각 메시지에 호출할 조치를 정의하는 데 사용됩니다. 조치가 처리되면 메시지가 큐에서 제거되고 정의된 메시지 이용자에게 전달되거나 메시지 토큰이 제공되며 이는 메시지를 검색하는 데 사용됩니다.
2. MQCB는 MQCTL로 이용을 시작하기 전에 콜백 루틴을 정의할 때 사용할 수 있거나 콜백 루틴으로부터 사용될 수 있습니다.
3. 콜백 루틴의 외부에서 MQCB를 사용하려면 먼저 MQCTL을 사용하여 메시지 이용을 일시중단한 후 이용을 재개해야 합니다.
4. MQCB는 IMS 어댑터 내에서 지원되지 않습니다.

## 메시지 이용자 콜백 순서

이용자의 라이프사이클 동안 키 포인트에서 콜백을 호출하도록 이용자를 구성할 수 있습니다. 예를 들면, 다음과 같습니다.

- 이용자가 먼저 등록하는 경우
- 연결이 시작된 경우
- 연결이 중지된 경우
- 이용자가 MQCLOSE에 의해 명시적 또는 내재적으로 등록 취소되는 경우

표 541. MQCTL 동사 정의	
동사	의미
MQCTL(START)	MQOP_START 조작을 사용하는 MQCTL 호출
MQCTL(STOP)	MQOP_STOP 조작을 사용하는 MQCTL 호출
MQCTL(WAIT)	MQOP_START_WAIT 조작을 사용하는 MQCTL 호출

이는 이용자가 이용자와 연관된 상태를 유지보수하도록 허용합니다. 호출이 애플리케이션에서 요청될 때 이용자 호출을 위한 규칙은 다음과 같습니다.

### REGISTER

항상 콜백의 첫 번째 호출 유형입니다.

항상 MQCB(REGISTER) 호출과 동일한 스레드에서 호출됩니다.

### 시작

항상 MQCTL(START) verb와 함께 동기적으로 호출됩니다.

- 모두 START 콜백은 MQCTL(START) verb 리턴 전에 완료됩니다.

THREAD\_AFFINITY가 요청되면 메시지 전달과 동일한 스레드에 있습니다.

예를 들어, 이전 콜백이 MQCTL(START) 동안 MQCTL(STOP)을 발행하면 start를 사용한 호출은 보증되지 않습니다.

### STOP

연결이 재시작될 때까지 이 호출 이후에 전달된 추가 메시지 또는 이벤트가 없습니다.

애플리케이션이 START, 메시지 또는 이벤트에 대해 이전에 호출된 경우 STOP은 보증됩니다.

### DEREGISTER

항상 콜백의 마지막 호출 유형입니다.

애플리케이션이 스레드 기반 초기화 및 정리를 START 및 STOP 호출에서 수행하는지 확인하십시오. REGISTER 및 DEREGISTER 콜백과 함께 비스레드 기반 초기화 및 정리를 수행할 수 있습니다.

명시된 사항 이외에 스레드의 수명 및 가용성에 대한 가정을 작성하지 마십시오. 예를 들어, DEREGISTER에 대한 마지막 호출을 넘어 남아 있는 스레드에 의존하지 마십시오. 마찬가지로 THREAD\_AFFINITY를 사용하도록 선택하지 않은 경우 연결이 시작될 때마다 스레드가 존재한다고 가정하지 마십시오.

애플리케이션에 스레드 특성에 대한 특별한 요구사항이 있는 경우 항상 스레드를 작성한 후 MQCTL(WAIT)을 사용할 수 있습니다. 이는 비동기 메시지 전달을 위해 스레드를 IBM MQ에 제공하는 효과를 가집니다.

## 메시지 이용자 연결 사용법

이용자의 라이프사이클 동안 키 포인트에서 콜백을 호출하도록 이용자를 구성할 수 있습니다. 예를 들면, 다음과 같습니다.

- 이용자가 먼저 등록하는 경우
- 연결이 시작된 경우
- 연결이 중지된 경우
- 이용자가 MQCLOSE에 의해 명시적 또는 내재적으로 등록 취소되는 경우

표 542. MQCTL 동사 정의	
동사	의미
MQCTL(START)	MQOP_START 조작을 사용하는 MQCTL 호출
MQCTL(STOP)	MQOP_STOP 조작을 사용하는 MQCTL 호출
MQCTL(WAIT)	MQOP_START_WAIT 조작을 사용하는 MQCTL 호출

이는 이용자가 이용자와 연관된 상태를 유지보수하도록 허용합니다. 호출이 애플리케이션에서 요청될 때 이용자 호출을 위한 규칙은 다음과 같습니다.

### REGISTER

항상 콜백의 첫 번째 호출 유형입니다.

항상 MQCB(REGISTER) 호출과 동일한 스레드에서 호출됩니다.

#### 시작

항상 MQCTL(START) verb와 함께 동기적으로 호출됩니다.

- 모두 START 콜백은 MQCTL(START) verb 리턴 전에 완료됩니다.

THREAD\_AFFINITY가 요청되면 메시지 전달과 동일한 스레드에 있습니다.

예를 들어, 이전 콜백이 MQCTL(START) 동안 MQCTL(STOP)을 발행하면 start를 사용한 호출은 보증되지 않습니다.

### STOP

연결이 재시작될 때까지 이 호출 이후에 전달된 추가 메시지 또는 이벤트가 없습니다.

애플리케이션이 START, 메시지 또는 이벤트에 대해 이전에 호출된 경우 STOP은 보증됩니다.

### DEREGISTER

항상 콜백의 마지막 호출 유형입니다.

애플리케이션이 스레드 기반 초기화 및 정리를 START 및 STOP 호출에서 수행하는지 확인하십시오. REGISTER 및 DEREGISTER 콜백과 함께 비스레드 기반 초기화 및 정리를 수행할 수 있습니다.

명시된 사항 이외에 스레드의 수명 및 가용성에 대한 가정을 작성하지 마십시오. 예를 들어, DEREGISTER에 대한 마지막 호출을 넘어 남아 있는 스레드에 의존하지 마십시오. 마찬가지로 THREAD\_AFFINITY를 사용하도록 선택하지 않은 경우 연결이 시작될 때마다 스레드가 존재한다고 가정하지 마십시오.

애플리케이션에 스레드 특성에 대한 특별한 요구사항이 있는 경우 항상 스레드를 작성한 후 MQCTL(WAIT)을 사용할 수 있습니다. 이는 비동기 메시지 전달을 위해 스레드를 IBM MQ에 제공하는 효과를 가집니다.

## C 호출

```
MQCB (Hconn, Operation, CallbackDesc, Hobj, MsgDesc,
GetMsgOpts, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```

MQHCONN Hconn;          /* Connection handle */
MQLONG  Operation;     /* Operation being processed */
MQCBD   CallbackDesc;  /* Callback descriptor */
MQHOBJ  HObj;          /* Object handle */
MQMD    MsgDesc        /* Message descriptor attributes */
MQGMO   GetMsgOpts     /* Message options */
MQLONG  CompCode;      /* Completion code */
MQLONG  Reason;        /* Reason code qualifying CompCode */

```

## COBOL 호출

```

CALL 'MQCB' USING HCONN, OPERATION, CBDESC, HOBJ, MSGDESC,
                GETMSGOPTS, COMPCODE, REASON.

```

매개변수를 다음과 같이 선언합니다.

```

** Connection handle
01 HCONN      PIC S9(9) BINARY.
** Operation
01 OPERATION  PIC S9(9) BINARY.
** Callback Descriptor
01 CBDESC     COPY CMQCBDV.
01 HOBJ       PIC S9(9) BINARY.
** Message Descriptor
01 MSGDESC    COPY CMQMDV.
** Get Message Options
01 GETMSGOPTS COPY CMQGMV.
** Completion code
01 COMPCODE   PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON     PIC S9(9) BINARY.

```

## PL/I 호출

```

call MQCB(Hconn, Operation, CallbackDesc, Hobj, MsgDesc, GetMsgOpts,
          CompCode, Reason)

```

매개변수를 다음과 같이 선언합니다.

```

dcl Hconn      fixed bin(31); /* Connection handle */
dcl Operation  fixed bin(31); /* Operation */
dcl CallbackDesc like MQCBD;  /* Callback Descriptor */
dcl Hobj       fixed bin(31); /* Object Handle */
dcl MsgDesc    like MQMD;     /* Message Descriptor */
dcl GetMsgOpts like MQGMO;    /* Get Message Options */
dcl CompCode   fixed bin(31); /* Completion code */
dcl Reason     fixed bin(31); /* Reason code qualifying CompCode */

```

## MQCB\_FUNCTION - 콜백 함수

MQCB\_FUNCTION 함수 호출은 이벤트 핸들링 및 비동기 메시지 이용을 위한 콜백 함수입니다.

MQCB\_FUNCTION 호출 정의는 콜백 함수에 전달되는 매개변수를 설명하기 위해 단독으로 제공됩니다. 큐 관리자가 제공한 MQCB\_FUNCTION이라고 하는 시작점이 없습니다.

호출되는 실제 함수의 스펙은 [MQCB](#) 호출에 대한 입력이며 [MQCBD](#) 구조를 통해 전달됩니다.

## 구문

```

MQCB_FUNCTION(Hconn, MsgDesc, GetMsgOpts, Buffer, Context)

```

## 매개변수

### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *Hconn*의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다. z/OS for CICS 애플리케이션에서 MQCONN 호출을 생략할 수 있으며 Hconn에 대해 다음 값을 지정할 수 있습니다.

### MQHC\_DEF\_CONN

기본 연결 핸들

### MsgDesc

유형: MQMD - 입력

이 구조는 검색된 메시지의 속성을 설명합니다.

자세한 내용은 412 페이지의 『MQMD - 메시지 디스크립터』의 내용을 참조하십시오.

전달된 MQMD의 버전은 사용자 함수를 정의한 MQCB 호출에서 전달된 버전과 동일한 버전입니다.

버전 4 MQGMO가 MQMD 대신 메시지 핸들을 리턴하도록 요청하는 데 사용된 경우 MQMD의 주소는 널 문자로 전달됩니다.

이 필드는 메시지 사용자 함수의 입력 필드입니다. 이벤트 핸들러 함수와는 관련이 없습니다.

### GetMsgOpts

유형: MQGMO - 입력

메시지 이용자의 조치를 제어하는 데 옵션이 사용됩니다. 이 매개변수는 또한 리턴된 메시지에 대한 추가 정보를 포함합니다.

세부사항은 MQGMO를 참조하십시오.

전달된 MQGMO의 버전은 지원되는 최신 버전입니다.

이 필드는 메시지 사용자 함수의 입력 필드입니다. 이벤트 핸들러 함수와는 관련이 없습니다.

### 버퍼

유형: MQBYTExBufferLength - 입력

이는 메시지 데이터가 포함된 영역입니다.

이 호출에 사용 가능한 메시지가 없거나 메시지가 메시지 데이터를 포함하지 않는 경우 *Buffer*의 주소는 널로 전달됩니다.

이 필드는 메시지 사용자 함수의 입력 필드입니다. 이벤트 핸들러 함수와는 관련이 없습니다.

### 컨텍스트

유형: MQCBC - 입출력(I/O)

이 구조는 콜백 함수에 컨텍스트 정보를 제공합니다. 자세한 내용은 279 페이지의 『MQCBC - 콜백 컨텍스트』의 내용을 참조하십시오.

## 사용법 참고

1. 콜백 루틴이 스레드를 지연하거나 차단할 수 있는 서비스를 사용하는 경우(예를 들어, 대기하는 MQGET) 기타 콜백의 디스패치를 지연시킬 수 있음에 주의하십시오.
2. 콜백 루틴의 각 호출에 대해 별도의 작업 단위가 자동으로 설정되지 않으므로 작업의 논리 배치가 처리될 때까지 루틴이 커밋 호출을 발행하거나 커밋을 지연시킬 수 있습니다. 작업 배치가 커밋될 때 이는 마지막 동기점 이후 호출된 모든 콜백 함수의 메시지를 커밋합니다.
3. CICS LINK 또는 CICS START가 호출하는 프로그램은 채널 컨테이너로 알려진 이름 지정된 오브젝트를 통해 CICS 서비스를 사용하여 매개변수를 검색합니다. 컨테이너 이름은 매개변수 이름과 동일합니다. 자세한 정보는 CICS 문서를 참조하십시오.
4. 콜백 루틴이 MQDISC 호출을 발행할 수 있지만 고유한 연결에는 발행할 수 없습니다. 예를 들어, 콜백 루틴이 연결을 작성한 경우 연결을 끊을 수도 있습니다.

5. 콜백 루틴은 일반적으로 매번 동일한 스레드로부터 호출되는 것에 의존해서는 안됩니다. 필요한 경우 연결이 시작될 때 MQCTLO\_THREAD\_AFFINITY를 사용하십시오.
6. 콜백 루틴이 0이 아닌 이유 코드를 수신하는 경우 적절한 조치를 수행해야 합니다.
7. MQCB\_FUNCTION은 IMS 어댑터 내에서 지원되지 않습니다.

## MQCLOSE - 오브젝트 닫기

MQCLOSE 호출은 오브젝트에 대한 액세스를 철회하며 MQOPEN 및 MQSUB 호출의 역호출입니다.

### 구문

MQCLOSE(*Hconn*, *Hobj*, *Options*, *CompCode*, *Reason*)

### 매개변수

#### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *Hconn*의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

CICS 용 z/OS 애플리케이션에서 MQCONN 호출을 생략하고 *Hconn*에 대해 다음 값을 지정할 수 있습니다.

#### MQHC\_DEF\_HCONN

기본 연결 핸들

#### Hobj

유형: MQHOBJ - 입출력(I/O)

이 핸들은 처리완료되고 있는 오브젝트를 나타냅니다. 오브젝트는 다음 유형 중 하나일 수 있습니다. *Hobj*의 값이 이전 MQOPEN 호출로 리턴되었습니다.

호출의 정상 완료 시 큐 관리자는 환경에 대한 올바른 핸들이 아닌 값에 이 매개변수를 설정합니다. 이 값은 다음과 같습니다.

#### MQHO\_UNUSABLE\_HOBJ

사용 불가능한 오브젝트 핸들.

z/OS에서 *Hobj*는 정의되지 않은 값으로 설정됩니다.

### 옵션

유형: MQLONG - 입력

이 매개변수는 오브젝트를 닫는 방법을 제어합니다.

보유하거나 삭제해야 하므로 영구적 동적 큐 및 구독만 둘 이상의 방법으로 닫을 수 있습니다. 이는 값 MQQDT\_PERMANENT\_DYNAMIC을 가진 **DefinitionType** 속성이 있는 큐입니다(804 페이지의 『[큐의 속성](#)』에서 설명하는 **DefinitionType** 속성 참조). 닫기 옵션은 이 토픽에서 요약됩니다.

지속 가능 구독을 보존하거나 제거할 수 있습니다. 이는 MQSO\_DURABLE 옵션으로 MQSUB 호출을 사용하여 작성됩니다.

관리 대상(MQSO\_MANAGED 옵션을 사용하여 MQSUB 호출에서 리턴된 **Hobj** 매개변수임)에 대해 핸들을 닫을 때 큐 관리자는 연관된 구독도 제거될 때 검색되지 않은 발행을 정리합니다. 구독은 MQSUB 호출에서 리턴된 **Hsub** 매개변수의 MQCO\_REMOVE\_SUB 옵션을 사용하여 제거됩니다. MQCO\_REMOVE\_SUB는 비지속 가능 구독에 대한 MQCLOSE의 기본 작동임을 참고하십시오.

비관리 목적지에 대한 핸들을 닫을 때 발행물이 전송된 큐를 정리할 책임이 있습니다. 먼저 MQCO\_REMOVE\_SUB를 사용하여 구독을 닫고 어떤 항목도 남지 않을 때까지 큐에서 메시지를 처리하십시오.

다음에서 하나의 옵션만 지정해야 합니다.

**동적 큐 옵션:** 이러한 옵션은 영구적 동적 큐를 닫는 방법을 제어합니다.

## MQCO\_DELETE

다음 중 하나가 참이면 큐는 삭제됩니다.

- 이는 이전 MQOPEN 호출로 작성된 영구적 동적 큐이며 큐에 메시지가 없고 큐에 대해 미결해 상태인 커밋하지 않은 가져오기 또는 넣기 요청이 없습니다(현재 태스크 또는 기타 태스크에 대해).
- 이는 *Hobj*를 리턴한 MQOPEN 호출로 작성된 임시 동적 큐입니다. 이 경우 큐의 모든 메시지는 제거됩니다.

*Hobj*가 MQSUB 호출에서 리턴된 경우를 포함한 다른 모든 경우에서 이유 코드 MQRC\_OPTION\_NOT\_VALID\_FOR\_TYPE과 함께 호출이 실패하고 오브젝트가 삭제되지 않습니다.

z/OS에서 큐가 논리적으로 삭제된 동적 큐이고 마지막 핸들인 경우 큐는 물리적으로 삭제됩니다. 추가적인 세부사항은 632 페이지의 『사용법 참고』의 내용을 참조하십시오.

## MQCO\_DELETE\_PURGE

다음 중 하나가 참인 경우 큐가 삭제되고 큐의 메시지가 제거됩니다.

- 이는 이전 MQOPEN 호출로 작성된 영구적 동적 큐이며 큐에 대해 미결해 상태인 커밋하지 않은 가져오기 또는 넣기 요청이 없습니다(현재 태스크 또는 기타 태스크에 대해).
- 이는 *Hobj*를 리턴한 MQOPEN 호출로 작성된 임시 동적 큐입니다.

*Hobj*가 MQSUB 호출에서 리턴된 경우를 포함한 다른 모든 경우에서 이유 코드 MQRC\_OPTION\_NOT\_VALID\_FOR\_TYPE과 함께 호출이 실패하고 오브젝트가 삭제되지 않습니다.

큐의 오브젝트 유형	MQCO_NONE	MQCO_DELETE	MQCO_DELETE_PURGE
큐 이외의 오브젝트	보유됨	올바르지 않음	올바르지 않음
사전 정의된 큐	보유됨	올바르지 않음	올바르지 않음
영구적 동적 큐	보유됨	비어 있고 보류 중인 업데이트가 없는 경우 삭제	메시지가 삭제됨, 보류 중인 업데이트가 없는 경우 큐가 삭제됨
임시 동적 큐(큐의 작성자가 발행한 호출)	삭제됨	삭제됨	삭제됨
임시 동적 큐(큐의 작성자가 발행하지 않은 호출)	보유됨	올바르지 않음	올바르지 않음
분배 목록	보유됨	올바르지 않음	올바르지 않음
관리되는 구독 목적지	보유됨	올바르지 않음	올바르지 않음
분배 목록(구독이 제거됨)	메시지가 삭제됨, 큐가 삭제됨	올바르지 않음	올바르지 않음

**구독 닫기 옵션:** 이 옵션은 핸들이 닫힌 경우 지속 가능 구독이 제거되는지 여부와 애플리케이션이 읽도록 여전히 대기 중인 발행이 정리되는지 여부를 제어합니다. 이러한 옵션은 MQSU 호출의 **Hsub** 매개변수에서 리턴된 오브젝트 핸들과 함께 사용하는 경우 올바릅니다.

## MQCO\_KEEP\_SUB

구독에 대한 핸들은 닫히지만 작성된 구독은 유지됩니다. 발행이 구독에서 지정된 대상에 계속 전송됩니다. 이 옵션은 구독이 MQSO\_DURABLE 옵션으로 작성된 경우에만 유효합니다.

구독이 지속 가능한 경우 MQCO\_KEEP\_SUB는 기본값입니다.

## MQCO\_REMOVE\_SUB

구독이 제거되고 구독에 대한 핸들이 닫힙니다.

MQSUB 호출의 **Hobj** 매개변수는 **Hsub** 매개변수의 닫기로 무효화되지 않으며 나머지 발행을 수신하기 위해 MQGET 또는 MQCB에 대해 계속 사용할 수 있습니다. MQSUB 호출의 **Hobj** 매개변수도 닫힌 경우 관리 대상이면 검색되지 않은 발행이 제거됩니다.



구독이 비지속 가능한 경우 MQCO\_REMOVE\_SUB는 기본값입니다.

MQCO\_REMOVE\_SUB의 완료는 조치가 완료됨을 의미하지 않습니다. 이 호출이 완료되었는지 확인하려면 분산 네트워크에 대한 비동기 명령이 완료되었는지 확인의 DELETE SUB 단계를 참조하십시오.

이러한 구독 닫기 옵션은 다음 테이블에서 요약됩니다.

표 544. 지속 가능한 구독 핸들을 닫지만 구독은 보유하는 옵션	
태스크	구독 클로저 옵션
MQOPENed 핸들에서 발행물 유지	MQCO_KEEP_SUB
MQOPENed 핸들에서 발행물 제거	조치가 허용되지 않음
MQSO_MANAGED 핸들에서 발행 보존	MQCO_KEEP_SUB
MQSO_MANAGED 핸들에서 발행 제거	조치가 허용되지 않음

지속 가능 구독 핸들을 닫아 구독을 해제하거나 지속 불가능한 구독 핸들을 닫아 구독을 해제하려면 다음 클로저 옵션을 사용하십시오.

표 545. 구독 해제 옵션	
태스크	구독 클로저 옵션
MQOPENed 핸들에서 발행물 유지	MQCO_REMOVE_SUB
MQOPENed 핸들에서 발행물 제거	조치가 허용되지 않음
MQSO_MANAGED 핸들에서 발행 보존	MQCO_REMOVE_SUB

**미리 읽기 옵션:** 다음 옵션은 애플리케이션이 요청하기 전에 클라이언트에 전송했고 애플리케이션이 아직 사용하지 않은 비지속 메시지에 대해 발생하는 사항을 제어합니다. 이 메시지는 MQCLOSE가 완료되기 전에 애플리케이션에서 요청하도록 대기 중인 클라이언트 미리 읽기 버퍼에 저장되고 큐에서 제거되거나 이용됩니다.

#### MQCO\_IMMEDIATE

이 오브젝트는 즉시 닫히며 요청된 애플리케이션이 제거되기 전 클라이언트에 전송한 메시지는 애플리케이션에서 사용할 수 없습니다. 이는 기본값입니다.

#### MQCO\_QUIESCE

오브젝트를 닫기 위한 요청이 작성되었지만 애플리케이션이 요청하기 전에 클라이언트에 전송된 메시지가 클라이언트 미리 읽기 버퍼에 아직 상주하는 경우 MQCLOSE 호출이 MQRC\_READ\_AHEAD\_MSGS의 경고와 함께 리턴되고 오브젝트 핸들이 유효한 상태로 남아 있습니다.

그런 다음 애플리케이션은 메시지를 검색하기 위해 오브젝트 핸들을 더 이상 사용할 수 없을 때까지 계속 사용한 후 오브젝트를 다시 닫을 수 있습니다. 애플리케이션이 요청하기 전에 클라이언트에 추가 메시지가 전송되지 않으며 이제 미리 읽기는 꺼집니다.

메시지가 마지막 MQGET 호출 및 MQCO\_IMMEDIATE가 사용된 경우 제거되는 다음 MQCLOSE 사이에 도달할 수 있으므로 애플리케이션은 클라이언트 미리 읽기 버퍼에서 추가 메시지가 없는 지점에 도달하려고 시도하는 대신 MQCO\_QUIESCE를 사용하는 것이 좋습니다.

비동기 콜백 함수 내에서 MQCO\_QUIESCE가 있는 MQCLOSE가 발행되면 메시지 동일한 동작의 메시지 미리 읽기가 적용됩니다. 경고 MQRC\_READ\_AHEAD\_MSGS가 리턴되면 콜백 함수가 한 번 이상 호출됩니다. 미리 읽은 마지막 남은 메시지가 콜백 함수에 전달된 경우 MQCBC ConsumerFlags 필드가 MQCBCF\_READA\_BUFFER\_EMPTY로 설정됩니다.

**기본 옵션:** 이전에 설명된 옵션 중 필요한 옵션이 없는 경우 다음 옵션을 사용할 수 있습니다.

#### MQCO\_NONE

선택적 닫기 처리가 필요하지 않습니다.

이는 다음에 대해 지정되어야 합니다.

- 큐 이외의 오브젝트

- 사전 정의된 큐
- 임시 동적 큐(*Hobj*가 큐를 작성한 MQOPEN 호출로 리턴된 핸들이 아닌 경우에만)
- 분배 목록

위의 모든 경우 오브젝트가 보유하고 삭제되지 않습니다.

이 옵션이 임시 동적 큐에 지정된 경우:

- *Hobj*를 리턴한 MQOPEN 호출에 의해 작성된 경우 큐는 삭제됩니다. 큐에 있는 메시지도 제거됩니다.
- 다른 모든 경우에 큐(및 큐의 메시지)가 유지됩니다.

이 옵션이 영구적 동적 큐에 지정되는 경우 큐는 유지되고 삭제되지 않습니다.

z/OS에서 큐가 논리적으로 삭제된 동적 큐이고 마지막 핸들인 경우 큐는 물리적으로 삭제됩니다. 추가적인 세부사항은 632 페이지의 『사용법 참고』의 내용을 참조하십시오.

### CompCode

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

#### **MQCC\_OK**

정상적으로 완료되었습니다.

#### **MQCC\_WARNING**

경고(일부 완료).

#### **MQCC\_FAILED**

호출에 실패했습니다.

### 원인

유형: MQLONG - 출력

나열된 이유 코드는 큐 관리자가 **Reason** 매개변수에 대해 리턴할 수 있는 코드입니다.

*CompCode*이(가) MQCC\_OK인 경우:

#### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_WARNING인 경우:

#### **MQRC\_INCOMPLETE\_GROUP**

(2241, X'8C1') 메시지 그룹이 완료되지 않았습니다.

#### **MQRC\_INCOMPLETE\_MSG**

(2242, X'8C2') 논리 메시지가 완료되지 않았습니다.

#### **MQRC\_READ\_AHEAD\_MSGS**

(nnnn, X'xxx') 클라이언트에 애플리케이션이 아직 이용하지 않은 미리 읽기 메시지가 있습니다.

*CompCode*이(가) MQCC\_FAILED인 경우:

#### **MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') 어댑터를 사용할 수 없습니다.

#### **MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

#### **MQRC\_API\_EXIT\_ERROR**

(2374, X'946') API 엑시트가 실패했습니다.

#### **MQRC\_API\_EXIT\_LOAD\_ERROR**

(2183, X'887') API 엑시트를 로드할 수 없습니다.

#### **MQRC\_ASID\_MISMATCH**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

#### **MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**MQRC\_CF\_NOT\_AVAILABLE**

(2345, X'929') 커플링 기능이 사용 불가능합니다.

**MQRC\_CF\_STRUC\_FAILED**

(2373, X'945') 커플링 기능 구조가 실패했습니다.

**MQRC\_CF\_STRUC\_IN\_USE**

(2346, X'92A') 커플링 기능 구조가 사용 중입니다.

**MQRC\_CICS\_WAIT\_FAILED**

(2140, X'85C') CICS에서 대기 요청이 거부되었습니다.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**MQRC\_CONNECTION\_NOT\_AUTHORIZED**

(2217, X'8A9') 연결할 권한이 부여되지 않았습니다.

**MQRC\_CONNECTION\_STOPPING**

(2203, X'89B') 연결이 종료됩니다.

**MQRC\_DB2\_NOT\_AVAILABLE**

(2342, X'926') Db2 서브시스템이 사용 불가능합니다.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**MQRC\_HOBJ\_ERROR**

(2019, X'7E3') 오브젝트 핸들이 올바르지 않습니다.

**MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

**MQRC\_OBJECT\_DAMAGED**

(2101, X'835') 오브젝트가 손상되었습니다.

**MQRC\_OPTION\_NOT\_VALID\_FOR\_TYPE**

(2045, X'7FD') MQOPEN 또는 MQCLOSE 호출에서 옵션이 오브젝트 유형에 대해 유효하지 않습니다.

**MQRC\_OPTIONS\_ERROR**

(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.

**MQRC\_PAGESET\_ERROR**

(2193, X'891') 페이지 세트 데이터 세트에 액세스하는 중에 오류가 발생했습니다.

**MQRC\_Q\_MGR\_NAME\_ERROR**

(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**

(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') 큐 관리자가 종료되었습니다.

**MQRC\_Q\_NOT\_EMPTY**

(2055, X'807') 큐에 하나 이상의 메시지 또는 커밋되지 않은 Put 또는 Get 요청이 있습니다.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**MQRC\_SECURITY\_ERROR**

(2063, X'80F') 보안 오류가 발생했습니다.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**MQRC\_SUPPRESSED\_BY\_EXIT**

(2109, X'83D') 엑시트 프로그램에서 호출이 억제되었습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

## 사용법 참고

1. 애플리케이션이 MQDISC 호출을 발행하거나 정상적으로 또는 비정상적으로 종료되는 경우 애플리케이션이 열었고 아직 열려 있는 오브젝트가 MQCO\_NONE 옵션으로 자동으로 닫힙니다.
2. 처리완료되는 오브젝트가 큐인 경우 다음 사항이 적용됩니다.

- 큐의 조작이 작업 단위의 부분으로 수행되는 경우 동기점의 결과에 영향을 주지 않고 동기점이 발생하기 전 또는 후에 큐를 닫을 수 있습니다. 큐가 트리거되는 경우 큐를 닫기 전에 롤백을 수행하면 트리거 메시지를 발행할 수 있습니다. 트리거 메시지에 대한 자세한 정보는 [트리거 메시지의 특성을 참조하십시오](#).
- 큐가 MQOO\_BROWSE 옵션으로 열린 경우 찾아보기 커서는 영구 삭제됩니다. 큐가 MQOO\_BROWSE 옵션으로 다시 열리는 경우 새 찾아보기 커서가 작성됩니다(MQOO\_BROWSE 참조).
- 메시지가 현재 MQCLOSE 호출 시 이 핸들에 대해 잠기는 경우 잠금이 해제됩니다(MQGMO\_LOCK 참조).
- z/OS에서 닫히는 큐 핸들에 대해 MQGMO\_SET\_SIGNAL 옵션을 사용한 미해결된 MQGET 요청이 있는 경우 해당 요청은 취소됩니다(MQGMO\_SET\_SIGNAL 참조). 동일한 큐이지만 서로 다른 핸들(Hobj)에 대해 제출된 신호 요청은 동적 큐가 삭제되지 않는 한 영향을 받지 않습니다. 동적 큐가 삭제되면 이 또한 취소됩니다.

3. 처리완료되는 오브젝트가 동적 큐인 경우 다음 사항이 적용됩니다(영구적 또는 임시).

- 동적 큐에 대해 해당 MQOPEN 호출에서 지정된 옵션에 상관없이 MQCO\_DELETE 및 MQCO\_DELETE\_PURGE 옵션을 지정할 수 있습니다.
- 동적 큐가 삭제되면 큐에 대해 미해결인 MQGMO\_WAIT 옵션이 있는 모든 MQGET 호출이 취소되고 이유 코드 MQRC\_Q\_DELETED가 리턴됩니다. [MQGMO\\_WAIT](#)를 참조하십시오.

애플리케이션이 삭제된 큐에 액세스할 수 없는 경우에도 큐는 시스템에서 제거되지 않고 큐를 참조하는 모든 핸들이 닫히고 큐에 영향을 미치는 모든 작업 단위가 커밋되거나 백아웃될 때까지 연관된 자원은 사용 가능하지 않습니다.

z/OS에서 논리적으로 삭제되었지만 아직 시스템에서 제거되지 않은 큐는 삭제된 큐와 동일한 이름이 있는 새 큐의 생성을 막습니다. 이 경우 이유 코드 MQRC\_NAME\_IN\_USE와 함께 MQOPEN 호출이 실패합니다. 또한 해당 큐는 애플리케이션이 액세스할 수 없는 경우에도 MQSC 명령을 사용하여 표시될 수 있습니다.

- 영구적 동적 큐가 삭제될 때 MQCLOSE 호출에서 지정된 Hobj 핸들이 큐를 작성한 MQOPEN 호출로 리턴된 핸들이 아닌 경우, MQOPEN 호출을 유효성 검증하는 데 사용된 사용자 ID에 큐를 삭제할 권한이 부여되었는지의 검사가 수행됩니다. MQOO\_ALTERNATE\_USER\_AUTHORITY 옵션이 MQOPEN 호출에서 지정된 경우 검사된 사용자 ID는 *AlternateUserId*입니다.

다음의 경우 이 검사는 수행되지 않습니다.

- 지정된 핸들은 큐를 작성한 MQOPEN 호출로 리턴된 핸들입니다.
- 삭제되고 있는 큐는 임시 동적 큐입니다.
- 임시 동적 큐가 닫힐 때 MQCLOSE 호출에서 지정된 Hobj 핸들이 큐를 작성한 MQOPEN 호출로 리턴된 핸들인 경우 큐가 삭제됩니다. 이는 MQCLOSE 호출에 지정된 가까운 옵션과 관계없이 발생합니다. 큐에 메시지가 있는 경우 제거되며 보고 메시지가 생성되지 않습니다.

큐에 영향을 주는 커밋되지 않은 작업 단위가 있는 경우 큐 및 해당 메시지가 삭제되지만 작업 단위를 실패하지 않습니다. 그러나 이전에 설명된 것처럼 각 작업 단위가 작업이 커밋되거나 백아웃될 때까지 작업 단위와 연관된 자원은 무효가 아닙니다.

4. 처리완료되는 오브젝트가 분배 목록인 경우 다음 사항이 적용됩니다.

- 분배 목록에 대한 유일하게 올바른 닫기 옵션은 MQCO\_NONE입니다. 다른 옵션이 지정되면 이유 코드 MQRC\_OPTIONS\_ERROR 또는 MQRC\_OPTION\_NOT\_VALID\_FOR\_TYPE과 함께 호출이 실패합니다.
- 분배 목록이 닫히면 개별 완료 코드 및 이유 코드가 목록의 큐에 대해 리턴되지 않습니다. 호출의 **CompCode** 및 **Reason** 매개변수만 진단을 위해 사용 가능합니다.

큐 중 하나를 처리완료하는 중에 실패가 발생하는 경우 큐 관리자는 처리를 계속하고 분배 목록에 남아있는 큐를 처리완료하기 위해 시도합니다. 호출의 **CompCode** 및 **Reason** 매개변수는 실패를 설명하는 정보를 리턴하도록 설정됩니다. 대부분의 큐가 성공적으로 닫혀도 완료 코드가 MQCC\_FAILED일 수 있습니다. 오류가 발생한 큐는 식별되지 않습니다.

둘 이상의 큐에 실패가 있는 경우 **CompCode** 및 **Reason** 매개변수에서 보고되는 실패가 정의되지 않습니다.

## C 호출

```
MQCLOSE (Hconn, &Hobj, Options, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN  Hconn;      /* Connection handle */
MQHOBJ   Hobj;       /* Object handle */
MQLONG   Options;    /* Options that control the action of MQCLOSE */
MQLONG   CompCode;   /* Completion code */
MQLONG   Reason;     /* Reason code qualifying CompCode */
```

## COBOL 호출

```
CALL 'MQCLOSE' USING HCONN, HOBJ, OPTIONS, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```
** Connection handle
01 HCONN    PIC S9(9) BINARY.
** Object handle
01 HOBJ     PIC S9(9) BINARY.
** Options that control the action of MQCLOSE
01 OPTIONS  PIC S9(9) BINARY.
** Completion code
01 COMPCODE PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON   PIC S9(9) BINARY.
```

## PL/I 호출

```
call MQCLOSE (Hconn, Hobj, Options, CompCode, Reason);
```

매개변수를 다음과 같이 선언합니다.

```
dcl Hconn      fixed bin(31); /* Connection handle */
dcl Hobj       fixed bin(31); /* Object handle */
dcl Options    fixed bin(31); /* Options that control the action of
                                MQCLOSE */
dcl CompCode   fixed bin(31); /* Completion code */
dcl Reason     fixed bin(31); /* Reason code qualifying CompCode */
```

## 상위 레벨 어셈블러 호출

```
CALL MQCLOSE,(HCONN,HOBJ,OPTIONS,COMPCODE,REASON)
```

매개변수를 다음과 같이 선언합니다.

```
HCONN      DS  F  Connection handle
HOBJ       DS  F  Object handle
OPTIONS    DS  F  Options that control the action of MQCLOSE
COMPCODE   DS  F  Completion code
REASON     DS  F  Reason code qualifying COMPCODE
```

## Visual Basic 호출

```
MQCLOSE Hconn, Hobj, Options, CompCode, Reason
```

매개변수를 다음과 같이 선언합니다.

```
Dim Hconn As Long 'Connection handle'  
Dim Hobj As Long 'Object handle'  
Dim Options As Long 'Options that control the action of MQCLOSE'  
Dim CompCode As Long 'Completion code'  
Dim Reason As Long 'Reason code qualifying CompCode'
```

## MQCMIT - 변경사항 커밋

MQCMIT 호출은 애플리케이션이 동기점에 도달했고 마지막 동기점 이후에 발생한 모든 메시지 가져오기 및 넣기가 영구적이 되도록 큐 관리자에 지시합니다.

작업 단위의 일부로 넣어진 메시지는 삭제되지만 작업 단위의 일부로 검색된 메시지는 큐에 복구됩니다.

- ▶ **z/OS** z/OS에서 호출은 배치 프로그램에 의해서만 사용됩니다(IMS 배치 DL/I 프로그램 포함).

## 구문

MQCMIT(*Hconn*, *CompCode*, *Reason*)

## 매개변수

### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *Hconn*의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

### CompCode

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_WARNING

경고(일부 완료).

#### MQCC\_FAILED

호출에 실패했습니다.

## 원인

유형: MQLONG - 출력

나열된 이유 코드는 큐 관리자가 **Reason** 매개변수에 대해 리턴할 수 있는 코드입니다.

*CompCode*이(가) MQCC\_OK인 경우:

#### MQRC\_NONE

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_WARNING인 경우:

#### MQRC\_BACKED\_OUT

(2003, X'7D3') 작업 단위가 백아웃되었습니다.

#### MQRC\_OUTCOME\_PENDING

(2124, X'84C') 커밋 조작의 결과가 보류 중입니다.

CompCode이(가) MQCC\_FAILED인 경우:

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

**MQRC\_API\_EXIT\_ERROR**

(2374, X'946') API 엑시트가 실패했습니다.

**MQRC\_ASID\_MISMATCH**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**MQRC\_CALL\_INTERRUPTED**

(2549, X'9F5') MQPUT 또는 MQCMIT가 인터럽트되었으며 다시 연결 처리가 명백한 결과를 재설정할 수 없습니다.

**MQRC\_CF\_STRUC\_IN\_USE**

(2346, X'92A') 커플링 기능 구조가 사용 중입니다.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**MQRC\_ENVIRONMENT\_ERROR**

(2012, X'7DC') 호출이 환경 내에서 유효하지 않습니다.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**MQRC\_OBJECT\_DAMAGED**

(2101, X'835') 오브젝트가 손상되었습니다.

**MQRC\_OUTCOME\_MIXED**

(2123, X'84B') 커밋 또는 백아웃 조작의 결과가 혼합되어 있습니다.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') 큐 관리자가 종료되었습니다.

**MQRC\_RECONNECT\_FAILED**

(2548, X'9F4') 다시 연결 후에 다시 연결 가능한 연결을 위해 핸들을 회복하는 중에 오류가 발생했습니다.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**MQRC\_STORAGE\_MEDIUM\_FULL**

(2192, X'890') 외부 스토리지 매체가 가득 찼습니다.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

## 사용법 참고

- 큐 관리자 자체가 작업 단위를 통합하는 경우에만 이 호출을 사용하십시오. 다음과 같습니다.
  - 변경사항이 IBM MQ 자원에만 영향을 미치는 경우, 로컬 작업 단위.
  - 변경사항이 다른 자원 관리자에 속한 자원과 IBM MQ 자원에 영향을 미칠 수 있는 경우, 글로벌 작업 단위.로컬 및 글로벌 작업 단위에 대한 세부사항은 610 페이지의 『MQBEGIN - 작업 단위 시작』의 내용을 참조하십시오.
- 큐 관리자가 작업 단위를 조정하지 않는 환경에서 적절한 백아웃 호출은 MQCMIT 대신에 사용되어야 합니다. 환경은 또한 정상적으로 종료된 애플리케이션으로 발생한 암시적 커밋도 지원할 수 있습니다.
  - z/OS에서는 다음 호출을 사용하십시오.



- 배치 프로그램(IMS 배치 DL/I 프로그램 포함)은 작업 단위가 IBM MQ 자원에만 영향을 미치는 경우 MQCMIT 호출을 사용할 수 있습니다. 그러나 작업 단위가 IBM MQ 자원 및 기타 자원 관리자에 속한 자원 모두에 영향을 미치는 경우(예: Db2) z/OS RRS(Recoverable Resource Service)가 제공하는 SRRRCMIT 호출을 사용하십시오. SRRRCMIT 호출은 RRS 통합에 사용 가능한 자원 관리자에 속한 자원에 변경사항을 커밋합니다.
  - CICS 애플리케이션은 EXEC CICS SYNCPOINT 명령을 사용하여 작업 단위를 명확하게 커밋해야 합니다. 또는 트랜잭션을 인코딩하면 작업 단위의 암시적 커밋이 발생합니다. MQCMIT 호출은 CICS 애플리케이션에 사용할 수 없습니다.
  - IMS 애플리케이션(일괄처리 DL/I 프로그램 이외)은 GU 및 CHPK와 같은 IMS 호출을 사용하여 작업 단위를 커밋해야 합니다. MQCMIT 호출은 IMS 애플리케이션(배치 DL/I 프로그램 제외)에 사용할 수 없습니다.
  - IBM i에서는 큐 관리자가 조정하는 로컬 작업 단위에 이 호출을 사용하십시오. 이는 커밋 정의가 작업 레벨에 존재하지 않아야 함을 의미합니다. 즉, **CMTSOPE(\*JOB)** 매개변수가 있는 STRCMTCTL 명령이 작업에 대해 실행되지 않았어야 합니다.
3. 애플리케이션이 작업 단위에서 커밋되지 않은 변경사항으로 종료되는 경우, 해당 변경사항의 배치는 애플리케이션이 정상으로 또는 비정상적으로 종료되는지 여부에 달려 있습니다. 추가 세부사항은 [MQDISC 사용법](#) [참고](#)를 참조하십시오.
  4. 애플리케이션이 논리 메시지의 세그먼트 또는 그룹에서 메시지를 넣거나 가져올 때 큐 관리자는 마지막 성공한 MQPUT 및 MQGET 호출에 대한 메시지 그룹 및 논리 메시지와 관련된 정보를 보유하고 있습니다. 이 정보는 큐 핸들과 연관되어 있으며 다음과 같은 내용을 포함합니다.
    - MQMD에 있는 *GroupId*, *MsgSeqNumber*, *Offset* 및 *MsgFlags* 필드의 값입니다.
    - 메시지가 작업 단위의 일부인지 여부.
    - MQPUT 호출의 경우: 메시지가 지속적인지 또는 비지속적인지 여부.

작업의 단위가 커밋될 때 큐 관리자는 그룹과 세그먼트 정보를 보유하고 애플리케이션이 현재 메시지 그룹 또는 논리 메시지에 메시지를 넣거나 가져오기를 계속할 수 있습니다.

작업의 단위가 커밋될 때 그룹과 세그먼트 정보를 보유하는 것은 애플리케이션이 여러 작업 단위 전체에 걸쳐 많은 세그먼트로 구성되어 대량 메시지 그룹 또는 대량 논리 메시지를 펼칠 수 있게 허용합니다. 로컬 큐 관리자에 제한된 큐 스토리지만 있는 경우 여러 작업 단위를 사용하는 것이 유용합니다. 그러나 애플리케이션은 시스템 장애가 발생하는 경우 올바른 지점에서 메시지 넣기 또는 가져오기를 다시 시작하도록 충분한 정보를 유지보수해야 합니다. 시스템 장애 후에 올바른 지점에서 다시 시작하는 방법에 대한 세부사항은 [MQPMO\\_LOGICAL\\_ORDER](#) 및 [MQGMO\\_LOGICAL\\_ORDER](#)를 참조하십시오.

나머지 사용 시 참고사항은 큐 관리자가 작업 단위를 통합할 때만 적용됩니다.
  5. 작업 단위에는 연결 핸들과 동일한 범위가 있습니다. 특정 작업 단위에 영향을 미치는 모든 IBM MQ 호출은 동일한 연결 핸들을 사용하여 수행해야 합니다. 다른 연결 핸들을 사용하여 실행된 호출(예: 다른 애플리케이션이 발행한 호출)은 다음 작업 단위에 영향을 줍니다. 연결 핸들의 범위에 대한 정보는 MQCONN에서 설명하는 **Hconn** 매개변수를 참조하십시오.
  6. 현재 작업 단위의 일부로서 넣거나 검색된 메시지만 이 호출의 영향을 받습니다.
  7. 작업 단위 내에서 MQGET, MQPUT 또는 MQPUT1 호출을 발행하지만 커밋 또는 백아웃 호출을 발행하지 않는 장기 실행 애플리케이션은 다른 애플리케이션에 사용 가능하지 않은 메시지로 큐를 채울 수 있습니다. 이를 예방하려면 관리자가 **MaxUncommittedMsgs** 큐 관리자 속성의 값을 제어 불가능한 애플리케이션이 큐를 채우는 것을 막을 수 있을 만큼 충분히 낮게, 하지만 예상되는 메시징 애플리케이션이 올바르게 작동할 수 있는 만큼 높게 설정해야 합니다.
  8.  AIX, Linux, and Windows 시스템에서 **Reason** 매개변수가 MQRC\_CONNECTION\_BROKEN(MQCC\_FAILED의 *CompCode* 사용) 또는 MQRC\_UNEXPECTED\_ERROR 인 경우 작업 단위가 성공적으로 커밋되었을 수 있습니다.

## C 호출

```
MQCMIT (Hconn, &CompCode, &Reason);
```



매개변수를 다음과 같이 선언합니다.

```
MQHCONN Hconn; /* Connection handle */
MQLONG CompCode; /* Completion code */
MQLONG Reason; /* Reason code qualifying CompCode */
```

## COBOL 호출

```
CALL 'MQCMIT' USING HCONN, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```
** Connection handle
01 HCONN PIC S9(9) BINARY.
** Completion code
01 COMPCODE PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON PIC S9(9) BINARY.
```

## PL/I 호출

```
call MQCMIT (Hconn, CompCode, Reason);
```

매개변수를 다음과 같이 선언합니다.

```
dcl Hconn fixed bin(31); /* Connection handle */
dcl CompCode fixed bin(31); /* Completion code */
dcl Reason fixed bin(31); /* Reason code qualifying CompCode */
```

## 상위 레벨 어셈블러 호출

```
CALL MQCMIT,(HCONN,COMPCODE,REASON)
```

매개변수를 다음과 같이 선언합니다.

```
HCONN DS F Connection handle
COMPCODE DS F Completion code
REASON DS F Reason code qualifying COMPCODE
```

## Visual Basic 호출

```
MQCMIT Hconn, CompCode, Reason
```

매개변수를 다음과 같이 선언합니다.

```
Dim Hconn As Long 'Connection handle'
Dim CompCode As Long 'Completion code'
Dim Reason As Long 'Reason code qualifying CompCode'
```

## MQCONN - 큐 관리자 연결

MQCONN 호출은 애플리케이션 프로그램을 큐 관리자에 연결합니다.

이는 후속 메시지 큐잉 호출에서 애플리케이션이 사용하는 큐 관리자 연결 핸들을 제공합니다.

- z/OS에서 CICS 애플리케이션이 이 호출을 발행할 필요가 없습니다. 이러한 애플리케이션은 CICS 시스템이 연결되는 큐 관리자에 자동으로 연결됩니다. 그러나 MQCONN 및 MQDISC 호출은 여전히 CICS 애플리케이션에서 허용됩니다.
- IBM i에서 애플리케이션은 MQCONN 또는 MQCONNX 호출을 사용하여 큐 관리자에 연결해야 하고 MQDISC 호출을 사용하여 큐 관리자에서 연결을 끊어야 합니다.

서버 전용 설치에는 클라이언트 연결을 설정할 수 없으며 클라이언트 전용 설치에는 로컬 연결을 설정할 수 없습니다.

## 구문

MQCONN(QMgrName, Hconn, CompCode, Reason)

## 매개변수

### QMgrName

유형: MQCHAR48 - 입력

이는 애플리케이션이 연결할 큐 관리자의 이름입니다. 이름에는 다음 문자가 포함될 수 있습니다.

- 대문자 알파벳(A - Z)
- 소문자 알파벳(a - z)
- 숫자(0 - 9)
- 마침표(.), 슬래시(/), 밑줄(\_), 퍼센트(%)

이름에 선두 문자 또는 임베드된 공백을 사용할 수 없으나 후미 문자 공백은 포함할 수 있습니다. 널 문자는 이름에서 중요한 데이터의 끝을 표시하는 데 사용할 수 있습니다. 널 및 그 다음에 오는 문자는 공백으로 처리됩니다. 다음 제한사항이 표시된 환경에서 적용됩니다.

- EBCDIC 가타카나를 사용하는 시스템에서는 소문자를 사용할 수 없습니다.
- z/OS에서 밑줄로 시작하거나 끝나는 이름은 조작 및 제어판에서 처리할 수 없습니다. 따라서 해당 이름을 사용하지 마십시오.
- IBM i에서 소문자, 슬래시 또는 퍼센트가 포함된 이름은 명령에서 지정할 때 따옴표로 묶어야 합니다.  
**QMgrName** 매개변수에서는 이러한 인용 부호를 지정하지 마십시오.

이름이 완전히 공백으로 구성된 경우 기본 큐 관리자의 이름이 사용됩니다. 그러나 IBM MQ MQI client 애플리케이션의 절에 설명된 공백 큐 관리자 이름의 사용에 유의하십시오.

*QMgrName* 에 지정된 이름은 연결 가능 큐 관리자의 이름이거나, 큐 관리자 그룹이 사용 중인 경우 큐 관리자 그룹의 이름이어야 합니다.

z/OS에서 연결 가능한 큐 관리자는 환경에서 판별됩니다.

- CICS의 경우 CICS 시스템이 연결되는 큐 관리자만 사용할 수 있습니다. **QMgrName** 매개변수를 지정해야 하지만 해당 값이 무시됩니다. 공백 문자는 적절한 옵션입니다.
- IMS의 경우 서브시스템 정의 테이블(CSQQDEFV)에 나열된 큐 관리자 및 IMS의 SSM 테이블에 나열된 큐 관리자만 연결할 수 있습니다(사용법 참고사항 6 참조).
- z/OS 배치 및 TSO의 경우 애플리케이션과 동일한 시스템에 상주하는 큐 관리자만 연결 가능합니다(사용법 참고 6 참조).

**큐 공유 그룹** 여러 큐 관리자가 존재하며 큐 공유 그룹을 형성하도록 구성된 시스템에서는 큐 관리자의 이름 대신 *QMgrName*에 대해 큐 공유 그룹의 이름을 지정할 수 있습니다. 이를 통해 애플리케이션은 큐 공유 그룹에서 사용 가능하고 애플리케이션과 동일한 z/OS 이미지에 있는 모든 큐 관리자에 연결할 수 있습니다. 시스템은 또는 공백 *QMgrName*을 사용하여 기본 큐 관리자 대신 큐 공유 그룹에 연결하도록 구성될 수 있습니다.

*QMgrName*이 큐 공유 그룹의 이름을 지정하지만 시스템에 해당 이름의 큐 관리자가 있는 경우 전자보다 후자에 우선적으로 연결이 설정됩니다. 해당 연결이 실패하는 경우에만 큐 공유 그룹의 큐 관리자 중 하나에 대한 연결을 시도합니다.

연결이 성공하면 MQCONN 또는 MQCONNX 호출에서 리턴되는 핸들을 사용하여 연결이 설정된 큐 관리자에 속한 모든 자원(공유 및 비공유 둘 다)에 액세스할 수 있습니다. 이러한 자원에 대한 액세스에는 일반적인 권한 제어가 적용됩니다.

애플리케이션이 현재 연결을 설정하기 위해 두 개의 MQCONN 또는 MQCONNX 호출을 발행하고 하나 또는 두 호출이 큐 공유 그룹의 이름을 지정하는 경우 두 번째 호출이 첫 번째 호출과 동일한 큐 관리자에 연결되면 완료 코드 MQCC\_WARNING 및 이유 코드 MQRC\_ALREADY\_CONNECTED를 리턴합니다.

큐 공유 그룹은 z/OS에서만 지원됩니다. 큐 공유 그룹에 대한 연결은 배치, RRS 배치, CICS, TSO 환경에서만 지원됩니다. CICS의 경우 CICS 시스템에 연결되는 큐 공유 그룹만 사용할 수 있습니다. QMgrName 매개변수를 지정해야 하지만 해당 값이 무시됩니다. 공백 문자는 적절한 옵션입니다.



**주의:** 큐 공유 그룹에 IMS를 연결할 수 없습니다.

**IBM MQ MQI client 애플리케이션:** IBM MQ MQI client 애플리케이션의 경우, 지정된 큐 관리자 이름의 클라이언트 연결 채널 정의마다 성공할 때까지 연결을 시도합니다. 그러나 큐 관리자에는 지정된 이름과 동일한 이름이 있어야 합니다. 모두 공백인 이름이 지정된 경우 연결이 성공할 때까지 모두 공백인 큐 관리자 이름을 가진 각 클라이언트 연결이 시도됩니다. 이 경우 큐 관리자의 실제 이름에 대응하는 검사는 수행되지 않습니다.

IBM MQ 클라이언트 애플리케이션은 z/OS에서 지원되지 않지만 z/OS는 IBM MQ 클라이언트 애플리케이션이 연결할 수 있는 IBM MQ 서버로 수행될 수 있습니다.

**IBM MQ MQI client 큐 관리자 그룹:** 지정된 이름이 별표(\*)로 시작하는 경우 연결되는 큐 관리자의 이름이 애플리케이션에서 지정한 이름과 다를 수 있습니다. 별표 없이 지정된 이름은 연결에 대해 적합한 큐 관리자 그룹을 정의합니다. 구현은 연결을 설정할 수 있는 큐 관리자를 찾을 때까지 차례로 하나씩 시도하여 그룹 중 하나를 선택합니다. 연결이 시도된 순서는 클라이언트 채널 가중치 및 후보 채널의 선호도 값에 영향을 받습니다. 그룹에서 큐 관리자 중 연결에 사용 가능한 큐 관리자가 없는 경우 호출이 실패합니다. 각 큐 관리자는 한 번씩만 시도됩니다. 이름에 대해 별표만 지정된 경우에는 구현에서 정의된 기본 큐 관리자 그룹이 사용됩니다.

큐 관리자 그룹은 MQ 클라이언트 환경에서 실행 중인 애플리케이션에 대해서만 지원됩니다. 클라이언트 외의 애플리케이션이 별표로 시작되는 큐 관리자 이름을 지정하면 호출에 실패합니다. 그룹 내의 각 큐 관리자와 통신하기 위해 동일한 큐 관리자 이름(별표 없이 지정된 이름)을 사용한 여러 개의 클라이언트 연결 채널 정의를 제공하여 그룹이 정의됩니다. 기본 그룹은 하나 이상의 클라이언트 연결 채널 정의를 제공하여 정의되며 각각 공백 큐 관리자 이름을 갖습니다(모두 공백인 이름을 지정하여 클라이언트 애플리케이션의 이름에 대해 하나의 별표를 지정하는 것과 동일한 효과를 갖습니다).

그룹의 하나의 큐 관리자에 연결한 후에 애플리케이션은 애플리케이션이 연결된 큐 관리자의 이름(로컬 큐 관리자)을 의미하는 오브젝트 디스크립터 및 메시지의 큐 관리자 이름 필드에서 일반적인 방법으로 공백을 지정할 수 있습니다. 애플리케이션이 이 이름을 알아야 하는 경우 MQINQ 호출을 사용하여 QMgrName 큐 관리자 속성을 조회하십시오.

연결 이름에 별표를 접두부로 사용하면 애플리케이션이 그룹에서 특정 큐 관리자에 대한 연결에 의존하지 않음을 나타냅니다. 적합한 애플리케이션은 다음과 같습니다.

- 메시지를 넣지만 메시지를 가져오지 않는 애플리케이션입니다.
- 요청 메시지를 넣은 후 임시 동적 큐에서 응답 메시지를 가져오는 애플리케이션입니다.

적합하지 않은 애플리케이션은 특정 큐 관리자의 특정 큐에서 메시지를 가져와야 하는 애플리케이션입니다. 이러한 애플리케이션은 이름에 별표를 접두부로 사용하지 마십시오.

별표를 지정하면 이름의 나머지 최대 길이는 47자입니다.

이 매개변수의 길이는 MQ\_Q\_MGR\_NAME\_LENGTH로 지정됩니다.

## Hconn

유형: MQHCONN - 출력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. 애플리케이션이 발행한 모든 후속 메시지 큐잉 호출에서 이를 지정하십시오. 이는 MQDISC 호출이 발행되거나 핸들의 범위를 정의하는 처리 단위가 종료될 때 유효하도록 사용이 중단됩니다.

IBM MQ는 이제 서버 패키지 및 클라이언트 패키지와 함께 mqm 라이브러리를 제공합니다. 즉, mqm 라이브러리에서 발견된 MQI 호출이 작성될 때 클라이언트 또는 서버 연결인지 여부를 확인하기 위해 연결 유형을 검사하고 올바른 기본 호출이 작성됩니다. 따라서 Hconn을 전달한 엑시트가 mqm 라이브러리에 대해 링크될 수 있지만 클라이언트 설치에서 사용됩니다.

핸들 범위: 리턴된 핸들의 범위는 큐 관리자에 연결하기 위해 사용된 호출(MQCONN 또는 MQCONNX)에 따라 다릅니다. 사용된 호출이 MQCONNX이면 핸들의 범위 또한 MQCNO 구조의 Options 필드에서 지정된 MQCNO\_HANDLE\_SHARE\_\* 옵션에 따라 다릅니다.

- 호출이 MQCONN이거나 MQCNO\_HANDLE\_SHARE\_NONE 옵션이 지정된 경우 리턴된 핸들이 비공유 핸들입니다.

비공유 핸들의 범위는 애플리케이션이 실행 중인 플랫폼에서 지원하는 병렬 처리의 가장 작은 단위입니다 (자세한 내용은 640 페이지의 표 546 참조). 이 핸들은 호출이 실행된 병렬 처리 단위 외부에서 유효하지 않습니다.

- MQCNO\_HANDLE\_SHARE\_BLOCK 또는 MQCNO\_HANDLE\_SHARE\_NO\_BLOCK 옵션을 지정하는 경우 리턴된 핸들은 공유 핸들입니다.

공유 핸들의 범위는 호출이 발행된 스레드를 소유한 프로세스입니다. 핸들은 해당 프로세스에 속하는 모든 스레드에서 사용할 수 있습니다. 일부 플랫폼은 스레드를 지원하지 않습니다.

- MQCC\_FAILED와 동일한 완료 코드와 함께 MQCONN 또는 MQCONNX 호출이 실패하는 경우 Hconn 값이 정의되지 않습니다.

플랫폼	비공유 핸들의 범위
z/OS	<ul style="list-style-type: none"> <li>• CICS: CICS 태스크</li> <li>• IMS: 태스크, 다음 동기점까지(태스크의 하위 태스크 제외)</li> <li>• z/OS 배치 및 TSO: 태스크(태스크의 하위 태스크 제외)</li> </ul>
IBM i	작업
AIX and Linux	스레드
32비트 Windows 애플리케이션	스레드
64비트 Windows 애플리케이션	스레드

z/OS for CICS 애플리케이션에서 리턴되는 값은 다음과 같습니다.

**MQHC\_DEF\_HCONN**

기본 연결 핸들

**CompCode**

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_WARNING**

경고(일부 완료).

**MQCC\_FAILED**

호출에 실패했습니다.

**원인**

유형: MQLONG - 출력

CompCode이(가) MQCC\_OK인 경우:

**MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_WARNING인 경우:

**MQRC\_ALREADY\_CONNECTED**

(2002, X'7D2') 애플리케이션이 이미 연결되어 있습니다.

**MQRC\_CLUSTER\_EXIT\_LOAD\_ERROR**

(2267, X'8DB') 클러스터 워크로드 엑시트를 로드하지 못했습니다.

**MQRC\_SSL\_ALREADY\_INITIALIZED**

(2391, X'957') SSL이 이미 초기화되었습니다.

*CompCode*이(가) MQCC\_FAILED인 경우:

**MQRC\_ADAPTER\_CONN\_LOAD\_ERROR**

(2129, X'851') 어댑터 연결 모듈을 로드할 수 없습니다.

**MQRC\_ADAPTER\_DEFS\_ERROR**

(2131, X'853') 어댑터 서브시스템 정의 모듈이 올바르지 않습니다.

**MQRC\_ADAPTER\_DEFS\_LOAD\_ERROR**

(2132, X'854') 어댑터 서브시스템 정의 모듈을 로드할 수 없습니다.

**MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') 어댑터를 사용할 수 없습니다.

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

**MQRC\_ADAPTER\_STORAGE\_SHORTAGE**

(2127, X'84F') 어댑터용 스토리지가 충분하지 않습니다.

**MQRC\_ANOTHER\_Q\_MGR\_CONNECTED**

(2103, X'837') 다른 큐 관리자가 이미 연결되어 있습니다.

**MQRC\_API\_EXIT\_ERROR**

(2374, X'946') API 엑시트가 실패했습니다.

**MQRC\_API\_EXIT\_INIT\_ERROR**

(2375, X'947') API 엑시트 초기화에 실패했습니다.

**MQRC\_API\_EXIT\_TERM\_ERROR**

(2376, X'948') API 엑시트 종료에 실패했습니다.

**MQRC\_ASID\_MISMATCH**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

**MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'7D5') 버퍼 길이 매개변수가 올바르지 않습니다.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**MQRC\_CONN\_ID\_IN\_USE**

(2160, X'870') 연결 ID가 이미 사용 중입니다.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**MQRC\_CONNECTION\_ERROR**

(2273, X'8E1') MQCONN 호출을 처리하는 중에 오류가 발생했습니다.

**MQRC\_CONNECTION\_NOT\_AVAILABLE**

(2568, X'A08') 큐 관리자가 현재 설치에서 요청된 연결 유형의 연결을 제공할 수 없는 경우 MQCONN 또는 MQCONNX 호출에서 발생합니다. 서버 전용 설치에서 클라이언트 연결을 설정할 수 없습니다. 클라이언트 전용 설치에서 로컬 연결을 설정할 수 없습니다.

**MQRC\_CONNECTION\_QUIESCING**

(2202, X'89A') 연결이 정지됩니다.

**MQRC\_CONNECTION\_STOPPING**

(2203, X'89B') 연결이 종료됩니다.

**MQRC\_CRYPTO\_HARDWARE\_ERROR**

(2382, X'94E') 암호화 하드웨어 구성 오류입니다.

**MQRC\_DUPLICATE\_RECOV\_COORD**

(2163, X'873') 복구 조정자가 존재합니다.

**MQRC\_ENVIRONMENT\_ERROR**

(2012, X'7DC') 호출이 환경 내에서 유효하지 않습니다.

또한 MQCONN 호출에서 CICS 또는 IMS 애플리케이션의 [329 페이지](#)의 『MQCSP - 보안 매개변수』 제어 블록을 전달합니다.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**MQRC\_HOST\_NOT\_AVAILABLE**

(2538, X'9EA') 큐 관리자에 연결하기 위해 클라이언트에서 MQCONN 호출이 발행되었으나 원격 시스템에 대화를 할당하려는 시도가 실패했습니다.

**MQRC\_INSTALLATION\_MISMATCH**

(2583, X'A17') 큐 관리자 설치와 선택된 라이브러리가 일치하지 않습니다.

**MQRC\_KEY\_REPOSITORY\_ERROR**

(2381, X'94D') 키 저장소가 올바르지 않습니다.

**MQRC\_MAX\_CONNS\_LIMIT\_REACHED**

(2025, X'7E9') 최대 연결 수에 도달했습니다.

**MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

**MQRC\_OPEN\_FAILED**

(2137, X'859') 오브젝트가 성공적으로 열리지 않았습니다.

**MQRC\_Q\_MGR\_NAME\_ERROR**

(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**

(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.

**MQRC\_Q\_MGR QUIESCING**

(2161, X'871') 큐 관리자가 정지됩니다.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') 큐 관리자가 종료되었습니다.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**MQRC\_SECURITY\_ERROR**

(2063, X'80F') 보안 오류가 발생했습니다.

**MQRC\_SSL\_INITIALIZATION\_ERROR**

(2393, X'959') SSL 초기화 오류입니다.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

## 사용법 참고

1. MQCONN 호출을 사용하여 연결이 작성된 큐 관리자를 로컬 큐 관리자라고 합니다.
2. 로컬 큐 관리자가 소유한 큐는 애플리케이션에 로컬 큐로 표시됩니다. 이러한 큐에 대해 메시지를 넣고 메시지를 가져올 수 있습니다.



로컬 큐 관리자가 속한 큐 공유 그룹이 소유한 공유 큐는 애플리케이션에 로컬 큐로 표시됩니다. 이러한 큐에 대해 메시지를 넣고 메시지를 가져올 수 있습니다.

리모트 큐 관리자가 소유한 큐는 리모트 큐로 표시됩니다. 이러한 큐에서 메시지를 넣을 수는 있지만 큐에서 메시지를 가져올 수는 없습니다.

3. 애플리케이션이 실행 중인 동안 큐 관리자가 실패하는 경우 애플리케이션이 MQCONN 호출을 다시 발행하여 후속 IBM MQ 호출에서 사용할 새 연결 핸들을 확보해야 합니다. 애플리케이션은 호출이 성공할 때까지 정기적으로 MQCONN 호출을 발행할 수 있습니다.

애플리케이션이 큐 관리자에 연결되었는지 여부를 확신하지 못하는 경우 애플리케이션은 이 안전하게 MQCONN 호출을 발행하여 연결 핸들을 확보할 수 있습니다. 애플리케이션이 이미 연결된 경우 리턴된 핸들이 이전 MQCONN 호출로 리턴된 핸들과 동일하지만 완료 코드 MQCC\_WARNING 및 이유 코드 MQRC\_ALREADY\_CONNECTED와 함께 리턴됩니다.

4. 애플리케이션이 IBM MQ 호출을 사용하여 완료된 경우 애플리케이션은 MQDISC 호출을 사용하여 큐 관리자에서 연결을 끊어야 합니다.
5. MQCC\_FAILED와 동일한 완료 코드와 함께 MQCONN 호출이 실패하는 경우 Hconn 값이 정의되지 않습니다.
6. z/OS의 경우:

- 배치, TSO, IMS 애플리케이션은 MQCONN 호출을 발행하여 다른 IBM MQ 호출을 사용해야 합니다. 이러한 애플리케이션은 동시에 둘 이상의 큐 관리자에 연결할 수 있습니다.

큐 관리자가 실패하면 애플리케이션은 큐 관리자가 재시작된 후에 다시 호출을 발행하여 새 연결 핸들을 확보해야 합니다.

이미 연결된 경우에도 IMS 애플리케이션은 MQCONN 호출을 반복적으로 발행할 수 있지만 이는 온라인 메시지 처리 프로그램(MPP)에 권장되지 않습니다.


- CICS 애플리케이션은 기타 IBM MQ 호출을 사용하기 위해 MQCONN 호출을 발행할 필요가 없지만 원하는 경우 이를 수행할 수 있습니다. MQCONN 호출 및 MQDISC 호출 모두 허용됩니다. 그러나 둘 이상의 큐 관리자에 동시에 연결할 수는 없습니다.

큐 관리자가 실패하면 큐 관리자가 재시작될 때 해당 애플리케이션이 자동으로 다시 연결되므로 MQCONN 호출을 발행할 필요가 없습니다.

7. z/OS에서 사용 가능한 큐 관리자를 정의하려면:

- 배치 애플리케이션의 경우, 시스템 프로그래머는 CSQBDEF 매크로를 사용하여 기본 큐 관리자 이름 또는 큐 공유 그룹 이름을 정의하는 모듈(CSQBDEFV)을 작성할 수 있습니다.
- IMS 애플리케이션의 경우 시스템 프로그래머는 CSQQDEFX 매크로를 사용하여 사용 가능한 큐 관리자의 이름을 정의하고 기본 큐 관리자를 지정하는 모듈(CSQQDEFV)을 작성할 수 있습니다.

또한 각 큐 관리자는 IMS 제어 영역 및 해당 큐 관리자에 액세스하는 각 종속 영역에 정의되어야 합니다. 이를 수행하려면 IMS.PROCLIB 라이브러리에서 서브시스템 멤버를 작성하고 적용 가능한 IMS 영역에 대해 서브시스템 멤버를 식별해야 합니다. 애플리케이션이 해당 IMS 영역의 서브시스템 멤버에서 정의되지 않은 큐 관리자에 연결하려고 시도하는 경우 애플리케이션이 이상종료됩니다.

 이러한 매크로 사용에 대한 추가 정보는 [고객 사용을 위해 의도된 매크로를 참조하십시오](#).

8. IBM i에서 비정상적으로 종료되는 프로그램은 자동으로 큐 관리자에서 연결 해제되지 않습니다. 완료 코드 MQCC\_WARNING 및 이유 코드 MQRC\_ALREADY\_CONNECTED를 리턴하는 MQCONN 또는 MQCONNX 호출을 사용할 수 있게 허용하도록 애플리케이션을 작성하십시오. 이 상황에서 정상으로 리턴된 연결 핸들을 사용하십시오.

## C 호출

```
MQCONN (QMgrName, &Hconn, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQCHAR48 QMgrName; /* Name of queue manager */
MQHCONN Hconn; /* Connection handle */
```

```
MQLONG  CompCode; /* Completion code */
MQLONG  Reason;   /* Reason code qualifying CompCode */
```

## COBOL 호출

```
CALL 'MQCONN' USING QMGRNAME, HCONN, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```
** Name of queue manager
01 QMGRNAME PIC X(48).
** Connection handle
01 HCONN PIC S9(9) BINARY.
** Completion code
01 COMPCODE PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON PIC S9(9) BINARY.
```

## PL/I 호출

```
call MQCONN (QMgrName, Hconn, CompCode, Reason);
```

매개변수를 다음과 같이 선언합니다.

```
dcl QMgrName char(48); /* Name of queue manager */
dcl Hconn fixed bin(31); /* Connection handle */
dcl CompCode fixed bin(31); /* Completion code */
dcl Reason fixed bin(31); /* Reason code qualifying CompCode */
```

## 상위 레벨 어셈블러 호출

```
CALL MQCONN,(QMGRNAME,HCONN,COMPCODE,REASON)
```

매개변수를 다음과 같이 선언합니다.

```
QMGRNAME DS CL48 Name of queue manager
HCONN DS F Connection handle
COMPCODE DS F Completion code
REASON DS F Reason code qualifying COMPCODE
```

## Visual Basic 호출

```
MQCONN QMgrName, Hconn, CompCode, Reason
```

매개변수를 다음과 같이 선언합니다.

```
Dim QMgrName As String*48 'Name of queue manager'
Dim Hconn As Long 'Connection handle'
Dim CompCode As Long 'Completion code'
Dim Reason As Long 'Reason code qualifying CompCode'
```

## MQCONNX - 연결 큐 관리자(확장)

MQCONNX 호출은 애플리케이션 프로그램을 큐 관리자에 연결합니다. 애플리케이션이 후속 IBM MQ 호출에서 사용된 큐 관리자 연결 핸들을 제공합니다.



MQCONNX 호출은 MQCONN 호출과 유사합니다. 단, MQCONNX는 호출이 작동하는 방법을 제어하기 위한 옵션을 지정하도록 허용합니다.

- 이 호출은 모든 IBM MQ 시스템에서 지원되며 IBM MQ 클라이언트가 해당 시스템에 연결되었습니다.

서버 전용 설치에는 클라이언트 연결을 설정할 수 없으며 클라이언트 전용 설치에는 로컬 연결을 설정할 수 없습니다.

## 구문

`MQCONNX(QMgrName, ConnectOpts, Hconn, CompCode, Reason)`

## 매개변수

### QMgrName

유형: MQCHAR48 - 입력

세부사항은 [637 페이지의 『MQCONN - 큐 관리자 연결』](#)에서 설명하는 **QMgrName** 매개변수를 참조하십시오.

### ConnectOpts

유형: MQCNO - 입출력(I/O)

자세한 내용은 [312 페이지의 『MQCNO - 연결 옵션』](#)의 내용을 참조하십시오.

### Hconn

유형: MQHCONN - 출력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. 애플리케이션이 발행한 모든 후속 메시지 큐잉 호출에서 이를 지정하십시오. 이는 MQDISC 호출이 발행되거나 핸들의 범위를 정의하는 처리 단위가 종료될 때 유효하도록 사용이 중단됩니다.

IBM MQ는 이제 서버 패키지 및 클라이언트 패키지와 함께 mqm 라이브러리를 제공합니다. 즉, mqm 라이브러리에서 발견된 MQI 호출이 작성될 때 클라이언트 또는 서버 연결인지 여부를 확인하기 위해 연결 유형을 검사하고 올바른 기본 호출이 작성됩니다. 따라서 *Hconn*을 전달한 엑시트가 mqm 라이브러리에 대해 링크될 수 있지만 클라이언트 설치에서 사용됩니다.

핸들 범위: 리턴된 핸들의 범위는 큐 관리자에 연결하기 위해 사용된 호출(MQCONN 또는 MQCONNX)에 따라 다릅니다. 사용된 호출이 MQCONNX이면 핸들의 범위 또한 MQCNO 구조의 *Options* 필드에서 지정된 MQCNO\_HANDLE\_SHARE\_\* 옵션에 따라 다릅니다.

- 호출이 MQCONN이거나 MQCNO\_HANDLE\_SHARE\_NONE 옵션이 지정된 경우 리턴된 핸들이 비공유 핸들입니다.

비공유 핸들의 범위는 애플리케이션이 실행 중인 플랫폼에서 지원하는 병렬 처리의 가장 작은 단위입니다 (자세한 내용은 [646 페이지의 표 547](#) 참조). 이 핸들은 호출이 실행된 병렬 처리 단위 외부에서 유효하지 않습니다.

- MQCNO\_HANDLE\_SHARE\_BLOCK 또는 MQCNO\_HANDLE\_SHARE\_NO\_BLOCK 옵션을 지정하는 경우 리턴된 핸들은 공유 핸들입니다.

공유 핸들의 범위는 호출이 발행된 스레드를 소유한 프로세스입니다. 핸들은 해당 프로세스에 속하는 모든 스레드에서 사용할 수 있습니다. 일부 플랫폼은 스레드를 지원하지 않습니다.

- MQCC\_FAILED와 동일한 완료 코드와 함께 MQCONN 또는 MQCONNX 호출이 실패하는 경우 Hconn 값이 정의되지 않습니다.

표 547. 다양한 플랫폼에서 비공유 핸들의 범위	
플랫폼	비공유 핸들의 범위
z/OS	<ul style="list-style-type: none"> <li>• CICS: CICS 태스크</li> <li>• IMS: 태스크, 다음 동기점까지(태스크의 하위 태스크 제외)</li> <li>• z/OS 배치 및 TSO: 태스크(태스크의 하위 태스크 제외)</li> </ul>
IBM i	작업
AIX and Linux	스레드
32비트 Windows 애플리케이션	스레드
64비트 Windows 애플리케이션	스레드

z/OS for CICS 애플리케이션에서 리턴되는 값은 다음과 같습니다.

### **MQHC\_DEF\_HCONN**

기본 연결 핸들

#### **CompCode**

유형: MQLONG - 출력

세부사항은 637 페이지의 『MQCONN - 큐 관리자 연결』에서 설명하는 **CompCode** 매개변수를 참조하십시오.

#### **원인**

유형: MQLONG - 출력

다음 코드는 MQCONN 및 MQCONNX 호출로 리턴될 수 있습니다. MQCONNX 호출로 리턴될 수 있는 추가 코드 목록은 다음 코드를 참조하십시오.

*CompCode*이(가) MQCC\_OK인 경우:

#### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_WARNING인 경우:

#### **MQRC\_ALREADY\_CONNECTED**

(2002, X'7D2') 애플리케이션이 이미 연결되어 있습니다.

#### **MQRC\_CLUSTER\_EXIT\_LOAD\_ERROR**

(2267, X'8DB') 클러스터 워크로드 엑시트를 로드하지 못했습니다.

#### **MQRC\_SSL\_ALREADY\_INITIALIZED**

(2391, X'957') SSL이 이미 초기화되었습니다.

*CompCode*이(가) MQCC\_FAILED인 경우:

#### **MQRC\_ADAPTER\_CONN\_LOAD\_ERROR**

(2129, X'851') 어댑터 연결 모듈을 로드할 수 없습니다.

#### **MQRC\_ADAPTER\_DEFS\_ERROR**

(2131, X'853') 어댑터 서브시스템 정의 모듈이 올바르지 않습니다.

#### **MQRC\_ADAPTER\_DEFS\_LOAD\_ERROR**

(2132, X'854') 어댑터 서브시스템 정의 모듈을 로드할 수 없습니다.

#### **MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') 어댑터를 사용할 수 없습니다.

#### **MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

**MQRC\_ADAPTER\_STORAGE\_SHORTAGE**

(2127, X'84F') 어댑터용 스토리지가 충분하지 않습니다.

**MQRC\_ANOTHER\_Q\_MGR\_CONNECTED**

(2103, X'837') 다른 큐 관리자가 이미 연결되어 있습니다.

**MQRC\_API\_EXIT\_ERROR**

(2374, X'946') API 엑시트가 실패했습니다.

**MQRC\_API\_EXIT\_INIT\_ERROR**

(2375, X'947') API 엑시트 초기화에 실패했습니다.

**MQRC\_API\_EXIT\_TERM\_ERROR**

(2376, X'948') API 엑시트 종료에 실패했습니다.

**MQRC\_ASID\_MISMATCH**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

**MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'7D5') 버퍼 길이 매개변수가 올바르지 않습니다.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**MQRC\_CONN\_ID\_IN\_USE**

(2160, X'870') 연결 ID가 이미 사용 중입니다.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**MQRC\_CONNECTION\_ERROR**

(2273, X'8E1') MQCONN 호출을 처리하는 중에 오류가 발생했습니다.

**MQRC\_CONNECTION\_NOT\_AVAILABLE**

(2568, X'A08') 큐 관리자가 현재 설치에서 요청된 연결 유형의 연결을 제공할 수 없는 경우 MQCONN 또는 MQCONNX 호출에서 발생합니다. 서버 전용 설치에서 클라이언트 연결을 설정할 수 없습니다. 클라이언트 전용 설치에서 로컬 연결을 설정할 수 없습니다.

**MQRC\_CONNECTION QUIESCING**

(2202, X'89A') 연결이 정지됩니다.

**MQRC\_CONNECTION\_STOPPING**

(2203, X'89B') 연결이 종료됩니다.

**MQRC\_CRYPTO\_HARDWARE\_ERROR**

(2382, X'94E') 암호화 하드웨어 구성 오류입니다.

**MQRC\_DUPLICATE\_RECOV\_COORD**

(2163, X'873') 복구 조정자가 존재합니다.

**MQRC\_ENVIRONMENT\_ERROR**

(2012, X'7DC') 호출이 환경 내에서 유효하지 않습니다.

또한 MQCONNX 호출에서 CICS 또는 IMS 애플리케이션의 [329 페이지](#)의 『MQCSP - 보안 매개변수』 제어 블록을 전달합니다.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**MQRC\_HOST\_NOT\_AVAILABLE**

(2538, X'9EA') 큐 관리자에 연결하기 위해 클라이언트에서 MQCONN 호출이 발행되었으나 원격 시스템에 대화를 할당하려는 시도가 실패했습니다.

**MQRC\_INSTALLATION\_MISMATCH**

(2583, X'A17') 큐 관리자 설치와 선택된 라이브러리가 일치하지 않습니다.

**MQRC\_KEY\_REPOSITORY\_ERROR**

(2381, X'94D') 키 저장소가 올바르지 않습니다.

**MQRC\_MAX\_CONNS\_LIMIT\_REACHED**

(2025, X'7E9') 최대 연결 수에 도달했습니다.

**MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

**MQRC\_OPEN\_FAILED**

(2137, X'859') 오브젝트가 성공적으로 열리지 않았습니다.

**MQRC\_Q\_MGR\_NAME\_ERROR**

(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**

(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.

**MQRC\_Q\_MGR QUIESCING**

(2161, X'871') 큐 관리자가 정지됩니다.

**MQRC\_Q\_MGR STOPPING**

(2162, X'872') 큐 관리자가 종료되었습니다.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**MQRC\_SECURITY\_ERROR**

(2063, X'80F') 보안 오류가 발생했습니다.

**MQRC\_SSL\_INITIALIZATION\_ERROR**

(2393, X'959') SSL 초기화 오류입니다.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

다음 추가 이유 코드는 MQCONNX 호출에 의해 리턴될 수 있습니다.

*CompCode*이(가) MQCC\_FAILED인 경우:

**MQRC\_AIR\_ERROR**

(2385, X'951') 인증 정보 레코드가 올바르지 않습니다.

**MQRC\_AUTH\_INFO\_CONN\_NAME\_ERROR**

(2387, X'953') 인증 정보 연결 이름이 올바르지 않습니다.

**MQRC\_AUTH\_INFO\_REC\_COUNT\_ERROR**

(2383, X'94F') 인증 정보 레코드 수가 올바르지 않습니다.

**MQRC\_AUTH\_INFO\_REC\_ERROR**

(2384, X'950') 인증 정보 레코드 필드가 올바르지 않습니다.

**MQRC\_AUTH\_INFO\_TYPE\_ERROR**

(2386, X'952') 인증 정보 유형이 올바르지 않습니다.

**MQRC\_CD\_ERROR**

(2277, X'8E5') 채널 정의가 올바르지 않습니다.

**MQRC\_CLIENT\_CONN\_ERROR**

(2278, X'8E6') 클라이언트 연결 필드가 유효하지 않습니다.

**MQRC\_CNO\_ERROR**

(2139, X'85B') 연결 옵션 구조가 올바르지 않습니다.

**MQRC\_CONN\_TAG\_IN\_USE**

(2271, X'8DF') 연결 태그가 사용 중입니다.

**MQRC\_CONN\_TAG\_NOT\_USABLE**

(2350, X'92E') 연결 태그를 사용할 수 없습니다.

**MQRC CSP 오류**

(2595, X'A23') MQCSP 구조가 올바르지 않습니다.

**V 9.4.0 MQRC\_FUNCTION\_NOT\_SUPPORTED**

(2298, X'8FA') 요청된 함수는 현재 환경에서 사용할 수 없습니다.

**MQRC\_LDAP\_PASSWORD\_ERROR**

(2390, X'956') LDAP 비밀번호가 올바르지 않습니다.

**MQRC\_LDAP\_USER\_NAME\_ERROR**

(2388, X'954') LDAP 사용자 이름 필드가 올바르지 않습니다.

**MQRC\_LDAP\_USER\_NAME\_LENGTH\_ERR**

(2389, X'955') LDAP 사용자 이름 길이가 올바르지 않습니다.

**MQRC\_OPTIONS\_ERROR**

(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.

**MQRC\_SCO\_ERROR**

(2380, X'94C') SSL 구성 옵션 구조가 올바르지 않습니다.

**MQRC\_SSL\_CONFIG\_ERROR**

(2392, X'958') SSL 구성 오류입니다.

**MQRC\_TOKEN\_TIMESTAMP\_NOT\_VALID**

(2064, X'810') 인증 토큰이 아직 유효하지 않거나 만료되었습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

## 사용법 참고

Visual Basic 프로그래밍 언어의 경우 다음과 같은 사항이 적용됩니다.

- **ConnectOpts** 매개변수가 MQCNO 유형으로 선언됩니다. 애플리케이션이 IBM MQ MQI client로 실행 중이고 클라이언트 연결 채널의 매개변수를 지정하려는 경우 애플리케이션이 MQCNO 구조 대신 호출에서 MQCNOCD 구조를 지정할 수 있도록 **ConnectOpts** 매개변수를 Any 유형으로 선언하십시오. 그러나 이는 올바른 데이터 유형인지 확인하기 위해 **ConnectOpts** 매개변수를 확인할 수 없음을 의미합니다.

## C 호출

```
MQCONN (QMgrName, &ConnectOpts, &Hconn, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```

MQCHAR48  QMgrName;      /* Name of queue manager */
MQCNO     ConnectOpts;  /* Options that control the action of MQCONN */
MQHCONN   Hconn;        /* Connection handle */
MQLONG    CompCode;     /* Completion code */
MQLONG    Reason;       /* Reason code qualifying CompCode */

```

## COBOL 호출

```
CALL 'MQCONN' USING QMGRNAME, CONNECTOPTS, HCONN, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```

** Name of queue manager
01 QMGRNAME      PIC X(48).
** Options that control the action of MQCONN
01 CONNECTOPTS.
   COPY CMQCNV.
** Connection handle
01 HCONN        PIC S9(9) BINARY.
** Completion code
01 COMPCODE     PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON       PIC S9(9) BINARY.

```

## PL/I 호출

```
call MQCONNX (QMgrName, ConnectOpts, Hconn, CompCode, Reason);
```

매개변수를 다음과 같이 선언합니다.

```
dc1 QMgrName    char(48);      /* Name of queue manager */
dc1 ConnectOpts like MQCNO;    /* Options that control the action of
                               MQCONNX */
dc1 Hconn       fixed bin(31); /* Connection handle */
dc1 CompCode    fixed bin(31); /* Completion code */
dc1 Reason      fixed bin(31); /* Reason code qualifying CompCode */
```

## 상위 레벨 어셈블러 호출

```
CALL MQCONNX, (QMGRNAME, CONNECTOPTS, HCONN, COMPCODE, REASON)
```

매개변수를 다음과 같이 선언합니다.

QMGRNAME	DS	CL48	Name of queue manager
CONNECTOPTS	CMQCNOA	,	Options that control the action of MQCONNX
HCONN	DS	F	Connection handle
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## Visual Basic 호출

```
MQCONNX QMgrName, ConnectOpts, Hconn, CompCode, Reason
```

매개변수를 다음과 같이 선언합니다.

```
Dim QMgrName As String*48 'Name of queue manager'
Dim ConnectOpts As MQCNO 'Options that control the action of'
                          'MQCONNX'
Dim Hconn As Long 'Connection handle'
Dim CompCode As Long 'Completion code'
Dim Reason As Long 'Reason code qualifying CompCode'
```

## MQCRTMH - 메시지 핸들 작성

MQCRTMH 호출은 메시지 핸들을 리턴합니다.

애플리케이션은 후속 메시지 큐잉 호출에서 MQCRTMH 호출을 사용할 수 있습니다.

- [MQSETMP](#) 호출을 사용하여 메시지 핸들의 특성을 설정하십시오.
- [MQINQMP](#) 호출을 사용하여 메시지 핸들의 특성 값을 조회하십시오.
- [MQDLTMP](#) 호출을 사용하여 메시지 핸들의 특성을 삭제하십시오.

MQPUT 및 MQPUT1 호출에서 메시지 핸들을 사용하여 메시지 핸들의 특성을 넣어 메시지의 특성과 연관시킬 수 있습니다. 이와 비슷하게 MQGET 호출에서 메시지 핸들을 지정하여, MQGET 호출이 완료될 때 메시지 핸들을 사용하여 검색할 메시지의 특성에 액세스할 수 있습니다.

[MQDLTMH](#)를 사용하여 메시지 핸들을 삭제하십시오.

## 구문

```
MQCRTMH(Hconn, CrtMsgHOpts, Hmsg, CompCode, Reason)
```

## 매개변수

### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *Hconn*의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다. 큐 관리자에 대한 연결이 유효하지 않고 IBM MQ 호출도 메시지 핸들에 작동하지 않는 경우 MQDLTMH가 내재적으로 호출되어 메시지를 삭제합니다.

또는 다음 값을 지정할 수 있습니다.

#### MQHC\_UNASSOCIATED\_HCONN

연결 핸들이 특정 큐 관리자에 대한 연결을 표시하지 않습니다.

이 값이 사용될 때 메시지 핸들은 할당된 스토리지를 릴리스하기 위해 MQDLTMH에 명시적 호출로 삭제되어야 합니다. IBM MQ는 내재적으로 메시지 핸들을 삭제하지 않습니다.

메시지 핸들을 작성하는 스레드에서 설정된 큐 관리자에 대해 하나 이상의 올바른 연결이 있어야 합니다. 그렇지 않으면 MQRC\_HCONN\_ERROR와 함께 호출이 실패합니다.

단일 시스템에 다중 설치가 있는 환경에서 MQHC\_UNASSOCIATED\_HCONN 값은 프로세스에 처음으로 로드된 설치에만 사용하도록 제한됩니다. 메시지 핸들이 다른 설치에 제공되면 이유 코드 MQRC\_HMSG\_NOT\_AVAILABLE이 리턴됩니다.

z/OS for CICS 애플리케이션에서 MQCONN 호출을 생략할 수 있으며 *Hconn*에 다음 값을 지정할 수 있습니다.

#### MQHC\_DEF\_CONN

기본 연결 핸들

### CrtMsgHOpts

유형: MQCMHO - 입력

MQCRTMH의 조치를 제어하는 옵션. 세부사항은 MQCMHO를 참조하십시오.

### Hmsg

유형: MQHMSG - 출력

출력에서 메시지 핸들의 특성을 설정, 조회 및 삭제하는 데 사용할 수 있는 메시지 핸들이 리턴됩니다. 초기에는 메시지에 특성이 없습니다.

메시지 핸들에는 연관된 메시지 디스크립터가 있습니다. 처음에는 기본값이 포함됩니다. MQSETMP 및 MQINQMP 호출을 사용하여 연관된 메시지 디스크립터 필드의 값을 설정하고 조회할 수 있습니다. MQDLTMP 호출은 메시지 디스크립터의 필드를 기본값으로 다시 재설정합니다.

*Hconn* 매개변수가 값 MQHC\_UNASSOCIATED\_HCONN으로 지정되면 리턴된 메시지 핸들은 처리 단위 내의 연결로 MQGET, MQPUT 또는 MQPUT1 호출에 사용될 수 있지만 한 번에 하나의 IBM MQ 호출만 사용할 수 있습니다. 두 번째 IBM MQ 호출이 동일한 메시지 핸들을 사용하려고 시도할 때 핸들이 사용 중인 경우 두 번째 IBM MQ 호출이 이유 코드 MQRC\_MSG\_HANDLE\_IN\_USE와 함께 실패합니다.

*Hconn* 매개변수가 MQHC\_UNASSOCIATED\_HCONN이 아닌 경우 리턴된 메시지 핸들은 지정된 연결에만 사용할 수 있습니다.

동일한 *Hconn* 매개변수 값은 이 메시지 핸들이 사용되는 후속 MQI 호출에서 사용되어야 합니다.

- MQDLTMH
- MQSETMP
- MQINQMP
- MQDLTMP
- MQMHBUF
- MQBUFMH

리턴된 메시지 핸들은 메시지 핸들에 대해 MQDLTMH 호출이 발행되거나 핸들의 범위를 정의하는 처리 단위가 종료될 때 유효하도록 사용이 중단됩니다. 메시지 핸들이 작성되고 큐 관리자에 대한 연결의 유효성이 중지될 때(예: MQDBC가 호출될 때) MQDLTMH가 내재적으로 호출됩니다.



## CompCode

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

### **MQCC\_OK**

정상적으로 완료되었습니다.

### **MQCC\_FAILED**

호출에 실패했습니다.

## 원인

유형: MQLONG - 출력

CompCode이(가) MQCC\_OK인 경우:

### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode이(가) MQCC\_FAILED인 경우:

### **MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'089C') 어댑터를 사용할 수 없습니다.

### **MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

### **MQRC\_ASID\_MISMATCH**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

### **MQRC\_CALL\_IN\_PROGRESS**

(2219, X'08AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

### **MQRC\_CMHO\_ERROR**

(2461, X'099D') 올바르지 않은 메시지 핸들 옵션 구조가 작성되었습니다.

### **MQRC\_CONNECTION\_BROKEN**

(2273, X'7D9') 큐 관리자에 대한 연결이 손실되었습니다.

### **MQRC\_HANDLE\_NOT\_AVAILABLE**

(2017, X'07E1') 사용 가능한 핸들이 없습니다.

### **MQRC\_HCONN\_ERROR**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

### **MQRC\_HMSG\_ERROR**

(2460, X'099C') 메시지 핸들 포인터가 올바르지 않습니다.

### **MQRC\_OPTIONS\_ERROR**

(2046, X'07FE') 옵션이 올바르지 않거나 일치하지 않습니다.

### **MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

### **MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드를 참조하십시오](#).

## C

```
MQCRTMH (Hconn, &CrtMsgHOpts, &Hmsg, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN  Hconn;          /* Connection handle */
MQCMHO   CrtMsgHOpts;   /* Options that control the action of MQCRTMH */
MQHMSG   Hmsg;          /* Message handle */
```



```
MQLONG  CompCode;    /* Completion code */
MQLONG  Reason;      /* Reason code qualifying CompCode */
```

## COBOL

```
CALL 'MQCRTMH' USING HCONN, CRTMSGHOPTS, HMSG, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```
** Connection handle
01 HCONN      PIC S9(9) BINARY.
** Options that control the action of MQCRTMH
01 CRTMSGHOPTS.
   COPY CMQCMHOV.
** Message handle
01 HMSG      PIC S9(18) BINARY.
** Completion code
01 COMPCODE  PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON    PIC S9(9) BINARY.
```

## PL/I

```
call MQCRTMH (Hconn, CrtMsgHOpts, Hmsg, CompCode, Reason);
```

매개변수를 다음과 같이 선언합니다.

```
dcl Hconn      fixed bin(31); /* Connection handle */
dcl CrtMsgHOpts like MQCMHO; /* Options that control the action of MQCRTMH */
dcl Hmsg       fixed bin(63); /* Message handle */
dcl CompCode   fixed bin(31); /* Completion code */
dcl Reason     fixed bin(31); /* Reason code qualifying CompCode */
```

## High Level Assembler

```
CALL MQCRTMH, (HCONN, CRTMSGHOPTS, HMSG, COMPCODE, REASON)
```

매개변수를 다음과 같이 선언합니다.

```
HCONN          DS      F  Connection handle
CRTMSGHOPTS    CMQCMHOA , Options that control the action of MQCRTMH
HMSG           DS      D  Message handle
COMPCODE       DS      F  Completion code
REASON         DS      F  Reason code qualifying COMPCODE
```

## MQCTL - 콜백 제어

MQCTL 호출은 연결을 위해 열린 오브젝트 핸들 및 콜백의 제어 조치를 수행합니다.

### 구문

```
MQCTL(Hconn, Operation, ControlOpts, CompCode, Reason)
```

### 매개변수

#### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *Hconn*의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

CICS 용 z/OS 애플리케이션에서 MQCONN 호출을 생략할 수 있으며 *Hconn*에 대해 다음 특수 값을 지정할 수 있습니다.

## **MQHC\_DEF\_HCONN**

기본 연결 핸들

### **Operation**

유형: MQLONG - 입력

지정된 오브젝트 핸들에 대해 정의된 콜백에서 처리 중인 조작입니다. 다음 옵션 중 하나 또는 하나만 지정해야 합니다.

### **MQOP\_START**

지정된 연결 핸들의 정의된 모든 메시지 사용자 함수에 대해 메시지 이용을 시작합니다.

콜백은 시스템에서 시작된 스레드에서 실행하며 모든 애플리케이션 스레드와 다릅니다.

이 작동은 시스템에 연결 핸들을 제공한 제어를 제공합니다. 사용자 스레드 이외의 스레드에서 실행할 수 있는 유일한 MQI 호출은 다음과 같습니다.

- 조작 MQOP\_STOP이 있는 MQCTL
- 조작 MQOP\_SUSPEND가 있는 MQCTL
- MQDISC - HConn 연결을 끊기 전에 조작 MQOP\_STOP이 있는 MQCTL을 수행합니다.

연결 핸들이 시작되는 동안 IBM MQ API 호출이 발행되고 호출이 메시지 사용자 함수에서 생성되지 않는 경우 MQRC\_HCONN\_ASYNC\_ACTIVE가 리턴됩니다.

메시지 이용자가 MQCBCT\_START\_CALL 중에 연결을 중지하는 경우 MQCTL 호출이 실패 이유 코드 MQRC\_CONNECTION\_STOPPED와 함께 리턴합니다.

이는 사용자 함수에 발행될 수 있습니다. 콜백 루틴과 동일한 연결의 경우 유일한 목적은 이전에 실행된 MQOP\_STOP 조작을 취소하는 것입니다.

이 옵션은 다음 환경에서 지원되지 않습니다. CICS on z/OS 또는 애플리케이션이 스레드되지 않은 IBM MQ 라이브러리와 바인드된 경우입니다.

### **MQOP\_START\_WAIT**

지정된 연결 핸들의 정의된 모든 메시지 사용자 함수에 대해 메시지 이용을 시작합니다.

동일한 스레드 및 제어에 실행하는 메시지 이용자는 MQCTL의 호출자로 리턴되지 않습니다.

- MQCTL MQOP\_STOP 또는 MQOP\_SUSPEND 조작의 사용으로 릴리스되었습니다. 또는
- 모든 사용자 루틴은 등록 취소되거나 일시중단되었습니다.

모든 이용자가 등록 해제되거나 일시중단되는 경우 암시적 MQOP\_STOP 조작이 실행됩니다.

이 옵션은 또한 현재 연결 핸들 또는 기타 연결 핸들에 대해 콜백 루틴에서 사용할 수 없습니다. 호출을 시도하는 경우 MQRC\_ENVIRONMENT\_ERROR와 함께 리턴됩니다.

MQOP\_START\_WAIT 조작 중에 언제든지 등록된, 일시중단되지 않은 이용자가 없는 경우 이유 코드 MQRC\_NO\_CALLBACKS\_ACTIVE와 함께 호출이 실패합니다.

MQOP\_START\_WAIT 조작 중에 연결이 일시중단되면 MQCTL 호출이 경고 이유 코드 MQRC\_CONNECTION\_SUSPENDED를 리턴합니다. 연결이 '시작된' 상태로 남아 있습니다.

애플리케이션이 MQOP\_STOP 또는 MQOP\_RESUME을 발행하도록 선택할 수 있습니다. 이 경우 MQOP\_RESUME 조작이 차단됩니다.

이 옵션은 단일 스레드 클라이언트에서 지원되지 않습니다.

### **MQOP\_STOP**

메시지의 사용을 중지하고 모든 이용자가 이 옵션이 완료하기 전에 해당 작동을 완료하도록 대기하십시오. 이 조작은 연결 핸들을 릴리스합니다.

콜백 루틴 내에서 발행되는 경우 이 옵션은 루틴 종료까지 적용되지 않습니다. 메시지에 대한 콜백 루틴이 이미 완료된 후와 콜백 루틴이 작성된 호출을 중지한 후(요청된 경우) 호출된 메시지 사용자 루틴이 없습니다.

콜백 루틴이 외부에서 발행되는 경우 콜백이 작성된 호출을 중지한 후(요청된 경우) 이미 읽기가 완료된 메시지의 사용자 루틴까지 발행자에게 리턴하지 않습니다. 그러나 콜백 자체는 등록된 상태로 남아 있습니다.

이 함수는 메시지 미리 읽기에 영향을 미치지 않습니다. 전달 가능한 추가 메시지가 있는지 여부를 판별하려면 콜백 함수 내에서 사용자가 MQCLOSE(MQCO\_QUIESCE)를 실행해야 합니다.

### **MQOP\_SUSPEND**

메시지 이용 일시정지. 이 조작은 연결 핸들을 릴리스합니다.

이는 애플리케이션에 대한 메시지 미리 읽기에 영향을 미치지 않습니다. 장기적으로 메시지 이용을 중지하려는 경우 큐를 닫고 이용을 계속할 때 이를 다시 여는 것을 고려하십시오.

콜백 루틴 내에서 발행되는 경우 이는 루틴 종료까지 적용되지 않습니다. 더 이상 메시지 사용자 루틴은 현재 루틴 엑시트 이후에 호출되지 않습니다.

호출이 외부에 발행되면 현재 사용자 루틴이 완료되어 더 이상 호출되지 않을 때까지 제어는 호출자로 돌아가지 않습니다.

### **MQOP\_RESUME**

메시지 이용 일시정지.

이 옵션은 일반적으로 주요 애플리케이션 스레드에서 발행되지만 동일한 루틴에서 발행된 이전 일시중단 요청을 취소시키기 위해서도 콜백루틴으로부터 사용될 수 있습니다.

MQOP\_START\_WAIT를 재개하기 위해 MQOP\_RESUME이 사용되는 경우 조작이 차단됩니다.

### **ControlOpts**

유형: MQCTLO - 입력

MQCTL의 조치를 제어하는 옵션

구조에 대한 세부사항은 MQCTLO의 내용을 참조하십시오.

### **CompCode**

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

#### **MQCC\_OK**

정상적으로 완료되었습니다.

#### **MQCC\_WARNING**

경고(일부 완료).

#### **MQCC\_FAILED**

호출에 실패했습니다.

### **원인**

유형: MQLONG - 출력

CompCode이(가) MQCC\_OK인 경우:

#### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode이(가) MQCC\_FAILED인 경우:

#### **MQRC\_ADAPTER\_CONV\_LOAD\_ERROR**

(2133, X'855') 데이터 변환 서비스 모듈을 로드할 수 없습니다.

#### **MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') 어댑터를 사용할 수 없습니다.

#### **MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

**MQRC\_API\_EXIT\_ERROR**  
(2374, X'946') API 엑시트가 실패했습니다.

**MQRC\_API\_EXIT\_LOAD\_ERROR**  
(2183, X'887') API 엑시트를 로드할 수 없습니다.

**MQRC\_ASID\_MISMATCH**  
(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

**MQRC\_BUFFER\_LENGTH\_ERROR**  
(2005, X'7D5') 버퍼 길이 매개변수가 올바르지 않습니다.

**MQRC\_CALLBACK\_LINK\_ERROR**  
(2487, X'9B7') 콜백루틴을 호출할 수 없음

**MQRC\_CALLBACK\_NOT\_REGISTERED**  
(2448, X'990') 등록된 콜백이 없기 때문에 등록 취소, 일시중단 또는 재개할 수 없습니다.

**MQRC\_CALLBACK\_ROUTINE\_ERROR**  
(2486, X'9B6') CallbackFunction 및 CallbackName이 MQOP\_REGISTER 호출에 지정되었습니다.  
CallbackFunction 또는 CallbackName이 지정되었지만 현재 등록된 콜백 함수와 일치하지 않습니다.

**MQRC\_CALLBACK\_TYPE\_ERROR**  
(2483, X'9B3') 올바르지 않은 콜백 유형 필드.

**MQRC\_CALL\_IN\_PROGRESS**  
(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**MQRC\_CBD\_ERROR**  
(2444, X'98C') 옵션 차단이 올바르지 않습니다.

**MQRC\_CBD\_OPTIONS\_ERROR**  
(2484, X'9B4') 올바르지 않은 MQCBD 옵션 필드입니다.

**MQRC\_CICS\_WAIT\_FAILED**  
(2140, X'85C') CICS에서 대기 요청이 거부되었습니다.

**MQRC\_CONNECTION\_BROKEN**  
(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**MQRC\_CONNECTION\_NOT\_AUTHORIZED**  
(2217, X'8A9') 연결할 권한이 부여되지 않았습니다.

**MQRC\_CONNECTION QUIESCING**  
(2202, X'89A') 연결이 정지됩니다.

**MQRC\_CONNECTION\_STOPPING**  
(2203, X'89B') 연결이 종료됩니다.

**MQRC\_CORREL\_ID\_ERROR**  
(2207, X'89F') 상관 ID 오류입니다.

**MQRC\_ENVIRONMENT\_ERROR**  
(2012, X'7DC') 호출이 환경 내에서 유효하지 않습니다.

**MQRC\_FUNCTION\_NOT\_SUPPORTED**  
(2298, X'8FA') 요청된 함수는 현재 환경에서 사용할 수 없습니다.

**MQRC\_GET\_INHIBITED**  
(2016, X'7E0') 큐에 대해 가져오기가 금지되었습니다.

**MQRC\_GLOBAL\_UOW\_CONFLICT**  
(2351, X'92F') 글로벌 작업 단위 충돌입니다.

**MQRC\_GMO\_ERROR**  
(2186, X'88A') 메시지 가져오기 옵션 구조가 올바르지 않습니다.

**MQRC\_HANDLE\_IN\_USE\_FOR\_UOW**  
(2353, X'931') 글로벌 작업 단위에 대해 사용 중인 핸들입니다.

**MQRC\_HCONN\_ERROR**  
(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**MQRC\_HOBJ\_ERROR**  
(2019, X'7E3') 오브젝트 핸들이 올바르지 않습니다.

**MQRC\_INCONSISTENT\_BROWSE**  
(2259, X'8D3') 찾아보기 지정에 일관성이 없습니다.

**MQRC\_INCONSISTENT\_UOW**  
(2245, X'8C5') 작업 단위 지정에 일관성이 없습니다.

**MQRC\_INVALID\_MSG\_UNDER\_CURSOR**  
(2246, X'8C6') 커서에 있는 메시지가 검색에 대해 유효하지 않습니다.

**MQRC\_LOCAL\_UOW\_CONFLICT**  
(2352, X'930') 글로벌 작업 단위가 로컬 작업 단위와 충돌합니다.

**MQRC\_MATCH\_OPTIONS\_ERROR**  
(2247, X'8C7') 일치 옵션이 올바르지 않습니다.

**MQRC\_MAX\_MSG\_LENGTH\_ERROR**  
(2485, X'9B5') 올바르지 않은 MaxMsgLength 필드

**MQRC\_MD\_ERROR**  
(2026, X'7EA') 메시지 디스크립터가 올바르지 않습니다.

**MQRC\_MODULE\_ENTRY\_NOT\_FOUND**  
(2497, X'9C1') 지정된 함수 시작점을 모듈에서 찾을 수 없습니다.

**MQRC\_MODULE\_INVALID**  
(2496, X'9C0') 모듈을 찾았지만 올바르지 않은 유형이거나(32비트/64비트) 올바른 dll이 아닙니다.

**MQRC\_MODULE\_NOT\_FOUND**  
(2495, X'9BF') 모듈을 검색 경로에서 찾을 수 없거나 로드 권한이 부여되지 않았습니다.

**MQRC\_MSG\_ID\_ERROR**  
(2206, X'89E') 메시지 ID 오류입니다.

**MQRC\_MSG\_SEQ\_NUMBER\_ERROR**  
(2250, X'8CA') 메시지 순서 번호가 올바르지 않습니다.

**MQRC\_MSG\_TOKEN\_ERROR**  
(2331, X'91B') 메시지 토큰 사용이 올바르지 않습니다.

**MQRC\_NOT\_OPEN\_FOR\_BROWSE**  
(2036, X'7F4') 큐가 읽기 전용으로 열려있지 않습니다.

**MQRC\_NOT\_OPEN\_FOR\_INPUT**  
(2037, X'7F5') 큐가 입력을 위해 열려있지 않습니다.

**MQRC\_OBJECT\_CHANGED**  
(2041, X'7F9') 오브젝트가 열린 후에 정의가 변경되었습니다.

**MQRC\_OBJECT\_DAMAGED**  
(2101, X'835') 오브젝트가 손상되었습니다.

**MQRC\_OPERATION\_ERROR**  
(2488, X'9B8') API 호출에 대해 올바르지 않은 조작 코드입니다.

**MQRC\_OPTIONS\_ERROR**  
(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.

**MQRC\_PAGESET\_ERROR**  
(2193, X'891') 페이지 세트 데이터 세트에 액세스하는 중에 오류가 발생했습니다.

**MQRC\_Q\_DELETED**  
(2052, X'804') 큐가 삭제되었습니다.

**MQRC\_Q\_INDEX\_TYPE\_ERROR**  
(2394, X'95A') 큐의 색인 유형이 올바르지 않습니다.

**MQRC\_Q\_MGR\_NAME\_ERROR**  
(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**  
(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.

**MQRC\_Q\_MGR QUIESCING**

(2161, X'871') 큐 관리자가 정지됩니다.

**MQRC\_Q\_MGR STOPPING**

(2162, X'872') 큐 관리자가 종료되었습니다.

**MQRC\_RESOURCE PROBLEM**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**MQRC\_SIGNAL\_OUTSTANDING**

(2069, X'815') 이 핸들의 미해결 신호.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**MQRC\_SUPPRESSED\_BY\_EXIT**

(2109, X'83D') 엑시트 프로그램에서 호출이 억제되었습니다.

**MQRC\_SYNCPOINT\_NOT\_AVAILABLE**

(2072, X'818') 동기점 지원을 사용할 수 없습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

**MQRC\_UOW\_ENLISTMENT\_ERROR**

(2354, X'932') 글로벌 작업 단위에서의 등록이 실패했습니다.

**MQRC\_UOW\_MIX\_NOT\_SUPPORTED**

(2355, X'933') 작업 단위 호출의 혼합은 지원되지 않습니다.

**MQRC\_UOW\_NOT\_AVAILABLE**

(2255, X'8CF') 사용할 큐 관리자에 대해 작업 단위를 사용할 수 없습니다.

**MQRC\_WAIT\_INTERVAL\_ERROR**

(2090, X'82A') MQGMO의 대기 간격이 올바르지 않습니다.

**MQRC\_WRONG\_GMO\_VERSION**


(2256, X'8D0') 올바르지 않은 버전의 MQGMO가 제공되었습니다.

**MQRC\_WRONG\_MD\_VERSION**


(2257, X'8D1') 올바르지 않은 버전의 MQMD가 제공되었습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

## 사용법 참고

1. 콜백 루틴은 호출하는 모든 서비스의 응답을 확인해야 하며 루틴이 해결할 수 없는 조건을 감지하는 경우 콜백 루틴에 대한 반복된 호출을 막기 위해 MQCB MQOP\_DEREGISTER 명령을 실행해야 합니다.
2. XA 트랜잭션 관리자가 IBM MQ에 대한 업데이트를 포함하여 글로벌 트랜잭션을 관리하는 애플리케이션에서 비동기 이용을 사용 중인 경우 다음 추가 사항을 고려해야 합니다.
  - a. 작성 후 **xa\_open**을 호출한 후에는 **HConn**에 대해 MQCTL(MQOP\_START)을 호출하는 것은 올바르지 않습니다.  
이유는 **HConn**이 XA 컨텍스트에 첨부되어 비동기 이용 메커니즘이 사용하는 별도의 스레드에서 액세스할 수 없기 때문입니다.
  - b. 해당 시나리오에서 MQCTL(MQOP\_START)을 호출하는 경우 이유 코드 MQRC\_ASYNC\_XA\_CONFLICT(2350)와 함께 호출이 실패합니다.
  - c. 작성 후 **xa\_open**을 호출한 이후 **HConn**에 대해 MQCTL(MQOP\_START\_WAIT)를 호출하는 것이 좋습니다.  
이유는 비동기 이용 메커니즘을 시작하는 이 방법이 이후 **HConn**에 대한 모든 콜백이 MQCTL 호출이 작성된 스레드에서 실행되도록 만들기 때문입니다. 그러므로 **HConn** 및 스레드 간의 링크가 손실되지 않습니다.
3.  z/OS에서 조작이 MQOP\_START인 경우:

- 비동기 콜백 루틴을 사용하는 프로그램은 z/OS UNIX System Services(z/OS UNIX)를 사용할 권한이 부여되어야 합니다.
- 비동기 콜백 루틴을 사용하는 LE(Language Environment) 프로그램은 LE 런타임 옵션 POSIX(ON)를 사용해야 합니다.
- 비동기 콜백 루틴을 사용하는 비LE 프로그램은 z/OS UNIX pthread\_create 인터페이스(호출 가능 서비스 BPX1PTC)를 사용하지 않아야 합니다.

4.  MQCTL은 IMS 어댑터 내에서 지원되지 않습니다.

참고: CICS에서 MQOP\_START가 지원되지 않습니다. 대신 MQOP\_START\_WAIT 함수 호출을 사용하십시오.

## C 호출

```
MQCTL (Hconn, Operation, &ControlOpts, &CompCode, &Reason)
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN  Hconn;          /* Connection handle */
MQLONG   Operation;     /* Operation being processed */
MQCTLO   ControlOpts   /* Options that control the action of MQCTL */
MQLONG   CompCode;     /* Completion code */
MQLONG   Reason;       /* Reason code qualifying CompCode */
```

## COBOL 호출

```
CALL 'MQCTL' USING HCONN, OPERATION, CTLOPTS, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```
** Connection handle
01 HCONN PIC S9(9) BINARY.
** Operation
01 OPERATION PIC S9(9) BINARY.
** Control Options
01 CTLOPTS.
   COPY CMQCTLOV.
** Completion code
01 COMPCODE PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON PIC S9(9) BINARY.
```

## PL/I 호출

```
call MQCTL(Hconn, Operation, Ctlopts, CompCode, Reason)
```

매개변수를 다음과 같이 선언합니다.

```
dcl Hconn          fixed bin(31); /* Connection handle */
dcl Operation     fixed bin(31); /* Operation */
dcl Ctlopts like  MQCTLO;       /* Options that control the action of MQCTL */
dcl CompCode     fixed bin(31); /* Completion code */
dcl Reason       fixed bin(31); /* Reason code qualifying CompCode */
```

## MQDISC - 큐 관리자 연결 끊기

MQDISC 호출은 큐 관리자 및 애플리케이션 프로그램 사이의 연결을 중단하고 MQCONN 또는 MQCONNX 호출의 역수입니다.

- z/OS에서 비동기 메시지 이용, 이벤트 핸들링 또는 콜백, 기본 제어 스레드를 사용하는 모든 애플리케이션은 인코딩 전에 MQDISC 호출을 발행해야 합니다. 자세한 정보는 [IBM MQ메시지의 비동기 이용을 참조하십시오](#).
- z/OS에서, CICS 애플리케이션이 큐 관리자로부터 연결을 끊기 위해 이 호출을 발행할 필요가 없습니다. CICS 애플리케이션이 이 호출을 수행하지 않는 경우 다음 중 하나를 지정하여 이전 MQCONN 호출이 작성되지 않으면 적용되지 않습니다.

MQCNO\_SERIALIZE\_CONN\_TAG\_Q\_MGR  
 MQCNO\_SERIALIZE\_CONN\_TAG\_QSG  
 MQCNO\_RESTRICT\_CONN\_TAG\_Q\_MGR 또는  
 MQCNO\_RESTRICT\_CONN\_TAG\_QSG

현재 열려 있는 오브젝트 모두를 닫히게 하는 옵션입니다.

## 구문

MQDISC(*Hconn*, *CompCode*, *Reason*)

## 매개변수

### Hconn

유형: MQHCONN - 입출력(I/O)

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *Hconn*의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

CICS 용 z/OS 애플리케이션에서 MQCONN 호출을 생략하고 *Hconn*에 대해 다음 값을 지정할 수 있습니다.

#### **MQHC\_DEF\_HCONN**

기본 연결 핸들

호출 완료 시 큐 관리자가 *Hconn*을 환경에 올바른 핸들이 아닌 값으로 설정합니다. 이 값은 다음과 같습니다.

#### **MQHC\_UNUSABLE\_HCONN**

사용 불가능한 연결 핸들입니다.

z/OS에서 *Hconn* 는 정의되지 않은 값으로 설정됩니다.

### CompCode

유형: MQLONG - 출력

완료 코드: 다음 코드 중 하나입니다.

#### **MQCC\_OK**

정상적으로 완료되었습니다.

#### **MQCC\_WARNING**

경고(일부 완료).

#### **MQCC\_FAILED**

호출에 실패했습니다.

### 원인

유형: MQLONG - 출력

*CompCode*이(가) MQCC\_OK인 경우:

#### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_WARNING인 경우:

#### **MQRC\_BACKED\_OUT**

(2003, X'7D3') 작업 단위가 백아웃되었습니다.



**MQRC\_CONN\_TAG\_NOT\_RELEASED**

(2344, X'928') 연결 태그가 릴리스되지 않습니다.

**MQRC\_OUTCOME\_PENDING**

(2124, X'84C') 커밋 조작의 결과가 보류 중입니다.

CompCode이(가) MQCC\_FAILED인 경우:

**MQRC\_ADAPTER\_DISC\_LOAD\_ERROR**

(2138, X'85A') 어댑터 연결 해제 모듈을 로드할 수 없습니다.

**MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') 어댑터를 사용할 수 없습니다.

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

**MQRC\_API\_EXIT\_ERROR**

(2374, X'946') API 엑시트가 실패했습니다.

**MQRC\_API\_EXIT\_INIT\_ERROR**

(2375, X'947') API 엑시트 초기화에 실패했습니다.

**MQRC\_API\_EXIT\_TERM\_ERROR**

(2376, X'948') API 엑시트 종료에 실패했습니다.

**MQRC\_ASID\_MISMATCH**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**MQRC\_CONNECTION\_STOPPING**

(2203, X'89B') 연결이 종료됩니다.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**MQRC\_OUTCOME\_MIXED**

(2123, X'84B') 커밋 또는 백아웃 조작의 결과가 혼합되어 있습니다.

**MQRC\_PAGESET\_ERROR**

(2193, X'891') 페이지 세트 데이터 세트에 액세스하는 중에 오류가 발생했습니다.

**MQRC\_Q\_MGR\_NAME\_ERROR**

(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**

(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') 큐 관리자가 종료되었습니다.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

## 사용법 참고

1. 해당 연결 아래에서 연결에 여전히 열린 오브젝트가 있는 경우 MQDISC 호출이 발행되면 큐 관리자가 MQCO\_NONE으로 설정된 닫기 옵션과 함께 해당 오브젝트를 닫습니다.

2. 애플리케이션이 작업 단위에서 커밋되지 않은 변경사항으로 종료되는 경우 해당 변경사항의 배치는 애플리케이션이 종료되는 방법에 따라 다릅니다.
  - a. 애플리케이션이 종료되기 전에 MQDISC 호출을 발행하는 경우:
    - 큐 관리자 통합 작업 단위의 경우 큐 관리자는 애플리케이션 대신 MQCMIT 호출을 발행합니다. 가능한 경우 작업 단위가 커밋되고 가능하지 않은 경우 백아웃됩니다.
    - 외부적으로 통합된 작업 단위의 경우 작업 단위의 상태에 변경이 없지만 큐 관리자는 일반적으로 작업 단위 조정자가 요청할 때 작업 단위를 커밋해야 함을 표시합니다.  
z/OS에서 CICS, IMS(배치 DL/1 프로그램 제외) 및 RRS 애플리케이션은 이와 같습니다.
  - b. 애플리케이션이 MQDISC 호출을 발행하지 않고 정상적으로 종료되는 경우 수행되는 조치는 환경에 따라 다릅니다.
    - z/OS에서(MQ Java 또는 MQ JMS 애플리케이션 제외) 참고 2a에서 설명하는 조치가 발생합니다.
    - 다른 모든 경우 참고 2c에서 설명하는 조치가 발생합니다.
 환경 간의 차이 때문에 이식하려는 해당 애플리케이션이 끝나기 전에 작업 단위를 커밋하거나 백아웃하도록 하십시오.
  - c. 애플리케이션이 MQDISC 호출을 발행하지 않고 비정상적으로 종료하는 경우 작업 단위가 백아웃됩니다.
3. z/OS에서 다음 사항이 적용됩니다.
  - CICS 시스템 자체가 큐 관리자에 연결하고 MQDISC 호출에는 이 연결에 대한 영향이 없으므로 CICS 애플리케이션은 큐 관리자에서 연결을 해제하기 위해 MQDISC 호출을 발행할 필요가 없습니다.
  - CICS, IMS(배치 DL/1 프로그램 제외), RRS 애플리케이션은 외부 작업 단위 조정자가 조정하는 작업 단위를 사용합니다. 결과적으로 MQDISC 호출은 호출이 발행될 때 존재하는 작업 단위(있는 경우)의 상태에 영향을 미치지 않습니다.  
그러나 MQDISC 호출은 애플리케이션이 발행한 이전 MQCONN 호출에 의한 연결과 연관된 연결 태그 ConnTag의 사용 종료를 표시합니다. MQDISC 호출이 발행될 때 연결 태그를 참조하는 활성 작업 단위가 있는 경우 호출이 완료 코드 MQCC\_WARNING 및 이유 코드 MQRC\_CONN\_TAG\_NOT\_RELEASED와 함께 완료됩니다. 외부 작업 단위 조정자가 작업 단위를 해석할 때까지 연결 태그가 재사용 가능하지 않게 됩니다.

참고: CICS에서 MQOP\_START가 지원되지 않습니다. 대신 MQOP\_START\_WAIT 함수 호출을 사용하십시오.

## C 호출

```
MQDISC (&Hconn, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN  Hconn;      /* Connection handle */
MQLONG   CompCode;  /* Completion code */
MQLONG   Reason;    /* Reason code qualifying CompCode */
```

## COBOL 호출

```
CALL 'MQDISC' USING HCONN, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```
** Connection handle
01 HCONN    PIC S9(9) BINARY.
** Completion code
01 COMPCODE PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON   PIC S9(9) BINARY.
```

## PL/I 호출

```
call MQDISC (Hconn, CompCode, Reason);
```

매개변수를 다음과 같이 선언합니다.

```
dcl Hconn      fixed bin(31); /* Connection handle */
dcl CompCode   fixed bin(31); /* Completion code */
dcl Reason     fixed bin(31); /* Reason code qualifying CompCode */
```

## System/390 어셈블리 호출

```
CALL MQDISC,(HCONN,COMPCODE,REASON)
```

매개변수를 다음과 같이 선언합니다.

```
HCONN      DS  F  Connection handle
COMPCODE   DS  F  Completion code
REASON     DS  F  Reason code qualifying COMPCODE
```

## Visual Basic 호출

```
MQDISC Hconn, CompCode, Reason
```

매개변수를 다음과 같이 선언합니다.

```
Dim Hconn      As Long 'Connection handle'
Dim CompCode   As Long 'Completion code'
Dim Reason     As Long 'Reason code qualifying CompCode'
```

## MQDLTMH - 메시지 핸들 삭제

MQDLTMH 호출은 메시지 핸들을 삭제하며 MQCRTMH 호출의 역수입니다.

### 구문

```
MQDLTMH(Hconn, Hmsg, DltMsgHOpts, CompCode, Reason)
```

### 매개변수

#### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다.

값이 **Hmsg** 매개변수에서 지정된 메시지 핸들을 작성하는 데 사용된 연결 핸들과 일치해야 합니다.

메시지 핸들이 MQHC\_UNASSOCIATED\_HCONN을 사용하여 작성된 경우 메시지 핸들을 삭제하는 스레드에서 올바른 연결을 설정해야 합니다. 그렇지 않으면 호출이 MQRC\_CONNECTION\_BROKEN과 함께 실패합니다.

#### Hmsg

유형: MQHMSG - 입출력(I/O)

이는 삭제될 메시지 핸들입니다. 값은 이전 MQCRTMH 호출에서 리턴합니다.

호출의 정상 완료 시 핸들은 환경에 올바르게 않은 값으로 설정됩니다. 이 값은 다음과 같습니다.

**MQHM\_UNUSABLE\_HMSG**

사용 불가능한 메시지 핸들.

동일한 메시지 핸들을 전달한 다른 IBM MQ 호출이 진행 중에 있는 경우 메시지 핸들은 삭제될 수 없습니다.

**DltMsgHOpts**

유형: MQDMHO - 입력

세부사항은 [MQDMHO](#)를 참조하십시오.

**CompCode**

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_FAILED**

호출에 실패했습니다.

**원인**

유형: MQLONG - 출력

*CompCode*이(가) MQCC\_OK인 경우:

**MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*가 MQCC\_FAILED인 경우:

**MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'089C') 어댑터를 사용할 수 없습니다.

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

**MQRC\_ASID\_MISMATCH**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'08AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'07D9') 큐 관리자에 대한 연결이 유실되었습니다.

**MQRC\_DMHO\_ERROR**

(2462, X'099E') 올바르지 않은 메시지 핸들 옵션 구조가 삭제되었습니다.

**MQRC\_HMSG\_ERROR**

(2460, X'099C') 메시지 핸들 포인터가 올바르지 않습니다.

**MQRC\_MSG\_HANDLE\_IN\_USE**

(2499, X'09C3') 메시지 핸들이 이미 사용 중입니다.

**MQRC\_OPTIONS\_ERROR**

(2046, X'07FE') 옵션이 올바르지 않거나 일치하지 않습니다.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

**C 호출**

```
MQDLTMH (Hconn, &Hmsg, &DltMsgHOpts, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN  Hconn;          /* Connection handle */
MQHMSG   Hmsg;          /* Message handle */
MQDMHO   DltMsgHOpts;   /* Options that control the action of MQDLTMH */
MQLONG   CompCode;      /* Completion code */
MQLONG   Reason;       /* Reason code qualifying CompCode */
```

## COBOL 호출

```
CALL 'MQDLTMH' USING HCONN, HMSG, DLTMGHOPTS, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```
** Connection handle
01  HCONN    PIC S9(9) BINARY.

** Options that control the action of MQDLTMH
01  DLTMGHOPTS.
COPY CMQDMHOL.

** Completion code
01  COMPCODE PIC S9(9) BINARY.

** Reason code qualifying COMPCODE
01  REASON   PIC S9(9) BINARY.
```

## PL/I 호출

```
call MQDLTMH (Hconn, Hmsg, DltMsgHOpts, CompCode, Reason);
```

매개변수를 다음과 같이 선언합니다.

```
dcl Hconn          /* Connection handle */
dcl Hmsg           /* Message handle */
dcl DltMsgHOpts like MQDMHO; /* Options that control the action of MQDLTMH */
dcl CompCode      /* Completion code */
dcl Reason        /* Reason code qualifying CompCode */
```

## 상위 레벨 어셈블러 호출

```
CALL MQDLTMH, (HCONN, HMSG, DLTMGHOPTS, COMPCODE, REASON)
```

매개변수를 다음과 같이 선언합니다.

```
HCONN      DS      F  Connection handle
HMSG       DS      D  Message handle
DLTMGHOPTS CMQDMHOA , Options that control the action of MQDLTMH
COMPCODE   DS      F  Completion code
REASON     DS      F  Reason code qualifying COMPCODE
```

## MQDLTMP - 메시지 특성 삭제

MQDLTMP 호출은 메시지 핸들의 특성을 삭제하며 MQSETMP 호출의 역수입니다.

## 구문

```
MQDLTMP(Hconn, Hmsg, DltPropOpts, Name, CompCode, Reason)
```

## 매개변수

### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. 값이 **Hmsg** 매개변수에서 지정된 메시지 핸들을 작성하는 데 사용된 연결 핸들과 일치해야 합니다.

메시지 핸들이 MQHC\_UNASSOCIATED\_HCONN을 사용하여 작성된 경우 메시지 핸들을 삭제하는 스레드에서 올바른 연결을 설정해야 합니다. 그렇지 않으면 호출이 MQRC\_CONNECTION\_BROKEN과 함께 실패합니다.

### Hmsg

유형: MQHMSG - 입력

이는 삭제될 특성을 포함하는 메시지 핸들입니다. 값은 이전 MQCRTMH 호출에서 리턴합니다.

### DltPropOpts

유형: MQDMPO - 입력

세부사항은 [MQDMPO](#) 데이터 유형을 참조하십시오.

### 이름

유형: MQCHARV - 입력

삭제할 특성 이름입니다. 특성 이름에 대한 추가 정보는 [특성 이름](#)을 참조하십시오.

와일드 카드는 특성 이름으로 허용되지 않습니다.

### CompCode

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

#### **MQCC\_OK**

정상적으로 완료되었습니다.

#### **MQCC\_WARNING**

경고(일부 완료).

#### **MQCC\_FAILED**

호출에 실패했습니다.

### 원인

유형: MQLONG - 출력

*CompCode*이(가) MQCC\_OK인 경우:

#### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_WARNING인 경우:

#### **MQRC\_PROPERTY\_NOT\_AVAILABLE**

(2471, X'09A7') 특성을 사용할 수 없습니다.

#### **MQRC\_RFH\_FORMAT\_ERROR**

(2421, X'0975') 특성을 포함하는 MQRFH2 폴더를 구문 분석할 수 없습니다.

*CompCode*이(가) MQCC\_FAILED인 경우:

#### **MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'089C') 어댑터를 사용할 수 없습니다.

#### **MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'0852') 어댑터 서비스 모듈을 로드할 수 없습니다.

#### **MQRC\_ASID\_MISMATCH**

(2157, X'086D') 1차 및 홈 ASID가 다릅니다.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'08AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'07D9') 큐 관리자에 대한 연결이 유실되었습니다.

**MQRC\_DMPO\_ERROR**

(2481, X'09B1') 삭제 메시지 특성 옵션 구조는 올바르지 않습니다.

**MQRC\_HMSG\_ERROR**

(2460, X'099C') 메시지 핸들이 올바르지 않습니다.

**MQRC\_MSG\_HANDLE\_IN\_USE**

(2499, X'09C3') 메시지 핸들이 이미 사용 중입니다.

**MQRC\_OPTIONS\_ERROR**

(2046, X'07FE') 옵션이 올바르지 않거나 일치하지 않습니다.

**MQRC\_PROPERTY\_NAME\_ERROR**

(2442, X'098A') 올바르지 않은 특성 이름입니다.

**MQRC\_SOURCE\_CCSID\_ERROR**

(2111, X'083F') 특성 이름 코드화 문자 세트 ID가 올바르지 않습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'0893') 예상치 못한 오류가 발생했습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드를 참조하십시오](#).

## C 호출

```
MQDLTMP (Hconn, Hmsg, &DltPropOpts, &Name, &CompCode, &Reason)
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN Hconn;          /* Connection handle */
MQHMSG  Hmsg;           /* Message handle */
MQDMPO  DltPropOpts;   /* Options that control the action of MQDLTMP */
MQCHARV Name;         /* Property name */
MQLONG  CompCode;     /* Completion code */
MQLONG  Reason;       /* Reason code qualifying CompCode */
```

## COBOL 호출

```
CALL 'MQDLTMP' USING HCONN, HMSG, DLTPROPOPTS, NAME, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```
** Connection handle
01 HCONN PIC S9(9) BINARY.
** Message handle
01 HMSG PIC S9(18) BINARY.
** Options that control the action of MQDLTMP
01 DLTPROPOPTS.
COPY CMQDMPOV.
** Property name
01 NAME.
COPY CMQCHRVV.
** Completion code
01 COMPCODE PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON PIC S9(9) BINARY.
```

## PL/I 호출

```
call MQDLTMP (Hconn, Hmsg, DltPropOpts, Name, CompCode, Reason);
```

매개변수를 다음과 같이 선언합니다.

```
dc1 Hconn      fixed bin(31); /* Connection handle */
dc1 Hmsg       fixed bin(63); /* Message handle */
dc1 DltPropOpts like MQDMPO; /* Options that control the action of MQDLTMP */
dc1 Name       like MQCHARV; /* Property name */
dc1 CompCode   fixed bin(31); /* Completion code */
dc1 Reason     fixed bin(31); /* Reason code qualifying CompCode */
```

## 상위 레벨 어셈블러 호출

```
CALL MQDLTMP, (HCONN,HMSG,DLTPROPOPTS,NAME,COMPCODE,REASON)
```

매개변수를 다음과 같이 선언합니다.

HCONN	DS	F	Connection handle
HMSG	DS	D	Message handle
DLTPROPOPTS	CMQDMPOA	,	Options that control the action of MQDLTMP
NAME	CMQCHARVA	,	Property name
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## MQGET - 메시지 가져오기

MQGET 호출은 MQOPEN 호출을 사용하여 열린 로컬 큐에서 메시지를 검색합니다.

### 구문

```
MQGET(Hconn, Hobj, MsgDesc, GetMsgOpts, BufferLength, Buffer, DataLength, CompCode, Reason)
```

### 매개변수

#### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *Hconn*의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

z/OS for CICS 애플리케이션에서 MQCONN 호출을 생략할 수 있으며 *Hconn*에 다음 값이 지정됩니다.

#### MQHC\_DEF\_HCONN

기본 연결 핸들

#### Hobj

유형: MQHOBJ - 입력

이 핸들은 메시지를 검색할 큐를 나타냅니다. *Hobj*의 값이 이전 MQOPEN 호출로 리턴되었습니다. 큐는 다음 옵션 중 하나 이상을 사용하여 열어야 합니다(세부사항은 [705 페이지의 『MQOPEN - 오브젝트 열기』](#) 참조).

- MQOO\_INPUT\_SHARED
- MQOO\_INPUT\_EXCLUSIVE
- MQOO\_INPUT\_AS\_Q\_DEF
- MQOO\_BROWSE



## MsgDesc

유형: MQMD - 입출력(I/O)

이 구조는 필요한 메시지의 속성 및 검색된 메시지의 속성을 나타냅니다. 자세한 내용은 [412 페이지의 『MQMD - 메시지 디스크립터』](#)의 내용을 참조하십시오.

*BufferLength*가 메시지 길이 미만인 경우 *MsgDesc*는 MQGMO\_ACCEPT\_TRUNCATED\_MSG가 **GetMsgOpts** 매개변수에 지정되었는지 여부에 관계없이 큐 관리자에 의해 채워집니다(MQGMO - Options field 참조).

애플리케이션이 버전-1 MQMD를 제공할 경우, 리턴된 메시지에서 애플리케이션 메시지 데이터에 접두부 MQMDE가 붙어 있지만 이는 MQMDE에 있는 하나 이상의 필드에 기본값이 아닌 값이 있는 경우에만 해당됩니다. MQMDE에 있는 모든 필드에 기본값이 있는 경우 MQMDE가 생략됩니다. MQMD에 있는 *Format* 필드의 MQFMT\_MD\_EXTENSION 형식 이름은 MQMDE가 있음을 나타냅니다.

올바른 메시지 핸들이 *MsgHandle* 필드에 제공되는 경우 애플리케이션은 MQMD 구조를 제공할 필요가 없습니다. 이 필드에서 제공되는 사항이 없는 경우 메시지 핸들과 연관된 디스크립터에서 메시지에 대한 디스크립터를 가져옵니다.

애플리케이션이 MQMD 구조 대신 메시지 핸들을 제공하며 MQGMO\_PROPERTIES\_FORCE\_MQRFH2를 지정하는 경우 호출은 이유 코드 MQRC\_MD\_ERROR와 함께 실패합니다. 호출은 애플리케이션이 MQMD 구조를 제공하지 않으며 MQGMO\_PROPERTIES\_AS\_Q\_DEF를 지정하고 **PropertyControl** 큐 속성인 MQPROP\_FORCE\_MQRFH2인 경우에도 이유 코드 MQRC\_MD\_ERROR와 함께 실패합니다.

일치 옵션이 지정되고 메시지 핸들과 연관된 메시지 디스크립터가 사용되고 있는 경우 일치에 사용되는 입력 필드는 메시지 핸들로부터 제공됩니다.

## GetMsgOpts

유형: MQGMO - 입출력(I/O)

자세한 내용은 [362 페이지의 『MQGMO - 메시지 가져오기 옵션』](#)의 내용을 참조하십시오.

## BufferLength

유형: MQLONG - 입력

이는 *Buffer* 영역의 길이(바이트)입니다. 데이터가 없는 메시지에 대해 또는 메시지가 큐에서 제거될 것이며 데이터가 제거된 경우(이 경우에는 MQGMO\_ACCEPT\_TRUNCATED\_MSG를 지정해야 함)에는 0을 지정하십시오.

**참고:** 큐에서 읽을 수 있는 가장 긴 메시지의 길이는 **MaxMsgLength** 큐 속성에서 제공됩니다. [804 페이지의 『큐의 속성』](#)의 내용을 참조하십시오.

## 버퍼

유형: MQBYTExBufferLength - 출력

이는 메시지 데이터를 포함할 영역입니다. 경계에 있는 버퍼를 메시지에 있는 데이터의 네이처에 적절하게 맞추십시오. 4바이트 맞추기는 대부분의 메시지에 적합하지만(IBM MQ 헤더 구조가 포함된 메시지 포함) 일부 메시지에는 더 엄격한 맞추기가 필요할 수 있습니다. 예를 들어, 64비트 2진 정수를 포함하는 메시지에는 8바이트 맞추기가 필요할 수 있습니다.

*BufferLength*가 메시지 길이 미만인 경우 최대한 많은 메시지가 **Buffer**로 이동됩니다. 이는 MQGMO\_ACCEPT\_TRUNCATED\_MSG가 **GetMsgOpts** 매개변수에 지정되었는지 여부에 관계없이 발생합니다(자세한 정보는 [MQGMO - Options field](#) 참조).

**Buffer**에 있는 데이터의 문자 세트 및 인코딩은 **MsgDesc** 매개변수에서 리턴되는 *CodedCharSetId* 및 *Encoding* 필드에 의해 제공됩니다. 이 값이 수신자에 필요한 값과 다른 경우 수신자는 애플리케이션 메시지 데이터를 필요한 문자 세트 및 인코딩으로 변환해야 합니다. MQGMO\_CONVERT 옵션을 메시지 데이터를 변환하기 위해 사용할 수 있습니다(필요한 경우 사용자 작성 엑시트와 함께). 이 옵션에 대한 세부사항은 [362 페이지의 『MQGMO - 메시지 가져오기 옵션』](#)의 내용을 참조하십시오.

**참고:** MQGET 호출의 다른 모든 매개변수는 로컬 큐 관리자의 문자 세트 및 인코딩에 있습니다(**CodedCharSetId** 큐 관리자 속성 및 MQENC\_NATIVE로 지정됨).

호출이 실패해도 버퍼의 콘텐츠가 여전히 변경되었을 수 있습니다.

C 프로그래밍 언어에서 매개변수는 void에 대한 포인터로 선언됩니다. 모든 유형의 데이터에 대한 주소를 매개변수로 지정할 수 있습니다.

**BufferLength** 매개변수가 0인 경우 *Buffer*는 참조되지 않습니다. 이 경우 C 또는 시스템/390 어셈블러로 작성된 프로그램이 전달한 매개변수 주소가 널일 수 있습니다.

### DataLength

유형: MQLONG - 출력

메시지에 있는 애플리케이션 데이터의 길이(바이트)입니다. 이 값이 *BufferLength*보다 큰 경우 *BufferLength* 바이트만 **Buffer** 매개변수에서 리턴됩니다(즉 메시지가 잘림). 값이 0인 경우 메시지는 애플리케이션 데이터를 포함하지 않습니다.

*BufferLength*가 메시지 길이 미만인 경우 *DataLength*는 MQGMO\_ACCEPT\_TRUNCATED\_MSG가 **GetMsgOpts** 매개변수에 지정되었는지 여부에 관계없이 큐 관리자에 의해 완료됩니다(자세한 정보는 MQGMO - Options field 참조). 이렇게 하면 애플리케이션이 메시지 데이터를 수용하는 데 필요한 버퍼의 크기를 판별하고 적절한 크기의 버퍼를 사용하여 호출을 다시 발행할 수 있습니다.

그러나 MQGMO\_CONVERT 옵션이 지정되고 변환된 메시지 데이터가 *Buffer*에 들어가기에 너무 긴 경우 *DataLength*에 대해 다음과 같은 값이 리턴됩니다.

- 큐 관리자 정의 형식의 경우, 변환되지 않은 데이터의 길이.
  - 이 경우 데이터의 네이처로 인해 데이터를 변환 중에 펼치는 경우 애플리케이션은 *DataLength*의 큐 관리자에 의해 리턴된 값보다 큰 버퍼를 할당해야 합니다.
- 애플리케이션 정의 형식의 경우, 데이터 변환 엑시트에 의해 리턴된 값.

### CompCode

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_WARNING

경고(일부 완료).

#### MQCC\_FAILED

호출에 실패했습니다.

### 원인

유형: MQLONG - 출력

나열된 이유 코드는 큐 관리자가 **Reason** 매개변수에 대해 리턴할 수 있는 코드입니다. 애플리케이션이 MQGMO\_CONVERT 옵션을 지정하며 메시지 데이터의 일부 또는 전부를 변환하기 위해 사용자 작성 엑시트가 호출된 경우 엑시트는 **Reason** 매개변수에 대해 어떤 값이 리턴될지 결정합니다. 결과적으로 기록된 값 이외의 값이 가능합니다.

*CompCode*이(가) MQCC\_OK인 경우:

#### MQRC\_NONE

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_WARNING인 경우:

#### MQRC\_CONVERTED\_MSG\_TOO\_BIG

(2120, X'848') 변환된 데이터가 버퍼에 비해 너무 큽니다.

#### MQRC\_CONVERTED\_STRING\_TOO\_BIG

(2190, X'88E') 변환된 문자열이 필드에 비해 너무 큽니다.

#### MQRC\_DBCS\_ERROR

(2150, X'866') DBCS 문자열이 올바르지 않습니다.

#### MQRC\_FORMAT\_ERROR

(2110, X'83E') 메시지 형식이 올바르지 않습니다.

- MQRC\_INCOMPLETE\_GROUP**  
(2241, X'8C1') 메시지 그룹이 완료되지 않았습니다.
- MQRC\_INCOMPLETE\_MSG**  
(2242, X'8C2') 논리 메시지가 완료되지 않았습니다.
- MQRC\_INCONSISTENT\_CCSDS**  
(2243, X'8C3') 메시지 세그먼트에 서로 다른 CCSID가 있습니다.
- MQRC\_INCONSISTENT\_ENCODINGS**  
(2244, X'8C4') 메시지 세그먼트에 서로 다른 인코딩이 있습니다.
- MQRC\_INCONSISTENT\_UOW**  
(2245, X'8C5') 작업 단위 지정에 일관성이 없습니다.
- MQRC\_MSG\_TOKEN\_ERROR**  
(2331, X'91B') 메시지 토큰의 사용이 올바르지 않습니다.
- MQRC\_NO\_MSG\_LOCKED**  
(2209, X'8A1') 잠긴 메시지가 없습니다.
- MQRC\_NOT\_CONVERTED**  
(2119, X'847') 메시지 데이터가 변환되지 않았습니다.
- MQRC\_OPTIONS\_CHANGED**  
(nnnn, X'xxx') 일관성 있어야 하는 옵션이 변경되었습니다.
- MQRC\_PARTIALLY\_CONVERTED**  
(2272, X'8E0') 메시지 데이터가 부분적으로 변환되었습니다.
- MQRC\_SIGNAL\_REQUEST\_ACCEPTED**  
(2070, X'816') 리턴된 메시지가 없습니다(신호 요청은 승인됨).
- MQRC\_SOURCE\_BUFFER\_ERROR**  
(2145, X'861') 소스 버퍼 매개변수가 올바르지 않습니다.
- MQRC\_SOURCE\_CCSD\_ERROR**  
(2111, X'83F') 소스 코드화 문자 세트 ID가 올바르지 않습니다.
- MQRC\_SOURCE\_DECIMAL\_ENC\_ERROR**  
(2113, X'841') 메시지의 팩형 10진수 인코딩이 인식되지 않습니다.
- MQRC\_SOURCE\_FLOAT\_ENC\_ERROR**  
(2114, X'842') 메시지의 부동 소수점 인코딩이 인식되지 않습니다.
- MQRC\_SOURCE\_INTEGER\_ENC\_ERROR**  
(2112, X'840') 소스 정수 인코딩이 인식되지 않습니다.
- MQRC\_SOURCE\_LENGTH\_ERROR**  
(2143, X'85F') 소스 길이 매개변수가 올바르지 않습니다.
- MQRC\_TARGET\_BUFFER\_ERROR**  
(2146, X'862') 대상 버퍼 매개변수가 올바르지 않습니다.
- MQRC\_TARGET\_CCSD\_ERROR**  
(2115, X'843') 대상 코드화 문자 세트 ID가 올바르지 않습니다.
- MQRC\_TARGET\_DECIMAL\_ENC\_ERROR**  
(2117, X'845') 수신자가 지정한 팩형 10진수 인코딩을 인식할 수 없습니다.
- MQRC\_TARGET\_FLOAT\_ENC\_ERROR**  
(2118, X'846') 수신자에 의해 지정되는 부동 소수점 인코딩이 인식되지 않습니다.
- MQRC\_TARGET\_INTEGER\_ENC\_ERROR**  
(2116, X'844') 대상 정수 인코딩이 인식되지 않습니다.
- MQRC\_TRUNCATED\_MSG\_ACCEPTED**  
(2079, X'81F') 잘린 메시지가 리턴되었습니다(처리 완료됨).
- MQRC\_TRUNCATED\_MSG\_FAILED**  
(2080, X'820') 잘린 메시지가 리턴되었습니다(처리 미완료).

CompCode이(가) MQCC\_FAILED인 경우:

**MQRC\_ADAPTER\_NOT\_AVAILABLE**  
(2204, X'89C') 어댑터를 사용할 수 없습니다.

**MQRC\_ADAPTER\_CONV\_LOAD\_ERROR**  
(2133, X'855') 데이터 변환 서비스 모듈을 로드할 수 없습니다.

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**  
(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

**MQRC\_API\_EXIT\_ERROR**  
(2374, X'946') API 엑시트가 실패했습니다.

**MQRC\_API\_EXIT\_LOAD\_ERROR**  
(2183, X'887') API 엑시트를 로드할 수 없습니다.

**MQRC\_ASID\_MISMATCH**  
(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

**MQRC\_BACKED\_OUT**  
(2003, X'7D3') 작업 단위가 백아웃되었습니다.

**MQRC\_BUFFER\_ERROR**  
(2004, X'7D4') 버퍼 매개변수가 올바르지 않습니다.

**MQRC\_BUFFER\_LENGTH\_ERROR**  
(2005, X'7D5') 버퍼 길이 매개변수가 올바르지 않습니다.

**MQRC\_CALL\_IN\_PROGRESS**  
(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**MQRC\_CF\_NOT\_AVAILABLE**  
(2345, X'929') 커플링 기능이 사용 불가능합니다.

**MQRC\_CF\_STRUC\_FAILED**  
(2373, X'945') 커플링 기능 구조가 실패했습니다.

**MQRC\_CF\_STRUC\_IN\_USE**  
(2346, X'92A') 커플링 기능 구조가 사용 중입니다.

**MQRC\_CF\_STRUC\_LIST\_HDR\_IN\_USE**  
(2347, X'92B') 커플링 기능 구조 목록 헤더가 사용 중입니다.

**MQRC\_CICS\_WAIT\_FAILED**  
(2140, X'85C') CICS에서 대기 요청이 거부되었습니다.

**MQRC\_CONNECTION\_BROKEN**  
(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**MQRC\_CONNECTION\_NOT\_AUTHORIZED**  
(2217, X'8A9') 연결할 권한이 부여되지 않았습니다.

**MQRC\_CONNECTION QUIESCING**  
(2202, X'89A') 연결이 정지됩니다.

**MQRC\_CONNECTION\_STOPPING**  
(2203, X'89B') 연결이 종료됩니다.

**MQRC\_CORREL\_ID\_ERROR**  
(2207, X'89F') 상관 ID 오류입니다.

**MQRC\_DATA\_LENGTH\_ERROR**  
(2010, X'7DA') 데이터 길이 매개변수가 올바르지 않습니다.

**MQRC\_DB2\_NOT\_AVAILABLE**  
(2342, X'926') Db2 서브시스템이 사용 불가능합니다.

**MQRC\_GET\_INHIBITED**  
(2016, X'7E0') 큐에 대해 가져오기가 금지되었습니다.

**MQRC\_GLOBAL\_UOW\_CONFLICT**  
(2351, X'92F') 글로벌 작업 단위 충돌입니다.

**MQRC\_GMO\_ERROR**  
(2186, X'88A') 메시지 가져오기 옵션 구조가 올바르지 않습니다.

**MQRC\_HANDLE\_IN\_USE\_FOR\_UOW**  
(2353, X'931') 글로벌 작업 단위에 대해 사용 중인 핸들입니다.

**MQRC\_HCONN\_ERROR**  
(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**MQRC\_HOBJ\_ERROR**  
(2019, X'7E3') 오브젝트 핸들이 올바르지 않습니다.

**MQRC\_INCONSISTENT\_BROWSE**  
(2259, X'8D3') 찾아보기 지정에 일관성이 없습니다.

**MQRC\_INCONSISTENT\_UOW**  
(2245, X'8C5') 작업 단위 지정에 일관성이 없습니다.

**MQRC\_INVALID\_MSG\_UNDER\_CURSOR**  
(2246, X'8C6') 커서에 있는 메시지가 검색에 대해 유효하지 않습니다.

**MQRC\_LOCAL\_UOW\_CONFLICT**  
(2352, X'930') 글로벌 작업 단위가 로컬 작업 단위와 충돌합니다.

**MQRC\_MATCH\_OPTIONS\_ERROR**  
(2247, X'8C7') 일치 옵션이 올바르지 않습니다.

**MQRC\_MD\_ERROR**  
(2026, X'7EA') 메시지 디스크립터가 올바르지 않습니다.

**MQRC\_MSG\_ID\_ERROR**  
(2206, X'89E') 메시지 ID 오류입니다.

**MQRC\_MSG\_SEQ\_NUMBER\_ERROR**  
(2250, X'8CA') 메시지 순서 번호가 올바르지 않습니다.

**MQRC\_MSG\_TOKEN\_ERROR**  
(2331, X'91B') 메시지 토큰 사용이 올바르지 않습니다.

**MQRC\_NO\_MSG\_AVAILABLE**  
(2033, X'7F1') 사용 가능한 메시지가 없습니다.

**MQRC\_NO\_MSG\_UNDER\_CURSOR**  
(2034, X'7F2') 찾아보기 커서가 메시지에 위치해 있지 않습니다.

**MQRC\_NOT\_OPEN\_FOR\_BROWSE**  
(2036, X'7F4') 큐가 읽기 전용으로 열려있지 않습니다.

**MQRC\_NOT\_OPEN\_FOR\_INPUT**  
(2037, X'7F5') 큐가 입력을 위해 열려있지 않습니다.

**MQRC\_OBJECT\_CHANGED**  
(2041, X'7F9') 오브젝트가 열린 후에 정의가 변경되었습니다.

**MQRC\_OBJECT\_DAMAGED**  
(2101, X'835') 오브젝트가 손상되었습니다.

**MQRC\_OPTIONS\_ERROR**  
(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.

**MQRC\_PAGESET\_ERROR**  
(2193, X'891') 페이지 세트 데이터 세트에 액세스하는 중에 오류가 발생했습니다.

**MQRC\_Q\_DELETED**  
(2052, X'804') 큐가 삭제되었습니다.

**MQRC\_Q\_INDEX\_TYPE\_ERROR**  
(2394, X'95A') 큐의 색인 유형이 올바르지 않습니다.

**MQRC\_Q\_MGR\_NAME\_ERROR**  
(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**  
(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.

**MQRC\_Q\_MGR QUIESCING**  
(2161, X'871') 큐 관리자가 정지됩니다.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') 큐 관리자가 종료되었습니다.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**MQRC\_SECOND\_MARK\_NOT\_ALLOWED**

(2062, X'80E') 메시지가 이미 표시되었습니다.

**MQRC\_SIGNAL\_OUTSTANDING**

(2069, X'815') 이 핸들의 미해결 신호.

**MQRC\_SIGNAL1\_ERROR**

(2099, X'833') 신호 필드가 올바르지 않습니다.

**MQRC\_STORAGE\_MEDIUM\_FULL**

(2192, X'890') 외부 스토리지 매체가 가득 찼습니다.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**MQRC\_SUPPRESSED\_BY\_EXIT**

(2109, X'83D') 엑시트 프로그램에서 호출이 억제되었습니다.

**MQRC\_SYNCPOINT\_LIMIT\_REACHED**

(2024, X'7E8') 현재 작업 단위 내에서 더 이상 메시지를 핸들링할 수 없습니다.

**MQRC\_SYNCPOINT\_NOT\_AVAILABLE**

(2072, X'818') 동기점 지원을 사용할 수 없습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

**MQRC\_UOW\_ENLISTMENT\_ERROR**

(2354, X'932') 글로벌 작업 단위에서의 등록이 실패했습니다.

**MQRC\_UOW\_MIX\_NOT\_SUPPORTED**

(2355, X'933') 작업 단위 호출의 혼합은 지원되지 않습니다.

**MQRC\_UOW\_NOT\_AVAILABLE**

(2255, X'8CF') 사용할 큐 관리자에 대해 작업 단위를 사용할 수 없습니다.

**MQRC\_WAIT\_INTERVAL\_ERROR**

(2090, X'82A') MQGMO의 대기 간격이 올바르지 않습니다.

**MQRC\_WRONG\_GMO\_VERSION**

(2256, X'8D0') 올바르지 않은 버전의 MQGMO가 제공되었습니다.

**MQRC\_WRONG\_MD\_VERSION**

(2257, X'8D1') 올바르지 않은 버전의 MQMD가 제공되었습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

## 사용법 참고

1. 검색된 메시지는 일반적으로 큐에서 삭제됩니다. 이 삭제는 MQGET 호출 자체의 부분 또는 동기점의 부분으로 발생할 수 있습니다.

찾아보기 옵션은 MQGMO\_BROWSE\_FIRST, MQGMO\_BROWSE\_NEXT, MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR입니다.

2. MQGMO\_LOCK 옵션이 찾아보기 옵션 중 하나로 지정되면 찾아본 메시지는 이 핸들에서만 인식되도록 잠깁니다.

MQGMO\_UNLOCK 옵션이 지정되는 경우 이전에 잠긴 메시지의 잠금이 해제됩니다. 이 경우에 검색된 메시지가 없으며 **MsgDesc**, **BufferLength**, **Buffer**, **DataLength** 매개변수가 확인 또는 대체되지 않습니다.

3. MQGET 호출을 발행하는 애플리케이션의 경우 애플리케이션이 비정상적으로 종료되거나 호출을 처리하는 중에 연결이 끊어지는 경우에는 검색된 메시지가 손실될 수 있습니다. 이 문제는 메시지가 큐에서 제거된 후 애플리케이션 대신 MQGET 호출을 발행하는 큐 관리자와 동일한 플랫폼에서 실행 중인 대리에서 대리가 애

플리케이션에 메시지를 리턴할 때까지 애플리케이션의 손실을 감지할 수 없기 때문에 발생합니다. 이 문제는 지속 메시지와 비지속 메시지 모두에 대해 발생할 수 있습니다.

이러한 방식으로 메시지가 손실될 위험을 제거하려면 항상 작업 단위 내에서 메시지를 검색하십시오. 즉, MQGET 호출에서 MQGMO\_SYNCPOINT 옵션을 지정하고 메시지 처리가 완료되었을 때 작업 단위를 커밋하거나 백아웃하기 위해 MQCMIT 또는 MQBACK 호출을 사용하십시오. MQGMO\_SYNCPOINT가 지정되고 클라이언트가 비정상적으로 종료되거나 연결이 중단된 경우 대리가 큐 관리자에서 작업 단위를 백아웃하고 메시지가 큐에 복원됩니다. 동기점에 대한 자세한 정보는 [IBM MQ 애플리케이션의 동기점 고려사항](#)을 참조하십시오.

이 상황은 큐 관리자와 동일한 플랫폼에서 실행 중인 애플리케이션 및 IBM MQ 클라이언트에 대해 발생할 수 있습니다.

4. 애플리케이션이 단일 작업 단위 내의 특정큐에 메시지 시퀀스를 배치하고 작업 단위를 성공적으로 커밋하는 경우 다음과 같이 메시지를 검색에 사용할 수 있게 됩니다.

- 큐가 비공유 큐(즉, 로컬 큐)이면 작업 단위 내의 모든 메시지를 동시에 사용할 수 있게 됩니다.
- 큐가 공유 큐이면 작업 단위 내의 메시지가 동시에 모두가 아닌 놓여진 순서대로 사용할 수 있게 됩니다. 시스템의 로드가 큰 경우 작업 단위에 있는 첫 번째 메시지 검색에는 성공하지만 작업 단위의 두 번째 또는 후속 메시지에 대한 MQGET 호출은 MQRC\_NO\_MSG\_AVAILABLE과 함께 실패할 수 있습니다. 이 문제가 발생하는 경우 애플리케이션은 잠시 대기한 후 조작을 다시 시도해야 합니다.

5. 애플리케이션이 메시지 그룹을 사용하지 않고 동일 큐에 메시지 순서를 넣는 경우, 특정 조건을 충족하는 경우 이러한 메시지의 순서가 보존됩니다. 세부사항은 [MQPUT 사용법 참고](#)를 참조하십시오. 다음 조건을 충족하면 메시지가 송신된 순서대로 수신 애플리케이션에 제공됩니다.

- 한 수신자만 큐에서 메시지를 가져옵니다.

큐에서 메시지를 가져오는 애플리케이션이 둘 이상 있는 경우 순서에 속하는 메시지를 식별하는 데 사용할 메커니즘을 송신자와 동의해야 합니다. 예를 들어, 송신자가 일련의 메시지에서 모든 CorrelId 필드를 메시지의 해당 순서에 고유한 값으로 설정할 수도 있습니다.

- 수신자가 고의로(예: 특정 MsgId 또는 CorrelId를 지정하여) 검색 순서를 변경하지 않습니다.

송신 애플리케이션이 메시지를 메시지 그룹으로 넣는 경우 수신 애플리케이션이 MQGET 호출에서 MQGMO\_LOGICAL\_ORDER 옵션을 지정하면 메시지는 수신 애플리케이션에 올바른 순서로 제공됩니다. 메시지 그룹에 대한 자세한 정보는 다음을 참조하십시오.

- [MQMD - MsgFlags 필드](#)
- [MQPMO\\_LOGICAL\\_ORDER](#)
- [MQGMO\\_LOGICAL\\_ORDER](#)

사용자가 동기점 아래에 있는 그룹의 메시지를 가져오는 경우 트랜잭션 완료를 시도하기 전에 전체 그룹이 처리되었는지 확인해야 합니다.

6. 애플리케이션은 **MsgDesc** 매개변수의 Feedback 필드에서 피드백 코드 MQFB\_QUIT를 테스트하고 이 값을 찾으면 종료해야 합니다. 자세한 정보는 [MQMD - 피드백 필드](#)를 참조하십시오.

7. Hobj로 식별된 큐가 MQOO\_SAVE\_ALL\_CONTEXT 옵션으로 열렸으며 MQGET 호출의 완료 코드가 MQCC\_OK 또는 MQCC\_WARNING인 경우 큐 핸들 Hobj와 연관된 컨텍스트가 검색된 메시지의 컨텍스트로 설정됩니다(MQGMO\_BROWSE\_FIRST, MQGMO\_BROWSE\_NEXT 또는 MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR 옵션이 설정되지 않은 경우 컨텍스트가 사용 불가능으로 표시됨).

MQPMO\_PASS\_IDENTITY\_CONTEXT 또는 MQPMO\_PASS\_ALL\_CONTEXT 옵션을 지정하여 후속 MQPUT 또는 MQPUT1 호출에서 저장된 컨텍스트를 사용할 수 있습니다. 이렇게 하면 수신된 메시지의 컨텍스트를 전체 또는 일부로 다른 메시지에 전송할 수 있습니다(예: 메시지를 다른 큐에 전달하는 경우). 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트](#)를 참조하십시오.

8. **GetMsgOpts** 매개변수에 MQGMO\_CONVERT 옵션이 포함된 경우 데이터가 **Buffer** 매개변수에 배치되기 전에 애플리케이션 메시지 데이터는 수신 애플리케이션에서 요청한 표현으로 변환됩니다.

- 메시지의 제어 정보에 있는 Format 필드는 애플리케이션 데이터의 구조를 식별하고 메시지의 제어 정보에 있는 CodedCharSetId 및 Encoding 필드는 해당 문자 세트 ID와 인코딩을 지정합니다.



- MQGET 호출을 발행하는 애플리케이션은 **MsgDesc** 매개변수의 CodedCharSetId 및 Encoding 필드에 애플리케이션 메시지 데이터를 변환할 문자 세트 ID 및 인코딩을 지정합니다.

메시지 데이터 변환이 필요한 경우 메시지의 제어 정보에 있는 Format 필드의 값에 따라 큐 관리자 자체에서 또는 사용자 작성 엑시트에서 변환이 수행됩니다.

- 다음 형식 이름은 큐 관리자에 의해 변환되는 형식으로 "내장" 형식이라고 합니다.
  - MQFMT\_ADMIN
  - MQFMT\_CICS(z/OS 전용)
  - MQFMT\_COMMAND\_1
  - MQFMT\_COMMAND\_2
  - MQFMT\_DEAD\_LETTER\_HEADER
  - MQFMT\_DIST\_HEADER
  - MQFMT\_EVENT 버전 1
  - MQFMT\_EVENT 버전 2(z/OS 전용)
  - MQFMT\_IMS
  - MQFMT\_IMS\_VAR\_STRING
  - MQFMT\_MD\_EXTENSION
  - MQFMT\_PCF
  - MQFMT\_REF\_MSG\_HEADER
  - MQFMT\_RF\_HEADER
  - MQFMT\_RF\_HEADER\_2
  - MQFMT\_STRING
  - MQFMT\_TRIGGER
  - MQFMT\_WORK\_INFO\_HEADER(z/OS 전용)
  - MQFMT\_XMIT\_Q\_HEADER
- 형식 이름 MQFMT\_NONE은 메시지의 데이터 네이처가 정의되어 있지 않음을 표시하는 특수 값입니다. 따라서, 메시지를 큐에서 검색하면 큐 관리자가 변환을 시도하지 않습니다.

**참고:** 형식 이름이 MQFMT\_NONE인 메시지에 대해 MQGET 호출에서 MQGMO\_CONVERT가 지정되었으며 메시지의 문자 세트 또는 인코딩이 **MsgDesc** 매개변수에서 지정된 것과 다른 경우 메시지가 **Buffer** 매개변수에서 리턴되지만(다른 오류가 없다고 가정) 호출은 완료 코드 MQCC\_WARNING 및 이유 코드 MQRC\_FORMAT\_ERROR와 함께 완료됩니다.

메시지 데이터의 네이처가 변환을 요구하지 않는 경우 또는 전송과 수신 애플리케이션이 두 애플리케이션 간에 메시지 데이터 전송 형식에 동의한 경우 MQFMT\_NONE을 사용할 수 있습니다.

- 그 외 형식 이름은 모두 변환을 위해 메시지를 사용자 작성 엑시트에 전달합니다. 엑시트의 이름은 환경 특정 추가 항목 없이 형식과 동일합니다. 사용자 지정 형식 이름은 IBM MQ 문자로 시작할 수 없습니다.

데이터 변환 엑시트에 대한 세부사항은 869 페이지의 『데이터 변환 엑시트』의 내용을 참조하십시오.

메시지의 사용자 데이터는 지원되는 문자 세트와 인코딩 간에 변환될 수 있습니다. 그런 메시지가 하나 이상의 IBM MQ 헤더 구조를 포함하는 경우 메시지는 큐 이름에서 올바른 문자에 대해 2바이트 문자 또는 멀티 바이트 문자가 있는 문자 세트에서 또는 문자 세트로 변환할 수 없습니다. 변환을 시도하면 이유 코드 MQRC\_SOURCE\_CCSD\_ERROR 또는 MQRC\_TARGET\_CCSD\_ERROR가 발생하며 메시지는 변환되지 않은 채 리턴됩니다. 유니코드 문자 세트 UTF-16은 이러한 문자 세트의 예입니다.

MQGET에서의 리턴에서 다음 이유 코드는 메시지가 성공적으로 변환되었음을 표시합니다.

- MQRC\_NONE

다음 이유 코드는 메시지가 변환되었을 수 있음을 표시합니다. 애플리케이션은 **MsgDesc** 매개변수에서 CodedCharSetId 및 Encoding 필드를 확인하여 다음을 찾아야 합니다.

- MQRC\_TRUNCATED\_MSG\_ACCEPTED



기타 모든 이유 코드는 메시지가 변환되지 않았음을 표시합니다.

**참고:** 이 이유 코드의 해석은 엑시트가 869 페이지의 『데이터 변환 엑시트』에서 설명한 처리 지침을 준수한 경우에만 사용자 작성 엑시트에서 수행한 변환에 대해서 적용됩니다.

9. 객체 지향 인터페이스를 사용하여 메시지를 가져오는 경우 MQGET 호출에 대해 메시지 데이터를 보유할 버퍼를 지정하도록 선택하지 않을 수 있습니다. 수신 메시지 버퍼의 크기를 제한하지 않고 오브젝트 지향 애플리케이션을 사용하여 메시지를 가져오는 경우 애플리케이션은 MQRC\_CONVERTED\_MSG\_TOO\_BIG으로 실패하지 않으며 변환된 메시지를 수신합니다. 이는 다음 환경에서 적용됩니다.

- .NET(완전히 관리되는 애플리케이션 포함)
- C++
- Java (IBM MQ classes for Java)

**참고:** 모든 클라이언트에서 sharingConversations의 값이 0이고 버퍼가 변환된 메시지를 수신하기에 너무 작은 경우 변환되지 않은 메시지가 리턴되며 이유 코드는 MQRC\_CONVERTED\_MSG\_TOO\_BIG입니다. sharingConversations에 대한 자세한 정보는 클라이언트 애플리케이션에서 공유 대화 사용을 참조하십시오.

10. 내장 형식의 경우 MQGMO\_CONVERT 옵션이 지정되면 메시지에 있는 문자열의 기본 변환을 큐 관리자가 수행할 수 있습니다. 기본 변환을 사용하면 큐 관리자가 문자열 데이터 변환 시 실제 문자 세트에 가까운 설치 지정 기본 문자 세트를 사용할 수 있습니다. 그 결과로 MQGET 호출은 MQCC\_WARNING 및 이유 코드 MQRC\_SOURCE\_CCSD\_ERROR 또는 MQRC\_TARGET\_CCSD\_ERROR와 함께 완료되는 대신 완료 코드 MQCC\_OK와 함께 성공할 수 있습니다.

**참고:** 근사치의 문자 세트를 사용하여 문자열 데이터를 변환하면 일부 문자가 올바르게 않게 변환되는 결과가 발생할 수 있습니다. 이를 피하려면, 실제 문자 세트 및 기본 문자 세트 모두에 공통인 문자열의 문자를 사용하십시오.

기본 변환은 MQMD와 MQMDE 구조의 문자 필드 및 애플리케이션 메시지 데이터 모두에 적용됩니다.

- 애플리케이션 메시지 데이터의 기본 변환은 다음 명령문이 모두 true인 경우에만 발생합니다.
  - 애플리케이션이 MQGMO\_CONVERT를 지정합니다.
  - 메시지가 지원되지 않는 문자 세트로(부터) 변환되어야 하는 데이터를 포함합니다.
  - 큐 관리자를 설치하거나 재시작할 때 기본 변환을 사용하도록 설정되었습니다.
- 큐 관리자에 기본 변환이 사용되는 경우, 필요에 따라 MQMD 및 MQMDE 구조의 문자 필드가 기본 변환됩니다. MQGET 호출의 애플리케이션이 MQGMO\_CONVERT 옵션을 지정하지 않는 경우에서 변환이 수행됩니다.

11. Visual Basic 프로그래밍 언어의 경우 다음 사항이 적용됩니다.

- **Buffer** 매개변수의 크기가 **BufferLength** 매개변수에서 지정한 길이 미만인 경우 이유 코드 MQRC\_STORAGE\_NOT\_AVAILABLE와 함께 호출이 실패합니다.
- **Buffer** 매개변수는 유형 String으로 선언됩니다. 큐에서 검색할 데이터 유형이 String이 아닌 경우 MQGET 대신 MQGETAny 호출을 사용합니다.

MQGETAny 호출은 임의의 데이터 유형을 검색할 수 있도록 **Buffer** 매개변수가 유형 Any로 선언되었다는 점을 제외하고는 MQGET 호출과 동일한 매개변수를 갖습니다. 그러나 이는 BufferLength 바이트 이상의 크기인지 확인하기 위해 Buffer를 확인할 수 없음을 의미합니다.

12. 미리 읽기가 사용 가능한 경우 모든 MQGET 옵션이 지원되는 것은 아닙니다. 다음 테이블은 허용되는 옵션 및 MQGET 호출 간 대체 가능 여부를 표시합니다.

	미리 읽기가 사용 가능한 경우 허용되며 MQGET 호출 간에 대체 가능	미리 읽기가 사용 가능한 경우 허용되지만 MQGET 호출 간에 대체할 수 없음 <sup>a</sup>	미리 읽기가 사용 가능한 경우 허용되지 않는 MQGET 옵션 <sup>b</sup>
MQGET MD 값	MsgId <sup>c</sup> CorrelId <sup>c</sup>	Encoding CodedCharSetId	

표 548. 미리 읽기가 사용 가능한 경우 허용되는 MQGET 옵션 (계속)			
	미리 읽기가 사용 가능한 경우 허용되며 MQGET 호출 간에 대체 가능	미리 읽기가 사용 가능한 경우 허용되지만 MQGET 호출 간에 대체될 수 없음 <sup>a</sup>	미리 읽기가 사용 가능한 경우 허용되지 않는 MQGET 옵션 <sup>b</sup>
MQGET MQGMO 옵션	MQGMO_WAIT MQGMO_NO_WAIT MQGMO_FAIL_IF QUIESCING MQGMO_BROWSE_FIRST <sup>d</sup> MQGMO_BROWSE_NEXT <sup>d</sup> MQGMO_BROWSE_MESSAGE_UNDER_CURSOR <sup>d</sup>	MQGMO_SYNCPOINT_IF_PERSISTENT MQGMO_NO_SYNCPOINT MQGMO_ACCEPT_TRUNCATED_MSG MQGMO_CONVERT MQGMO_LOGICAL_ORDER MQGMO_COMPLETE_MSG MQGMO_ALL_MSGS_AVAILABLE MQGMO_ALL_SEGMENTS_AVAILABLE MQGMO_MARK_BROWSE_HANDLE MQGMO_MARK_BROWSE_CO_OP MQGMO_UNMARK_BROWSE_CO_OP MQGMO_UNMARK_BROWSE_HANDLE MQGMO_UNMARKED_BROWSE_MSG MQGMO_PROPERTIES_FORCE_MQRFH2 MQGMO_NO_PROPERTIES MQGMO_PROPERTIES_IN_HANDLE MQGMO_PROPERTIES_COMPATIBILITY	MQGMO_SET_SIGNAL MQGMO_SYNCPOINT MQGMO_MARK_SKIP_BACKOUT MQGMO_MSG_UNDER_CURSOR <sup>d</sup> MQGMO_LOCK MQGMO_UNLOCK
MQGMO 값		MsgHandle	

- a. 이 옵션이 MQGET 호출 간에 대체되면 MQRC\_OPTIONS\_CHANGED 이유 코드가 리턴됩니다.
  - b. 이 옵션을 첫 번째 MQGET 호출에 지정할 경우 미리 읽기를 사용할 수 없습니다. 이 옵션을 후속 MQGET 호출에 지정할 경우 이유 코드 MQRC\_OPTIONS\_ERROR가 리턴됩니다.
  - c. 클라이언트 애플리케이션은 MsgId 및 CorrelId 값이 MQGET 호출 간에 대체된 경우, 이전 값을 가진 메시지가 이미 클라이언트에 전송되어 이용(또는 자동으로 제거)될 때까지 클라이언트 미리 읽기 버퍼에 남아있는지 알아야 합니다.
  - d. 첫 번째 MQGET 호출은 메시지를 찾아볼지 아니면 미리 읽기가 가능할 때 큐에서 가져올지 여부를 판별합니다. 애플리케이션이 찾아보기와 가져오기의 결합을 사용하려고 할 경우 MQRC\_OPTIONS\_CHANGED 이유 코드가 리턴됩니다.
  - e. MQGMO\_MSG\_UNDER\_CURSOR는 미리 읽기에 사용할 수 없습니다. 메시지를 찾아보거나 미리 읽기가 가능할 때 가져올 수 있지만, 이 둘을 결합할 수는 없습니다.
13. 애플리케이션은 커밋되지 않은 메시지를 가져오기와 동일한 로컬 작업 단위에 넣는 경우에만 이를 파괴적으로 가져올 수 있습니다. 애플리케이션은 커밋되지 않은 메시지를 비파괴적으로 가져올 수 없습니다.
  14. 찾아보기 커서에서 메시지는 작업 단위로 검색할 수 있습니다. 이 방식으로는 커밋되지 않은 메시지를 검색할 수 없습니다.

## C 호출

```
MQGET (Hconn, Hobj, &MsgDesc, &GetMsgOpts, BufferLength, Buffer,
       &DataLength, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN  Hconn;           /* Connection handle */
MQHOBJ   Hobj;           /* Object handle */
MQMD     MsgDesc;       /* Message descriptor */
MQGMO    GetMsgOpts;    /* Options that control the action of MQGET */
MQLONG   BufferLength;   /* Length in bytes of the Buffer area */
MQBYTE   Buffer[n];     /* Area to contain the message data */
MQLONG   DataLength;    /* Length of the message */
MQLONG   CompCode;     /* Completion code */
MQLONG   Reason;       /* Reason code qualifying CompCode */
```

## COBOL 호출

```
CALL 'MQGET' USING HCONN, HOBJ, MSGDESC, GETMSGOPTS, BUFFERLENGTH,
BUFFER, DATALENGTH, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```
** Connection handle
01 HCONN PIC S9(9) BINARY.
```

```

** Object handle
01 HOBJ          PIC S9(9) BINARY.
** Message descriptor
01 MSGDESC.
   COPY CMQMDV.
** Options that control the action of MQGET
01 GETMSGOPTS.
   COPY CMQGMV.
** Length in bytes of the BUFFER area
01 BUFFERLENGTH PIC S9(9) BINARY.
** Area to contain the message data
01 BUFFER        PIC X(n).
** Length of the message
01 DATALENGTH  PIC S9(9) BINARY.
** Completion code
01 COMPCODE     PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON       PIC S9(9) BINARY.

```

## PL/I 호출

```

call MQGET (Hconn, Hobj, MsgDesc, GetMsgOpts, BufferLength, Buffer,
            DataLength, CompCode, Reason);

```

매개변수를 다음과 같이 선언합니다.

```

dcl Hconn          fixed bin(31); /* Connection handle */
dcl Hobj           fixed bin(31); /* Object handle */
dcl MsgDesc       like MQMD;     /* Message descriptor */
dcl GetMsgOpts    like MQGMO;    /* Options that control the action of
                                MQGET */
dcl BufferLength   fixed bin(31); /* Length in bytes of the Buffer
                                area */
dcl Buffer         char(n);       /* Area to contain the message data */
dcl DataLength    fixed bin(31); /* Length of the message */
dcl CompCode      fixed bin(31); /* Completion code */
dcl Reason        fixed bin(31); /* Reason code qualifying CompCode */

```

## 상위 레벨 어셈블러 호출

```

CALL MQGET, (HCONN,HOBJ,MSGDESC,GETMSGOPTS,BUFFERLENGTH,
            BUFFER,DATALENGTH,COMPCODE,REASON)

```

매개변수를 다음과 같이 선언합니다.

HCONN	DS	F	Connection handle
HOBJ	DS	F	Object handle
MSGDESC	CMQMDA	,	Message descriptor
GETMSGOPTS	CMQGMOA	,	Options that control the action of MQGET
BUFFERLENGTH	DS	F	Length in bytes of the BUFFER area
BUFFER	DS	CL(n)	Area to contain the message data
DATALENGTH	DS	F	Length of the message
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## Visual Basic 호출

```

MQGET Hconn, Hobj, MsgDesc, GetMsgOpts, BufferLength, Buffer,
DataLength, CompCode, Reason

```

매개변수를 다음과 같이 선언합니다.

```

Dim Hconn          As Long 'Connection handle'
Dim Hobj           As Long 'Object handle'

```

Dim MsgDesc	As MQMD	'Message descriptor'
Dim GetMsgOpts	As MQGMO	'Options that control the action of MQGET'
Dim BufferLength	As Long	'Length in bytes of the Buffer area'
Dim Buffer	As String	'Area to contain the message data'
Dim DataLength	As Long	'Length of the message'
Dim CompCode	As Long	'Completion code'
Dim Reason	As Long	'Reason code qualifying CompCode'

## MQINQ - 오브젝트 속성 조회

MQINQ 호출은 오브젝트의 속성이 포함된 문자열의 세트 및 정수의 배열을 리턴합니다.

올바른 오브젝트의 유형은 다음과 같습니다.

- 큐 관리자
- 큐
- 이름 목록
- 프로세스 정의

## 구문

MQINQ (*Hconn*, *Hobj*, *SelectorCount*, *Selectors*, *IntAttrCount*, *IntAttrs*, *CharAttrLength*, *CharAttrs*, *CompCode*, *Reason*)

## 매개변수

### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *Hconn* 값은 이전 MQCONN 또는 MQCONNX 호출로 리턴됩니다.

z/OS for CICS 애플리케이션에서 MQCONN 호출을 생략할 수 있으며 *Hconn*에 대해 다음 값을 지정할 수 있습니다.

#### MQHC\_DEF\_HCONN

기본 연결 핸들

### Hobj

유형: MQHOBJ - 입력

이 핸들은 필요한 속성을 포함한 오브젝트(임의 유형)를 표시합니다. 핸들은 MQOO\_INQUIRE 옵션을 지정한 이전 MQOPEN 호출로 리턴해야 합니다.

### SelectorCount

유형: MQLONG - 입력

이는 *Selectors* 배열에서 제공되는 선택자의 수입니다. 리턴될 속성의 수입니다. 0은 올바른 값입니다. 허용된 최대 수는 256입니다.

### Selectors

유형: MQLONG x *SelectorCount* - 입력

이는 **SelectorCount** 속성 선택자의 배열입니다. 각 선택자는 필요한 값이 있는 속성(정수 또는 문자)을 식별합니다.

각 선택자는 *Hobj*이(가) 표시하는 오브젝트 유형에 대해 유효해야 합니다. 그렇지 않으면 호출이 완료 코드 MQCC\_FAILED 및 이유 코드 MQRC\_SELECTOR\_ERROR과(와) 함께 실패합니다.

특수한 유형의 큐인 경우

- 선택자가 모든 유형의 큐에 유효하지 않은 경우 호출은 완료 코드 MQCC\_FAILED 및 이유 코드 MQRC\_SELECTOR\_ERROR과(와) 함께 실패합니다.

- 선택자가 오브젝트 유형이 아닌 다른 유형의 큐에만 적용되는 경우 호출은 완료 코드 MQCC\_WARNING 및 이유 코드 MQRC\_SELECTOR\_NOT\_FOR\_TYPE과(와) 함께 성공합니다.
- 조회되는 큐가 클러스터 큐인 경우 올바른 선택자는 큐가 해석되는 방법에 따라 다릅니다. 추가 세부사항은 693 페이지의 『사용법 참고』의 내용을 참조하십시오.

선택자는 원하는 순서대로 지정할 수 있습니다. 정수 속성 선택자(MQIA\_\* 선택자)에 해당하는 속성 값은 해당 선택자가 *Selectors*에서 발생하는 순서와 동일한 순서로 *IntAttrs*에 리턴됩니다. 문자 속성 선택자(MQCA\_\* 선택자)에 해당하는 속성 값은 해당 선택자가 발생하는 순서와 동일한 순서로 *CharAttrs*에 리턴됩니다. MQIA\_\* 선택자는 MQCA\_\* 선택자와 인터리브될 수 있습니다. 각 유형 내의 상대적인 순서만 중요합니다.

**참고:**

1. 정수 및 문자 속성 선택자는 두 가지 다른 범위 내에서 할당됩니다. MQIA\_\* 선택자는 MQIA\_FIRST에서 MQIA\_LAST까지의 범위 내에 있고 MQCA\_\* 선택자는 MQCA\_FIRST에서 MQCA\_LAST까지의 범위 내에 있습니다.  
각 범위에 대해 MQIA\_LAST\_USED 및 MQCA\_LAST\_USED 상수는 큐 관리자가 허용하는 최대값을 정의합니다.
2. 모든 MQIA\_\* 선택자가 먼저 발생하는 경우, 동일한 요소 번호를 사용하여 *Selectors* 및 *IntAttrs* 배열의 해당 요소를 처리할 수 있습니다.
3. **SelectorCount** 매개변수가 0인 경우 *Selectors*은(는) 참조되지 않습니다. 이 경우 C 또는 S/390 어셈블러에서 작성된 프로그램이 전달한 매개변수 주소는 널일 수 있습니다.

조회할 수 있는 속성이 다음 표에 나와 있습니다. MQCA\_\* 선택자의 경우, *CharAttrs*에서 결과 문자열의 길이(바이트 단위)를 정의하는 상수가 소괄호로 제공됩니다.

그 다음에 오는 테이블은 다음과 같이 선택자를 오브젝트별로 알파벳순으로 나열합니다.

- 큐의 681 페이지의 표 549 MQINQ 속성 선택자
- 이름 목록의 684 페이지의 표 550 MQINQ 속성 선택자
- 프로세스 정의의 684 페이지의 표 551 MQINQ 속성 선택자
- 큐 관리자의 684 페이지의 표 552 MQINQ 속성 선택자


표 549. 큐의 MQINQ 속성 선택자		
선택기	필드 길이	설명
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	가장 최근 대체 날짜
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	가장 최근 대체 시간
MQCA_BACKOUT_REQ_Q_NAME	MQ_Q_NAME_LENGTH	초과 백아웃 리큐 이름
MQCA_BASE_Q_NAME	MQ_Q_NAME_LENGTH	알리어스가 해석되는 큐의 이름
 MQCA_CF_STRUC_NAME	MQ_CF_STRUC_NAME_LENGTH	커플링 기능 구조 이름
MQCA_CLUS_CHL_NAME	MQ_CHANNEL_NAME_LENGTH	이 큐를 전송 큐로 사용하는 클러스터 송신자 채널의 이름입니다.
MQCA_CLUSTER_NAME	MQ_CLUSTER_NAME_LENGTH	클러스터 이름
MQCA_CLUSTER_NAMELIST	MQ_NAMELIST_NAME_LENGTH	클러스터 이름 목록
MQCA_CREATION_DATE	MQ_CREATION_DATE_LENGTH	큐 작성 날짜

표 549. 큐의 MQINQ 속성 선택자 (계속)

선택기	필드 길이	설명
MQCA_CREATION_TIME	MQ_CREATION_TIME_LENGTH	큐 작성 시간
MQCA_CUSTOM	MQ_CUSTOM_LENGTH	새 기능의 사용자 정의 속성
MQCA_INITIATION_Q_NAME	MQ_Q_NAME_LENGTH	이니시에이션 큐 이름
MQCA_PROCESS_NAME	MQ_PROCESS_NAME_LENGTH	프로세스 정의의 이름
MQCA_Q_DESC	MQ_Q_DESC_LENGTH	큐 설명
MQCA_Q_NAME	MQ_Q_NAME_LENGTH	큐 이름
MQCA_REMOTE_Q_MGR_NAME	MQ_Q_MGR_NAME_LENGTH	리모트 큐 관리자의 이름
MQCA_REMOTE_Q_NAME	MQ_Q_NAME_LENGTH	리모트 큐 관리자에서 알려진 리모트 큐의 이름
 MQCA_STORAGE_CLASS	MQ_STORAGE_CLASS_LENGTH	스토리지 클래스의 이름
MQCA_TRIGGER_DATA	MQ_TRIGGER_DATA_LENGTH	트리거 데이터
MQCA_XMIT_Q_NAME	MQ_Q_NAME_LENGTH	전송 큐 이름
 MQIA_ACCOUNTING_Q	MQLONG	큐에 대한 회계 데이터의 콜렉션 제어
MQIA_BACKOUT_THRESHOLD	MQLONG	백아웃 임계값
MQIA_CLWL_Q_PRIORITY	MQLONG	큐의 우선순위
MQIA_CLWL_Q_RANK	MQLONG	큐의 순위
MQIA_CLWL_USEQ	MQLONG	리모트 큐 사용
MQIA_CURRENT_Q_DEPTH	MQLONG	큐의 메시지 수
MQIA_DEF_BIND	MQLONG	기본 바인딩
MQIA_DEF_INPUT_OPEN_OPTION	MQLONG	기본 open-for-input 옵션
MQIA_DEF_PERSISTENCE	MQLONG	기본 메시지 지속성
MQIA_DEF_PRIORITY	MQLONG	기본 메시지 우선순위
MQIA_DEFINITION_TYPE	MQLONG	큐 정의 유형
 MQIA_DIST_LISTS	MQLONG	분배 목록 지원
MQIA_HARDEN_GET_BACKOUT	MQLONG	백아웃 수를 기록할지 여부
 MQIA_INDEX_TYPE	MQLONG	큐에 대해 유지보수되는 색인 유형
MQIA_INHIBIT_GET	MQLONG	가져오기 조작의 허용 여부
MQIA_INHIBIT_PUT	MQLONG	넣기 조작의 허용 여부
MQIA_MAX_MSG_LENGTH	MQLONG	최대 메시지 길이
MQIA_MAX_Q_DEPTH	MQLONG	큐에서 허용되는 최대 메시지 수

표 549. 큐의 MQINQ 속성 선택자 (계속)

선택기	필드 길이	설명
MQIA_MSG_DELIVERY_SEQUENCE	MQLONG	메시지 우선순위가 관련되는지 여부
MQIA_NPM_CLASS	MQLONG	비지속 메시지의 안전성 레벨
MQIA_OPEN_INPUT_COUNT	MQLONG	입력을 위해 큐를 연 MQOPEN 호출의 수
MQIA_OPEN_OUTPUT_COUNT	MQLONG	출력을 위해 큐를 연 MQOPEN 호출의 수
MQIA_PROPERTY_CONTROL	MQLONG	특성 제어 속성
 MQIA_Q_DEPTH_HIGH_EVENT	MQLONG	큐 용량 상한 이벤트에 대한 제어 속성
 MQIA_Q_DEPTH_HIGH_LIMIT	MQLONG	큐 용량에 대한 상한
 MQIA_Q_DEPTH_LOW_EVENT	MQLONG	큐 용량 하한 이벤트에 대한 제어 속성
 MQIA_Q_DEPTH_LOW_LIMIT	MQLONG	큐 용량에 대한 하한
 MQIA_Q_DEPTH_MAX_EVENT	MQLONG	큐 용량 최대 이벤트에 대한 제어 속성
 MQIA_Q_SERVICE_INTERVAL	MQLONG	큐 서비스 간격의 한계
 MQIA_Q_SERVICE_INTERVAL_EVENT	MQLONG	큐 서비스 간격 이벤트를 위한 제어 속성
MQIA_Q_TYPE	MQLONG	큐 유형
 MQIA_QSG_DISP	MQLONG	큐 공유 그룹 속성 지정
MQIA_RETENTION_INTERVAL	MQLONG	큐 보유 간격
 MQIA_SCOPE	MQLONG	큐 정의 범위
MQIA_SHAREABILITY	MQLONG	입력을 위해 큐를 공유할 수 있는지 여부
 MQIA_STATISTICS_Q	MQLONG	큐에 대한 통계 데이터의 콜렉션 제어
MQIA_TRIGGER_CONTROL	MQLONG	트리거 제어
MQIA_TRIGGER_DEPTH	MQLONG	트리거 용량
MQIA_TRIGGER_MSG_PRIORITY	MQLONG	트리거에 대한 임계값 메시지 우선순위
MQIA_TRIGGER_TYPE	MQLONG	트리거 유형
MQIA_USAGE	MQLONG	사용법



표 550. 이름 목록의 MQINQ 속성 선택자		
선택기	필드 길이	설명
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	최근 대체 날짜
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	최근 대체 시간
MQCA_NAMELIST_DESC	MQ_NAMELIST_DESC_LENGTH	이름 목록 설명
MQCA_NAMELIST_NAME	MQ_NAMELIST_NAME_LENGTH	이름 목록 오브젝트의 이름
 MQIA_NAMELIST_TYPE	MQLONG	이름 목록 유형
MQCA_NAMES	MQ_Q_NAME_LENGTH x Number of names in the list	이름 목록의 이름
MQIA_NAME_COUNT	MQLONG	이름 목록에 있는 이름 수
 MQIA_QSG_DISP	MQLONG	큐 공유 그룹 속성 지정


표 551. 프로세스 정의의 MQINQ 속성 선택자		
선택기	필드 길이	설명
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	최근 대체 날짜
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	최근 대체 시간
MQCA_APPL_ID	MQ_PROCESS_APPL_ID_LENGTH	애플리케이션 ID
MQCA_ENV_DATA	MQ_PROCESS_ENV_DATA_LENGTH	환경 데이터.
MQCA_PROCESS_DESC	MQ_PROCESS_DESC_LENGTH	프로세스 정의에 대한 설명
MQCA_PROCESS_NAME	MQ_PROCESS_NAME_LENGTH	프로세스 정의의 이름
MQCA_USER_DATA	MQ_PROCESS_USER_DATA_LENGTH	사용자 데이터
MQIA_APPL_TYPE	MQLONG	애플리케이션 유형
 MQIA_QSG_DISP	MQLONG	큐 공유 그룹 속성 지정

표 552. 큐 관리자의 MQINQ 속성 선택자		
선택기	필드 길이	설명
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	최근 대체 날짜
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	최근 대체 시간
MQCA_CHANNEL_AUTO_DEF_EXIT	MQ_EXIT_NAME_LENGTH	자동 채널 정의 엑시트 이름
MQCA_CHINIT_SERVICE_PARM		IBM에서 사용하도록 예약됨
MQCA_CLUSTER_WORKLOAD_DATA	MQ_EXIT_DATA_LENGTH	클러스터 워크로드 엑시트에 전달된 데이터
MQCA_CLUSTER_WORKLOAD_EXIT	MQ_EXIT_NAME_LENGTH	클러스터 워크로드 엑시트의 이름



표 552. 큐 관리자의 MQINQ 속성 선택자 (계속)

선택기	필드 길이	설명
MQCA_COMMAND_INPUT_Q_NAME	MQ_Q_NAME_LENGTH	시스템 명령 입력 큐 이름
MQCA_CUSTOM	MQ_CUSTOM_LENGTH	새 기능의 사용자 정의 속성
MQCA_DEAD_LETTER_Q_NAME	MQ_Q_NAME_LENGTH	데드-레터 큐의 이름입니다.
MQCA_DEF_XMIT_Q_NAME	MQ_Q_NAME_LENGTH	기본 전송 큐 이름입니다.
 MQCA_DNS_GROUP	MQ_DNS_GROUP_NAME_LENGTH	결합할 큐 공유 그룹에 대한 인바운드 전송을 핸들링하는 TCP 리스너에 대한 그룹의 이름입니다. 이름은 워크로드 관리자 동적 도메인 이름 서비스를 사용할 때 적용됩니다.
 MQCA_IGQ_USER_ID	MQ_USER_ID_LENGTH	그룹 내 큐잉 사용자 ID입니다.
MQCA_INITIAL_KEY	MQ_INITIAL_KEY_LENGTH	암호 보호 시스템의 초기 키 공백이 아닌 경우 ***** 를 리턴하고, 기본 초기 키가 사용 중인 경우 공백을 리턴합니다.
 MQCA_INSTALLATION_DESC	MQ_INSTALLATION_DESC_LENGTH	연관된 설치에 대한 설명
 MQCA_INSTALLATION_NAME	MQ_INSTALLATION_NAME_LENGTH	큐 관리자와 연관된 설치의 이름
 MQCA_INSTALLATION_PATH	MQ_INSTALLATION_PATH_LENGTH	연관된 IBM MQ가 설치된 경로
 MQCA_LU_GROUP_NAME	MQ_LU_NAME_LENGTH	사용할 큐 공유 그룹의 인바운드 전송을 핸들링하는 LU 6.2 리스너의 일반 LU 이름
 MQCA_LU_NAME	MQ_LU_NAME_LENGTH	아웃바운드 LU 6.2 전송에 사용할 LU의 이름. 이 이름을 리스너가 인바운드 전송에 사용하는 동일한 LU로 설정함
 MQCA_LU62_ARM_SUFFIX	MQ_ARM_SUFFIX_LENGTH	이 채널 시작기에 대한 LUADD를 지명하는 SYS1.PARMLIB 멤버 APPCPM <i>xxx</i> 의 접미부
MQCA_PARENT	MQ_Q_MGR_NAME_LENGTH	이 큐 관리자의 상위로 지정된 계층적으로 연결된 큐 관리자의 이름
MQCA_Q_MGR_DESC	MQ_Q_MGR_DESC_LENGTH	큐 관리자 설명입니다.
MQCA_Q_MGR_IDENTIFIER	MQ_Q_MGR_IDENTIFIER_LENGTH	큐 관리자 ID(H)
MQCA_Q_MGR_NAME	MQ_Q_MGR_NAME_LENGTH	로컬 큐 관리자의 이름
 MQCA_QSG_NAME	MQ_QSG_NAME_LENGTH	큐 공유 그룹 이름
MQCA_REPOSITORY_NAME	MQ_CLUSTER_NAME_LENGTH	큐 관리자가 저장소 서비스를 제공하는 클러스터의 이름

표 552. 큐 관리자의 MQINQ 속성 선택자 (계속)

선택기	필드 길이	설명
MQCA_REPOSITORY_NAMELIST	MQ_NAMELIST_NAME_LENGTH	큐 관리자가 저장소 서비스를 제공하는 클러스터의 이름을 포함하는 이름 목록 오브젝트의 이름
MQCA_SSL_KEY_REPO_PASSWORD	MQ_SSL_ENCRYPT_KEY_REPO_PWD_LEN	키 저장소 비밀번호 공백이 아닌 경우 ***** 를 리턴하고, 설정되지 않은 경우 공백을 리턴합니다. 저장하기 전에 설정할 때 암호화됩니다.
 MQCA_TCP_NAME	MQ_TCP_NAME_LENGTH	사용 중인 TCP/IP 시스템의 이름
 MQIA_ACCOUNTING_CONN_OVERRIDE	MQLONG	회계 설정 대체
 MQIA_ACCOUNTING_INTERVAL	MQLONG	중간 회계 레코드 작성 빈도
 MQIA_ACCOUNTING_MQI	MQLONG	MQI 데이터에 대한 회계 정보의 콜렉션 제어
 MQIA_ACCOUNTING_Q	MQLONG	큐에 대한 회계 정보의 콜렉션 제어
 MQIA_ACTIVE_CHANNELS	MQLONG	언제든지 활성화할 수 있는 최대 채널 수
 MQIA_ADOPTNEWMCA_CHECK	MQLONG	MCA 채택 여부를 판별하기 위해 검사하는 요소입니다. 이미 활성 상태인 MCA와 동일한 이름의 새 인바운드 채널이 감지된 경우 검사가 수행됩니다.
 MQIA_ADOPTNEWMCA_INTERVAL	MQLONG	새 채널에서 고아 채널이 종료될 때까지 대기하는 시간(초)
 MQIA_ADOPTNEWMCA_TYPE	MQLONG	AdoptNewMCACheck 매개변수와 일치하는 새 인바운드 채널 요청이 감지된 경우 특정 채널 유형의 고아 MCA 인스턴스를 자동으로 다시 시작할지 여부
 MQIA_AUTHORITY_EVENT	MQLONG	권한 이벤트에 대한 제어 속성
 MQIA_BRIDGE_EVENT	MQLONG	IMS 브릿지 이벤트에 대한 제어 속성
 MQIA_CHANNEL_AUTO_DEF	MQLONG	자동 채널 정의에 대한 제어 속성
 MQIA_CHANNEL_AUTO_DEF_EVENT	MQLONG	자동 채널 정의 이벤트에 대한 제어 속성

표 552. 큐 관리자의 MQINQ 속성 선택자 (계속)





선택기	필드 길이	설명
MQIA_CHANNEL_EVENT	MQLONG	채널 이벤트에 대한 제어 속성
 MQIA_CHINIT_ADAPTERS	MQLONG	IBM MQ 호출 처리에 사용할 어댑터 하위 태스크의 수
 MQIA_CHINIT_DISPATCHERS	MQLONG	채널 시작기에 사용할 디스패처의 수
 MQIA_CHINIT_TRACE_AUTO_START	MQLONG	채널 시작기 추적을 자동으로 시작할지 여부
 MQIA_CHINIT_TRACE_TABLE_SIZE	MQLONG	채널 시작기의 추적 데이터 공간 크기(MB)
MQIA_CLUSTER_WORKLOAD_LENGTH	MQLONG	클러스터 워크로드 길이
MQIA_CLWL_MRU_CHANNELS	MQLONG	클러스터 워크로드 밸런싱에 대해 가장 최근에 사용된 채널의 수입니다.
MQIA_CLWL_USEQ	MQLONG	리모트 큐 사용
MQIA_CODED_CHAR_SET_ID	MQLONG	코드화 문자 세트 ID
MQIA_COMMAND_EVENT	MQLONG	명령 이벤트에 대한 제어 속성
MQIA_COMMAND_LEVEL	MQLONG	큐 관리자가 지원하는 명령 레벨
 MQIA_CONFIGURATION_EVENT	MQLONG	구성 이벤트에 대한 제어 속성
MQIA_DEF_CLUSTER_XMIT_Q_TYPE	MQLONG	클러스터 송신자 채널에 사용할 기본 전송 큐 유형.
 MQIA_DIST_LISTS	MQLONG	분배 목록 지원
 MQIA_DNS_WLM	MQLONG	큐 공유 그룹에 대한 인바운드 전송을 핸들링하는 TCP 리스너를 Workload Manager for Dynamic Domain Name Services에 등록할지 여부
 MQIA_EXPIRY_INTERVAL	MQLONG	만기된 메시지에 대한 스캔 간의 간격입니다.
 MQIA_GROUP_UR	MQLONG	GROUP 복구 단위가 이 큐 관리자에서 사용 가능한지 여부에 대한 제어 속성. GROUP 복구 단위 처리는 큐 관리자가 큐 공유 그룹의 멤버인 경우에만 사용 가능합니다.
 MQIA_IGQ_PUT_AUTHORITY	MQLONG	그룹 내 큐잉 넣기 권한입니다.
 MQIA_INHIBIT_EVENT	MQLONG	금지 이벤트에 대한 제어 속성

표 552. 큐 관리자의 MQINQ 속성 선택자 (계속)











선택기	필드 길이	설명
 MQIA_INTRA_GROUP_QUEUING	MQLONG	그룹 내 큐잉 지원입니다.
 MQIA_LISTENER_TIMER	MQLONG	APPC 또는 TCP/IP가 실패한 경우 IBM MQ의 리스너 다시 시작 시도 사이의 시간 간격(초)
 MQIA_LOCAL_EVENT	MQLONG	로컬 이벤트에 대한 제어 속성
 MQIA_LOGGER_EVENT	MQLONG	금지 이벤트에 대한 제어 속성
 MQIA_LU62_CHANNELS	MQLONG	LU 6.2 전송 프로토콜을 사용하는, 현재 상태일 수 있는 최대 채널 수나 연결할 수 있는 최대 클라이언트 수
MQIA_MSG_MARK_BROWSE_INTERVAL	MQLONG	큐 관리자가 찾아보기 메시지에서 표시를 자동으로 제거할 수 있을 때까지의 시간 간격(밀리초)입니다.  <b>주의:</b> 절대로 이 값을 기본값 5000 미만으로 설정하지 마십시오.
 MQIA_MAX_CHANNELS	MQLONG	현재 상태일 수 있는 최대 채널 수(연결된 클라이언트가 있는 서버 연결 채널 포함)
MQIA_MAX_HANDLES	MQLONG	핸들의 최대 수입입니다.
MQIA_MAX_MSG_LENGTH	MQLONG	최대 메시지 길이
MQIA_MAX_PRIORITY	MQLONG	최대 우선순위입니다.
MQIA_MAX_UNCOMMITTED_MSGS	MQLONG	작업 단위 내에서 커밋되지 않은 최대 메시지 수입입니다.
 MQIA_OUTBOUND_PORT_MAX	MQLONG	MQIA_OUTBOUND_PORT_MIN을(를) 사용하는 경우, 전송 채널을 바인딩할 때 사용할 포트 번호의 범위를 정의합니다.
 MQIA_OUTBOUND_PORT_MIN	MQLONG	MQIA_OUTBOUND_PORT_MAX을(를) 사용하는 경우, 전송 채널을 바인딩할 때 사용할 포트 번호의 범위를 정의합니다.
 MQIA_PERFORMANCE_EVENT	MQLONG	성능 이벤트에 대한 제어 속성
MQIA_PLATFORM	MQLONG	큐 관리자가 상주하는 플랫폼
MQIA_PROT_POLICY_CAPABILITY	MQLONG	Advanced Message Security의 보안 기능이 큐 관리자에 사용 가능한지 여부를 표시합니다.
MQIA_PUBSUB_MAXMSG_RETRY_COUNT	MQLONG	동기점에서 실패한 명령 메시지의 재처리 시도 수



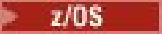
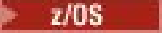










표 552. 큐 관리자의 MQINQ 속성 선택자 (계속)		
선택기	필드 길이	설명
MQIA_PUBSUB_MODE	MQLONG	발행/구독 엔진 및 큐된 발행/구독 인터페이스가 실행 중인지 여부입니다.  API(Application Programming Interface)를 사용하여 발행/구독할 애플리케이션에는 발행/구독 엔진이 필요합니다. 큐된 발행/구독 인터페이스가 모니터링하는 큐에서는 큐된 발행/구독 인터페이스를 실행해야 합니다.
MQIA_PUBSUB_NP_MSG	MQLONG	미배달 입력 메시지를 제거할지(또는 유지할지) 여부입니다.
MQIA_PUBSUB_NP_RESP	MQLONG	미배달 응답 메시지의 작동 제어
MQIA_PUBSUB_SYNC_PT	MQLONG	지속적(또는 모든) 메시지를 동기점에서 처리하는지 여부
 MQIA_QMGR_CFCONLOS	MQLONG	큐 관리자가 관리 구조 또는 CFCONLOS가 ASQMGR(으)로 설정된 CF 구조에 대한 연결이 끊긴 경우에 수행할 조치를 지정합니다.
 MQIA_RECEIVE_TIMEOUT	MQLONG	비활성 상태로 되돌아가기 전에 TCP/IP 채널이 파트너로부터 데이터(하트비트 포함)를 수신하기 위해 기다리는 대략적인 시간입니다. 값은 MQIA_RECEIVE_TIMEOUT_TYPE에 의해 규정된 숫자입니다.
 MQIA_RECEIVE_TIMEOUT_MIN	MQLONG	비활성 상태로 되돌아가기 전에 TCP/IP 채널이 파트너로부터 데이터(하트비트 포함)를 수신하기 위해 기다리는 최소 시간
 MQIA_RECEIVE_TIMEOUT_TYPE	MQLONG	비활성 상태로 되돌아가기 전에 TCP/IP 채널이 파트너로부터 데이터(하트비트 포함)를 수신하기 위해 기다리는 대략적인 시간입니다. MQIA_RECEIVE_TIMEOUT_TYPE은(는) MQIA_RECEIVE_TIMEOUT에 적용되는 규정자입니다.
 MQIA_REMOTE_EVENT	MQLONG	리모트 이벤트에 대한 제어 속성
 MQIA_SECURITY_CASE	MQLONG	보안 프로파일의 경우입니다.
MQIA_SSL_EVENT	MQLONG	채널 이벤트에 대한 제어 속성
MQIA_SSL_FIPS_REQUIRED	MQLONG	암호화를 위해 FIPS 인증 알고리즘만 사용
MQIA_SSL_RESET_COUNT	MQLONG	TLS 키 재설정 수
 MQIA_START_STOP_EVENT	MQLONG	시작 중지 이벤트에 대한 제어 속성
MQIA_STATISTICS_AUTO_CLUSSDR	MQLONG	클러스터 송신자 채널에 대한 통계 모니터링 정보의 콜렉션 제어

표 552. 큐 관리자의 MQINQ 속성 선택자 (계속)		
선택기	필드 길이	설명
MQIA_STATISTICS_CHANNEL	MQLONG	채널에 대한 통계 데이터의 콜렉션 제어
 MQIA_STATISTICS_INTERVAL	MQLONG	통계 모니터링 데이터 작성 빈도
 MQIA_STATISTICS_MQI	MQLONG	큐 관리자에 대한 통계 모니터링 정보의 콜렉션 제어
 MQIA_STATISTICS_Q	MQLONG	큐에 대한 통계 데이터의 콜렉션 제어
MQIA_SYNCPOINT	MQLONG	동기점 가용성
 MQIA_TCP_CHANNELS	MQLONG	TCP/IP 전송 프로토콜을 사용하는, 현재 상태일 수 있는 최대 채널 수나 연결할 수 있는 최대 클라이언트 수
 MQIA_TCP_KEEP_ALIVE	MQLONG	연결의 다른 측 끝이 여전히 사용 가능한지 확인하기 위해 TCP KEEPALIVE 기능을 사용할지 여부
 MQIA_TCP_STACK_TYPE	MQLONG	채널 시작기가 TCPNAME에 지정된 TCP/IP 주소 공간만 사용할 수 있는지 또는 선택적으로 선택된 TCP/IP 주소에 바인드할 수 있는지 여부
 MQIA_TRACE_ROUTE_RECORDING	MQLONG	추적 라우트 정보의 기록 제어
MQIA_TREE_LIFE_TIME	MQLONG	사용하지 않는 비관리 토픽의 지속 시간
MQIA_TRIGGER_INTERVAL	MQLONG	트리거 간격

### IntAttrCount

유형: MQLONG - 입력

이는 *IntAttrs* 배열에 있는 요소의 수입입니다. 0은 올바른 값입니다.

*IntAttrCount*가 **Selectors** 매개변수에 있는 MQIA\_\* 선택자 수 이상인 경우, 요청된 모든 정수 속성이 리턴됩니다.

### IntAttrs

유형: MQLONG x *IntAttrCount* - 출력

이는 *IntAttrCount* 정수 속성 값의 배열입니다.

정수 속성 값은 **Selectors** 매개변수의 MQIA\_\* 선택자와 동일한 순서로 리턴됩니다. 배열에 MQIA\_\* 선택자 수보다 많은 요소가 포함된 경우, 초과 요소는 변경되지 않습니다.

*Hobj*이(가) 큐를 나타내지만 속성 선택자가 해당 큐 유형에 적용되지 않는 경우, 특정 값 MQIAV\_NOT\_APPLICABLE이(가) 리턴됩니다. *IntAttrs* 배열의 해당 요소에 대해 리턴됩니다.

**IntAttrCount** 또는 **SelectorCount** 매개변수가 0이면 *IntAttrs*은(는) 참조되지 않습니다. 이 경우 C 또는 S/390 어셈블러에서 작성된 프로그램이 전달한 매개변수 주소는 널일 수 있습니다.

### CharAttrLength

유형: MQLONG - 입력

이는 **CharAttrs** 매개변수의 길이(바이트)입니다.

CharAttrLength는 요청된 문자 속성의 길이의 합계 이상이어야 합니다(선택자 참조). 0은 올바른 값입니다.

### CharAttrs

유형: MQCHAR x CharAttrLength - 출력

함께 연결된 문자 속성이 리턴되는 버퍼입니다. 버퍼의 길이는 **CharAttrLength** 매개변수로 제공됩니다.

문자 속성은 **Selectors** 매개변수의 MQCA\_\* 선택자와 동일한 순서로 리턴됩니다. 각 속성 문자열의 길이는 각 속성에 대해 고정되며(Selectors 선택자 참조) 필요한 경우 해당 값은 공백으로 오른쪽까지 채워집니다. 요청된 모든 문자 속성 및 채우기를 포함하는 데 필요한 것보다 더 많은 버퍼를 제공할 수 있습니다. 리턴된 마지막 속성 값을 초과하는 바이트는 변경되지 않습니다.

Hobj이(가) 큐를 나타내지만 속성 선택자가 해당 유형의 큐에 적용되지 않는 경우, 전체적으로 별표(\*)로 구성된 문자열이 리턴됩니다. 별표는 CharAttrs에서 해당 속성의 값으로 리턴됩니다.

CharAttrLength 또는 **SelectorCount** 매개변수가 0이면 CharAttrs은(는) 참조되지 않습니다. 이 경우 C 또는 S/390 어셈블러에서 작성된 프로그램이 전달한 매개변수 주소는 널일 수 있습니다.

### CompCode

유형: MQLONG - 출력

완료 코드:

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_WARNING

경고(일부 완료).

#### MQCC\_FAILED

호출에 실패했습니다.

### 원인

유형: MQLONG - 출력

CompCode이(가) MQCC\_OK인 경우:

#### MQRC\_NONE

(0, X'000') 보고할 이유가 없습니다.

CompCode이(가) MQCC\_WARNING인 경우:

#### MQRC\_CHAR\_ATTRS\_TOO\_SHORT

(2008, X'7D8') 문자 속성에 사용할 수 있는 공간이 충분하지 않습니다.

#### MQRC\_INT\_ATTR\_COUNT\_TOO\_SMALL

(2022, X'7E6') 정수 속성에 사용할 수 있는 공간이 충분하지 않습니다.

#### MQRC\_SELECTOR\_NOT\_FOR\_TYPE

(2068, X'814') 큐 유형에 적용할 수 없는 선택자입니다.

CompCode이(가) MQCC\_FAILED인 경우:

#### MQRC\_ADAPTER\_NOT\_AVAILABLE

(2204, X'89C') 어댑터를 사용할 수 없습니다.

#### MQRC\_ADAPTER\_SERV\_LOAD\_ERROR

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

#### MQRC\_API\_EXIT\_ERROR

(2374, X'946') API 엑시트가 실패했습니다.

#### MQRC\_API\_EXIT\_LOAD\_ERROR

(2183, X'887') API 엑시트를 로드할 수 없습니다.

#### MQRC\_ASID\_MISMATCH

(2157, X'86D') 기본 및 홈 ASID가 서로 다릅니다.

**MQRC\_CALL\_IN\_PROGRESS**  
 (2219, X'8AB') 이전 호출이 완료되기 전에 MQI 호출이 입력되었습니다.

**MQRC\_CF\_STRUC\_FAILED**  
 (2373, X'945') 결합 기능(CF) 구조가 실패했습니다.

**MQRC\_CF\_STRUC\_IN\_USE**  
 (2346, X'92A') 결합 기능(CF) 구조가 사용 중입니다.

**MQRC\_CHAR\_ATTR\_LENGTH\_ERROR**  
 (2006, X'7D6') 문자 속성의 길이가 올바르지 않습니다.

**MQRC\_CHAR\_ATTRS\_ERROR**  
 (2007, X'7D7') 문자 속성 문자열이 올바르지 않습니다.

**MQRC\_CICS\_WAIT\_FAILED**  
 (2140, X'85C') CICS이(가) 대기 요청을 거부했습니다.

**MQRC\_CONNECTION\_BROKEN**  
 (2009, X'7D9') 큐 관리자에 대한 연결이 끊어졌습니다.

**MQRC\_CONNECTION\_NOT\_AUTHORIZED**  
 (2217, X'8A9') 연결할 권한이 없습니다.

**MQRC\_CONNECTION\_STOPPING**  
 (2203, X'89B') 연결이 종료되었습니다.

**MQRC\_HCONN\_ERROR**  
 (2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**MQRC\_HOBJ\_ERROR**  
 (2019, X'7E3') 오브젝트 핸들이 유효하지 않습니다.

**MQRC\_INT\_ATTR\_COUNT\_ERROR**  
 (2021, X'7E5') 정수 속성의 수가 올바르지 않습니다.

**MQRC\_INT\_ATTRS\_ARRAY\_ERROR**  
 (2023, X'7E7') 정수 속성 배열이 올바르지 않습니다.

**MQRC\_NOT\_OPEN\_FOR\_INQUIRE**  
 (2038, X'7F6') 조회를 위해 큐가 열리지 않습니다.

**MQRC\_OBJECT\_CHANGED**  
 (2041, X'7F9') 열린 이후 오브젝트 정의가 변경되었습니다.

**MQRC\_OBJECT\_DAMAGED**  
 (2101, X'835') 오브젝트가 손상되었습니다.

**MQRC\_PAGESET\_ERROR**  
 (2193, X'891') 페이지 세트 데이터 세트에 액세스하는 중에 오류가 발생했습니다.

**MQRC\_Q\_DELETED**  
 (2052, X'804') 큐가 삭제되었습니다.

**MQRC\_Q\_MGR\_NAME\_ERROR**  
 (2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알려지지 않았습니다.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**  
 (2059, X'80B') 큐 관리자를 연결할 수 없습니다.

**MQRC\_Q\_MGR\_STOPPING**  
 (2162, X'872') 큐 관리자가 종료되었습니다.

**MQRC\_RESOURCE\_PROBLEM**  
 (2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**MQRC\_SELECTOR\_COUNT\_ERROR**  
 (2065, X'811') 선택자 수가 올바르지 않습니다.

**MQRC\_SELECTOR\_ERROR**  
 (2067, X'813') 속성 선택자가 올바르지 않습니다.



#### **MQRC\_SELECTOR\_LIMIT\_EXCEEDED**

(2066, X'812') 선택자의 수가 너무 큼니다.

#### **MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

#### **MQRC\_SUPPRESSED\_BY\_EXIT**

(2109, X'83D') 엑시트 프로그램에 의해 호출이 억제되었습니다.

#### **MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예기치 않은 오류가 발생했습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

## 사용법 참고

1. 리턴되는 값이 선택된 속성의 스냅샷입니다. 애플리케이션이 리턴된 값에 대해 수행하기 전에 속성이 그대로 남아 있도록 보장할 수 없습니다.
2. 모델 큐를 열면 동적 로컬 큐가 작성됩니다. 모델 큐를 열어 해당 속성을 조회하는 경우에도 동적 로컬 큐가 작성됩니다.

동적 큐의 속성은 동적 큐가 작성될 때 모델 큐의 속성과 대부분 동일합니다. 이 큐에서 MQINQ 호출을 사용하면 큐 관리자는 동적 큐의 속성을 리턴하지만 모델 큐의 속성은 리턴하지 않습니다. 동적 큐가 상속하는 모델 큐의 속성에 대한 세부사항은 [806 페이지의 표 561](#)의 내용을 참조하십시오.

3. 조회되는 오브젝트가 알리어스 큐인 경우 MQINQ 호출이 리턴하는 속성 값은 알리어스 큐의 속성입니다. 이는 알리어스가 해석하는 기본 큐 또는 토픽의 속성이 아닙니다.
4. 조회하는 오브젝트가 클러스터 큐이면 조회할 수 있는 속성은 큐를 여는 방법에 따라 다릅니다.

- 하나 이상의 입력, 찾아보기 또는 설정 조작 및 조회 작업에 대해 클러스터 큐를 열 수 있습니다. 이를 수행하려면 클러스터 큐의 로컬 인스턴스를 성공적으로 열 수 있어야 합니다. 이 경우 조회할 수 있는 속성은 로컬 큐에 올바른 속성입니다.

입력, 찾아보기 또는 설정이 지정되지 않은 조회를 위해 클러스터 큐가 열려 있는 경우, 로컬 큐에만 유효하고 클러스터 큐에는 유효하지 않은 속성을 조회하려고 시도하면 호출이 완료 코드 MQCC\_WARNING 및 이유 코드 MQRC\_SELECTOR\_NOT\_FOR\_TYPE(2068)을 리턴합니다.

- 연결된 큐 관리자의 기본 큐 관리자 이름을 전달하는 동안 조회할 클러스터 큐를 열 수 있습니다.

이를 수행하려면 클러스터 큐의 로컬 인스턴스를 성공적으로 열 수 있어야 합니다. 기본 큐 관리자가 전달되지 않은 경우, 로컬 큐에만 유효하고 클러스터 큐에는 유효하지 않은 속성을 조회하려고 시도하면 호출이 완료 코드 MQCC\_WARNING 및 이유 코드 MQRC\_SELECTOR\_NOT\_FOR\_TYPE(2068)을 리턴합니다.

- 조회 전용 또는 조회 및 출력을 위해 클러스터 큐를 여는 경우 나열된 속성만 조회할 수 있습니다. 이 경우 **QType** 속성에는 MQQT\_CLUSTER 값이 있습니다.

- MQCA\_Q\_DESC
- MQCA\_Q\_NAME
- MQIA\_DEF\_BIND
- MQIA\_DEF\_PERSISTENCE
- MQIA\_DEF\_PRIORITY
- MQIA\_INHIBIT\_PUT
- MQIA\_Q\_TYPE

고정된 바인딩 없이 클러스터 큐를 열 수 있습니다. MQOPEN 호출에 지정된 MQOO\_BIND\_NOT\_FIXED로 이를 열 수 있습니다. 또는 MQOO\_BIND\_AS\_Q\_DEF(를) 지정하고 큐의 **DefBind** 속성을 MQBND\_BIND\_NOT\_FIXED(으)로 설정하십시오. 고정된 바인딩 없이 클러스터 큐를 열면 큐에 대한 연속 MQINQ 호출이 클러스터 큐의 다른 인스턴스를 조회할 수 있습니다. 하지만 일반적으로 모든 인스턴스에 동일한 속성 값이 사용됩니다.

- 알리어스 큐 오브젝트를 클러스터에 대해 정의할 수 있습니다. TARGETYPE 및 TARGET이 클러스터 속성이 아니므로 알리어스 큐에서 MQOPEN 프로세스를 수행하는 프로세스는 알리어스가 해석하는 오브젝트를 인식하지 못합니다.

초기 MQOPEN 중에 알리어스 큐는 클러스터의 큐 및 큐 관리자를 해석합니다. 이름 해석은 리모트 큐 관리자에서 다시 이루어지며 여기에서 알리어스 큐의 TARGETYPE이 해석됩니다.

알리어스 큐가 토픽 알리어스를 해석하면 알리어스 큐에 넣은 메시지의 발행이 이 리모트 큐 관리자에서 이루어집니다.

클러스터 큐를 참조하십시오.

- 속성 수를 조회한 후 MQSET 호출을 사용하여 일부를 설정할 수 있습니다. 조회 및 설정을 효율적으로 프로그래밍하려면 선택자 배열의 시작 부분에 설정할 속성을 배치하십시오. 이를 수행하려는 경우 MQSET에 더 적은 수의 동일한 배열을 사용할 수 있습니다.
- 경고 상황 중 둘 이상이 발생하는 경우(**CompCode** 매개변수 참조), 리턴되는 이유 코드는 다음 목록의 첫 번째 이유 코드입니다.
  - MQRC\_SELECTOR\_NOT\_FOR\_TYPE
  - MQRC\_INT\_ATTR\_COUNT\_TOO\_SMALL
  - MQRC\_CHAR\_ATTRS\_TOO\_SHORT
- 다음 토픽에는 오브젝트 속성에 대한 정보가 있습니다.
  - 804 페이지의 『큐의 속성』
  - 834 페이지의 『이름 목록에 대한 속성』
  - 836 페이지의 『프로세스 정의에 대한 속성』
  - 768 페이지의 『큐 관리자의 속성』

## C 호출

```
MQINQ (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs, CharAttrLength, CharAttrs, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN  Hconn;          /* Connection handle */
MQHOBJ   Hobj;          /* Object handle */
MQLONG   SelectorCount; /* Count of selectors */
MQLONG   Selectors[n];  /* Array of attribute selectors */
MQLONG   IntAttrCount;  /* Count of integer attributes */
MQLONG   IntAttrs[n];   /* Array of integer attributes */
MQLONG   CharAttrLength; /* Length of character attributes buffer */
MQCHAR   CharAttrs[n];  /* Character attributes */
MQLONG   CompCode;      /* Completion code */
MQLONG   Reason;        /* Reason code qualifying CompCode */
```

## COBOL 호출

```
CALL 'MQINQ' USING HCONN, HOBJ, SELECTORCOUNT, SELECTORS-TABLE,
                  INTATTRCOUNT, INTATTRS-TABLE, CHARATTRLENGTH,
                  CHARATTRS, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```
** Connection handle
01 HCONN          PIC S9(9) BINARY.
** Object handle
01 HOBJ          PIC S9(9) BINARY.
** Count of selectors
01 SELECTORCOUNT PIC S9(9) BINARY.
```

```

** Array of attribute selectors
01 SELECTORS-TABLE.
02 SELECTORS PIC S9(9) BINARY OCCURS n TIMES.
** Count of integer attributes
01 INTATTRCOUNT PIC S9(9) BINARY.
** Array of integer attributes
01 INTATTRS-TABLE.
02 INTATTRS PIC S9(9) BINARY OCCURS n TIMES.
** Length of character attributes buffer
01 CHARATTRLENGTH PIC S9(9) BINARY.
** Character attributes
01 CHARATTRS PIC X(n).
** Completion code
01 COMPCODE PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON PIC S9(9) BINARY.

```

## PL/I 호출

```
call MQINQ (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount,
           IntAttris, CharAttrLength, CharAttris, CompCode, Reason);
```

매개변수를 다음과 같이 선언합니다.

```

dcl Hconn          fixed bin(31); /* Connection handle */
dcl Hobj           fixed bin(31); /* Object handle */
dcl SelectorCount  fixed bin(31); /* Count of selectors */
dcl Selectors(n)   fixed bin(31); /* Array of attribute selectors */
dcl IntAttrCount   fixed bin(31); /* Count of integer attributes */
dcl IntAttris(n)   fixed bin(31); /* Array of integer attributes */
dcl CharAttrLength fixed bin(31); /* Length of character attributes
                                   buffer */
dcl CharAttris     char(n);       /* Character attributes */
dcl CompCode       fixed bin(31); /* Completion code */
dcl Reason         fixed bin(31); /* Reason code qualifying
                                   CompCode */

```

## 상위 레벨 어셈블러 호출

```
CALL MQINQ, (HCONN, HOBJ, SELECTORCOUNT, SELECTORS, INTATTRCOUNT, X
           INTATTRS, CHARATTRLENGTH, CHARATTRS, COMPCODE, REASON)
```

매개변수를 다음과 같이 선언합니다.

```

HCONN      DS F      Connection handle
HOBJ       DS F      Object handle
SELECTORCOUNT DS F      Count of selectors
SELECTORS  DS (n)F   Array of attribute selectors
INTATTRCOUNT DS F      Count of integer attributes
INTATTRS   DS (n)F   Array of integer attributes
CHARATTRLENGTH DS F      Length of character attributes buffer
CHARATTRS  DS CL(n)  Character attributes
COMPCODE   DS F      Completion code
REASON     DS F      Reason code qualifying COMPCODE

```

## Visual Basic 호출

```
MQINQ Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttris,
      CharAttrLength, CharAttris, CompCode, Reason
```

매개변수를 다음과 같이 선언합니다.

```

Dim Hconn As Long 'Connection handle'
Dim Hobj As Long 'Object handle'

```

Dim SelectorCount	As Long	'Count of selectors'
Dim Selectors	As Long	'Array of attribute selectors'
Dim IntAttrCount	As Long	'Count of integer attributes'
Dim IntAttrs	As Long	'Array of integer attributes'
Dim CharAttrLength	As Long	'Length of character attributes buffer'
Dim CharAttrs	As String	'Character attributes'
Dim CompCode	As Long	'Completion code'
Dim Reason	As Long	'Reason code qualifying CompCode'

## MQINQMP - 메시지 특성 조회

MQINQMP 호출은 메시지의 특성 값을 리턴합니다.

### 구문

MQINQMP(*Hconn*, *Hmsg*, *InqPropOpts*, *Name*, *PropDesc*, *Type*, *ValueLength*, *Value*, *DataLength*, *CompCode*, *Reason*)

### 매개변수

#### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *Hconn* 값이 **Hmsg** 매개변수에 지정된 메시지 핸들 작성에 사용된 연결 핸들과 일치해야 합니다.

메시지 핸들이 MQHC\_UNASSOCIATED\_HCONN을 사용하여 작성된 경우 메시지 핸들의 특성을 조회하는 스레드에서 올바른 연결을 설정해야 합니다. 그렇지 않으면 호출이 MQRC\_CONNECTION\_BROKEN과 함께 실패합니다.

#### Hmsg

유형: MQHMSG - 입력

조회할 메시지 핸들입니다. 이 값은 이전 **MQCRTMH** 호출을 통해 리턴됩니다.

#### InqPropOpts

유형: MQIMPO - 입출력(I/O)

자세한 정보는 [MQIMPO](#) 데이터 유형을 참조하십시오.

#### 이름

유형: MQCHARV - 입출력(I/O)

조회할 특성의 이름입니다.

이 이름과 일치하는 특성을 찾을 수 없는 경우 이유 MQRC\_PROPERTY\_NOT\_AVAILABLE과 함께 호출이 실패합니다.

특성 이름의 끝에 와일드카드 문자 퍼센트 부호(%)를 사용할 수 있습니다. 와일드카드는 마침표(.) 문자를 포함하여 0개 이상의 문자와 일치합니다. 와일드카드를 사용하면 애플리케이션에서 많은 특성의 값을 조회할 수 있습니다. 첫 번째 일치하는 특성을 가져오기 위한 옵션 MQIMPO\_INQ\_FIRST 및 다음 일치하는 특성을 가져오기 위한 옵션 MQIMPO\_INQ\_NEXT와 함께 MQINQMP를 호출하십시오. 사용 가능한 일치 특성이 더 이상 없는 경우 MQRC\_PROPERTY\_NOT\_AVAILABLE과 함께 호출이 실패합니다. InqPropOpts 구조의 *ReturnedName* 필드가 리턴된 특성 이름의 오프셋 또는 주소로 초기화되는 경우, 일치한 특성의 이름과 함께 MQINQMP의 리턴에서 완료됩니다. InqPropOpts 구조에서 *ReturnedName*의 *VSBufSize* 필드가 리턴된 특성 이름의 길이 미만인 경우 완료 코드는 이유 MQRC\_PROPERTY\_NAME\_TOO\_BIG과 함께 MQCC\_FAILED로 설정됩니다.

알려진 동의어가 있는 특성은 다음과 같이 리턴됩니다.

1. 접두부 "mqps."가 있는 특성이 IBM MQ 특성 이름으로 리턴됩니다. 예를 들어, "MQTopicString"은 "mqps.Top"이 아닌 리턴된 이름입니다.
2. 접두부 "jms." 또는 "mcd."가 있는 특성이 JMS 헤더 필드 이름으로 리턴됩니다. 예를 들어, "JMSExpiration"은 "jms.Exp"가 아닌 리턴된 이름입니다.

3. 접두부 "usr."가 있는 특성이 해당 접두부 없이 리턴됩니다. 예를 들면, "Color"가 "usr.Color" 대신 리턴됩니다.

동义어가 있는 특성은 한 번만 리턴됩니다.

C 프로그래밍 언어에서 다음 매크로 변수는 모든 속성을 조회한 다음 "usr."로 시작하는 모든 속성을 조회하기 위해 정의됩니다.

#### **MQPROP\_INQUIRE\_ALL**

메시지의 모든 특성을 조회합니다.

MQPROP\_INQUIRE\_ALL은 다음 방법으로 사용할 수 있습니다.

```
MQCHARV Name = {MQPROP_INQUIRE_ALL};
```

#### **MQPROP\_INQUIRE\_ALL\_USR**

"usr"로 시작하는 메시지의 모든 특성을 조회하십시오. 리턴되는 이름은 "usr." 접두부 없이 리턴됩니다.

MQIMP\_INQ\_NEXT가 지정되지만 이름이 이전 호출 이후로 변경되었거나 이 호출이 첫 번째 호출인 경우 MQIMPO\_INQ\_FIRST를 의미합니다.

특성 이름 사용에 대한 자세한 정보는 [특성 이름 및 특성 이름 제한사항](#)을 참조하십시오.

#### **PropDesc**

입력: MQPD - 출력

이 구조는 특성이 지원되지 않을 때 일어나는 일, 특성이 속한 메시지 컨텍스트, 특성이 복사되어야 하는 메시지 등 특성의 속성을 정의하는 데 사용됩니다. 이 구조에 대한 자세한 정보는 [MQPD](#)를 참조하십시오.

#### **유형**

유형: MQLONG - 입출력(I/O)

MQINQMP 호출의 리턴에서 이 매개변수는 *Value* 데이터 유형으로 설정됩니다. 데이터 유형은 다음 중 하나일 수 있습니다.

#### **MQTYPE\_BOOLEAN**

부울.

#### **MQTYPE\_BYTE\_STRING**

바이트 문자열.

#### **MQTYPE\_INT8**

8비트 서명 정수입니다.

#### **MQTYPE\_INT16**

16비트 서명 정수입니다.

#### **MQTYPE\_INT32**

32비트 서명 정수입니다.

#### **MQTYPE\_INT64**

64비트 서명 정수입니다.

#### **MQTYPE\_FLOAT32**

32비트 부동 소수점 숫자입니다.

#### **MQTYPE\_FLOAT64**

64비트 부동 소수점 숫자입니다.

#### **MQTYPE\_STRING**

문자열.

#### **MQTYPE\_NULL**

특성이 존재하지만 값이 널입니다.

특성 값의 데이터 유형이 인식되지 않는 경우 MQTYPE\_STRING이 리턴되고 값의 문자열 표현은 *Value* 영역에 배치됩니다. 데이터 유형의 문자열 표현은 *InqPropOpts* 매개변수의 *TypeString* 필드에서 찾을 수 있습니다. 경고 완료 코드는 이유 MQRC\_PROP\_TYPE\_NOT\_SUPPORTED와 함께 리턴됩니다.

또한 옵션 MQIMPO\_CONVERT\_TYPE이 지정되면 특성 값의 변환이 요청됩니다. 리턴되는 특성에 대해 원하는 데이터 유형을 지정하려면 *Type*을 입력으로 사용하십시오. 데이터 유형 변환에 대한 세부사항은 MQIMPO 구조의 MQIMPO\_CONVERT\_TYPE 옵션에 대한 설명을 참조하십시오.

유형 변환을 요청하지 않은 경우, 입력에 다음 값을 사용할 수 있습니다.

#### **MQTYPE\_AS\_SET**

데이터 유형을 변환하지 않고 특성 값이 리턴됩니다.

#### **ValueLength**

유형: MQLONG - 입력

값 영역의 길이(바이트)입니다. 값이 리턴될 필요가 없는 특성에 대해서는 0을 지정하십시오. 이들 특성은 널 값이나 빈 문자열을 갖도록 애플리케이션이 설계한 특성일 수 있습니다. MQIMPO\_QUERY\_LENGTH 옵션이 지정된 경우에도 0을 지정하십시오. 이 경우 값이 리턴되지 않습니다.

#### **값**

유형: MQBYTEx *ValueLength* - 출력

조회한 특성 값을 포함할 영역입니다. 버퍼는 리턴되는 값에 적절한 경계에 맞춰야 합니다. 그렇게 하지 않으면 나중에 값에 액세스하는 경우 오류가 발생합니다.

*ValueLength*가 특성 값의 길이 미만인 경우 최대한 많은 특성 값이 *Value*로 이동되고 완료 코드 MQCC\_FAILED 및 이유 MQRC\_PROPERTY\_VALUE\_TOO\_BIG과 함께 호출이 실패합니다.

*Value*에서 데이터의 문자 세트는 InqPropOpts 매개변수의 ReturnedCCSID 필드로 지정됩니다. *Value*의 데이터 인코딩은 InqPropOpts 매개변수의 ReturnedEncoding 필드로 지정됩니다.

C 프로그래밍 언어에서 매개변수는 pointer-to-void로서 선언됩니다. 임의의 데이터 유형의 주소를 매개변수로 지정할 수 있습니다.

*ValueLength* 매개변수가 0인 경우 *Value*가 참조되지 않고 C 또는 System/390 어셈블러에서 작성된 프로그램이 전달하는 해당 값이 널일 수 있습니다.

#### **DataLength**

유형: MQLONG - 출력

*Value* 영역에 리턴되는 실제 특성 값의 길이(바이트)입니다.

*DataLength*가 특성 값 길이 미만인 경우에도 *DataLength*는 MQINQMP 호출에서 리턴 시 채워집니다. 따라서 애플리케이션이 특성 값을 수용하는 데 필요한 버퍼의 크기를 판별하고 적절한 크기의 버퍼를 사용하여 호출을 다시 실행할 수 있게 됩니다.

다음 값을 리턴할 수도 있습니다.

*Type* 매개변수가 MQTYPE\_STRING 또는 MQTYPE\_BYTE\_STRING으로 설정되는 경우:

#### **MQVL\_EMPTY\_STRING**

특성이 존재하지만, 문자나 바이트를 포함하지 않습니다.

#### **CompCode**

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

#### **MQCC\_OK**

정상적으로 완료되었습니다.

#### **MQCC\_WARNING**

경고(일부 완료).

#### **MQCC\_FAILED**

호출에 실패했습니다.

#### **원인**

유형: MQLONG - 출력

*CompCode*이(가) MQCC\_OK인 경우:

**MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_WARNING인 경우:

**MQRC\_PROP\_NAME\_NOT\_CONVERTED**

(2492, X'09BC') 리턴된 특성 이름이 변환되지 않습니다.

**MQRC\_PROP\_VALUE\_NOT\_CONVERTED**

(2466, X'09A2') 특성 값이 변환되지 않습니다.

**MQRC\_PROP\_TYPE\_NOT\_SUPPORTED**

(2467, X'09A3') 특성 데이터 유형이 지원되지 않습니다.

**MQRC\_RFH\_FORMAT\_ERROR**

(2421, X'0975') 특성을 포함하는 MQRFH2 폴더를 구문 분석할 수 없습니다.

*CompCode*이(가) MQCC\_FAILED인 경우:

**MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'089C') 어댑터를 사용할 수 없습니다.

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'0852') 어댑터 서비스 모듈을 로드할 수 없습니다.

**MQRC\_ASID\_MISMATCH**

(2157, X'086D') 1차 및 홈 ASID가 다릅니다.

**MQRC\_BUFFER\_ERROR**

(2004, X'07D4') 값 매개변수가 올바르지 않습니다.

**MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'07D5') 값 길이 매개변수가 올바르지 않습니다.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'08AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'07D9') 큐 관리자에 대한 연결이 유실되었습니다.

**MQRC\_DATA\_LENGTH\_ERROR**

(2010, X'07DA') 데이터 길이 매개변수가 올바르지 않습니다.

**MQRC\_IMPO\_ERROR**

(2464, X'09A0') 메시지 특성 조회 옵션 구조가 올바르지 않습니다.

**MQRC\_HMSG\_ERROR**

(2460, X'099C') 메시지 핸들이 올바르지 않습니다.

**MQRC\_MSG\_HANDLE\_IN\_USE**

(2499, X'09C3') 메시지 핸들이 이미 사용 중입니다.

**MQRC\_OPTIONS\_ERROR**

(2046, X'07F8') 옵션이 올바르지 않거나 일치하지 않습니다.

**MQRC\_PD\_ERROR**

(2482, X'09B2') 특성 디스크립터 구조가 올바르지 않습니다.

**MQRC\_PROP\_CONV\_NOT\_SUPPORTED**

(2470, X'09A6') 실제 항목에서 요청된 데이터 유형으로 변환할 수 없습니다.

**MQRC\_PROPERTY\_NAME\_ERROR**

(2442, X'098A') 올바르지 않은 특성 이름입니다.

**MQRC\_PROPERTY\_NAME\_TOO\_BIG**

(2465, X'09A1') 리턴된 이름 버퍼에 비해 특성 이름이 너무 큼니다.

**MQRC\_PROPERTY\_NOT\_AVAILABLE**

(2471, X'09A7') 특성을 사용할 수 없습니다.

**MQRC\_PROPERTY\_VALUE\_TOO\_BIG**

(2469, X'09A5') 값 영역에 비해 특성 값이 너무 큼니다.

**MQRC\_PROP\_NUMBER\_FORMAT\_ERROR**

(2472, X'09A8') 숫자 형식 오류가 값 데이터에서 발견되었습니다.

**MQRC\_PROPERTY\_TYPE\_ERROR**

(2473, X'09A9') 요청된 특성 유형이 올바르지 않습니다.

**MQRC\_SOURCE\_CCSID\_ERROR**

(2111, X'083F') 특성 이름 코드화 문자 세트 ID가 올바르지 않습니다.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'0871') 사용 가능한 스토리지가 충분하지 않습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'0893') 예상치 못한 오류가 발생했습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

**C 호출**

```
MQINQMP (Hconn, Hmsg, &InqPropOpts, &Name, &PropDesc, &Type,
ValueLength, Value, &DataLength, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN Hconn;          /* Connection handle */
MQHMSG Hmsg;            /* Message handle */
MQIMPO InqPropOpts;    /* Options that control the action of MQINQMP */
MQCHARV Name;         /* Property name */
MQPD PropDesc;        /* Property descriptor */
MQLONG Type;          /* Property data type */
MQLONG ValueLength;   /* Length in bytes of the Value area */
MQBYTE Value[n];     /* Area to contain the property value */
MQLONG DataLength;   /* Length of the property value */
MQLONG CompCode;     /* Completion code */
MQLONG Reason;       /* Reason code qualifying CompCode */
```

**COBOL 호출**

```
CALL 'MQINQMP' USING HCONN, HMSG, INQMSGOPTS, NAME, PROPDESC, TYPE,
VALUELENGTH, VALUE, DATALENGTH, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```
** Connection handle
01 HCONN          PIC S9(9) BINARY.
** Message handle
01 HMSG           PIC S9(18) BINARY.
** Options that control the action of MQINQMP
01 INQMSGOPTS.
   COPY CMQIMPOV.
** Property name
01 NAME.
   COPY CMQCHRVV.
** Property descriptor
01 PROPDESC.
   COPY CMQPDV.
** Property data type
01 TYPE          PIC S9(9) BINARY.
** Length in bytes of the VALUE area
01 VALUELENGTH  PIC S9(9) BINARY.
** Area to contain the property value
01 VALUE        PIC X(n).
** Length of the property value
01 DATALENGTH  PIC S9(9) BINARY.
** Completion code
01 COMPCODE     PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON       PIC S9(9) BINARY.
```



## PL/I 호출

```
call MQINQMP (Hconn, Hmsg, InqPropOpts, Name, PropDesc, Type,
ValueLength, Value, DataLength, CompCode, Reason);
```

매개변수를 다음과 같이 선언합니다.

```
dc1 Hconn          fixed bin(31); /* Connection handle */
dc1 Hmsg           fixed bin(63); /* Message handle */
dc1 InqPropOpts    like MQIMPO; /* Options that control the action of MQINQMP */
dc1 Name           like MQCHARV; /* Property name */
dc1 PropDesc       like MQPDA; /* Property descriptor */
dc1 Type           fixed bin (31); /* Property data type */
dc1 ValueLength    fixed bin (31); /* Length in bytes of the Value area */
dc1 Value          char (n); /* Area to contain the property value */
dc1 DataLength     fixed bin (31); /* Length of the property value */
dc1 CompCode       fixed bin (31); /* Completion code */
dc1 Reason         fixed bin (31); /* Reason code qualifying CompCode */
```

## 상위 레벨 어셈블러 호출

```
CALL MQINQMP, (HCONN, HMSG, INQMSGOPTS, NAME, PROPDSC, TYPE,
VALUELENGTH, VALUE, DATALENGTH, COMPCODE, REASON)
```

매개변수를 다음과 같이 선언합니다.

HCONN	DS	F	Connection handle
HMSG	DS	D	Message handle
INQMSGOPTS	CMQIMPOA	,	Options that control the action of MQINQMP
NAME	CMQCHRVA	,	Property name
PROPDSC	CMQPDA	,	Property descriptor
TYPE	DS	F	Property data type
VALUELENGTH	DS	F	Length in bytes of the VALUE area
VALUE	DS	CL(n)	Area to contain the property value
DATALENGTH	DS	F	Length of the property value
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## MQMHBUF - 버퍼로 메시지 핸들 변환

MQMHBUF 호출은 메시지 핸들을 버퍼로 변환하며 MQBUFMH 호출의 역호출입니다.

### 구문

```
MQMHBUF(Hconn, Hmsg, MsgHBufOpts, Name, MsgDesc, BufferLength, Buffer, DataLength, CompCode,
Reason)
```

### 매개변수

#### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *Hconn* 값이 **Hmsg** 매개변수에 지정된 메시지 핸들 작성에 사용된 연결 핸들과 일치해야 합니다.

메시지 핸들이 MQHC\_UNASSOCIATED\_HCONN을 사용하여 작성된 경우 메시지 핸들을 삭제하는 스레드에서 올바른 연결을 설정해야 합니다. 올바른 연결이 설정되지 않는 경우 MQRC\_CONNECTION\_BROKEN과 함께 호출이 실패합니다.

#### Hmsg

유형: MQHMSG - 입력

이는 버퍼가 필요한 메시지 핸들입니다. 값은 이전 MQCRTMH 호출에서 리턴합니다.

## MsgHBufOpts

유형: MQMHBO - 입력

MQMHBO 구조에서는 애플리케이션이 메시지 핸들에서 버퍼가 생성되는 방법을 제어하는 옵션을 지정할 수 있습니다.

자세한 내용은 [464 페이지의 『MQMHBO - 버퍼에 대한 메시지 핸들 옵션』](#)의 내용을 참조하십시오.

## 이름

유형: MQCHARV - 입력

버퍼에 넣을 특성의 이름.

이름과 일치하는 특성을 찾을 수 없는 경우 MQRC\_PROPERTY\_NOT\_AVAILABLE과 함께 호출이 실패합니다.

와일드카드를 사용하면 둘 이상의 특성을 버퍼에 넣을 수 있습니다. 이를 수행하려면 특성 이름의 끝에 와일드카드 문자 '%'를 사용하십시오. 이 와일드카드는 '.'를 포함한 0개 이상의 문자와 사용할 수 없습니다.

C 프로그래밍 언어에서 다음 매크로 변수는 모든 특성과 'usr'로 시작하는 모든 특성을 조회하기 위해 정의됩니다.

### MQPROP\_INQUIRE\_ALL

메시지의 모든 특성을 버퍼에 넣으십시오.

### MQPROP\_INQUIRE\_ALL\_USR

문자 'usr.'로 시작하는 메시지의 모든 특성을 버퍼에 넣습니다.

특성 이름 사용에 대한 자세한 정보는 [특성 이름 및 특성 이름 제한사항](#)을 참조하십시오.

## MsgDesc

유형: MQMD - 입출력(I/O)

MsgDesc 구조는 버퍼 영역의 콘텐츠를 설명합니다.

출력에서, 호출이 작성한 대로 버퍼 영역에 있는 데이터의 인코딩, 문자 세트 ID 및 형식을 올바르게 설명하도록 *Encoding*, *CodedCharSetId* 및 *Format* 필드가 설정됩니다.

이 구조의 데이터는 애플리케이션의 문자 세트와 인코딩에 있습니다.

## BufferLength

유형: MQLONG - 입력

BufferLength는 버퍼 영역의 길이(바이트)입니다.

## 버퍼

유형: MQBYTEExBufferLength - 출력

Buffer는 메시지 특성을 포함하는 영역을 정의합니다. 4바이트 경계에서 버퍼를 맞추어야 합니다.

BufferLength가 Buffer에서 특성을 저장하는 데 필요한 길이 미만인 경우 MQMHBUF가 MQRC\_PROPERTY\_VALUE\_TOO\_BIG과 함께 실패합니다.

호출이 실패해도 버퍼의 콘텐츠를 변경할 수 있습니다.

## DataLength

유형: MQLONG - 출력

DataLength는 버퍼에서 리턴된 특성의 길이(바이트)입니다. 값이 0이면 Name에서 지정된 값과 일치하는 특성이 없으며 이유 코드 MQRC\_PROPERTY\_NOT\_AVAILABLE과 함께 호출이 실패합니다.

BufferLength가 버퍼에서 특성을 저장하는 데 필요한 길이 미만인 경우 MQMHBUF 호출이 MQRC\_PROPERTY\_VALUE\_TOO\_BIG과 함께 실패하지만 값은 여전히 DataLength에 입력됩니다. 그러면 애플리케이션이 특성을 수용하는 데 필요한 버퍼 크기를 판별하도록 허용한 후 필요한 BufferLength로 호출을 재발행하십시오.

## CompCode

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_FAILED**

호출에 실패했습니다.

**원인**

유형: MQLONG - 출력

*CompCode*를 규정하는 이유 코드입니다.

*CompCode*이(가) MQCC\_OK인 경우:

**MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_FAILED인 경우:

**MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'089C') 어댑터를 사용할 수 없습니다.

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

**MQRC\_ASID\_MISMATCH**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

**MQRC\_MHBO\_ERROR**

(2501, X'095C') 메시지 핸들 버퍼 옵션 구조가 올바르지 않습니다.

**MQRC\_BUFFER\_ERROR**

(2004, X'07D4') 버퍼 매개변수가 올바르지 않습니다.

**MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'07D5') 버퍼 길이 매개변수가 올바르지 않습니다.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'08AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'07D9') 큐 관리자에 대한 연결이 유실되었습니다.

**MQRC\_DATA\_LENGTH\_ERROR**

(2010, X'07DA') 데이터 길이 매개변수가 올바르지 않습니다.

**MQRC\_HMSG\_ERROR**

(2460, X'099C') 메시지 핸들이 올바르지 않습니다.

**MQRC\_MD\_ERROR**

(2026, X'07EA') 메시지 디스크립터가 올바르지 않습니다.

**MQRC\_MSG\_HANDLE\_IN\_USE**

(2499, X'09C3') 메시지 핸들이 이미 사용 중입니다.

**MQRC\_OPTIONS\_ERROR**

(2046, X'07FE') 옵션이 올바르지 않거나 일치하지 않습니다.

**MQRC\_PROPERTY\_NAME\_ERROR**

(2442, X'098A') 특성 이름이 올바르지 않습니다.

**MQRC\_PROPERTY\_NOT\_AVAILABLE**

(2471, X'09A7') 특성을 사용할 수 없습니다.

**MQRC\_PROPERTY\_VALUE\_TOO\_BIG**

(2469, X'09A5') 지정된 특성을 포함하기에 BufferLength 값이 너무 작습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

## C 호출

```
MQMHBUF (Hconn, Hmsg, &MsgHBufOpts, &Name, &MsgDesc, BufferLength, Buffer,  
&DataLength, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN Hconn;          /* Connection handle */  
MQHMSG Hmsg;            /* Message handle */  
MQMHBO MsgHBufOpts;    /* Options that control the action of MQMHBUF */  
MQCHARV Name;          /* Property name */  
MQMD MsgDesc;          /* Message descriptor */  
MQLONG BufferLength;    /* Length in bytes of the Buffer area */  
MQBYTE Buffer[n];       /* Area to contain the properties */  
MQLONG DataLength;     /* Length of the properties */  
MQLONG CompCode;       /* Completion code */  
MQLONG Reason;         /* Reason code qualifying CompCode */
```

## 사용법 참고

MQMHBUF는 메시지 핸들을 버퍼로 변환합니다.

메시지 특성 API를 사용하여 특정 특성에 액세스하기 위해 이를 MQGET API 엑시트와 함께 사용할 수 있으며 메시지 핸들이 아닌 MQRFH2 헤더를 사용하도록 설계된 애플리케이션에 이러한 항목을 버퍼로 다시 전달할 수 있습니다.

이 호출은 MQBUFMH 호출의 역으로, 버퍼의 메시지 특성을 메시지 핸들에 구문 분석하는 데 사용할 수 있습니다.

## COBOL 호출

```
CALL 'MQMHBUF' USING HCONN, HMSG, MSGHBUFOPTS, NAME, MSGDESC,  
                    BUFFERLENGTH, BUFFER, DATALENGTH, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```
** Connection handle  
01 HCONN          PIC S9(9) BINARY.  
** Message handle  
01 HMSG          PIC S9(18) BINARY.  
** Options that control the action of MQMHBUF  
01 MSGHBUFOPTS.  
   COPY CMQMHBV.  
** Property name  
01 NAME  
   COPY CMQCHRVV.  
** Message descriptor  
01 MSGDESC  
   COPY CMQMDV.  
** Length in bytes of the Buffer area */  
01 BUFFERLENGTH PIC S9(9) BINARY.  
** Area to contain the properties  
01 BUFFER       PIC X(n).  
** Length of the properties  
01 DATALENGTH PIC S9(9) BINARY.  
** Completion code  
01 COMPCODE     PIC S9(9) BINARY.  
** Reason code qualifying COMPCODE  
01 REASON       PIC S9(9) BINARY.
```

## PL/I 호출

```
call MQMHBUF (Hconn, Hmsg, MsgHBufOpts, Name, MsgDesc, BufferLength, Buffer,  
DataLength, CompCode, Reason);
```

매개변수를 다음과 같이 선언합니다.

```
dc1 Hconn          fixed bin(31); /* Connection handle */
dc1 Hmsg           fixed bin(63); /* Message handle */
dc1 MsgHBufOpts   like MQMHBO; /* Options that control the action of MQMHBUF */
dc1 Name           like MQCHARV; /* Property name */
dc1 MsgDesc       like MQMD; /* Message descriptor */
dc1 BufferLength   fixed bin(31); /* Length in bytes of the Buffer area */
dc1 Buffer         char(n); /* Area to contain the properties */
dc1 DataLength    fixed bin(31); /* Length of the properties */
dc1 CompCode      fixed bin(31); /* Completion code */
dc1 Reason        fixed bin(31); /* Reason code qualifying CompCode */
```

## 상위 레벨 어셈블러 호출

```
CALL MQMHBUF, (HCONN, HMSG, MSGHBUFOPTS, NAME, MSGDESC, BUFFERLENGTH,
              BUFFER, DATALENGTH, COMPCODE, REASON)
```

매개변수를 다음과 같이 선언합니다.

HCONN	DS	F	Connection handle
HMSG	DS	D	Message handle
MSGHBUFOPTS	CMQMHBOA	,	Options that control the action of MQMHBUF
NAME	CMQCHRVA	,	Property name
MSGDESC	CMQMDA	,	Message descriptor
BUFFERLENGTH	DS	F	Length in bytes of the BUFFER area
BUFFER	DS	CL(n)	Area to contain the properties
DATALENGTH	DS	F	Length of the properties
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## MQOPEN - 오브젝트 열기

MQOPEN 호출은 오브젝트에 대한 액세스를 설정합니다.

올바른 오브젝트의 유형은 다음과 같습니다.

- 큐(분배 목록 포함)
- 이름 목록
- 프로세스 정의
- 큐 관리자
- 토픽

## 구문


MQOPEN(*Hconn*, *ObjDesc*, *Options*, *Hobj*, *CompCode*, *Reason*)

## 매개변수

### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. Hconn의 값은 이전 MQCONN 또는 MQCONNx 호출에 의해 리턴되었습니다.

 z/OS for CICS 애플리케이션에서 MQCONN 호출을 생략할 수 있으며 Hconn에 다음 값이 지정됩니다.

### MQHC\_DEF\_HCONN

기본 연결 핸들

## ObjDesc

유형: MQOD - 입출력(I/O)

이는 열리는 오브젝트를 식별하는 구조입니다. 세부사항은 466 페이지의 『MQOD - 오브젝트 디스크립터』의 내용을 참조하십시오.

**ObjDesc** 매개변수의 ObjectName 필드가 모델 큐의 이름인 경우 동적 로컬 큐는 모델 큐의 속성으로 작성됩니다. **Options** 매개변수에서 지정하는 모든 옵션에 발생합니다. MQOPEN 호출로 리턴된 Hobj를 사용하는 후속 조작용 모델 큐가 아니라 새 동적 큐에서 수행됩니다. 이는 MQINQ 및 MQSET 호출에 대해서도 마찬가지입니다. **ObjDesc** 매개변수에서 모델 큐의 이름은 작성된 동적 큐의 이름으로 대체됩니다. 동적 큐의 유형은 모델 큐의 **DefinitionType** 속성 값에 따라 판별됩니다(804 페이지의 『큐의 속성』 참조). 동적 큐에 적용할 수 있는 닫기 옵션에 대한 정보는 MQCLOSE 호출의 설명을 참조하십시오.

## 옵션

유형: MQLONG - 입력

다음 옵션 중 최소 하나를 지정해야 합니다.

- MQOO\_BROWSE
- MQOO\_INPUT\_\*(다음 중 하나만 해당됨)
- MQOO\_INQUIRE
- MQOO\_OUTPUT
- MQOO\_SET
- MQOO\_BIND\_\*(다음 중 하나만 해당됨)

이러한 옵션에 대한 세부사항은 다음 테이블을 참조하십시오. 기타 옵션은 필요에 따라 지정될 수 있습니다. 둘 이상의 옵션을 지정하려면 값을 함께 추가하거나(동일한 상수를 두 번 이상 추가하지 않음), 비트 단위 OR 연산을 사용하여 값을 결합하십시오(프로그래밍 언어가 비트 연산을 지원하는 경우).올바르지 않은 결합이 설명되어 있습니다. 다른 모든 결합은 유효합니다. ObjDesc에 의해 지정된 오브젝트 유형에 해당되는 옵션만 허용됩니다.

옵션	알리어스 <sup>1</sup>	로컬 및 모델	원격	로컬이 아닌 클러스터	분배 목록	토픽
<u>MQOO_INPUT_AS_Q_DEF</u>	예	예	아니오	아니오	아니오	아니오
<u>MQOO_INPUT_SHARED</u>	예	예	아니오	아니오	아니오	아니오
<u>MQOO_INPUT_EXCLUSIVE</u>	예	예	아니오	아니오	아니오	아니오
<u>MQOO_OUTPUT</u>	예	예	예	예	예	예
<u>MQOO_BROWSE</u>	예	예	아니오	아니오	아니오	아니오
<u>MQOO_CO_OP</u>	예	예	아니오	아니오	아니오	아니오
<u>MQOO_INQUIRE</u>	예	예	<sup>2</sup>	예	아니오	아니오
<u>MQOO_SET</u>	예	예	<sup>2</sup>	아니오	아니오	아니오
<u>MQOO_BIND_ON_OPEN</u> <sup>3</sup>	예	예	예	예	예	아니오
<u>MQOO_BIND_NOT_FIXED</u> <sup>3</sup>	예	예	예	예	예	아니오
<u>MQOO_BIND_ON_GROUP</u> <sup>3</sup>	예	예	예	예	예	아니오

표 553. 큐와 토픽에 대한 올바른 MQOPEN 옵션 (계속)

옵션	알리어스 <sup>1</sup>	로컬 및 모델	원격	로컬이 아닌 클러스터	분배 목록	토픽
<u>MQOO_BIND_AS_Q_DEF</u> <sup>3</sup>	예	예	예	예	예	아니오
<u>MQOO_SAVE_ALL_CONTEXT</u>	예	예	아니오	아니오	아니오	아니오
<u>MQOO_PASS_IDENTITY_CONTEXT</u>	예	예	예	예	예	<u>4</u>
<u>MQOO_PASS_ALL_CONTEXT</u>	예	예	예	예	예	예
<u>MQOO_SET_IDENTITY_CONTEXT</u>	예	예	예	예	예	<u>4</u>
<u>MQOO_SET_ALL_CONTEXT</u>	예	예	예	예	예	예
<u>MQOO_NO_READ_AHEAD</u>	예	예	아니오	아니오	아니오	아니오
<u>MQOO_READ_AHEAD</u>	예	예	아니오	아니오	아니오	아니오
<u>MQOO_READ_AHEAD_AS_Q_DEF</u>	예	예	아니오	아니오	아니오	아니오
<u>MQOO_ALTERNATE_USER_AUTHORITY</u>	예	예	예	예	예	예
<u>MQOO_FAIL_IF QUIESCING</u>	예	예	예	예	예	예
<u>MQOO_RESOLVE_LOCAL_Q</u>	예	예	예	예	아니오	아니오
<u>MQOO_RESOLVE_LOCAL_TOPIC</u>	아니오	아니오	아니오	아니오	아니오	예
<u>MQOO_NO_MULTICAST</u>	아니오	아니오	아니오	아니오	아니오	예

**참고:**

1. 알리어스에 대한 옵션 검증은 알리어스가 해석되는 큐에 대한 옵션의 검증에 따라 결정됩니다.
2. 이 옵션은 리모트 큐의 로컬 정의에 대해서만 유효합니다.
3. 이 옵션은 모든 큐 유형에 대해 지정할 수 있지만 큐가 클러스터 큐가 아니면 무시됩니다. 그러나 알리어스 큐가 클러스터 내에 없는 경우에도 **DefBind** 큐 속성이 기본 큐를 대체합니다.
4. 이러한 속성은 토픽과 함께 사용될 수 있으나 모든 구독자에게 전송된 컨텍스트 필드가 아니라 보류된 메시지에 대한 컨텍스트 세트에만 영향을 미칩니다.

**액세스 옵션:** 다음 옵션은 오브젝트에서 수행할 수 있는 조작 유형을 제어합니다.

**MQOO\_INPUT\_AS\_Q\_DEF**

큐가 정의한 디폴트를 사용하여 메시지를 가져오기 위해 큐를 엽니다.

해당 큐가 후속 MQSET 호출에 사용하도록 열립니다. 액세스 유형은 **DefInputOpenOption** 큐 속성 값에 따라 공유 또는 독점입니다. 자세한 정보는 804 페이지의 『큐의 속성』의 내용을 참조하십시오.

이 옵션은 로컬, 알리어스 및 모델 큐에서만 올바르며, 리모트 큐, 분배 목록과 큐에 없는 오브젝트에는 올바르지 않습니다.

**MQOO\_INPUT\_SHARED**

공유 액세스로 메시지를 가져오기 위해 큐를 엽니다.

해당 큐가 후속 MQSET 호출에 사용하도록 열립니다. 현재 이 애플리케이션 또는 다른 애플리케이션에서 MQOO\_INPUT\_SHARED를 사용하여 큐가 열려 있는 경우 호출에 성공하지만 큐가 현재 MQOO\_INPUT\_EXCLUSIVE를 사용하여 열려 있으면 이유 코드 MQRC\_OBJECT\_IN\_USE와 함께 호출에 실패합니다.

이 옵션은 로컬, 알리어스 및 모델 큐에서만 올바르며, 리모트 큐, 분배 목록과 큐에 없는 오브젝트에는 올바르지 않습니다.

## MQOO\_INPUT\_EXCLUSIVE

배타적 액세스를 메시지를 가져오기 위해 큐를 엽니다.

해당 큐가 후속 MQSET 호출에 사용하도록 열립니다. 임의의 유형(MQOO\_INPUT\_SHARED 또는 MQOO\_INPUT\_EXCLUSIVE)의 입력에 대해 현재 이 애플리케이션 또는 다른 애플리케이션에서 큐가 열려 있는 경우 이유 코드 MQRC\_OBJECT\_IN\_USE와 함께 호출에 실패합니다.

이 옵션은 로컬, 알리어스 및 모델 큐에서만 올바르며, 리모트 큐, 분배 목록과 큐에 없는 오브젝트에는 올바르지 않습니다.

## MQOO\_OUTPUT

메시지를 발행하기 위해 메시지 넣기, 토픽 또는 토픽 문자열에 대한 큐를 엽니다.

후속 MQPUT 호출과 함께 사용하기 위해 큐 또는 토픽이 열립니다.

**InhibitPut** 큐 속성이 MQQA\_PUT\_INHIBITED로 설정된 경우 속성이 이 값으로 설정된 동안 후속 MQPUT 호출이 실패하더라도 이 옵션이 설정된 MQOPEN 호출은 성공할 수 있습니다.

이 옵션은 분배 목록 및 토픽을 포함하여 모든 유형의 큐에 대해 유효합니다.

다음은 이러한 옵션에 적용되는 참고입니다.

- 이러한 옵션 중 하나만 지정될 수 있습니다.
- **InhibitGet** 큐 속성이 MQQA\_GET\_INHIBITED로 설정된 경우 속성이 이 값으로 설정된 동안 후속 MQGET 호출이 실패하더라도 이러한 옵션이 설정된 MQOPEN 호출은 성공할 수 있습니다.
- 큐가 공유할 수 없음으로 정의된 경우 즉, **Shareability** 큐 속성 값이 MQQA\_NOT\_SHAREABLE인 경우 공유 액세스에 대해 큐를 열려는 시도는 독점 액세스를 큐를 열려는 시도로 처리됩니다.
- 이러한 옵션 중 하나를 사용하여 알리어스 큐가 열린 경우, 알리어스 큐가 해석되는 기본 큐에 대해 독점 사용(또는 다른 애플리케이션의 독점 사용 여부)에 대한 테스트가 수행됩니다.
- **ObjectQMgrName**이 큐 관리자 알리어스의 이름인 경우에는 이 옵션이 올바르지 않습니다. 이는 큐 관리자 알리어스에 사용되는 리모트 큐의 로컬 정의에서 **RemoteQMgrName** 속성의 값이 로컬 큐 관리자의 이름인 경우에도 적용됩니다.

## MQOO\_BROWSE

메시지를 열람하기 위해 큐를 엽니다.

다음 옵션 중 하나를 사용하여 후속 MQGET 호출과 함께 사용하기 위해 큐가 열립니다.

- MQGMO\_BROWSE\_FIRST
- MQGMO\_BROWSE\_NEXT
- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR

큐가 현재 MQOO\_INPUT\_EXCLUSIVE에 대해 열린 경우에도 허용됩니다. MQOO\_BROWSE 옵션이 지정된 MQOPEN 호출은 찾아보기 커서를 설정하고 이를 큐의 첫 번째 메시지 앞에 논리적으로 배치합니다. 추가 정보는 [MQGMO - Options field](#)를 참조하십시오.

이 옵션은 로컬, 알리어스 및 모델 큐에서만 올바르며, 리모트 큐, 분배 목록과 큐에 없는 오브젝트에는 올바르지 않습니다. **ObjectQMgrName**이 큐 관리자 알리어스의 이름인 경우에도 이 옵션이 올바르지 않습니다. 이는 큐 관리자 알리어스에 사용되는 리모트 큐의 로컬 정의에서 **RemoteQMgrName** 속성의 값이 로컬 큐 관리자의 이름인 경우에도 적용됩니다.

## MQOO\_CO\_OP

핸들 세트의 협동 멤버로 엽니다.

이 옵션은 MQOO\_BROWSE 옵션과 함께 사용하는 경우에만 유효합니다. MQOO\_BROWSE 없이 지정되면 MQOPEN이 MQRC\_OPTIONS\_ERROR와 함께 리턴됩니다.

리턴되는 핸들은 다음 옵션 중 하나를 가진 후속 MQGET 호출에 대한 협동 핸들 세트의 멤버인 것으로 간주됩니다.

- MQGMO\_MARK\_BROWSE\_CO\_OP
- MQGMO\_UNMARKED\_BROWSE\_MSG



- MQGMO\_UNMARK\_BROWSE\_CO\_OP

이 옵션은 로컬, 알리어스 및 모델 큐에서만 올바르며, 리모트 큐, 분배 목록과 큐에 없는 오브젝트에는 올바르지 않습니다.

### MQOO\_INQUIRE

속성을 조회할 오브젝트를 엽니다.

큐, 이름 목록, 프로세스 정의 또는 큐 관리자가 후속 MQINQ 호출과 함께 사용하기 위해 열립니다.

이 옵션은 분배 목록을 제외한 모든 유형의 오브젝트에 대해 유효합니다. ObjectQMgrName이 큐 관리자 알리어스의 이름인 경우에는 이 옵션이 올바르지 않습니다. 이는 큐 관리자 알리어스에 사용되는 리모트 큐의 로컬 정의에서 RemoteQMgrName 속성의 값이 로컬 큐 관리자의 이름인 경우에도 적용됩니다.

### MQOO\_SET

속성을 설정하기 위해 큐를 엽니다.

후속 MQSET 호출과 함께 사용하기 위해 큐가 열립니다.

이 옵션은 분배 목록을 제외한 모든 유형의 큐에 대해 유효합니다. ObjectQMgrName이 리모트 큐의 로컬 정의 이름인 경우에는 이 옵션이 유효하지 않습니다. 이는 큐 관리자 알리어스에 사용되는 리모트 큐의 로컬 정의에서 RemoteQMgrName 속성의 값이 로컬 큐 관리자의 이름인 경우에도 적용됩니다.

**바인딩 옵션:** 다음 옵션은 열리는 오브젝트가 클러스터 큐인 경우에 적용됩니다. 이러한 옵션은 클러스터 큐의 인스턴스에 대한 큐 핸들 바인딩을 제어합니다.

### MQOO\_BIND\_ON\_OPEN

로컬 큐 관리자는 큐가 열릴 때 목적지 큐의 인스턴스에 대해 큐 핸들을 바인딩합니다. 따라서 이 핸들을 사용하여 넣는 모든 메시지가 동일한 라우트로 목적지 큐의 동일한 인스턴스에 전송됩니다.

이 옵션은 큐에 대해서만 유효하며 클러스터 큐에만 영향을 미칩니다. 클러스터 큐가 아닌 큐에 대해 지정된 경우에는 옵션이 무시됩니다.

### MQOO\_BIND\_NOT\_FIXED

로컬 큐 관리자가 큐 핸들을 목적지 큐의 인스턴스에 바인딩하는 것을 중지합니다. 따라서 이 핸들을 사용하는 후속 MQPUT 호출이 목적지 큐의 다른 인스턴스에 메시지를 전송하거나 다른 라우트로 동일한 인스턴스에 메시지를 전송합니다. 네트워크 조건에 따라 로컬 큐 관리자, 리모트 큐 관리자 또는 메시지 채널 에이전트(MCA)에 의해 인스턴스가 나중에 변경되도록 선택할 수도 있습니다.

**참고:** 트랜잭션을 완료하기 위해 메시지 시리즈를 교환해야 하는 클라이언트 및 서버 애플리케이션은 MQOO\_BIND\_NOT\_FIXED(DefBind의 값이 MQBND\_BIND\_NOT\_FIXED인 경우에는 MQOO\_BIND\_AS\_Q\_DEF)를 사용할 수 없습니다. 시리즈 내의 후속 메시지가 서버 애플리케이션의 다른 인스턴스에 송신될 수 있기 때문입니다.

클러스터 큐에 대해 MQOO\_BROWSE 또는 MQOO\_INPUT\_\* 옵션 중 하나가 지정된 경우에는 큐 관리자가 클러스터 큐의 로컬 인스턴스를 선택하도록 강제 실행됩니다. 결과적으로 MQOO\_BIND\_NOT\_FIXED가 지정된 경우에도 큐 핸들의 바인딩이 고정됩니다.

MQOO\_INQUIRE가 MQOO\_BIND\_NOT\_FIXED와 함께 지정된 경우 일반적으로 모든 인스턴스가 동일한 속성 값을 가지더라도 해당 핸들을 사용하는 후속 MQINQ 호출이 클러스터 큐의 다른 인스턴스를 조회할 수 있습니다.

MQOO\_BIND\_NOT\_FIXED는 큐에 대해서만 유효하며 클러스터 큐에만 영향을 미칩니다. 클러스터 큐가 아닌 큐에 대해 지정된 경우에는 옵션이 무시됩니다.

### MQOO\_BIND\_ON\_GROUP

애플리케이션을 통해 메시지 그룹이 모두 동일한 목적지 인스턴스에 할당되도록 요청할 수 있습니다.

이 옵션은 큐에 대해서만 유효하며 클러스터 큐에만 영향을 미칩니다. 클러스터 큐가 아닌 큐에 대해 지정된 경우에는 옵션이 무시됩니다.

### MQOO\_BIND\_AS\_Q\_DEF

로컬 큐 관리자가 DefBind 큐 속성에서 정의된 방법으로 큐 핸들을 바인딩합니다. 이 속성의 값은 MQBND\_BIND\_ON\_OPEN, MQBND\_BIND\_NOT\_FIXED 또는 MQBND\_BIND\_ON\_GROUP입니다.

MQOO\_BIND\_ON\_OPEN, MQOO\_BIND\_NOT\_FIXED 또는 MQOO\_BIND\_ON\_GROUP이 지정되지 않은 경우 MQOO\_BIND\_AS\_Q\_DEF가 기본값입니다.

MQOO\_BIND\_AS\_Q\_DEF는 프로그램 문서화를 지원합니다. 이 옵션은 기타 두 바인드 옵션 중 하나와 함께 사용하기 위한 용도가 아니며 값이 0이므로 이러한 사용을 감지할 수 없습니다.

**컨텍스트 옵션:** 다음 옵션은 메시지 컨텍스트의 처리를 제어합니다.

#### **MQOO\_SAVE\_ALL\_CONTEXT**

컨텍스트 정보는 이 큐 핸들과 연관됩니다. 이 정보는 이 핸들을 사용하여 검색된 메시지의 컨텍스트에서 설정됩니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어](#)를 참조하십시오.

이 컨텍스트 정보는 MQPUT 또는 MQPUT1 호출을 사용하여 큐에 넣는 메시지에 전달할 수 있습니다. 485 페이지의 『MQPMO - 메시지 넣기 옵션』에서 설명한 MQPMO\_PASS\_IDENTITY\_CONTEXT 및 MQPMO\_PASS\_ALL\_CONTEXT 옵션을 참조하십시오.

메시지가 성공적으로 검색될 때까지 큐에 넣는 메시지에 컨텍스트를 전달할 수 없습니다.

MQGMO\_BROWSE\_\* 찾아보기 옵션 중 하나를 사용하여 검색된 메시지는 **MsgDesc** 매개변수의 컨텍스트 필드가 찾아보기 후에 설정되는 경우에도 컨텍스트 정보를 저장하지 않습니다.

이 옵션은 로컬, 알리어스 및 모델 큐에서만 올바르며, 리모트 큐, 분배 목록과 큐에 없는 오브젝트에는 올바르지 않습니다. MQOO\_INPUT\_\* 옵션 중 하나를 지정해야 합니다.

#### **MQOO\_PASS\_IDENTITY\_CONTEXT**

이는 메시지를 큐에 넣을 때 **PutMsgOpts** 매개변수에서 MQPMO\_PASS\_IDENTITY\_CONTEXT 옵션을 지정할 수 있도록 허용합니다. 이로 인해 MQOO\_SAVE\_ALL\_CONTEXT 옵션을 사용하여 열린 입력 큐의 ID 컨텍스트 정보를 메시지에 제공할 수 있습니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어](#)를 참조하십시오.

MQOO\_OUTPUT 옵션을 지정해야 합니다.

이 옵션은 분배 목록을 포함하여 모든 유형의 큐에 유효합니다.

#### **MQOO\_PASS\_ALL\_CONTEXT**

이는 메시지를 큐에 넣을 때 **PutMsgOpts** 매개변수에서 MQPMO\_PASS\_ALL\_CONTEXT 옵션을 지정할 수 있도록 허용합니다. 이로 인해 MQOO\_SAVE\_ALL\_CONTEXT 옵션을 사용하여 열린 입력 큐의 ID 및 원본 컨텍스트 정보를 메시지에 제공할 수 있습니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어](#)를 참조하십시오.

이 옵션은 MQOO\_PASS\_IDENTITY\_CONTEXT를 지정할 필요가 없음을 의미합니다. MQOO\_OUTPUT 옵션을 지정해야 합니다.

이 옵션은 분배 목록을 포함하여 모든 유형의 큐에 유효합니다.

#### **MQOO\_SET\_IDENTITY\_CONTEXT**

이는 메시지를 큐에 넣을 때 **PutMsgOpts** 매개변수에서 MQPMO\_SET\_IDENTITY\_CONTEXT 옵션을 지정할 수 있도록 허용합니다. 이로 인해 MQPUT 또는 MQPUT1 호출에서 지정된 **MsgDesc** 매개변수에 포함된 ID 컨텍스트 정보를 메시지에 제공할 수 있습니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어](#)를 참조하십시오.

이 옵션은 MQOO\_PASS\_IDENTITY\_CONTEXT를 지정할 필요가 없음을 의미합니다. MQOO\_OUTPUT 옵션을 지정해야 합니다.

이 옵션은 분배 목록을 포함하여 모든 유형의 큐에 유효합니다.

#### **MQOO\_SET\_ALL\_CONTEXT**

이는 메시지를 큐에 넣을 때 **PutMsgOpts** 매개변수에서 MQPMO\_SET\_ALL\_CONTEXT 옵션을 지정할 수 있도록 허용합니다. 이로 인해 MQPUT 또는 MQPUT1 호출에서 지정된 **MsgDesc** 매개변수에 포함된 ID 및 원본 컨텍스트 정보를 메시지에 제공할 수 있습니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어](#)를 참조하십시오.

이 옵션은 다음 옵션을 의미하므로 지정할 필요가 없습니다.

- MQOO\_PASS\_IDENTITY\_CONTEXT
- MQOO\_PASS\_ALL\_CONTEXT
- MQOO\_SET\_IDENTITY\_CONTEXT

MQOO\_OUTPUT 옵션을 지정해야 합니다.

이 옵션은 분배 목록을 포함하여 모든 유형의 큐에 유효합니다.

#### 미리 읽기 옵션:

MQOO\_READ\_AHEAD와 함께 MQOPEN을 호출할 경우 특정 조건이 충족되면 IBM MQ 클라이언트에서는 미리 읽기만 가능합니다. 이러한 조건은 다음과 같습니다.

- 클라이언트 애플리케이션이 컴파일되고 스레드된 IBM MQ MQI 클라이언트 라이브러리에 대해 링크되어야 합니다.
- 클라이언트 채널이 TCP/IP 프로토콜을 사용해야 합니다.
- 채널이 클라이언트 및 서버 채널 정의 모두에서 0이 아닌 SharingConversations(SHARECNV) 설정을 사용해야 합니다.

다음 옵션은 애플리케이션이 요청하기 전에 비지속 메시지가 클라이언트에 전송되는지 여부를 제어합니다. 다음은 미리 읽기 옵션에 적용되는 참고입니다.

- 이러한 옵션 중 하나만 지정될 수 있습니다.
- 이러한 옵션은 로컬, 알리어스, 모델 큐에 대해서만 유효합니다. 이는 리모트 큐, 분배 목록, 토픽 또는 큐 관리자에 대해서는 유효하지 않습니다.
- 이러한 옵션을 MQOO\_INQUIRE 또는 MQOO\_SET과 함께 지정하는 것이 오류는 아니지만 MQOO\_BROWSE, MQOO\_INPUT\_SHARED, MQOO\_INPUT\_EXCLUSIVE 중 하나도 지정한 경우에만 적용할 수 있습니다.
- 애플리케이션이 IBM MQ 클라이언트로 실행 중이지 않은 경우 이러한 옵션은 무시됩니다.

#### **MQOO\_NO\_READ\_AHEAD**

비지속 메시지는 애플리케이션이 요청하기 전에 클라이언트로 전송되지 않습니다.

#### **MQOO\_READ\_AHEAD**

애플리케이션이 요청하기 전에 비지속 메시지가 클라이언트에 송신됩니다.

#### **MQOO\_READ\_AHEAD\_AS\_Q\_DEF**

미리 읽기 작동은 열리는 큐의 기본 미리 읽기 속성에 의해 판별됩니다. 이는 기본값입니다.

**기타 옵션:** 다음 옵션은 권한 검사, 큐 관리자가 정지 중일 때 발생하는 현상, 로컬 큐 이름 해석 여부 및 멀티 캐스트를 제어합니다.


#### **MQOO\_ALTERNATE\_USER\_AUTHORITY**

**ObjDesc** 매개변수의 *AlternateUserId* 필드에는 이 MQOPEN 호출을 유효성 검증하는 데 사용할 사용자 ID가 포함되어 있습니다. 애플리케이션을 실행 중인 사용자 ID에 오브젝트를 열 수 있는 권한이 부여되었는지 여부에 상관없이 이 *AlternateUserId*가 지정된 액세스 옵션이 있는 오브젝트를 열 수 있는 권한이 있는 경우에만 호출이 성공합니다. 이는 지정된 어떠한 컨텍스트 옵션에도 적용되지 않지만 애플리케이션이 실행 중인 사용자 ID에 대해서는 항상 검사가 수행됩니다.

이 옵션은 모든 유형의 오브젝트에 대해 유효합니다.

#### **MQOO\_FAIL\_IF QUIESCING**

큐 관리자가 정지 상태에 있는 경우 MQOPEN 호출이 실패합니다.

 z/OS에서 CICS 또는 IMS 애플리케이션의 경우 이 옵션은 또한 연결이 정지 상태일 때 MQOPEN 호출이 실패하도록 강제 실행합니다.

이 옵션은 모든 유형의 오브젝트에 대해 유효합니다.

클라이언트 채널에 대한 정보는 [IBM MQ MQI clients](#)의 내용을 참조하십시오.

#### **MQOO\_RESOLVE\_LOCAL\_Q**

열려 있는 로컬 큐의 이름으로 MQOD 구조의 ResolvedQName을 채웁니다. 마찬가지로, ResolvedQMgrName은 로컬 큐를 호스트하는 로컬 큐 관리자의 이름으로 채워집니다. MQOD 구조가 버전 3 미만이면 오류가 리턴되지 않고 MQOO\_RESOLVE\_LOCAL\_Q가 무시됩니다.

로컬, 알리어스 또는 모델 큐가 열린 경우 로컬 큐가 항상 리턴되지만, 이 상황은 그런 경우가 아닙니다 (예: 리모트 큐 또는 비로컬 큐가 MQOO\_RESOLVE\_LOCAL\_Q 옵션 없이 열림). 리모트 큐 정의에서 또는 유사한 리모트 클러스터 큐로 ResolvedQName 및 RemoteQMgrName이 발견됩니다.

예를 들어, 열기를 수행할 때 MQOO\_RESOLVE\_LOCAL\_Q를 지정하면 리모트 큐 ResolvedQName이 메시지를 넣는 전송 큐가 됩니다. ResolvedQMgrName은 전송 큐를 호스트하는 로컬 큐 관리자의 이름으로 채워집니다.

큐에 대한 찾아보기, 입력 또는 출력 권한이 부여된 경우 이 플래그를 MQOPEN 호출에 지정하기 위한 필수 권한이 있습니다. 특별한 권한이 필요하지는 않습니다.

이 옵션은 큐 및 큐 관리자에만 유효합니다.

#### **MQOO\_RESOLVE\_LOCAL\_TOPIC**

MQOD 구조의 ResolvedQName을 열려 있는 관리 토픽의 이름으로 채웁니다.

#### **MQOO\_NO\_MULTICAST**

발행 메시지는 멀티캐스트를 사용하여 전송되지 않습니다.

이 옵션은 MQOO\_OUTPUT 옵션이 있는 경우에만 유효합니다. MQOO\_OUTPUT 없이 지정되면 MQOPEN이 MQRC\_OPTIONS\_ERROR와 함께 리턴됩니다.

이 옵션은 토픽에 대해서만 유효합니다.

### **Hobj**

유형: MQHOBJ - 출력

이 핸들은 오브젝트에 설정된 액세스를 나타냅니다. 이는 오브젝트를 운영하는 후속 IBM MQ 호출에서 지정되어야 합니다. 이는 MQCLOSE 호출이 발행되거나 핸들의 범위를 정의하는 처리 단위가 종료될 때 유효성이 중단됩니다.

리턴되는 오브젝트 핸들의 범위는 호출에서 지정된 연결 핸들의 범위와 동일합니다. 핸들 범위에 대한 정보는 MQCONN - Hconn 매개변수를 참조하십시오.

### **CompCode**

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

#### **MQCC\_OK**

정상적으로 완료되었습니다.

#### **MQCC\_WARNING**

경고(일부 완료).

#### **MQCC\_FAILED**

호출에 실패했습니다.

### **원인**

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드입니다.

CompCode이(가) MQCC\_OK인 경우:

#### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode이(가) MQCC\_WARNING인 경우:

#### **MQRC\_MULTIPLE\_REASONS**

(2136, X'858') 다중 이유 코드가 리턴되었습니다.

CompCode이(가) MQCC\_FAILED인 경우:

#### **MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') 어댑터를 사용할 수 없습니다.

#### **MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

**MQRC\_ALIAS\_BASE\_Q\_TYPE\_ERROR**  
(2001, X'7D1') 알리어스 기본 큐가 올바른 유형이 아닙니다.

**MQRC\_API\_EXIT\_ERROR**  
(2374, X'946') API 엑시트가 실패했습니다.

**MQRC\_API\_EXIT\_LOAD\_ERROR**  
(2183, X'887') API 엑시트를 로드할 수 없습니다.

**MQRC\_ASID\_MISMATCH**  
(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

**MQRC\_CALL\_IN\_PROGRESS**  
(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**MQRC\_CF\_NOT\_AVAILABLE**  
(2345, X'929') 커플링 기능이 사용 불가능합니다.

**MQRC\_CF\_STRUC\_AUTH\_FAILED**  
(2348, X'92C') 커플링 기능 구조 권한 검사에 실패했습니다.

**MQRC\_CF\_STRUC\_ERROR**  
(2349, X'92D') 커플링 기능 구조가 올바르지 않습니다.

**MQRC\_CF\_STRUC\_FAILED**  
(2373, X'945') 커플링 기능 구조가 실패했습니다.

**MQRC\_CF\_STRUC\_IN\_USE**  
(2346, X'92A') 커플링 기능 구조가 사용 중입니다.

**MQRC\_CF\_STRUC\_LIST\_HDR\_IN\_USE**  
(2347, X'92B') 커플링 기능 구조 목록 헤더가 사용 중입니다.

**MQRC\_CICS\_WAIT\_FAILED**  
(2140, X'85C') CICS에서 대기 요청이 거부되었습니다.

**MQRC\_CLUSTER\_EXIT\_ERROR**  
(2266, X'8DA') 클러스터 워크로드 엑시트가 실패했습니다.

**MQRC\_CLUSTER\_PUT\_INHIBITED**  
(2268, X'8DC') 클러스터 내의 모든 큐에 대해 금지된 넣기 호출입니다.

**MQRC\_CLUSTER\_RESOLUTION\_ERROR**  
(2189, X'88D') 클러스터 이름을 해석하지 못했습니다.

**MQRC\_CLUSTER\_RESOURCE\_ERROR**  
(2269, X'8DD') 클러스터 자원에 오류가 있습니다.

**MQRC\_CONNECTION\_BROKEN**  
(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**MQRC\_CONNECTION\_NOT\_AUTHORIZED**  
(2217, X'8A9') 연결할 권한이 부여되지 않았습니다.

**MQRC\_CONNECTION QUIESCING**  
(2202, X'89A') 연결이 정지됩니다.

**MQRC\_CONNECTION\_STOPPING**  
(2203, X'89B') 연결이 종료됩니다.

**MQRC\_DB2\_NOT\_AVAILABLE**  
(2342, X'926') Db2 서브시스템이 사용 불가능합니다.

**MQRC\_DEF\_XMIT\_Q\_TYPE\_ERROR**  
(2198, X'896') 기본 전송 큐가 로컬이 아닙니다.

**MQRC\_DEF\_XMIT\_Q\_USAGE\_ERROR**  
(2199, X'897') 기본 전송 큐 사용법 오류입니다.

**MQRC\_DYNAMIC\_Q\_NAME\_ERROR**  
(2011, X'7DB') 동적 큐의 이름이 올바르지 않습니다.

**MQRC\_HANDLE\_NOT\_AVAILABLE**  
(2017, X'7E1') 사용 가능한 핸들이 없습니다.

**MQRC\_HCONN\_ERROR**  
(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**MQRC\_HOBJ\_ERROR**  
(2019, X'7E3') 오브젝트 핸들이 올바르지 않습니다.

**MQRC\_MULTIPLE\_REASONS**  
(2136, X'858') 다중 이유 코드가 리턴되었습니다.

**MQRC\_NAME\_IN\_USE**  
(2201, X'899') 이름이 사용 중입니다.

**MQRC\_NAME\_NOT\_VALID\_FOR\_TYPE**  
(2194, X'892') 오브젝트 이름이 오브젝트 유형에 대해 유효하지 않습니다.

**MQRC\_NOT\_AUTHORIZED**  
(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

**MQRC\_OBJECT\_ALREADY\_EXISTS**  
(2100, X'834') 오브젝트가 존재합니다.

**MQRC\_OBJECT\_DAMAGED**  
(2101, X'835') 오브젝트가 손상되었습니다.

**MQRC\_OBJECT\_IN\_USE**  
(2042, X'7FA') 오브젝트가 이미 충돌하는 옵션으로 열렸습니다.

**MQRC\_OBJECT\_LEVEL\_INCOMPATIBLE**  
(2360, X'938') 오브젝트 레벨이 호환 가능하지 않습니다.

**MQRC\_OBJECT\_NAME\_ERROR**  
(2152, X'868') 오브젝트 이름이 올바르지 않습니다.

**MQRC\_OBJECT\_NOT\_UNIQUE**  
(2343, X'927') 오브젝트가 고유하지 않습니다.

**MQRC\_OBJECT\_Q\_MGR\_NAME\_ERROR**  
(2153, X'869') 오브젝트 큐 관리자 이름이 올바르지 않습니다.

**MQRC\_OBJECT\_RECORDS\_ERROR**  
(2155, X'86B') 오브젝트 레코드가 올바르지 않습니다.

**MQRC\_OBJECT\_STRING\_ERROR**  
(2441, X'0989') Objectstring 필드가 올바르지 않습니다.

**MQRC\_OBJECT\_TYPE\_ERROR**  
(2043, X'7FB') 오브젝트 유형이 유효하지 않습니다.

**MQRC\_OD\_ERROR**  
(2044, X'7FC') 오브젝트 디스크립터 구조가 유효하지 않습니다.

**MQRC\_OPTION\_NOT\_VALID\_FOR\_TYPE**  
(2045, X'7FD') 옵션이 오브젝트 유형에 대해 유효하지 않습니다.

**MQRC\_OPTIONS\_ERROR**  
(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.

**MQRC\_PAGESET\_ERROR**  
(2193, X'891') 페이지 세트 데이터 세트에 액세스하는 중에 오류가 발생했습니다.

**MQRC\_PAGESET\_FULL**  
(2192, X'890') 외부 스토리지 매체가 가득 찼습니다.

**MQRC\_Q\_DELETED**  
(2052, X'804') 큐가 삭제되었습니다.

**MQRC\_Q\_MGR\_NAME\_ERROR**  
(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**  
(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.

**MQRC\_Q\_MGR\_QUIESCING**  
(2161, X'871') 큐 관리자가 정지됩니다.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') 큐 관리자가 종료되었습니다.

**MQRC\_Q\_TYPE\_ERROR**

(2057, X'809') 큐 유형이 유효하지 않습니다.

**MQRC\_RECS\_PRESENT\_ERROR**

(2154, X'86A') 현재 레코드 수가 유효하지 않습니다.

**MQRC\_REMOTE\_Q\_NAME\_ERROR**

(2184, X'888') 리모트 큐 이름이 유효하지 않습니다.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**MQRC\_RESPONSE\_RECORDS\_ERROR**

(2156, X'86C') 응답 레코드가 올바르지 않습니다.

**MQRC\_SECURITY\_ERROR**

(2063, X'80F') 보안 오류가 발생했습니다.

**MQRC\_SELECTOR\_SYNTAX\_ERROR**

2459 (X'099B') MQOPEN, MQPUT1 또는 MQSUB 호출이 발생되었으나 구문 오류를 포함한 선택 문자 열이 지정되었습니다.

**MQRC\_STOPPED\_BY\_CLUSTER\_EXIT**

(2188, X'88C') 클러스터 워크로드 엑시트에 의해 호출이 거부되었습니다.

**MQRC\_STORAGE\_MEDIUM\_FULL**

(2192, X'890') 외부 스토리지 매체가 가득 찼습니다.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**MQRC\_SUPPRESSED\_BY\_EXIT**

(2109, X'83D') 엑시트 프로그램에서 호출이 억제되었습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

**MQRC\_UNKNOWN\_ALIAS\_BASE\_Q**

(2082, X'822') 알 수 없는 알리아스 기본 큐입니다.

**MQRC\_UNKNOWN\_DEF\_XMIT\_Q**

(2197, X'895') 알 수 없는 기본 전송 큐입니다.

**MQRC\_UNKNOWN\_OBJECT\_NAME**

(2085, X'825') 알 수 없는 오브젝트 이름입니다.

**MQRC\_UNKNOWN\_OBJECT\_Q\_MGR**

(2086, X'826') 알 수 없는 오브젝트 큐 관리자입니다.

**MQRC\_UNKNOWN\_REMOTE\_Q\_MGR**

(2087, X'827') 알 수 없는 리모트 큐 관리자입니다.

**MQRC\_UNKNOWN\_XMIT\_Q**

(2196, X'894') 알 수 없는 전송 큐입니다.

**MQRC\_WRONG\_CF\_LEVEL**

(2366, X'93E') 커플링 기능 구조의 레벨이 올바르지 않습니다.

**MQRC\_XMIT\_Q\_TYPE\_ERROR**

(2091, X'82B') 전송 큐가 로컬이 아닙니다.

**MQRC\_XMIT\_Q\_USAGE\_ERROR**

(2092, X'82C') 사용법이 올바르지 않은 전송 큐입니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

**일반 사용 참고사항**

1. 열린 오브젝트는 다음 중 하나입니다.




- 다음에 대한 큐입니다.
  - 메시지 가져오기 또는 찾아보기(MQGET 호출 사용)
  - 메시지 넣기(MQPUT 호출 사용)
  - 큐의 속성 조회(MQINQ 호출 사용)
  - 큐의 속성 설정(MQSET 호출 사용)

이름 지정된 큐가 모델 큐이면 동적 로컬 큐가 작성됩니다. 705 페이지의 『MQOPEN - 오브젝트 열기』에 설명된 **ObjDesc** 매개변수를 참조하십시오.

분배 목록은 큐 목록을 포함하는 특수 유형의 큐 오브젝트입니다. 메시지를 넣기 위해 열 수 있으나 메시지를 가져오거나 찾아보는 목적 또는 속성을 조회하거나 설정하는 목적으로는 열 수 없습니다. 자세한 정보는 사용 시 참고사항 8을 참조하십시오.

QSGDISP(GROUP)가 있는 큐는 MQOPEN 또는 MQPUT1 호출에 사용할 수 없는 특수 유형의 큐 정의입니다.

- 목록에서 큐의 이름에 대해 조회할 이름 목록입니다(MQINQ 호출 사용).
  - 프로세스 속성에 대해 조회할 프로세스 정의입니다(MQINQ 호출 사용).
  - 로컬 큐 관리자의 속성에 대해 조회할 큐 관리자입니다(MQINQ 호출 사용).
  - 메시지를 발행할 주제입니다(MQPUT 호출 사용).
2. 애플리케이션이 동일한 오브젝트를 두 번 이상 열 수 있습니다. 각각 열 때마다 다른 오브젝트 핸들이 리턴됩니다. 리턴되는 각 핸들은 해당 열기가 수행된 함수에 사용될 수 있습니다.
  3. 열리는 오브젝트가 클러스터 큐가 아닌 큐인 경우 로컬 큐 관리자 내의 모든 이름 해석이 MQOPEN 호출 시에 발생합니다. 이는 다음과 같습니다.
    - 리모트 큐 로컬 정의의 이름을 리모트 큐 관리자 이름 및 리모트 큐 관리자에 큐가 인식되는 이름으로 해석
    - 리모트 큐 관리자 이름을 로컬 전송 큐의 이름으로 해석

-  z/OS에만 해당, IGO 에이전트에서 사용하는 공유 전송 큐 이름에 대한 리모트 큐 관리자 이름의 해석(로컬 및 리모트 큐 관리자가 동일한 큐 공유 그룹에 속하는 경우에만 적용됨)
- 기본 큐 또는 토픽 오브젝트의 이름에 대한 알리어스 해석입니다.

그러나, 핸들에 대한 후속 MQINQ 또는 MQSET 호출은 열린 이름에만 관련이 있고 이름 해석 이후 발생한 오브젝트에는 관련이 없습니다. 예를 들어, 열린 오브젝트가 알리어스인 경우 MQINQ 호출로 리턴된 속성은 알리어스의 속성이며 알리어스가 해석하는 토픽 오브젝트 또는 기본 큐의 속성이 아닙니다.

열리는 오브젝트가 클러스터 큐이면 이름 해석이 MQOPEN 호출 시 발생하거나 나중까지 지연될 수 있습니다. 해석이 발생하는 시점은 MQOPEN 호출에서 지정된 MQOO\_BIND\_\* 옵션에 의해 제어됩니다.

- MQOO\_BIND\_ON\_OPEN
- MQOO\_BIND\_NOT\_FIXED
- MQOO\_BIND\_AS\_Q\_DEF
- MQOO\_BIND\_ON\_GROUP

클러스터 큐의 이름 해석에 대한 자세한 정보는 이름 해석의 내용을 참조하십시오.

4. 오브젝트 핸들 및 찾아보기 옵션 중 하나를 지정하는 MQGET 호출과 함께 사용하기 위해 MQOO\_BROWSE 옵션이 있는 MQOPEN 호출이 찾아보기 커서를 설정합니다. 그러면 큐의 콘텐츠를 대체하지 않고 스캔할 수 있습니다. MQGMO\_MSG\_UNDER\_CURSOR 옵션을 사용하여 찾아보기로 발견한 메시지를 큐에서 제거할 수 있습니다.

동일한 큐에 여러 개의 MQOPEN 요청을 발행하면 단일 애플리케이션을 상대로 여러 찾아보기 커서가 활성화될 수 있습니다.

5. 트리거 모니터에 의해 시작된 애플리케이션은 애플리케이션이 시작될 때 애플리케이션과 연관된 큐의 이름이 전달됩니다. 이 큐 이름은 큐를 열기 위해 **ObjDesc** 매개변수에서 지정할 수 있습니다. 추가 세부사항은 585 페이지의 『MQTMC2 - 트리거 메시지 2(문자 형식)』의 내용을 참조하십시오.



## 미리 읽기 옵션

MQOO\_READ\_AHEAD와 함께 MQOPEN을 호출할 경우 특정 조건이 충족되면 IBM MQ 클라이언트에서는 미리 읽기만 가능합니다. 이러한 조건은 다음과 같습니다.

- 클라이언트 애플리케이션이 컴파일되고 스레드된 IBM MQ MQI 클라이언트 라이브러리에 대해 링크되어야 합니다.
- 클라이언트 채널이 TCP/IP 프로토콜을 사용해야 합니다.
- 채널이 클라이언트 및 서버 채널 정의 모두에서 0이 아닌 SharingConversations(SHARECNV) 설정을 사용해야 합니다.

다음은 미리 읽기 옵션의 사용에 적용되는 참고입니다.

1. 미리 읽기 옵션은 MQOO\_BROWSE, MQOO\_INPUT\_SHARED, MQOO\_INPUT\_EXCLUSIVE 옵션 중 하나가 지정된 경우에도 적용됩니다. 미리 읽기 옵션이 MQOO\_INQUIRE 또는 MQOO\_SET 옵션과 함께 지정된 경우에는 오류가 발생하지 않습니다.
2. 첫 번째 MQGET 호출에서 사용되는 옵션이 미리 읽기와 함께 사용하도록 지원되지 않는 경우에는 미리 읽기가 요청될 때 설정되지 않습니다. 또한 클라이언트가 미리 읽기를 지원하지 않는 큐 관리자에 연결되는 경우에도 미리 읽기를 사용할 수 없습니다.
3. 애플리케이션이 IBM MQ 클라이언트로 실행 중이지 않은 경우 미리 읽기 옵션은 무시됩니다.

## 클러스터 큐

다음은 클러스터 큐 사용에 적용되는 참고입니다.

1. 클러스터 큐가 처음으로 열리고 로컬 큐 관리자가 전체 저장소 큐 관리자가 아닌 경우, 로컬 큐 관리자는 전체 저장소 큐 관리자에서 클러스터 큐에 대한 정보를 가져옵니다. 네트워크가 사용 중인 경우 로컬 큐 관리자가 저장소 큐 관리자로부터 필요한 정보를 수신하는 데 몇 초 정도 소요될 수 있습니다. 결과적으로 MQOPEN 호출을 발행하는 애플리케이션은 MQOPEN 호출로부터의 리턴을 제어하기 전에 최대 10초까지 대기해야 할 수도 있습니다. 로컬 큐 관리자가 이 시간 안에 클러스터 큐에 대해 필요한 정보를 수신하지 않는 경우 이유 코드 MQRC\_CLUSTER\_RESOLUTION\_ERROR와 함께 호출이 실패합니다.

2. 클러스터 큐가 열리고 클러스터 내에 여러 큐 인스턴스가 있는 경우 열리는 인스턴스는 MQOPEN 호출에서 지정된 옵션에 따라 다릅니다.

- 옵션을 지정하는 경우, 다음을 포함하십시오.

- MQOO\_BROWSE
- MQOO\_INPUT\_AS\_Q\_DEF
- MQOO\_INPUT\_EXCLUSIVE
- MQOO\_INPUT\_SHARED
- MQOO\_SET

열린 클러스터 큐의 인스턴스는 로컬 인스턴스여야 합니다. 큐의 로컬 인스턴스가 없으면 MQOPEN 호출이 실패합니다.

- 지정된 옵션에 위에서 설명한 옵션이 하나도 포함되지 않는 경우 다음 중 하나 또는 둘 다를 포함합니다.

- MQOO\_INQUIRE
- MQOO\_OUTPUT

열린 인스턴스는 로컬 인스턴스가 있으면 로컬 인스턴스이며 없는 경우에는 리모트 인스턴스입니다 (CLWLUSEQ 기본값을 사용하는 경우). 그러나 클러스터 워크로드 엑시트를 통해 큐 관리자가 선택한 인스턴스를 대체할 수 있습니다(해당 항목이 있는 경우).

3. 큐에 대한 구독이 있으나 전체 저장소에 의해 수신확인되지 않는 경우 오브젝트가 클러스터 내에 없으며 이유 코드 MQRC\_OBJECT\_NAME과 함께 호출이 실패합니다.

클러스터 큐에 대한 자세한 정보는 [클러스터 큐](#)를 참조하십시오.

## 분배 목록

다음은 분배 목록의 사용에 적용되는 참고입니다.

분배 목록은 다음 환경에서 지원됩니다.

-  AIX
-  IBM i
-  Linux
-  Windows

및 해당 시스템에 연결된 IBM MQ MQI clients

1. MQOD 구조 내의 필드는 분배 목록을 열 때 다음과 같이 설정되어야 합니다.

- Version은 MQOD\_VERSION\_2 이상이어야 합니다.
- ObjectType은 MQOT\_Q여야 합니다.
- ObjectName은 공백 또는 널 문자열이어야 합니다.
- ObjectQMgrName은 공백 또는 널 문자열이어야 합니다.
- RecsPresent는 0보다 커야 합니다.
- ObjectRecOffset 및 ObjectRecPtr 중 하나는 0이고 다른 하나는 0이 아니어야 합니다.
- ResponseRecOffset 및 ResponseRecPtr 중 하나 이하가 0이 아니어야 합니다.
- ObjectRecOffset 또는 ObjectRecPtr에 의해 처리되는 RecsPresent 오브젝트 레코드가 있어야 합니다. 오브젝트 레코드는 열리는 목적지 큐의 이름으로 설정되어야 합니다.
- ResponseRecOffset 및 ResponseRecPtr 중 하나가 0이 아니면 RecsPresent 응답 레코드가 있어야 합니다. 호출이 이유 코드 MQRC\_MULTIPLE\_REASONS로 완료되면 이는 큐 관리자에 의해 설정됩니다.

RecsPresent가 0인 경우 분배 목록에 없는 단일 큐를 열기 위해 버전-2 MQOD를 사용할 수 있습니다.

2. **Options** 매개변수에서는 다음 열기 옵션만 올바릅니다.

- MQOO\_OUTPUT
- MQOO\_PASS\_\*\_CONTEXT
- MQOO\_SET\_\*\_CONTEXT
- MQOO\_ALTERNATE\_USER\_AUTHORITY
- MQOO\_FAIL\_IF\_QUIESCING

3. 분배 목록의 목적지 큐는 로컬, 알리아스 또는 리모트 큐가 될 수 있으나 모델 큐는 될 수 없습니다. 모델 큐가 지정된 경우 이유 코드 MQRC\_Q\_TYPE\_ERROR와 함께 해당 큐를 여는 데 실패합니다. 그러나 이것 때문에 목록의 다른 큐를 열지 못하는 것은 아닙니다.

4. 완료 코드 및 이유 코드 매개변수는 다음과 같이 설정됩니다.

- 분배 목록에 있는 큐에 대한 열기 조작이 같은 방법으로 모두 성공하거나 실패하면 완료 코드 및 이유 코드 매개변수가 공용 결과를 설명하도록 설정됩니다. MQRR 응답 레코드(애플리케이션에서 제공하는 경우)는 이 경우에 설정되지 않습니다.

예를 들어, 모든 열기가 성공하면 완료 코드가 MQCC\_OK로 설정되고 이유 코드가 MQRC\_NONE으로 설정됩니다. 존재하는 큐가 없어 모든 열기가 실패하면 매개변수가 MQCC\_FAILED 및 MQRC\_UNKNOWN\_OBJECT\_NAME으로 설정됩니다.

- 분배 목록에 있는 큐에 대한 열기 조작이 모두 같은 방법으로 성공하거나 실패하지 않는 경우
  - 하나 이상의 열기가 성공하면 완료 코드 매개변수가 MQCC\_WARNING으로 설정되고 모두 실패하면 MQCC\_FAILED로 설정됩니다.
  - 이유 코드 매개변수는 MQRC\_MULTIPLE\_REASONS로 설정됩니다.

- 응답 레코드(애플리케이션에 의해 제공되는 경우)가 분배 목록 내의 큐에 대해 개별 완료 코드 및 이유 코드로 설정됩니다.
5. 분배 목록이 성공적으로 열린 경우 호출이 리턴하는 핸들 `Hobj`를 후속 `MQPUT` 호출에 사용하여 분배 목록 내의 큐에 메시지를 넣을 수 있으며 `MQCLOSE` 호출에 사용하여 분배 목록에 대한 액세스를 철회할 수 있습니다. 분배 목록에 대한 유일하게 올바른 닫기 옵션은 `MQCO_NONE`입니다.  
분배 목록에 메시지를 넣는 데 `MQPUT1` 호출을 사용할 수도 있습니다. 목록에서 큐를 정의하는 `MQOD` 구조가 해당 호출에서 매개변수로 지정됩니다.
  6. 애플리케이션이 허용되는 최대 핸들 수를 초과했는지 검사할 때 분배 목록에서 성공적으로 열린 각 목적지는 별도의 핸들로 계수됩니다(**MaxHandles** 큐 관리자 속성 참조). 이는 분배 목록 내의 목적지 중 둘 이상이 동일한 물리적 큐로 해석된 경우에도 적용됩니다. 분배 목록에 대한 `MQOPEN` 또는 `MQPUT1` 호출로 인해 애플리케이션이 사용 중인 핸들 수가 `MaxHandles`를 초과하는 경우 이유 코드 `MQRC_HANDLE_NOT_AVAILABLE`과 함께 호출이 실패합니다.
  7. 목적지가 성공적으로 열릴 때마다 **OpenOutputCount** 속성의 값이 1씩 증가합니다. 분배 목록 내의 목적지 중 둘 이상이 동일한 물리적 큐로 해석된 경우 해당 큐는 해당 큐에 대해 해석하는 분배 목록 내의 목적지의 수만큼 증가하는 **OpenOutputCount** 속성을 가집니다.
  8. 핸들을 올바르게 않게 만드는 큐 정의의 변경사항으로 인해 큐가 개별적으로 열린 경우(예: 해석 경로 변경), 분배 목록 핸들이 올바르게 되지 않습니다. 그러나 분배 목록 핸들이 후속 `MQPUT` 호출에서 사용될 때 이는 특정 큐의 실패를 유발합니다.
  9. 분배 목록은 단 하나의 목적지만 포함할 수 있습니다.

## 리모트 큐

다음은 리모트 큐의 사용에 적용되는 참고입니다.

리모트 큐는 이 호출의 **ObjDesc** 매개변수에서 두 가지 방법 중 하나를 사용하여 지정될 수 있습니다.

- `ObjectName`에 리모트 큐의 로컬 정의 이름을 지정하는 방법입니다. 이 경우 `ObjectQMgrName`은 로컬 큐 관리자를 참조하며 공백 또는 (C 프로그래밍 언어에서) 널 문자열로 지정될 수 있습니다.

로컬 큐 관리자에 의해 수행되는 보안 유효성 검증은 사용자가 리모트 큐의 로컬 정의를 열 수 있는 권한이 있는지 확인합니다.

- `ObjectName`에 리모트 큐 관리자에 알려진 대로 리모트 큐의 이름을 지정하는 방법입니다. 이 경우 `ObjectQMgrName`은 리모트 큐 관리자의 이름입니다.

로컬 큐 관리자에 의해 수행되는 보안 유효성 검증은 사용자가 이름 해석 프로세스에서 발생하는 전송 큐에 메시지를 송신할 수 있는 권한이 있는지 확인합니다.

두 경우 모두 다음 사항이 적용됩니다.

- 사용자에게 큐에 메시지를 넣을 권한이 부여되었는지 확인하기 위해 로컬 큐 관리자가 리모트 큐 관리자에 메시지를 전송하지 않습니다.
- 메시지가 리모트 큐 관리자에 도착할 때 메시지를 생성한 사용자에게 권한이 없으므로 리모트 큐 관리자가 이를 거부할 수 있습니다.

자세한 정보는 466 페이지의 『MQOD - 오브젝트 디스크립터』에 설명된 `ObjectName` 및 `ObjectQMgrName` 필드를 참조하십시오.

## 오브젝트

### 보안

다음은 `MQOPEN` 사용 시 보안과 관련된 참고입니다.

`MQOPEN` 호출 발행 시 액세스 허용 전에 애플리케이션이 실행 중인 사용자 ID에 적절할 레벨의 권한이 있는지 확인하기 위해 큐 관리자가 보안 검사를 수행합니다. 권한 검사는 이름이 해석된 후에 발생하는 이름이 아니라 열리는 오브젝트의 이름에서 수행됩니다.

열리는 오브젝트가 토픽 오브젝트를 가리키는 알리어스 큐인 경우 토픽 오브젝트가 직접 사용된 경우처럼 토픽에 대한 보안 검사를 수행하기 전에 큐 관리자가 알리어스 큐 이름에서 보안 검사를 수행합니다.

열리는 오브젝트가 토픽 오브젝트인 경우 `ObjectName`만 사용하거나 기본 `ObjectName`과 함께 또는 단독으로 `ObjectString`을 사용하는지 여부에 상관없이 큐 관리자는 `ObjectName`에서 지정된 토픽 오브젝트 내에서 가져온 결과 토픽 문자열을 사용하여 보안 검사를 수행합니다. 또한 필요한 경우 `ObjectString`에서 제공된 문자열과 결합한 다음 토픽 트리 내의 해당 지점 또는 그 위 지점에 있는 가장 가까운 토픽 오브젝트를 발견하여 보안 검사를 수행합니다. 이 때, `ObjectName`에서 지정된 토픽 오브젝트와 동일하지 않을 수 있습니다.

열리는 오브젝트가 모델 큐인 경우, 큐 관리자가 모델 큐의 이름 및 작성되는 동적 큐의 이름 둘 다에 대해 전체 보안 검사를 수행합니다. 그럼 다음, 결과 동적 큐를 명시적으로 열면 동적 큐의 이름에 대해 추가적인 자원 보안 검사가 수행됩니다.

**z/OS** z/OS에서는 보안이 사용 가능한 경우에만 큐 관리자가 보안 검사를 수행합니다. 보안 검사에 대한 자세한 정보는 z/OS에서 보안 설정을 참조하십시오.

## 속성

다음은 속성에 관한 참고입니다.

오브젝트의 속성은 애플리케이션이 오브젝트를 여는 동안 변경될 수 있습니다. 대부분의 경우, 애플리케이션에서 이를 표시하지 않지만 특정 속성은 큐 관리자가 핸들을 더 이상 올바르게 않은 것으로 표시합니다. 이러한 속성은 다음과 같습니다.

- 오브젝트의 이름 해석에 영향을 미치는 모든 속성. 이 속성은 사용되는 열기 옵션에 상관없이 적용되며 다음을 포함합니다.
  - 열린 알리어스 큐의 **BaseQName** 속성 변경사항
  - 열린 알리어스 큐의 **TargetType** 속성 변경사항
  - 이 큐에 대해 열린 모든 핸들 또는 큐 관리자 알리어스로 이 정의를 통해 해석되는 큐에 대해 **RemoteQName** 또는 **RemoteQMgrName** 큐 속성에 대한 변경입니다.
  - 현재 리모트 큐에 대해 열린 핸들이 다른 전송 큐를 해석하도록 하거나 전혀 해석하지 못하도록 만드는 모든 변경입니다. 예를 들어, 다음과 같습니다.
    - 정의가 큐 또는 큐 관리자 알리어스에 사용되는지 여부에 상관없이 리모트 큐 로컬 정의의 **XmitQName** 속성 변경사항.
  - **z/OS** z/OS 에서만, **IntraGroupQueuing** 큐 관리자 속성의 값에 대한 변경 또는 공유 전송 큐의 정의에 대한 변경 (**SYSTEM.QSG.TRANSMIT.QUEUE**) 가 IGQ 에이전트에서 사용됩니다.

새 전송 큐의 작성에 대한 하나의 예외가 있습니다. 핸들을 열 때 핸들이 있으면 이 큐로 해석되어야 하지만 기본 전송 큐로 해석된 핸들은 올바르게 않은 것으로 간주되지 않습니다.
  - **DefXmitQName** 큐 관리자 속성 변경사항. 이 경우, 이전에 명명된 큐로 해석된 모든 열기 핸들(기본 전송 큐란 이유만으로 해당 큐로 해석됨)이 올바르게 않은 것으로 표시됩니다. 그 외 이유로 이 큐로 해석된 핸들은 영향을 받지 않습니다.
- 현재 이 큐 또는 이 큐에 대해 해석하는 큐에 대해 **MQOO\_INPUT\_SHARED** 액세스를 제공하는 핸들이 두 개 이상일 때 **Shareability** 큐 속성입니다. 해당되는 경우 이 큐 또는 이 큐로 해석되는 큐에 대해 열려 있는 모든 핸들은 열기 옵션에 상관없이 유효하지 않음으로 표시됩니다.

**z/OS** z/OS에서는 하나 이상의 핸들이 큐에 대해 **MQOO\_INPUT\_SHARED** 또는 **MQOO\_INPUT\_EXCLUSIVE** 액세스를 제공하는 경우 이전에 설명한 핸들이 올바르게 않은 것으로 표시됩니다.

- 열기 옵션에 상관없이, 이 큐 또는 이 큐로 해석되는 큐에 열려 있는 모든 핸들에 대한 **Usage** 큐 속성.

핸들이 올바르게 않은 것으로 표시되면 이 핸들을 사용하는 모든 후속 호출(**MQCLOSE** 제외)이 이유 코드 **MQRC\_OBJECT\_CHANGED**와 함께 실패합니다. 애플리케이션이 원래 핸들을 사용하여 **MQCLOSE** 호출을 발행한 후 큐를 다시 열어야 합니다. 이전의 성공적인 호출에서 이전 핸들에 대해 커미트되지 않은 업데이트는 애플리케이션 논리에 따라 여전히 커미트되거나 백아웃된 상태일 수 있습니다.

속성을 변경하면 이런 현상이 발생할 수 있으므로 호출의 특수 강제 실행 버전을 사용하십시오.

## C 호출

```
MQOPEN (Hconn, &ObjDesc, Options, &Hobj, &CompCode,  
&Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN Hconn; /* Connection handle */  
MQOD ObjDesc; /* Object descriptor */  
MQLONG Options; /* Options that control the action of MQOPEN */  
MQHOBJ Hobj; /* Object handle */  
MQLONG CompCode; /* Completion code */  
MQLONG Reason; /* Reason code qualifying CompCode */
```

## COBOL 호출

```
CALL 'MQOPEN' USING HCONN, OBJDESC, OPTIONS, HOBJ, COMPCODE, REASON
```

매개변수를 다음과 같이 선언합니다.

```
** Connection handle  
01 HCONN PIC S9(9) BINARY.  
** Object descriptor  
01 OBJDESC.  
COPY CMQODV.  
** Options that control the action of MQOPEN  
01 OPTIONS PIC S9(9) BINARY.  
** Object handle  
01 HOBJ PIC S9(9) BINARY.  
** Completion code  
01 COMPCODE PIC S9(9) BINARY.  
** Reason code qualifying COMPCODE  
01 REASON PIC S9(9) BINARY.
```

## PL/I 호출

```
call MQOPEN (Hconn, ObjDesc, Options, Hobj, CompCode, Reason);
```

매개변수를 다음과 같이 선언합니다.

```
dcl Hconn fixed bin(31); /* Connection handle */  
dcl ObjDesc like MQOD; /* Object descriptor */  
dcl Options fixed bin(31); /* Options that control the action of  
MQOPEN */  
dcl Hobj fixed bin(31); /* Object handle */  
dcl CompCode fixed bin(31); /* Completion code */  
dcl Reason fixed bin(31); /* Reason code qualifying CompCode */
```

## 상위 레벨 어셈블러 호출

```
CALL MQOPEN, (HCONN,OBJDESC,OPTIONS,HOBJ,COMPCODE,REASON)
```

매개변수를 다음과 같이 선언합니다.

```
HCONN DS F Connection handle  
OBJDESC CMQODA , Object descriptor  
OPTIONS DS F Options that control the action of MQOPEN  
HOBJ DS F Object handle  
COMPCODE DS F Completion code  
REASON DS F Reason code qualifying COMPCODE
```

## Visual Basic 호출

### Windows

MQOPEN Hconn, ObjDesc, Options, Hobj, CompCode, Reason

매개변수를 다음과 같이 선언합니다.

```
Dim Hconn As Long 'Connection handle'  
Dim ObjDesc As MQOD 'Object descriptor'  
Dim Options As Long 'Options that control the action of MQOPEN'  
Dim Hobj As Long 'Object handle'  
Dim CompCode As Long 'Completion code'  
Dim Reason As Long 'Reason code qualifying CompCode'
```

## MQPUT - 넣기 메시지

MQPUT 호출은 큐 또는 분배 목록에서 또는 토픽에 하나의 메시지를 넣습니다. 큐, 분배 목록 또는 토픽은 이미 열려 있는 상태여야 합니다.

### 구문

MQPUT(*Hconn, Hobj, MsgDesc, PutMsgOpts, BufferLength, Buffer, CompCode, Reason*)

### 매개변수

#### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. Hconn의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.



z/OS for CICS 애플리케이션에서 MQCONN 호출을 생략할 수 있으며 Hconn에 다음 값이 지정됩니다.

#### MQHC\_DEF\_HCONN

기본 연결 핸들

#### Hobj

유형: MQHOBJ - 입력

이 핸들은 메시지가 추가되는 큐 또는 메시지가 발행되는 토픽을 나타냅니다. Hobj의 값은 MQOO\_OUTPUT 옵션을 지정한 이전 MQOPEN 호출에 의해 리턴되었습니다.

#### MsgDesc

유형: MQMD - 입출력(I/O)

이 구조는 송신하는 메시지의 속성을 설명하며, Put 요청이 완료된 후 메시지에 대한 정보를 수신합니다. 세부사항은 412 페이지의 『MQMD - 메시지 디스크립터』을(를) 참조하십시오.

애플리케이션이 버전-1 MQMD를 제공하는 경우 버전-1이 아니라 버전-2 MQMD에 있는 필드에 값을 지정하기 위해 메시지 데이터에 MQMDE 구조를 접두부로 지정할 수 있습니다. MQMD의 *Format* 필드를 MQFMT\_MD\_EXTENSION으로 설정하여 MQMDE가 있음을 표시해야 합니다. 자세한 내용은 458 페이지의 『MQMDE - 메시지 디스크립터 확장』의 내용을 참조하십시오.

올바른 메시지 핸들이 MQPMO 구조의 OriginalMsgHandle 또는 NewMsgHandle 필드에서 제공되는 경우 애플리케이션이 MQMD 구조를 제공할 필요는 없습니다. 해당 필드 중 하나에서 제공되는 사항이 없는 경우 메시지 핸들과 연관된 디스크립터에서 메시지에 대한 디스크립터를 가져옵니다.

API 엑시트를 사용하거나 사용하려는 경우 MQMD 구조를 명시적으로 제공하고 메시지 핸들과 연관된 메시지 디스크립터를 사용하지 않는 것이 좋습니다. 이는 MQPUT 또는 MQPUT1 호출과 연관된 API 엑시트가 MQPUT 또는 MQPUT1 요청을 완료하기 위해 큐 관리자가 어떤 MQMD 값을 사용하는지 확인할 수 없기 때문입니다.



## PutMsgOpts

유형: MQPMO - 입출력(I/O)

자세한 내용은 485 페이지의 『MQPMO - 메시지 넣기 옵션』의 내용을 참조하십시오.

## BufferLength

유형: MQLONG - 입력

Buffer에서 메시지의 길이입니다. 올바른 값은 0이며, 메시지에 애플리케이션 데이터가 없음을 표시합니다. BufferLength에 대한 상한은 다양한 요인에 따라 다릅니다.

- 목적지가 로컬 큐이거나 로컬 큐로 해석되면 상한은 다음의 여부에 따라 다릅니다.
  - 로컬 큐 관리자가 세그먼트화를 지원합니다.
  - 전송 애플리케이션은 큐 관리자가 메시지를 세그먼트화하도록 하는 플래그를 지정합니다. 이 플래그는 MQMF\_SEGMENTATION\_ALLOWED이며 버전-2 MQMD 또는 버전-1 MQMD와 함께 사용된 MQMDE에서 지정할 수 있습니다.

해당 조건이 모두 충족되면 BufferLength는 999,999,999에서 MQMD의 Offset 필드 값을 뺀 값을 초과할 수 없습니다. 그러므로 넣을 수 있는 가장 긴 논리 메시지는 999,999,999바이트입니다(Offset이 0인 경우). 그러나 애플리케이션이 실행 중인 운영 체제 또는 환경에서 도입되는 자원 제한조건으로 하한이 발생할 수 있습니다.

이전 조건 중 하나 또는 둘 모두가 충족되지 않는 경우 BufferLength는 더 작은 큐의 MaxMsgLength 속성 및 큐 관리자의 MaxMsgLength 속성을 초과할 수 없습니다.

- 대상이 리모트 큐이거나 리모트 큐로 해석되는 경우, 로컬 큐의 조건이 적용됩니다. 목적지 큐에 도달하기 위해 메시지가 통과해야 하는 각 큐 관리자와 다음 큐에서 특히 그렇습니다.

1. 로컬 큐 관리자에 임시로 메시지를 저장하는 데 사용되는 로컬 송신 큐
2. 로컬 및 목적지 큐 관리자 간 라우트에 있는 큐 관리자에 메시지를 저장하는 데 사용되는 중간 전송 큐 (있는 경우)
3. 목적지 큐 관리자에 있는 목적지 큐

따라서 넣을 수 있는 가장 긴 메시지는 이 큐 및 큐 관리자의 가장 제한적인 값에 따라 통제됩니다.

메시지가 전송 큐에 있을 때 추가 정보가 메시지 데이터와 함께 있으며, 이로 인해 이동할 수 있는 애플리케이션 데이터의 양이 줄어듭니다. 이 상황에서 BufferLength에 대한 한계를 판별할 때 전송 큐의 MaxMsgLength 값에서 MQ\_MSG\_HEADER\_LENGTH 바이트를 빼십시오.

**참고:** 메시지를 넣을 때 조건 1을 준수하는 실패만 동시에 진단할 수 있습니다(이유 코드 MQRC\_MSG\_TOO\_BIG\_FOR\_Q 또는 MQRC\_MSG\_TOO\_BIG\_FOR\_Q\_MGR). 조건 2 또는 3을 충족하지 않는 경우, 메시지가 중간 큐 관리자 또는 목적지 큐 관리자에서 데드 레터(미배달 메시지) 큐로 경로 재지정됩니다. 이 경우, 송신자의 요청이 있으면 보고 메시지가 생성됩니다.

## 버퍼

유형: MQBYTExBufferLength - 입력

송신할 애플리케이션 데이터가 있는 버퍼입니다. 버퍼는 메시지에 있는 데이터의 네이처에 적절한 경계에 맞춰야 합니다. 4바이트 맞추기는 대부분의 메시지에 적합하지만(IBM MQ 헤더 구조가 포함된 메시지 포함) 일부 메시지는 더 엄격한 맞추기가 필요할 수 있습니다. 예를 들어, 64비트 2진 정수를 포함하는 메시지에는 8바이트 맞추기가 필요할 수 있습니다.

Buffer에 문자 또는 숫자 데이터가 포함된 경우 MsgDesc 매개변수의 CodedCharSetId 및 Encoding 필드를 데이터에 적합한 값으로 설정하십시오. 이를 통해 메시지 수신자는 필요한 경우 데이터를 수신자가 사용하는 문자 세트 및 인코딩으로 변환할 수 있습니다.

**참고:** MQPUT 호출의 다른 모든 매개변수는 로컬 큐 관리자의 문자 세트 및 인코딩에 있어야 합니다 (CodedCharSetId 큐 관리자 속성 및 MQENC\_NATIVE로 지정됨).

C 프로그래밍 언어에서 매개변수는 pointer-to-void로서 선언됩니다. 임의의 데이터 유형의 주소를 매개변수로 지정할 수 있습니다.

BufferLength 매개변수가 0인 경우 Buffer는 참조되지 않습니다. 이 경우 C 또는 시스템/390 어셈블러로 작성된 프로그램이 전달한 매개변수 주소가 널일 수 있습니다.

## CompCode

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

### **MQCC\_OK**

정상적으로 완료되었습니다.

### **MQCC\_WARNING**

경고(일부 완료).

### **MQCC\_FAILED**

호출에 실패했습니다.

## 원인

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드입니다.

CompCode이(가) MQCC\_OK인 경우:

### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode이(가) MQCC\_WARNING인 경우:

### **MQRC\_INCOMPLETE\_GROUP**

(2241, X'8C1') 메시지 그룹이 완료되지 않았습니다.

### **MQRC\_INCOMPLETE\_MSG**

(2242, X'8C2') 논리 메시지가 완료되지 않았습니다.

### **MQRC\_INCONSISTENT\_PERSISTENCE**

(2185, X'889') 불일치하는 지속성 스펙.

### **MQRC\_INCONSISTENT\_UOW**

(2245, X'8C5') 작업 단위 지정에 일관성이 없습니다.

### **MQRC\_MULTIPLE\_REASONS**

(2136, X'858') 다중 이유 코드가 리턴되었습니다.

### **MQRC\_PRIORITY\_EXCEEDS\_MAXIMUM**

(2049, X'801') 메시지 우선순위가 지원되는 최대 값을 초과합니다.

### **MQRC\_UNKNOWN\_REPORT\_OPTION**

(2104, X'838') 메시지 디스크립터의 보고서 옵션이 인식되지 않습니다.

CompCode이(가) MQCC\_FAILED인 경우:

### **MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') 어댑터를 사용할 수 없습니다.

### **MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

### **MQRC\_ALIAS\_TARGTYPE\_CHANGED**

(2480, X'09B0') 구독 대상 유형이 큐에서 토픽 오브젝트로 변경되었습니다.

### **MQRC\_API\_EXIT\_ERROR**

(2374, X'946') API 엑시트가 실패했습니다.

### **MQRC\_API\_EXIT\_LOAD\_ERROR**

(2183, X'887') API 엑시트를 로드할 수 없습니다.

### **MQRC\_ASID\_MISMATCH**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

### **MQRC\_BACKED\_OUT**

(2003, X'7D3') 작업 단위가 백아웃되었습니다.

### **MQRC\_BUFFER\_ERROR**

(2004, X'7D4') 버퍼 매개변수가 올바르지 않습니다.



**MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'7D5') 버퍼 길이 매개변수가 올바르지 않습니다.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**MQRC\_CALL\_INTERRUPTED**

(2549, X'9F5') MQPUT 또는 MQCMIT가 인터럽트되었으며 다시 연결 처리가 명백한 결과를 재설정할 수 없습니다.

**MQRC\_CF\_NOT\_AVAILABLE**

(2345, X'929') 커플링 기능이 사용 불가능합니다.

**MQRC\_CF\_STRUC\_FAILED**

(2373, X'945') 커플링 기능 구조가 실패했습니다.

**MQRC\_CF\_STRUC\_IN\_USE**

(2346, X'92A') 커플링 기능 구조가 사용 중입니다.

**MQRC\_CFGR\_ERROR**

(2416, X'970') 메시지 데이터의 PCF 그룹 매개변수 구조 MQCFGR이 올바르지 않습니다.

**MQRC\_CFH\_ERROR**

(2235, X'8BB') PCF 헤더 구조가 올바르지 않습니다.

**MQRC\_CFIF\_ERROR**

(2414, X'96E') 메시지 데이터의 PCF 정수 필터 매개변수 구조가 올바르지 않습니다.

**MQRC\_CFIL\_ERROR**

(2236, X'8BC') PCF 정수 목록 매개변수 구조 또는 PCIF\*64 정수 목록 매개변수 구조가 올바르지 않습니다.

**MQRC\_CFIN\_ERROR**

(2237, X'8BD') PCF 정수 매개변수 구조 또는 PCIF\*64 정수 매개변수 구조가 올바르지 않습니다.

**MQRC\_CFSF\_ERROR**

(2415, X'96F') 메시지 데이터의 PCF 문자열 필터 매개변수 구조가 올바르지 않습니다.

**MQRC\_CFSL\_ERROR**

(2238, X'8BE') PCF 문자열 목록 매개변수 구조가 올바르지 않습니다.

**MQRC\_CFST\_ERROR**

(2239, X'8BF') PCF 문자열 매개변수 구조가 올바르지 않습니다.

**MQRC\_CICS\_WAIT\_FAILED**

(2140, X'85C') CICS에서 대기 요청이 거부되었습니다.

**MQRC\_CLUSTER\_EXIT\_ERROR**

(2266, X'8DA') 클러스터 워크로드 엑시트가 실패했습니다.

**MQRC\_CLUSTER\_RESOLUTION\_ERROR**

(2189, X'88D') 클러스터 이름을 해석하지 못했습니다.

**MQRC\_CLUSTER\_RESOURCE\_ERROR**

(2269, X'8DD') 클러스터 자원에 오류가 있습니다.

**MQRC\_COD\_NOT\_VALID\_FOR\_XCF\_Q**

(2106, X'83A') XCF 큐에 대해 COD 보고서 옵션이 올바르지 않습니다.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**MQRC\_CONNECTION\_NOT\_AUTHORIZED**

(2217, X'8A9') 연결할 권한이 부여되지 않았습니다.

**MQRC\_CONNECTION\_QUIESCING**

(2202, X'89A') 연결이 정지됩니다.

**MQRC\_CONNECTION\_STOPPING**

(2203, X'89B') 연결이 종료됩니다.

**MQRC\_CONTENT\_ERROR**

2554 (X'09FA') 확장된 메시지 선택자가 있는 구독자에 메시지를 전달해야 하는지 여부를 판별하기 위해 메시지 콘텐츠를 구문 분석할 수 없습니다.

**MQRC\_CONTEXT\_HANDLE\_ERROR**

(2097, X'831') 참조되는 큐 핸들이 컨텍스트를 저장하지 않습니다.

**MQRC\_CONTEXT\_NOT\_AVAILABLE**

(2098, X'832') 참조되는 큐 핸들에 대해 컨텍스트를 사용할 수 없습니다.

**MQRC\_DATA\_LENGTH\_ERROR**

(2010, X'7DA') 데이터 길이 매개변수가 올바르지 않습니다.

**MQRC\_DH\_ERROR**

(2135, X'857') 분배 헤더 구조가 올바르지 않습니다.

**MQRC\_DLH\_ERROR**

(2141, X'85D') 데드 레터 헤더 구조가 올바르지 않습니다.

**MQRC\_EPH\_ERROR**

(2420, X'974') 임베드된 PCF 구조가 올바르지 않습니다.

**MQRC\_EXPIRY\_ERROR**

(2013, X'7DD') 만기 시간이 올바르지 않습니다.

**MQRC\_FEEDBACK\_ERROR**

(2014, X'7DE') 피드백 코드가 올바르지 않습니다.

**MQRC\_GLOBAL\_UOW\_CONFLICT**

(2351, X'92F') 글로벌 작업 단위 충돌입니다.

**MQRC\_GROUP\_ID\_ERROR**

(2258, X'8D2') 그룹 ID가 올바르지 않습니다.

**MQRC\_HANDLE\_IN\_USE\_FOR\_UOW**

(2353, X'931') 글로벌 작업 단위에 대해 사용 중인 핸들입니다.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**MQRC\_HEADER\_ERROR**

(2142, X'85E') MQ 헤더 구조가 올바르지 않습니다.

**MQRC\_HOBJ\_ERROR**

(2019, X'7E3') 오브젝트 핸들이 올바르지 않습니다.

**MQRC\_IIH\_ERROR**

(2148, X'864') IMS 정보 헤더 구조가 올바르지 않습니다.

**MQRC\_INCOMPLETE\_GROUP**

(2241, X'8C1') 메시지 그룹이 완료되지 않았습니다.

**MQRC\_INCOMPLETE\_MSG**

(2242, X'8C2') 논리 메시지가 완료되지 않았습니다.

**MQRC\_INCONSISTENT\_PERSISTENCE**

(2185, X'889') 불일치하는 지속성 스펙.

**MQRC\_INCONSISTENT\_UOW**

(2245, X'8C5') 작업 단위 지정에 일관성이 없습니다.

**MQRC\_LOCAL\_UOW\_CONFLICT**

(2352, X'930') 글로벌 작업 단위가 로컬 작업 단위와 충돌합니다.

**MQRC\_MD\_ERROR**

(2026, X'7EA') 메시지 디스크립터가 올바르지 않습니다.

**MQRC\_MDE\_ERROR**

(2248, X'8C8') 메시지 디스크립터 확장이 올바르지 않습니다.

**MQRC\_MISSING\_REPLY\_TO\_Q**

(2027, X'7EB') 누락된 응답 대상 큐 또는 MQPMO\_SUPPRESS\_REPLYTO가 사용되었습니다.

**MQRC\_MISSING\_WIH**  
(2332, X'91C') 메시지 데이터가 MQWIH로 시작하지 않습니다.

**MQRC\_MSG\_FLAGS\_ERROR**  
(2249, X'8C9') 메시지 플래그가 올바르지 않습니다.

**MQRC\_MSG\_SEQ\_NUMBER\_ERROR**  
(2250, X'8CA') 메시지 순서 번호가 올바르지 않습니다.

**MQRC\_MSG\_TOO\_BIG\_FOR\_Q**  
(2030, X'7EE') 메시지 길이가 큐의 최대 길이보다 깁니다.

**MQRC\_MSG\_TOO\_BIG\_FOR\_Q\_MGR**  
(2031, X'7EF') 메시지 길이가 큐 관리자의 최대값보다 큽니다.

**MQRC\_MSG\_TYPE\_ERROR**  
(2029, X'7ED') 메시지 디스크립터의 메시지 유형이 올바르지 않습니다.

**MQRC\_MULTIPLE\_REASONS**  
(2136, X'858') 다중 이유 코드가 리턴되었습니다.

**MQRC\_NO\_DESTINATIONS\_AVAILABLE**  
(2270, X'8DE') 목적지 큐가 사용 가능하지 않습니다.

**MQRC\_NOT\_OPEN\_FOR\_OUTPUT**  
(2039, X'7F7') 출력을 큐를 열지 못했습니다.

**MQRC\_NOT\_OPEN\_FOR\_PASS\_ALL**  
(2093, X'82D') 모든 컨텍스트 전달을 위해 큐를 열지 못했습니다.

**MQRC\_NOT\_OPEN\_FOR\_PASS\_IDENT**  
(2094, X'82E') ID 컨텍스트 전달을 위해 큐를 열지 못했습니다.

**MQRC\_NOT\_OPEN\_FOR\_SET\_ALL**  
(2095, X'82F') 모든 컨텍스트 설정을 위해 큐를 열지 못했습니다.

**MQRC\_NOT\_OPEN\_FOR\_SET\_IDENT**  
(2096, X'830') ID 컨텍스트 설정을 위해 큐를 열지 못했습니다.

**MQRC\_OBJECT\_CHANGED**  
(2041, X'7F9') 오브젝트가 열린 후에 정의가 변경되었습니다.

**MQRC\_OBJECT\_DAMAGED**  
(2101, X'835') 오브젝트가 손상되었습니다.

**MQRC\_OFFSET\_ERROR**  
(2251, X'8CB') 메시지 세그먼트 오프셋이 올바르지 않습니다.

**MQRC\_OPEN\_FAILED**  
(2137, X'859') 오브젝트가 성공적으로 열리지 않았습니다.

**MQRC\_OPTIONS\_ERROR**  
(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.

**MQRC\_ORIGINAL\_LENGTH\_ERROR**  
(2252, X'8CC') 원래 길이가 올바르지 않습니다.

**MQRC\_PAGESET\_ERROR**  
(2193, X'891') 페이지 세트 데이터 세트에 액세스하는 중에 오류가 발생했습니다.

**MQRC\_PAGESET\_FULL**  
(2192, X'890') 외부 스토리지 매체가 가득 찼습니다.

**MQRC\_PCF\_ERROR**  
(2149, X'865') PCF 구조가 올바르지 않습니다.

**MQRC\_PERSISTENCE\_ERROR**  
(2047, X'7FF') 지속이 올바르지 않습니다.

**MQRC\_PERSISTENT\_NOT\_ALLOWED**  
(2048, X'800') 큐가 지속 메시지를 지원하지 않습니다.

**MQRC\_PMO\_ERROR**  
(2173, X'87D') 메시지 넣기 옵션 구조가 올바르지 않습니다.

**MQRC\_PMO\_RECORD\_FLAGS\_ERROR**

(2158, X'86E') 메시지 넣기 레코드 플래그가 올바르지 않습니다.

**MQRC\_PRIORITY\_ERROR**

(2050, X'802') 메시지 우선순위가 올바르지 않습니다.

**MQRC\_PUBLICATION\_FAILURE**

(2502, X'9C6') 발행이 구독자에 전달되지 않았습니다.

**MQRC\_PUT\_INHIBITED**

(2051, X'803') 넣기 호출이 큐, 이 큐가 해결하는 큐 또는 토픽에 대해 금지됩니다.

**MQRC\_PUT\_MSG\_RECORDS\_ERROR**

(2159, X'86F') 메시지 넣기 레코드가 올바르지 않습니다.

**MQRC\_PUT\_NOT\_RETAINED**

(2479, X'09AF') 발행을 보유할 수 없습니다.

**MQRC\_Q\_DELETED**

(2052, X'804') 큐가 삭제되었습니다.

**MQRC\_Q\_FULL**

(2053, X'805') 큐에 이미 최대 수의 메시지가 포함되어 있습니다.

**MQRC\_Q\_MGR\_NAME\_ERROR**

(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**

(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.

**MQRC\_Q\_MGR QUIESCING**

(2161, X'871') 큐 관리자가 정지됩니다.

**MQRC\_Q\_MGR STOPPING**

(2162, X'872') 큐 관리자가 종료되었습니다.

**MQRC\_Q\_SPACE\_NOT\_AVAILABLE**

(2056, X'808') 큐에 사용할 수 있는 공간이 디스크에 없습니다.

**MQRC\_RECONNECT\_FAILED**

(2548, X'9F4') 다시 연결 후에 다시 연결 가능한 연결을 위해 핸들을 회복하는 중에 오류가 발생했습니다.

**MQRC\_RECS\_PRESENT\_ERROR**

(2154, X'86A') 현재 레코드 수가 유효하지 않습니다.

**MQRC\_REPORT\_OPTIONS\_ERROR**

(2061, X'80D') 메시지 디스크립터의 보고 옵션이 올바르지 않습니다.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**MQRC\_RESPONSE\_RECORDS\_ERROR**

(2156, X'86C') 응답 레코드가 올바르지 않습니다.

**MQRC\_RFH\_ERROR**

(2334, X'91E') MQRFH 또는 MQRFH2 구조가 올바르지 않습니다.

**MQRC\_RMH\_ERROR**

(2220, X'8AC') 참조 메시지 헤더 구조가 올바르지 않습니다.

**MQRC\_SEGMENT\_LENGTH\_ZERO**

(2253, X'8CD') 메시지 세그먼트의 데이터 길이가 0입니다.

**MQRC\_SEGMENTS\_NOT\_SUPPORTED**

(2365, X'93D') 세그먼트가 지원되지 않습니다.

**MQRC\_SELECTION\_NOT\_AVAILABLE**

2551 (X'09F7') 발행에 가능한 구독자가 있지만 큐 관리자가 구독자에 발행을 전송할지 여부를 확인할 수 없습니다.

**MQRC\_STOPPED\_BY\_CLUSTER\_EXIT**

(2188, X'88C') 클러스터 워크로드 엑시트에 의해 호출이 거부되었습니다.

**MQRC\_STORAGE\_CLASS\_ERROR**

(2105, X'839') 스토리지 클래스 오류입니다.

**MQRC\_STORAGE\_MEDIUM\_FULL**

(2192, X'890') 외부 스토리지 매체가 가득 찼습니다.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**MQRC\_SUPPRESSED\_BY\_EXIT**

(2109, X'83D') 엑시트 프로그램에서 호출이 억제되었습니다.

**MQRC\_SYNCPOINT\_LIMIT\_REACHED**

(2024, X'7E8') 현재 작업 단위 내에서 더 이상 메시지를 핸들링할 수 없습니다.

**MQRC\_SYNCPOINT\_NOT\_AVAILABLE**

(2072, X'818') 동기점 지원을 사용할 수 없습니다.

**MQRC\_TM\_ERROR**

(2265, X'8D9') 트리거 메시지 구조가 올바르지 않습니다.

**MQRC\_TMC\_ERROR**

(2191, X'88F') 문자 트리거 메시지 구조가 올바르지 않습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

**MQRC\_UOW\_ENLISTMENT\_ERROR**

(2354, X'932') 글로벌 작업 단위에서의 등록이 실패했습니다.

**MQRC\_UOW\_MIX\_NOT\_SUPPORTED**

(2355, X'933') 작업 단위 호출의 혼합은 지원되지 않습니다.

**MQRC\_UOW\_NOT\_AVAILABLE**

(2255, X'8CF') 사용할 큐 관리자에 대해 작업 단위를 사용할 수 없습니다.

**MQRC\_WIH\_ERROR**

(2333, X'91D') MQWIH 구조가 올바르지 않습니다.

**MQRC\_WRONG\_MD\_VERSION**

(2257, X'8D1') 올바르지 않은 버전의 MQMD가 제공되었습니다.

**MQRC\_XQH\_ERROR**

(2260, X'8D4') 전송 큐 헤더 구조가 올바르지 않습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

## 토픽 사용법 참고

1. 다음 참고사항이 토픽 사용에 적용됩니다.

a. MQPUT를 사용하여 메시지를 토픽에 발행할 때 구독자 큐의 문제(예: 가득 참) 때문에 해당 토픽에 대한 하나 이상의 구독자에게 발행물을 제공할 수 없으면 MQPUT 호출에 리턴되는 이유 코드와 전달 동작은 TOPIC의 PMSGDLV 또는 NPMSGDLV 속성 설정에 따라 다릅니다. MQRO\_DEAD\_LETTER\_Q가 지정될 때 데드-레터 큐에 발행 전달 또는 MQRO\_DISCARD\_MSG가 지정될 때 메시지 제거는 성공적인 메시지 전달로 간주됨을 참고하십시오. 발행이 전달되지 않는 경우 MQPUT은 MQRC\_PUBLICATION\_FAILURE와 함께 리턴됩니다. 이러한 상태는 다음의 경우에 발생할 수 있습니다.

- (메시지 지속성에 따라) PMSGDLV 또는 NPMSGDLV를 ALL로 설정한 상태에서 메시지가 TOPIC에 발행되고, 구독(지속 또는 비지속)에 발행물을 수신할 수 없는 큐가 있습니다.
- (메시지 지속성에 따라) PMSGDLV 또는 NPMSGDLV를 ALLDUR로 설정한 상태에서 메시지가 TOPIC에 발행되고, 지속 구독에 발행물을 수신할 수 없는 큐가 있습니다.

다음 경우 발행을 일부 구독자에게 전달할 수 없어도 MQPUT은 MQRC\_NONE과 함께 리턴할 수 있습니다.

- (메시지 지속성에 따라) PMSGDLV 또는 NPMSGDLV를 ALLAVAIL로 설정한 상태에서 메시지가 TOPIC에 발행되고, 구독(지속 또는 비지속)에 발행물을 수신할 수 없는 큐가 있습니다.

- (메시지 지속성에 따라) PMSGDLV 또는 NPMMSGDLV를 ALLDUR로 설정한 상태에서 메시지가 TOPIC에 발행되고, 비지속 구독에 발행물을 수신할 수 없는 큐가 있습니다.

발행 메시지를 올바른 구독자 큐로 전달할 수 없는 경우 USEDLDQ 토픽 속성을 사용하여 데드-레터 큐를 사용할지 여부를 판별할 수 있습니다. USEDLDQ 사용에 대한 자세한 정보는 [DEFINE TOPIC](#)의 내용을 참조하십시오.

- b. 사용 중인 토픽에 대한 구독자가 없으면 발행한 메시지가 큐에 송신되지 않고 제거됩니다. 메시지가 지속적이지 비지속적인지 여부나 무제한 만기인지 만기 시간이 있는지 여부에 상관없이 구독자가 없으면 제거됩니다. 이에 대한 예외로, 메시지가 보유되는 경우에는 메시지가 구독자의 큐로 송신되지 않더라도 새 구독 또는 MQSUBRQ를 사용하여 보유된 발행물을 요청하는 구독자에게 전달할 토픽을 위해 저장됩니다.

## MQPUT 및 MQPUT1

큐에 메시지를 넣기 위해 MQPUT 및 MQPUT1 호출 모두를 사용할 수 있습니다. 사용할 호출은 환경에 따라 다릅니다.

- MQPUT 호출을 사용하여 동일한 큐에 다중 메시지를 배치하십시오.

MQOO\_OUTPUT 옵션을 지정하는 MQOPEN 호출이 먼저 발행되고 하나 이상의 MQPUT 요청이 큐에 메시지를 추가하도록 발행됩니다. 마지막으로 큐가 MQCLOSE 호출로 닫힙니다. 이렇게 하면 MQPUT1 호출을 반복 사용하는 것보다 성능이 높아집니다.

- MQPUT1 호출을 사용하여 큐에 하나의 메시지만 넣으십시오.

이 호출은 MQOPEN, MQPUT 및 MQCLOSE 호출을 단일 호출로 캡슐화하며, 실행되어야 하는 호출의 수를 최소화합니다.

## 목적지 큐

다음은 목적지 큐의 사용에 적용되는 참고입니다.

1. 애플리케이션이 메시지 그룹을 사용하지 않고 동일 큐에 메시지 순서를 넣는 경우, 조건이 상세하고 충족되면 해당 메시지의 순서가 유지됩니다. 일부 조건은 로컬 및 리모트 목적지 큐 모두에 적용됩니다. 기타 조건은 리모트 목적지 큐에만 적용됩니다.


### 로컬 및 리모트 목적지 큐에 적용되는 조건

- 모든 MQPUT 호출이 동일한 작업 단위 내에 있거나 작업 단위 내에 있는 호출이 없습니다.


메시지를 단일 작업 단위 내의 특정 큐에 넣을 때 기타 애플리케이션의 메시지는 큐의 메시지 순서 사이에 배치될 수 있습니다.

- 모든 MQPUT 호출은 동일한 오브젝트 핸들 *Hobj*를 사용하여 작성됩니다.

일부 환경에서 동일한 애플리케이션에서 호출이 작성될 때 다른 오브젝트 핸들이 사용되는 경우에도 메시지 순서가 유지됩니다. 동일한 애플리케이션의 의미는 환경에 의해 판별됩니다.

-  z/OS에서 애플리케이션은 다음과 같습니다.

- CICS의 경우 CICS 태스크
- IMS의 경우 태스크
- z/OS 배치의 경우 태스크

-  IBM i에서는 애플리케이션이 작업입니다.

-  AIX, Linux, and Windows에서 애플리케이션은 스레드입니다.

- 모든 메시지는 우선순위가 같습니다.

- 이러한 메시지는 MQOO\_BIND\_NOT\_FIXED가 지정된(또는 DefBind 큐 속성의 값이 MQBND\_BIND\_NOT\_FIXED인 경우 MQOO\_BIND\_AS\_Q\_DEF가 적용되는) 클러스터 큐에 넣어지지 않습니다.

### 리모트 목적지 큐에 적용되는 추가 조건

- 송신 큐 관리자에서 목적지 큐 관리자로 이동하는 유일한 경로입니다.

순서에서 일부 메시지가 다른 경로로 이동할 수 있는 경우(예를 들어, 메시지 크기를 기반으로 한 재구성, 트래픽 밸런싱 또는 경로 선택으로 인해) 목적지 큐 관리자의 메시지 순서를 보장할 수 없습니다.

- 메시지를 송신, 중간 또는 목적지 큐 관리자의 데드-레터 큐에 일시적으로 넣지 않습니다.

하나 이상의 메시지를 데드-레터 큐에 일시적으로 넣는 경우(예: 전송 큐 또는 목적지 큐가 일시적으로 가득 참) 메시지가 순서대로 목적지 큐에 도착하지 않을 수 있습니다.

- 메시지는 모두 지속적이거나 모두 지속적이지 않습니다.

전송 및 목적지 큐 관리자 사이에서 라우트의 채널에 MQNPMMS\_FAST로 설정된 해당

**NonPersistentMsgSpeed** 속성이 있는 경우 비지속 메시지가 지속 메시지 앞으로 건너뛸 수 있으며 이로 인해 비지속 메시지와 관련된 지속 메시지의 순서가 유지되지 않을 수 있습니다. 그러나, 서로 연관된 지속 메시지의 순서와 서로 연관된 비지속 메시지의 순서는 보존됩니다.

해당 조건이 충족되지 않는 경우 메시지 그룹을 사용하여 메시지 순서를 유지할 수 있지만 메시지 그룹 지원을 사용하기 위해 전송 및 수신 애플리케이션 모두 필요합니다. 메시지 그룹에 대한 자세한 정보는 다음을 참조하십시오.

- [MQMD - MsgFlags 필드](#)
- [MQPMO\\_LOGICAL\\_ORDER](#)
- [MQGMO\\_LOGICAL\\_ORDER](#)

## 분배 목록

다음은 분배 목록의 사용에 적용되는 참고입니다.

분배 목록은 다음 환경에서 지원됩니다.

-  AIX
-  IBM i
-  Linux
-  Windows

및 해당 시스템에 연결된 IBM MQ MQI clients

1. 버전-1 또는 버전-2 MQPMO를 사용하여 분배 목록에 메시지를 넣을 수 있습니다. 버전-1 MQPMO(또는 0과 같은 RecsPresent가 있는 버전-2 MQPMO)를 사용하는 경우 애플리케이션은 넣기 메시지 레코드 또는 응답 레코드를 제공할 수 없습니다. 메시지가 분배 목록에서 일부 큐로 전송되는 경우 오류가 발생하는 큐를 식별할 수 없습니다.

애플리케이션이 넣기 메시지 레코드 또는 응답 레코드를 제공하는 경우 Version 필드를 MQPMO\_VERSION\_2로 설정하십시오.

또한 버전-2 MQPMO를 사용해서 RecsPresent가 0이 아닌지 확인하여 분배 목록에 있지 않은 단일 큐에 메시지를 전송할 수 있습니다.

2. 완료 코드 및 이유 코드 매개변수는 다음과 같이 설정됩니다.

- 분배 목록에 있는 큐에 넣기 작업이 같은 식으로 모두 성공하거나 실패하면, 공통 결과를 설명하는 완료 코드 및 이유 코드 매개변수가 설정됩니다. MQRR 응답 레코드(애플리케이션에서 제공하는 경우)는 이 경우에 설정되지 않습니다.

예를 들어, 모든 넣기가 성공하면 완료 코드 및 이유 코드가 MQCC\_OK 및 MQRC\_NONE으로 설정됩니다. 넣기에 대해 모든 큐가 금지되어 모든 넣기가 실패하는 경우 매개변수가 MQCC\_FAILED 및 MQRC\_PUT\_INHIBITED로 설정됩니다.

- 분배 목록에 있는 큐에 넣기 작업이 모두 같은 식으로 성공하거나 실패하지 않는 경우:
  - 완료 코드 매개변수는 하나 이상의 넣기가 성공한 경우 MQCC\_WARNING으로 설정되고 모두 실패한 경우 MQCC\_FAILED로 설정됩니다.
  - 이유 코드 매개변수는 MQRC\_MULTIPLE\_REASONS로 설정됩니다.

- 응답 레코드(애플리케이션에 의해 제공되는 경우)가 분배 목록 내의 큐에 대해 개별 완료 코드 및 이유 코드로 설정됩니다.

해당 목적지에 대한 열기가 실패하여 목적지에 넣기가 실패하는 경우 응답 레코드의 필드가 MQCC\_FAILED 및 MQRC\_OPEN\_FAILED로 설정됩니다. 해당 목적지는 InvalidDestCount에 포함됩니다.

3. 분배 목록에 있는 목적지가 로컬 큐로 해석되면 메시지를 해당 큐에 (분배 목록 메시지가 아닌) 일반 양식으로 넣습니다. 둘 이상의 목적지가 동일한 로컬 큐로 해석되면 하나의 메시지를 이러한 각 목적지의 큐에 넣습니다.

분배 목록의 목적지가 리모트 큐로 해석되는 경우, 메시지가 적절한 전송 큐에 놓입니다. 여러 목적지가 동일한 전송 큐에 해석되는 경우 해당 목적지가 애플리케이션이 제공한 목적지의 목록에 인접하지 않은 경우에도 해당 목적지가 포함된 단일 분배 목록 메시지를 전송 큐에 배치할 수 있습니다. 그러나 전송 큐가 분배 목록 메시지를 지원하는 경우에만 이를 수행할 수 있습니다(DistLists 참조).

전송 큐가 분배 목록을 지원하지 않으면 일반 양식의 메시지 사본 한 개가 해당 전송 큐를 사용하는 각 목적지의 전송 큐에 놓입니다.

애플리케이션 메시지 데이터가 있는 분배 목록이 전송 큐에 너무 큰 경우 분배 목록 메시지는 더 작은 목적지가 포함된 더 작은 분배 목록 메시지로 각각 분할됩니다. 애플리케이션 메시지 데이터가 큐에만 적합한 경우, 분배 목록 메시지를 사용할 수 없고 큐 관리자가 전송 큐를 사용하는 각 목적지에 대해 일반 양식의 메시지 사본을 한 개 생성합니다.

다른 목적지에 다른 메시지 우선순위 또는 메시지 지속성이 있는 경우(이는 애플리케이션이 MQPRI\_PRIORITY\_AS\_Q\_DEF 또는 MQPER\_PERSISTENCE\_AS\_Q\_DEF를 지정하는 경우 발생할 수 있음) 메시지는 동일한 분배 목록 메시지에 보유되지 않습니다. 대신, 다른 우선순위 및 지속성 값을 수용하는 데 필요한 수만큼 분배 목록 메시지를 큐 관리자에서 생성합니다.

4. 분배 목록에 넣기로 인해 다음이 발생할 수 있습니다.

- 하나의 분배 목록 메시지, 또는
- 다수의 작은 분배 목록 메시지, 또는
- 분배 목록 메시지와 일반 메시지의 혼합, 또는
- 일반 메시지만.

해당 항목 중 발생하는 항목은 다음의 여부에 따라 다릅니다.

- 목록에 있는 목적지가 로컬, 리모트 또는 혼합입니다.
- 목적지의 메시지 우선순위와 메시지 지속성이 같습니다.
- 전송 큐가 분배 목록 메시지를 보유할 수 있습니다.
- 전송 큐의 최대 메시지 길이가 분배 목록 양식의 메시지를 수용할 만큼 충분한 크기입니다.

그러나 위의 사항 중 어느 것이 발생하든 상관없이 결과로 발생하는 각 실제 메시지(즉 넣기로 인해 발생한 각 일반 메시지 또는 분배 목록 메시지)는 다음의 경우 한 개의 메시지로만 계수됩니다.

- 애플리케이션이 작업 단위에서 허용되는 최대 메시지 수를 초과했는지 여부를 확인합니다 (MaxUncommittedMsgs 큐 관리자 속성 참조).
- 트리거 조건을 충족하는지 여부를 확인합니다.
- 큐 용량을 늘리고 큐의 최대 큐 용량을 초과하는지 여부를 확인합니다.

5. 핸들을 올바르게 않게 만드는 큐 정의의 변경사항으로 인해 큐가 개별적으로 열린 경우(예: 해석 경로 변경), 분배 목록 핸들이 올바르게 되지 않습니다. 그러나 분배 목록 핸들이 후속 MQPUT 호출에서 사용될 때 이는 특정 큐의 실패를 유발합니다.

## 헤더

애플리케이션 메시지 데이터의 시작 부분에 하나 이상의 IBM MQ 헤더 구조가 있는 상태에서 메시지를 넣으면 큐 관리자가 헤더 구조에 특정 검사를 수행하여 해당 구조가 올바르게 확인합니다. 큐 관리자가 오류를 감지하면 적절한 이유 코드와 함께 호출이 실패합니다. 수행되는 검사는 존재하는 특정 구조에 따라 다릅니다.



- 검사는 버전-2 이상 MQMD가 MQPUT 또는 MQPUT1 호출에서 사용되는 경우에만 수행됩니다. 메시지 데이터의 시작 시 MQMDE가 있어도 버전-1 MQMD가 사용되는 경우 검사는 수행되지 않습니다.
- 로컬 큐 관리자에 지원되지 않는 구조 및 메시지의 첫 번째 MQDLH를 따르는 구조는 유효성 검증되지 않습니다.
- MQDH 및 MQMDE 구조는 큐 관리자가 완전히 유효성 검증하지 않습니다.
- 다른 구조는 큐 관리자가 부분적으로 유효성 검증합니다(일부 필드 검사).

큐 관리자가 수행하는 일반 검사는 다음을 포함합니다.

- **StrucId** 필드는 유효해야 합니다.
- **Version** 필드는 유효해야 합니다.
- **StrucLength** 필드는 구조의 부분을 구성하는 변수 길이 데이터 및 구조를 포함할 만큼 충분히 큰 값을 지정해야 합니다.
- **CodedCharSetId** 필드는 0이 아니거나 올바르지 않은 음수 값이어야 합니다 (MQCCSI\_DEFAULT, MQCCSI\_EMBEDDED, MQCCSI\_Q\_MGR 및 MQCCSI\_UNDEFINED는 대부분의 IBM MQ 헤더 구조에서 올바르지 않음).
- 호출의 **BufferLength** 매개변수는 구조를 포함할 만큼 충분히 큰 값을 지정해야 합니다(구조는 메시지 끝을 넘어서 확장할 수 없음).

구조에 대한 일반 검사 외에도 다음 조건을 충족해야 합니다.

- PCF 메시지에서 구조 길이의 합계는 MQPUT 또는 MQPUT1 호출에서 **BufferLength** 매개변수가 지정하는 길이와 같아야 합니다. PCF 메시지는 형식 이름이 MQFMT\_ADMIN, MQFMT\_EVENT 또는 MQFMT\_PCF인 메시지입니다.
- IBM MQ 구조를 자를 수 없습니다(잘린 구조가 허용되는 다음 상황에서는 제외).
  - 보고 메시지인 메시지입니다.
  - PCF 메시지.
  - MQDLH 구조를 포함하는 메시지. (첫 번째 MQDLH 뒤의 구조는 자를 수 있지만, MQDLH 앞의 구조는 자를 수 없습니다.)
- IBM MQ 구조는 2개 이상의 세그먼트로 분할되면 안됩니다. 구조 전체가 하나의 세그먼트에 포함되어야 합니다.

## 버퍼

Visual Basic 프로그래밍 언어의 경우 다음 사항이 적용됩니다.

- **Buffer** 매개변수의 크기가 **BufferLength** 매개변수에서 지정한 길이 미만인 경우 이유 코드 MQRC\_BUFFER\_LENGTH\_ERROR와 함께 호출이 실패합니다.
- **Buffer** 매개변수는 유형 **String**으로 선언됩니다. 큐에 배치될 데이터의 유형이 **String**이 아닌 경우 MQPUT 대신 MQPUTAny 호출을 사용하십시오.

MQPUTAny 호출은 임의의 데이터 유형을 큐에 배치할 수 있도록 **Buffer** 매개변수가 유형 Any로 선언되었다는 점을 제외하고는 MQPUT 호출과 동일한 매개변수를 갖습니다. 그러나 이는 BufferLength 바이트 이상의 크기인지 확인하기 위해 Buffer를 확인할 수 없음을 의미합니다.

## C 호출

```
MQPUT (Hconn, Hobj, &MsgDesc, &PutMsgOpts, BufferLength, Buffer,
      &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN  Hconn;          /* Connection handle */
MQHOBJ   Hobj;           /* Object handle */
MQMD     MsgDesc;       /* Message descriptor */
```

```

MQPMO    PutMsgOpts;    /* Options that control the action of MQPUT */
MQLONG   BufferLength;  /* Length of the message in Buffer */
MQBYTE   Buffer[n];     /* Message data */
MQLONG   CompCode;     /* Completion code */
MQLONG   Reason;       /* Reason code qualifying CompCode */

```

## COBOL 호출

```

CALL 'MQPUT' USING HCONN, HOBJ, MSGDESC, PUTMSGOPTS, BUFFERLENGTH,
                BUFFER, COMPCODE, REASON.

```

매개변수를 다음과 같이 선언합니다.

```

** Connection handle
01 HCONN          PIC S9(9) BINARY.
** Object handle
01 HOBJ          PIC S9(9) BINARY.
** Message descriptor
01 MSGDESC.
   COPY CMQMDV.
** Options that control the action of MQPUT
01 PUTMSGOPTS.
   COPY CMQPMOV.
** Length of the message in BUFFER
01 BUFFERLENGTH PIC S9(9) BINARY.
** Message data
01 BUFFER        PIC X(n).
** Completion code
01 COMPCODE      PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON        PIC S9(9) BINARY.

```

## PL/I 호출

```

call MQPUT (Hconn, Hobj, MsgDesc, PutMsgOpts, BufferLength, Buffer,
            CompCode, Reason);

```

매개변수를 다음과 같이 선언합니다.

```

dcl Hconn          fixed bin(31); /* Connection handle */
dcl Hobj           fixed bin(31); /* Object handle */
dcl MsgDesc       like MQMD;     /* Message descriptor */
dcl PutMsgOpts    like MQPMO;    /* Options that control the action of
                                MQPUT */
dcl BufferLength   fixed bin(31); /* Length of the message in Buffer */
dcl Buffer         char(n);       /* Message data */
dcl CompCode      fixed bin(31); /* Completion code */
dcl Reason        fixed bin(31); /* Reason code qualifying CompCode */

```

## 상위 레벨 어셈블러 호출

```

CALL MQPUT, (HCONN,HOBJ,MSGDESC,PUTMSGOPTS,BUFFERLENGTH, X
            BUFFER,COMPCODE,REASON)

```

매개변수를 다음과 같이 선언합니다.

```

HCONN      DS      F      Connection handle
HOBJ       DS      F      Object handle
MSGDESC    CMQMDA   ,      Message descriptor
PUTMSGOPTS CMQPMOA  ,      Options that control the action of MQPUT
BUFFERLENGTH DS     F      Length of the message in BUFFER
BUFFER     DS      CL(n)  Message data
COMPCODE   DS      F      Completion code
REASON     DS      F      Reason code qualifying COMPCODE

```

## Visual Basic 호출

### Windows

```
MQPUT Hconn, Hobj, MsgDesc, PutMsgOpts, BufferLength, Buffer, CompCode, Reason
```

매개변수를 다음과 같이 선언합니다.

```
Dim Hconn As Long 'Connection handle'  
Dim Hobj As Long 'Object handle'  
Dim MsgDesc As MQMD 'Message descriptor'  
Dim PutMsgOpts As MQPMO 'Options that control the action of MQPUT'  
Dim BufferLength As Long 'Length of the message in Buffer'  
Dim Buffer As String 'Message data'  
Dim CompCode As Long 'Completion code'  
Dim Reason As Long 'Reason code qualifying CompCode'
```

## MQPUT1 - 하나의 메시지 넣기

MQPUT1 호출은 큐 또는 분배 목록에서 또는 토픽에 하나의 메시지를 넣습니다.

큐, 분배 목록 또는 토픽을 열 필요가 없습니다.

### 구문


MQPUT1(*Hconn, ObjDesc, MsgDesc, PutMsgOpts, BufferLength, Buffer, CompCode, Reason*)

### 매개변수

#### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *Hconn*의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

 z/OS for CICS 애플리케이션에서 MQCONN 호출을 생략할 수 있으며 *Hconn*에 다음 값이 지정됩니다.

#### MQHC\_DEF\_HCONN

기본 연결 핸들

#### ObjDesc

유형: MQOD - 입출력(I/O)

이는 메시지가 추가되는 큐 또는 메시지가 발행되는 토픽을 식별하는 구조입니다. 세부사항은 [466 페이지의 『MQOD - 오브젝트 디스크립터』](#)을(를) 참조하십시오.

구조가 큐인 경우 사용자에게 출력에 대해 큐를 열 권한이 부여되어야 합니다. 큐는 모델 큐가 아니어야 합니다.

#### MsgDesc

유형: MQMD - 입출력(I/O)

이 구조는 송신하는 메시지의 속성을 설명하고, 넣기 요청이 완료된 후 피드백 정보를 수신합니다. 세부사항은 [412 페이지의 『MQMD - 메시지 디스크립터』](#)을(를) 참조하십시오.

애플리케이션이 버전-1 MQMD를 제공하는 경우 버전-1이 아니라 버전-2 MQMD에 있는 필드에 값을 지정하기 위해 메시지 데이터에 MQMDE 구조를 접두부로 지정할 수 있습니다. MQMD의 Format 필드를 MQFMT\_MD\_EXTENSION으로 설정하여 MQMDE가 있음을 표시하십시오. 자세한 내용은 [458 페이지의 『MQMDE - 메시지 디스크립터 확장』](#)의 내용을 참조하십시오.

올바른 메시지 핸들이 MQGMO 구조의 MsgHandle 필드나 MQPMO 구조의 OriginalMsgHandle 또는 NewMsgHandle 필드에서 제공되는 경우 애플리케이션이 MQMD 구조를 제공할 필요는 없습니다. 해당 필

드 중 하나에서 제공되는 사항이 없는 경우 메시지 핸들과 연관된 디스크립터에서 메시지에 대한 디스크립터를 가져옵니다.

### PutMsgOpts

유형: MQPMO - 입출력(I/O)

자세한 내용은 485 페이지의 『MQPMO - 메시지 넣기 옵션』의 내용을 참조하십시오.

### BufferLength

유형: MQLONG - 입력

Buffer에서 메시지의 길이입니다. 올바른 값은 0이며, 메시지에 애플리케이션 데이터가 없음을 표시합니다. 상한은 다양한 요인에 따라 다릅니다. **BufferLength** 매개변수에 대한 설명은 722 페이지의 『MQPUT - 넣기 메시지』의 내용을 참조하십시오.

### 버퍼

유형: MQBYTExBufferLength - 입력

송신할 애플리케이션 메시지 데이터가 있는 버퍼입니다. 경계에 있는 버퍼를 메시지에 있는 데이터의 네이처에 적절하게 맞추십시오. 4바이트 맞추기는 대부분의 메시지에 적합하지만(IBM MQ 헤더 구조가 포함된 메시지 포함) 일부 메시지에는 더 엄격한 맞추기가 필요할 수 있습니다. 예를 들어, 64비트 2진 정수를 포함하는 메시지에는 8바이트 맞추기가 필요할 수 있습니다.

Buffer에 문자 또는 숫자 데이터가 포함된 경우 **MsgDesc** 매개변수의 **CodedCharSetId** 및 **Encoding** 필드를 데이터에 적합한 값으로 설정하십시오. 이를 통해 메시지 수신자는 필요한 경우 데이터를 수신자가 사용하는 문자 세트 및 인코딩으로 변환할 수 있습니다.

**참고:** MQPUT1 호출의 다른 모든 매개변수는 로컬 큐 관리자의 문자 세트 및 인코딩에 있어야 합니다 (**CodedCharSetId** 큐 관리자 속성 및 MQENC\_NATIVE로 지정됨).

C 프로그래밍 언어에서 매개변수는 pointer-to-void로서 선언됩니다. 임의의 데이터 유형의 주소를 매개변수로 지정할 수 있습니다.

**BufferLength** 매개변수가 0인 경우 Buffer는 참조되지 않습니다. 이 경우 C 또는 시스템/390 어셈블러로 작성된 프로그램이 전달한 매개변수 주소가 널일 수 있습니다.

### CompCode

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_WARNING

경고(일부 완료).

#### MQCC\_FAILED

호출에 실패했습니다.

### 원인

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드입니다.

CompCode이(가) MQCC\_OK인 경우:

#### MQRC\_NONE

(0, X'000') 보고할 이유가 없습니다.

CompCode이(가) MQCC\_WARNING인 경우:

#### MQRC\_MULTIPLE\_REASONS

(2136, X'858') 다중 이유 코드가 리턴되었습니다.

#### MQRC\_INCOMPLETE\_GROUP

(2241, X'8C1') 메시지 그룹이 완료되지 않았습니다.

**MQRC\_INCOMPLETE\_MSG**

(2242, X'8C2') 논리 메시지가 완료되지 않았습니다.

**MQRC\_PRIORITY\_EXCEEDS\_MAXIMUM**

(2049, X'801') 메시지 우선순위가 지원되는 최대 값을 초과합니다.

**MQRC\_UNKNOWN\_REPORT\_OPTION**

(2104, X'838') 메시지 디스크립터의 보고서 옵션이 인식되지 않습니다.

CompCode이(가) MQCC\_FAILED인 경우:

**MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') 어댑터를 사용할 수 없습니다.

**MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

**MQRC\_ALIAS\_BASE\_Q\_TYPE\_ERROR**

(2001, X'7D1') 알리어스 기본 큐가 올바른 유형이 아닙니다.

**MQRC\_API\_EXIT\_ERROR**

(2374, X'946') API 엑시트가 실패했습니다.

**MQRC\_API\_EXIT\_LOAD\_ERROR**

(2183, X'887') API 엑시트를 로드할 수 없습니다.

**MQRC\_ASID\_MISMATCH**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

**MQRC\_BACKED\_OUT**

(2003, X'7D3') 작업 단위가 백아웃되었습니다.

**MQRC\_BUFFER\_ERROR**

(2004, X'7D4') 버퍼 매개변수가 올바르지 않습니다.

**MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'7D5') 버퍼 길이 매개변수가 올바르지 않습니다.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**MQRC\_CF\_NOT\_AVAILABLE**

(2345, X'929') 커플링 기능이 사용 불가능합니다.

**MQRC\_CF\_STRUC\_AUTH\_FAILED**

(2348, X'92C') 커플링 기능 구조 권한 검사에 실패했습니다.

**MQRC\_CF\_STRUC\_ERROR**

(2349, X'92D') 커플링 기능 구조가 올바르지 않습니다.

**MQRC\_CF\_STRUC\_FAILED**

(2373, X'945') 커플링 기능 구조가 실패했습니다.

**MQRC\_CF\_STRUC\_IN\_USE**

(2346, X'92A') 커플링 기능 구조가 사용 중입니다.

**MQRC\_CF\_STRUC\_LIST\_HDR\_IN\_USE**

(2347, X'92B') 커플링 기능 구조 목록 헤더가 사용 중입니다.

**MQRC\_CFGR\_ERROR**

(2416, X'970') 메시지 데이터의 PCF 그룹 매개변수 구조 MQCFGR이 올바르지 않습니다.

**MQRC\_CFH\_ERROR**

(2235, X'8BB') PCF 헤더 구조가 올바르지 않습니다.

**MQRC\_CFIF\_ERROR**

(2414, X'96E') 메시지 데이터의 PCF 정수 필터 매개변수 구조가 올바르지 않습니다.

**MQRC\_CFIL\_ERROR**

(2236, X'8BC') PCF 정수 목록 매개변수 구조 또는 PCIF\*64 정수 목록 매개변수 구조가 올바르지 않습니다.

**MQRC\_CFIN\_ERROR**  
(2237, X'8BD') PCF 정수 매개변수 구조 또는 PCIF\*64 정수 매개변수 구조가 올바르지 않습니다.

**MQRC\_CFSF\_ERROR**  
(2415, X'96F') 메시지 데이터의 PCF 문자열 필터 매개변수 구조가 올바르지 않습니다.

**MQRC\_CFSL\_ERROR**  
(2238, X'8BE') PCF 문자열 목록 매개변수 구조가 올바르지 않습니다.

**MQRC\_CFST\_ERROR**  
(2239, X'8BF') PCF 문자열 매개변수 구조가 올바르지 않습니다.

**MQRC\_CICS\_WAIT\_FAILED**  
(2140, X'85C') CICS에서 대기 요청이 거부되었습니다.

**MQRC\_CLUSTER\_EXIT\_ERROR**  
(2266, X'8DA') 클러스터 워크로드 엑시트가 실패했습니다.

**MQRC\_CLUSTER\_RESOLUTION\_ERROR**  
(2189, X'88D') 클러스터 이름을 해석하지 못했습니다.

**MQRC\_CLUSTER\_RESOURCE\_ERROR**  
(2269, X'8DD') 클러스터 자원에 오류가 있습니다.

**MQRC\_COD\_NOT\_VALID\_FOR\_XCF\_Q**  
(2106, X'83A') XCF 큐에 대해 COD 보고서 옵션이 올바르지 않습니다.

**MQRC\_CONNECTION\_BROKEN**  
(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**MQRC\_CONNECTION\_NOT\_AUTHORIZED**  
(2217, X'8A9') 연결할 권한이 부여되지 않았습니다.

**MQRC\_CONNECTION QUIESCING**  
(2202, X'89A') 연결이 정지됩니다.

**MQRC\_CONNECTION\_STOPPING**  
(2203, X'89B') 연결이 종료됩니다.

**MQRC\_CONTENT\_ERROR**  
2554 (X'09FA) 확장된 메시지 선택자가 있는 구독자에 메시지를 전달할 수 있는지 여부를 판별하기 위해 메시지 콘텐츠를 구문 분석할 수 없습니다.

**MQRC\_CONTEXT\_HANDLE\_ERROR**  
(2097, X'831') 참조되는 큐 핸들이 컨텍스트를 저장하지 않습니다.

**MQRC\_CONTEXT\_NOT\_AVAILABLE**  
(2098, X'832') 참조되는 큐 핸들에 대해 컨텍스트를 사용할 수 없습니다.

**MQRC\_DATA\_LENGTH\_ERROR**  
(2010, X'7DA') 데이터 길이 매개변수가 올바르지 않습니다.

**MQRC\_DB2\_NOT\_AVAILABLE**  
(2342, X'926') Db2 서브시스템이 사용 불가능합니다.

**MQRC\_DEF\_XMIT\_Q\_TYPE\_ERROR**  
(2198, X'896') 기본 전송 큐가 로컬이 아닙니다.

**MQRC\_DEF\_XMIT\_Q\_USAGE\_ERROR**  
(2199, X'897') 기본 전송 큐 사용법 오류입니다.

**MQRC\_DH\_ERROR**  
(2135, X'857') 분배 헤더 구조가 올바르지 않습니다.

**MQRC\_DLH\_ERROR**  
(2141, X'85D') 데드 레터 헤더 구조가 올바르지 않습니다.

**MQRC\_EPH\_ERROR**  
(2420, X'974') 임베드된 PCF 구조가 올바르지 않습니다.

**MQRC\_EXPIRY\_ERROR**  
(2013, X'7DD') 만기 시간이 올바르지 않습니다.

**MQRC\_FEEDBACK\_ERROR**  
(2014, X'7DE') 피드백 코드가 올바르지 않습니다.

**MQRC\_GLOBAL\_UOW\_CONFLICT**  
(2351, X'92F') 글로벌 작업 단위 충돌입니다.

**MQRC\_GROUP\_ID\_ERROR**  
(2258, X'8D2') 그룹 ID가 올바르지 않습니다.

**MQRC\_HANDLE\_IN\_USE\_FOR\_UOW**  
(2353, X'931') 글로벌 작업 단위에 대해 사용 중인 핸들입니다.

**MQRC\_HANDLE\_NOT\_AVAILABLE**  
(2017, X'7E1') 사용 가능한 핸들이 없습니다.

**MQRC\_HCONN\_ERROR**  
(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**MQRC\_HEADER\_ERROR**  
(2142, X'85E') IBM MQ 헤더 구조가 올바르지 않습니다.

**MQRC\_IIH\_ERROR**  
(2148, X'864') IMS 정보 헤더 구조가 올바르지 않습니다.

**MQRC\_LOCAL\_UOW\_CONFLICT**  
(2352, X'930') 글로벌 작업 단위가 로컬 작업 단위와 충돌합니다.

**MQRC\_MD\_ERROR**  
(2026, X'7EA') 메시지 디스크립터가 올바르지 않습니다.

**MQRC\_MDE\_ERROR**  
(2248, X'8C8') 메시지 디스크립터 확장이 올바르지 않습니다.

**MQRC\_MISSING\_REPLY\_TO\_Q**  
(2027, X'7EB') 응답 대상 큐가 누락되었습니다.

**MQRC\_MISSING\_WIH**  
(2332, X'91C') 메시지 데이터가 MQWIH로 시작하지 않습니다.

**MQRC\_MSG\_FLAGS\_ERROR**  
(2249, X'8C9') 메시지 플래그가 올바르지 않습니다.

**MQRC\_MSG\_SEQ\_NUMBER\_ERROR**  
(2250, X'8CA') 메시지 순서 번호가 올바르지 않습니다.

**MQRC\_MSG\_TOO\_BIG\_FOR\_Q**  
(2030, X'7EE') 메시지 길이가 큐의 최대 길이보다 깁니다.

**MQRC\_MSG\_TOO\_BIG\_FOR\_Q\_MGR**  
(2031, X'7EF') 메시지 길이가 큐 관리자의 최대값보다 큼니다.

**MQRC\_MSG\_TYPE\_ERROR**  
(2029, X'7ED') 메시지 디스크립터의 메시지 유형이 올바르지 않습니다.

**MQRC\_MULTIPLE\_REASONS**  
(2136, X'858') 다중 이유 코드가 리턴되었습니다.

**MQRC\_NO\_DESTINATIONS\_AVAILABLE**  
(2270, X'8DE') 목적지 큐가 사용 가능하지 않습니다.

**MQRC\_NOT\_AUTHORIZED**  
(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

**MQRC\_OBJECT\_DAMAGED**  
(2101, X'835') 오브젝트가 손상되었습니다.

**MQRC\_OBJECT\_IN\_USE**  
(2042, X'7FA') 오브젝트가 이미 충돌하는 옵션으로 열렸습니다.

**MQRC\_OBJECT\_LEVEL\_INCOMPATIBLE**  
(2360, X'938') 오브젝트 레벨이 호환 가능하지 않습니다.

**MQRC\_OBJECT\_NAME\_ERROR**  
(2152, X'868') 오브젝트 이름이 올바르지 않습니다.

**MQRC\_OBJECT\_NOT\_UNIQUE**  
(2343, X'927') 오브젝트가 고유하지 않습니다.

**MQRC\_OBJECT\_Q\_MGR\_NAME\_ERROR**  
(2153, X'869') 오브젝트 큐 관리자 이름이 올바르지 않습니다.

**MQRC\_OBJECT\_RECORDS\_ERROR**  
(2155, X'86B') 오브젝트 레코드가 올바르지 않습니다.

**MQRC\_OBJECT\_TYPE\_ERROR**  
(2043, X'7FB') 오브젝트 유형이 유효하지 않습니다.

**MQRC\_OD\_ERROR**  
(2044, X'7FC') 오브젝트 디스크립터 구조가 유효하지 않습니다.

**MQRC\_OFFSET\_ERROR**  
(2251, X'8CB') 메시지 세그먼트 오프셋이 올바르지 않습니다.

**MQRC\_OPTIONS\_ERROR**  
(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.

**MQRC\_ORIGINAL\_LENGTH\_ERROR**  
(2252, X'8CC') 원래 길이가 올바르지 않습니다.

**MQRC\_PAGESET\_ERROR**  
(2193, X'891') 페이지 세트 데이터 세트에 액세스하는 중에 오류가 발생했습니다.

**MQRC\_PAGESET\_FULL**  
(2192, X'890') 외부 스토리지 매체가 가득 찼습니다.

**MQRC\_PCF\_ERROR**  
(2149, X'865') PCF 구조가 올바르지 않습니다.

**MQRC\_PERSISTENCE\_ERROR**  
(2047, X'7FF') 지속이 올바르지 않습니다.

**MQRC\_PERSISTENT\_NOT\_ALLOWED**  
(2048, X'800') 큐가 지속 메시지를 지원하지 않습니다.

**MQRC\_PMO\_ERROR**  
(2173, X'87D') 메시지 넣기 옵션 구조가 올바르지 않습니다.

**MQRC\_PMO\_RECORD\_FLAGS\_ERROR**  
(2158, X'86E') 메시지 넣기 레코드 플래그가 올바르지 않습니다.

**MQRC\_PRIORITY\_ERROR**  
(2050, X'802') 메시지 우선순위가 올바르지 않습니다.

**MQRC\_PUBLICATION\_FAILURE**  
(2502, X'9C6') 발행이 구독자에 전달되지 않았습니다.

**MQRC\_PUT\_INHIBITED**  
(2051, X'803') 큐에 대해 금지된 Put 호출입니다.

**MQRC\_PUT\_MSG\_RECORDS\_ERROR**  
(2159, X'86F') 메시지 넣기 레코드가 올바르지 않습니다.

**MQRC\_Q\_DELETED**  
(2052, X'804') 큐가 삭제되었습니다.

**MQRC\_Q\_FULL**  
(2053, X'805') 큐에 이미 최대 수의 메시지가 포함되어 있습니다.

**MQRC\_Q\_MGR\_NAME\_ERROR**  
(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**  
(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.

**MQRC\_Q\_MGR QUIESCING**  
(2161, X'871') 큐 관리자가 정지됩니다.

**MQRC\_Q\_MGR\_STOPPING**  
(2162, X'872') 큐 관리자가 종료되었습니다.



**MQRC\_Q\_SPACE\_NOT\_AVAILABLE**

(2056, X'808') 큐에 사용할 수 있는 공간이 디스크에 없습니다.

**MQRC\_Q\_TYPE\_ERROR**

(2057, X'809') 큐 유형이 유효하지 않습니다.

**MQRC\_RECS\_PRESENT\_ERROR**

(2154, X'86A') 현재 레코드 수가 유효하지 않습니다.

**MQRC\_REMOTE\_Q\_NAME\_ERROR**

(2184, X'888') 리모트 큐 이름이 유효하지 않습니다.

**MQRC\_REPORT\_OPTIONS\_ERROR**

(2061, X'80D') 메시지 디스크립터의 보고 옵션이 올바르지 않습니다.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**MQRC\_RESPONSE\_RECORDS\_ERROR**

(2156, X'86C') 응답 레코드가 올바르지 않습니다.

**MQRC\_RFH\_ERROR**

(2334, X'91E') MQRFH 또는 MQRFH2 구조가 올바르지 않습니다.

**MQRC\_RMH\_ERROR**

(2220, X'8AC') 참조 메시지 헤더 구조가 올바르지 않습니다.

**MQRC\_SECURITY\_ERROR**

(2063, X'80F') 보안 오류가 발생했습니다.

**MQRC\_SEGMENT\_LENGTH\_ZERO**

(2253, X'8CD') 메시지 세그먼트의 데이터 길이가 0입니다.

**MQRC\_SELECTION\_NOT\_AVAILABLE**

2551 (X'09F7') 발행에 가능한 구독자가 있지만 큐 관리자가 구독자에 발행을 전송할지 여부를 확인할 수 없습니다.

**MQRC\_STOPPED\_BY\_CLUSTER\_EXIT**

(2188, X'88C') 클러스터 워크로드 엑시트에 의해 호출이 거부되었습니다.

**MQRC\_STORAGE\_CLASS\_ERROR**

(2105, X'839') 스토리지 클래스 오류입니다.

**MQRC\_STORAGE\_MEDIUM\_FULL**

(2192, X'890') 외부 스토리지 매체가 가득 찼습니다.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**MQRC\_SUPPRESSED\_BY\_EXIT**

(2109, X'83D') 엑시트 프로그램에서 호출이 억제되었습니다.

**MQRC\_SYNCPOINT\_LIMIT\_REACHED**

(2024, X'7E8') 현재 작업 단위 내에서 더 이상 메시지를 핸들링할 수 없습니다.

**MQRC\_SYNCPOINT\_NOT\_AVAILABLE**

(2072, X'818') 동기점 지원을 사용할 수 없습니다.

**MQRC\_TM\_ERROR**

(2265, X'8D9') 트리거 메시지 구조가 올바르지 않습니다.

**MQRC\_TMC\_ERROR**

(2191, X'88F') 문자 트리거 메시지 구조가 올바르지 않습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

**MQRC\_UNKNOWN\_ALIAS\_BASE\_Q**

(2082, X'822') 알 수 없는 알리아스 기본 큐입니다.

**MQRC\_UNKNOWN\_DEF\_XMIT\_Q**

(2197, X'895') 알 수 없는 기본 전송 큐입니다.

**MQRC\_UNKNOWN\_OBJECT\_NAME**

(2085, X'825') 알 수 없는 오브젝트 이름입니다.

**MQRC\_UNKNOWN\_OBJECT\_Q\_MGR**

(2086, X'826') 알 수 없는 오브젝트 큐 관리자입니다.

**MQRC\_UNKNOWN\_REMOTE\_Q\_MGR**

(2087, X'827') 알 수 없는 리모트 큐 관리자입니다.

**MQRC\_UNKNOWN\_XMIT\_Q**

(2196, X'894') 알 수 없는 전송 큐입니다.

**MQRC\_UOW\_ENLISTMENT\_ERROR**

(2354, X'932') 글로벌 작업 단위에서의 등록이 실패했습니다.

**MQRC\_UOW\_MIX\_NOT\_SUPPORTED**

(2355, X'933') 작업 단위 호출의 혼합은 지원되지 않습니다.

**MQRC\_UOW\_NOT\_AVAILABLE**

(2255, X'8CF') 사용할 큐 관리자에 대해 작업 단위를 사용할 수 없습니다.

**MQRC\_WIH\_ERROR**

(2333, X'91D') MQWIH 구조가 올바르지 않습니다.

**MQRC\_WRONG\_CF\_LEVEL**

(2366, X'93E') 커플링 기능 구조의 레벨이 올바르지 않습니다.

**MQRC\_WRONG\_MD\_VERSION**

(2257, X'8D1') 올바르지 않은 버전의 MQMD가 제공되었습니다.

**MQRC\_XMIT\_Q\_TYPE\_ERROR**

(2091, X'82B') 전송 큐가 로컬이 아닙니다.

**MQRC\_XMIT\_Q\_USAGE\_ERROR**

(2092, X'82C') 사용법이 올바르지 않은 전송 큐입니다.

**MQRC\_XQH\_ERROR**

(2260, X'8D4') 전송 큐 헤더 구조가 올바르지 않습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드를 참조하십시오](#).

## 사용법 참고

1. MQPUT 및 MQPUT1 호출을 사용하여 메시지를 큐에 넣을 수 있습니다. 사용할 호출은 환경에 따라 다릅니다.

- MQPUT 호출을 사용하여 동일한 큐에 다중 메시지를 배치하십시오.

MQOO\_OUTPUT 옵션을 지정하는 MQOPEN 호출이 먼저 발행되고 하나 이상의 MQPUT 요청이 큐에 메시지를 추가하도록 발행됩니다. 마지막으로 큐가 MQCLOSE 호출로 닫힙니다. 이렇게 하면 MQPUT1 호출을 반복 사용하는 것보다 성능이 높아집니다.

- MQPUT1 호출을 사용하여 큐에 하나의 메시지만 넣으십시오.

이 호출은 MQOPEN, MQPUT 및 MQCLOSE 호출을 단일 호출로 캡슐화하며, 실행되어야 하는 호출의 수를 최소화합니다.

2. 애플리케이션이 메시지 그룹을 사용하지 않고 동일 큐에 메시지 순서를 넣는 경우, 특정 조건을 충족하는 경우 이러한 메시지의 순서가 보존됩니다. 그러나, 대부분의 환경에서는 MQPUT1 호출이 이 조건을 충족하지 않으므로 메시지 순서가 보존되지 않습니다. 이 환경에서는 MQPUT 호출을 대신 사용해야 합니다. 세부사항은 [MQPUT 사용법 참고](#)를 참조하십시오.

3. MQPUT1 호출을 사용하여 메시지를 분배 목록에 넣을 수 있습니다. 이에 관한 일반적인 정보는 MQOPEN 및 MQPUT 호출에 대한 사용 시 참고사항을 참조하십시오.

분배 목록은 다음 환경에서 지원됩니다.

-  AIX
-  IBM i
-  Linux

- **Windows** Windows

및 해당 시스템에 연결된 IBM MQ 클라이언트

MQPUT1 호출을 사용하는 경우 다음 차이점이 적용됩니다.

- 애플리케이션이 MQRR 응답 레코드를 제공하는 경우 MQOD 구조를 사용하여 제공해야 하며 MQPMO 구조를 사용하여 제공할 수 없습니다.
  - 이유 코드 MQRC\_OPEN\_FAILED는 응답 레코드에서 MQPUT1로 리턴되지 않습니다. 큐를 여는 데 실패하는 경우 해당 큐의 응답 레코드는 열기 조작에서 발생한 이유 코드를 포함합니다.  
  
큐의 열기 조작이 완료 코드 MQCC\_WARNING과 함께 성공하는 경우 해당 큐의 응답 레코드에서 완료 코드 및 이유 코드가 넣기 조작에서 발행한 완료 및 이유 코드로 대체됩니다.  
  
MQOPEN 및 MQPUT 호출과 같이 큐 관리자는 호출의 결과가 분배 목록의 모든 큐에 대해 동일하지 않은 경우에만 응답 레코드(제공된 경우)를 설정합니다. 이는 이유 코드 MQRC\_MULTIPLE\_REASONS와 함께 완료된 호출로 표시됩니다.
- MQPUT1 호출이 클러스터 큐에서 메시지를 넣는 데 사용되는 경우 호출은 MQOO\_BIND\_NOT\_FIXED가 MQOPEN 호출에서 지정된 것처럼 작동합니다.
  - 애플리케이션 메시지 데이터의 시작 부분에 하나 이상의 IBM MQ 헤더 구조가 있는 상태에서 메시지를 넣으면 큐 관리자가 헤더 구조에 특정 검사를 수행하여 해당 구조가 올바른지 확인합니다. 이에 대한 자세한 정보는 MQPUT 호출에 대한 사용 시 참고사항을 참조하십시오.
  - 경고 상황 중 둘 이상이 발생하는 경우(**CompCode** 매개변수 참조), 리턴되는 이유 코드는 다음 목록의 첫 번째 이유 코드입니다.
    - MQRC\_MULTIPLE\_REASONS
    - MQRC\_INCOMPLETE\_MSG
    - MQRC\_INCOMPLETE\_GROUP
    - MQRC\_PRIORITY\_EXCEEDS\_MAXIMUM 또는 MQRC\_UNKNOWN\_REPORT\_OPTION
  - Visual Basic 프로그래밍 언어의 경우 다음 사항이 적용됩니다.
    - **Buffer** 매개변수의 크기가 **BufferLength** 매개변수에서 지정한 길이 미만인 경우 이유 코드 MQRC\_BUFFER\_LENGTH\_ERROR와 함께 호출이 실패합니다.
    - **Buffer** 매개변수는 유형 String으로 선언됩니다. 큐에 배치될 데이터의 유형이 String이 아닌 경우 MQPUT1 대신 MQPUT1Any 호출을 사용하십시오.  
  
MQPUT1Any 호출은 임의의 데이터 유형을 큐에 배치할 수 있도록 **Buffer** 매개변수가 유형 Any로 선언되었다는 점을 제외하고는 MQPUT1 호출과 동일한 매개변수를 갖습니다. 그러나 이는 BufferLength 바이트 이상의 크기인지 확인하기 위해 Buffer를 확인할 수 없음을 의미합니다.
  - MQPUT1 호출이 MQPMO\_SYNCPOINT로 발행되면 기본 작동이 변경되어 Put 조작이 비동기적으로 완료됩니다. 이 결과, MQOD 및 MQMD 구조의 특정 필드를 필요로 하지만 지금 정의되지 않은 값이 포함되어 있는 일부 애플리케이션의 작동이 변경될 수 있습니다. 애플리케이션은 MQPMO\_SYNC\_RESPONSE를 지정하여 넣기 조작이 동시에 수행되고 적절한 모든 필드 값이 완료되는지 확인할 수 있습니다.

## C 호출

```
MQPUT1 (Hconn, &ObjDesc, &MsgDesc, &PutMsgOpts,
        BufferLength, Buffer, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN  Hconn;           /* Connection handle */
MQOD      ObjDesc;        /* Object descriptor */
MQMD      MsgDesc;       /* Message descriptor */
MQPMO     PutMsgOpts;    /* Options that control the action of MQPUT1 */
MQLONG    BufferLength;   /* Length of the message in Buffer */
MQBYTE    Buffer[n];     /* Message data */
```

```

MQLONG  CompCode;      /* Completion code */
MQLONG  Reason;        /* Reason code qualifying CompCode */

```

## COBOL 호출

```

CALL 'MQPUT1' USING HCONN, OBJDESC, MSGDESC, PUTMSGOPTS,
                  BUFFERLENGTH, BUFFER, COMPCODE, REASON.

```

매개변수를 다음과 같이 선언합니다.

```

** Connection handle
01 HCONN      PIC S9(9) BINARY.
** Object descriptor
01 OBJDESC.
   COPY CMQODV.
** Message descriptor
01 MSGDESC.
   COPY CMQMDV.
** Options that control the action of MQPUT1
01 PUTMSGOPTS.
   COPY CMQPMOV.
** Length of the message in BUFFER
01 BUFFERLENGTH PIC S9(9) BINARY.
** Message data
01 BUFFER      PIC X(n).
** Completion code
01 COMPCODE    PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON     PIC S9(9) BINARY.

```

## PL/I 호출

```

call MQPUT1 (Hconn, ObjDesc, MsgDesc, PutMsgOpts, BufferLength, Buffer,
            CompCode, Reason);

```

매개변수를 다음과 같이 선언합니다.

```

dcl Hconn      fixed bin(31); /* Connection handle */
dcl ObjDesc    like MQOD;    /* Object descriptor */
dcl MsgDesc    like MQMD;    /* Message descriptor */
dcl PutMsgOpts like MQPMO;    /* Options that control the action of
                               MQPUT1 */
dcl BufferLength fixed bin(31); /* Length of the message in Buffer */
dcl Buffer      char(n);      /* Message data */
dcl CompCode   fixed bin(31); /* Completion code */
dcl Reason     fixed bin(31); /* Reason code qualifying CompCode */

```

## 상위 레벨 어셈블러 호출

```

CALL MQPUT1, (HCONN,OBJDESC,MSGDESC,PUTMSGOPTS,BUFFERLENGTH, X
             BUFFER,COMPCODE,REASON)

```

매개변수를 다음과 같이 선언합니다.

HCONN	DS	F	Connection handle
OBJDESC	CMQODA	,	Object descriptor
MSGDESC	CMQMDA	,	Message descriptor
PUTMSGOPTS	CMQPMOA	,	Options that control the action of MQPUT1
BUFFERLENGTH	DS	F	Length of the message in BUFFER
BUFFER	DS	CL(n)	Message data
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## Visual Basic 호출

### Windows

```
MQPUT1 Hconn, ObjDesc, MsgDesc, PutMsgOpts, BufferLength, Buffer,  
CompCode, Reason
```

매개변수를 다음과 같이 선언합니다.

```
Dim Hconn As Long 'Connection handle'  
Dim ObjDesc As MQOD 'Object descriptor'  
Dim MsgDesc As MQMD 'Message descriptor'  
Dim PutMsgOpts As MQPMO 'Options that control the action of MQPUT1'  
Dim BufferLength As Long 'Length of the message in Buffer'  
Dim Buffer As String 'Message data'  
Dim CompCode As Long 'Completion code'  
Dim Reason As Long 'Reason code qualifying CompCode'
```

## MQSET - 오브젝트 속성 설정

MQSET 호출을 사용하여 핸들이 나타내는 오브젝트의 속성을 변경하십시오. 오브젝트는 큐여야 합니다.

### 구문


```
MQSET(Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs, CharAttrLength, CharAttrs, Compcode,  
Reason)
```

### 매개변수

#### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. Hconn의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

 z/OS for CICS 애플리케이션에서 MQCONN 호출을 생략할 수 있으며 Hconn에 다음 값이 지정됩니다.

#### MQHC\_DEF\_HCONN

기본 연결 핸들

#### Hobj

유형: MQHOBJ - 입력

이 핸들은 설정될 속성을 가진 큐 오브젝트를 나타냅니다. 핸들이 MQOO\_SET 옵션을 지정한 이전 MQOPEN 호출에 의해 리턴되었습니다.

#### SelectorCount

유형: MQLONG - 입력

이는 Selectors 배열에서 제공되는 선택자의 수입입니다. 설정되는 속성의 수입입니다. 0은 올바른 값입니다. 허용된 최대 수는 256입니다.

#### Selectors

유형: MQLONGxSelectorCount - 입력

이는 **SelectorCount** 속성 선택자의 배열입니다. 각 선택자는 설정되는 값을 사용하여 속성(정수 또는 문자)을 식별합니다.

각 선택자는 Hobj가 나타내는 큐 유형에 대해 유효해야 합니다. 아래에 나열된 대로 특정 MQIA\_\* 및 MQCA\_\* 값만 허용됩니다.

임의의 순서로 선택자를 지정할 수 있습니다. 정수 속성 선택자(MQIA\_\* 선택자)에 해당하는 속성 값은 Selectors에서 해당 선택자가 발생한 순서와 동일한 순서로 IntAttrs에서 지정되어야 합니다. 문자 속

성 선택자(MQCA\_\* 선택자)에 해당하는 속성 값은 해당 선택자가 발생한 순서와 동일한 순서로 CharAttrs에서 지정되어야 합니다. MQIA\_\* 선택자는 MQCA\_\* 선택자로 인터리브될 수 있습니다. 각 유형 내의 상대 순서만 중요합니다.

동일한 선택자를 두 번 이상 지정할 수 있습니다. 지정하는 경우 특정 선택자에 대해 지정한 마지막 값이 적용됩니다.

**참고:**

1. 정수 및 문자 속성 선택자는 두 가지 다른 범위에서 할당됩니다. MQIA\_\* 선택자는 MQIA\_FIRST에서 MQIA\_LAST 사이의 범위에 상주하며 MQCA\_\* 선택자는 MQCA\_FIRST에서 MQCA\_LAST 사이의 범위에 상주합니다.

각 범위에 대해 상수 MQIA\_LAST\_USED 및 MQCA\_LAST\_USED는 큐 관리자가 승인하는 최고값을 정의합니다.

2. 모든 MQIA\_\* 선택자가 처음 발생하면 동일한 요소 번호를 사용하여 Selectors 및 IntAttrs 배열에서 해당 요소를 처리할 수 있습니다.

3. **SelectorCount** 매개변수가 0인 경우 Selectors가 참조되지 않고 C 또는 System/390 어셈블리어에서 작성된 프로그램이 전달하는 매개변수 주소가 널일 수 있습니다.

설정할 수 있는 속성이 다음 표에 나와 있습니다. 그 외 속성은 이 호출을 사용하여 설정할 수 없습니다.

MQCA\_\* 속성 선택자의 경우 CharAttrs에서 필요한 문자열의 길이(바이트)를 정의하는 상수가 괄호를 사용하여 제공됩니다.

표 554. 큐에 대한 MQSET 속성 선택자		
선택기	설명	참고
MQCA_TRIGGER_DATA	트리거 데이터입니다(MQ_TRIGGER_DATA_LENGTH).	
MQIA_DIST_LISTS	분배 목록 지원.	1
MQIA_INHIBIT_GET	가져오기 조작성 허용되는지 여부.	
MQIA_INHIBIT_PUT	넣기 조작성 허용되는지 여부.	
MQIA_TRIGGER_CONTROL	트리거 제어.	
MQIA_TRIGGER_DEPTH	트리거 용량.	
MQIA_TRIGGER_MSG_PRIORITY	트리거에 대한 메시지 우선순위 임계값입니다.	
MQIA_TRIGGER_TYPE	트리거 유형.	

**참고:**

1. 다음 플랫폼에서만 지원됩니다.

-  AIX
-  IBM i
-  Linux
-  Windows

및 해당 시스템에 연결된 IBM MQ MQI clients

**IntAttrCount**

유형: MQLONG - 입력

이는 IntAttrs 배열의 요소 수이며 최소한 **Selectors** 매개변수 내의 MQIA\_\* 선택자 수여야 합니다. 요소가 없으면 0이 올바른 값입니다.

**IntAttrs**

유형: MQLONGxIntAttrCount - 입력

이는 IntAttrCount 정수 속성 값의 배열입니다. 이러한 속성 값은 Selectors 배열의 MQIA\_\* 선택자와 동일한 순서여야 합니다.

**IntAttrCount** 또는 **SelectorCount** 매개변수가 0인 경우 IntAttrs가 참조되지 않고 C 또는 System/390 어셈블러에서 작성된 프로그램이 전달하는 매개변수 주소가 널일 수 있습니다.

### CharAttrLength

유형: MQLONG - 입력

이는 **CharAttrs** 매개변수의 길이(바이트)이며 최소한 Selectors 배열에서 지정된 문자 속성 길이의 합계여야 합니다. Selectors에서 MQCA\_\* 선택자가 없는 경우 0은 올바른 값입니다.

### CharAttrs

유형: MQCHAR x CharAttrLength - 입력

함께 연결된 문자 속성 값을 포함하는 버퍼입니다. 버퍼의 길이는 **CharAttrLength** 매개변수로 제공됩니다.

문자 속성은 Selectors 배열의 MQCA\_\* 선택자와 동일한 순서로 지정해야 합니다. 각 문자 속성의 길이는 고정됩니다(선택자 참조). 속성에 대해 설정할 값이 속성의 정의된 길이보다 더 적은 비공백 문자를 포함하는 경우, CharAttrs의 값을 오른쪽에 공백으로 채워 속성 값이 속성의 정의된 길이와 일치하도록 하십시오.

**CharAttrLength** 또는 **SelectorCount** 매개변수가 0인 경우 CharAttrs가 참조되지 않고 C 또는 System/390 어셈블러에서 작성된 프로그램이 전달하는 매개변수 주소가 널일 수 있습니다.

### CompCode

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

#### **MQCC\_OK**

정상적으로 완료되었습니다.

#### **MQCC\_FAILED**

호출에 실패했습니다.

### 원인

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드입니다.

CompCode이(가) MQCC\_OK인 경우:

#### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode이(가) MQCC\_FAILED인 경우:

#### **MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') 어댑터를 사용할 수 없습니다.

#### **MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

#### **MQRC\_API\_EXIT\_ERROR**

(2374, X'946') API 엑시트가 실패했습니다.

#### **MQRC\_API\_EXIT\_LOAD\_ERROR**

(2183, X'887') API 엑시트를 로드할 수 없습니다.

#### **MQRC\_ASID\_MISMATCH**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

#### **MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

#### **MQRC\_CF\_NOT\_AVAILABLE**

(2345, X'929') 커플링 기능이 사용 불가능합니다.

**MQRC\_CF\_STRUC\_FAILED**  
(2373, X'945') 커플링 기능 구조가 실패했습니다.

**MQRC\_CF\_STRUC\_IN\_USE**  
(2346, X'92A') 커플링 기능 구조가 사용 중입니다.

**MQRC\_CF\_STRUC\_LIST\_HDR\_IN\_USE**  
(2347, X'92B') 커플링 기능 구조 목록 헤더가 사용 중입니다.

**MQRC\_CHAR\_ATTR\_LENGTH\_ERROR**  
(2006, X'7D6') 문자 속성의 길이가 올바르지 않습니다.

**MQRC\_CHAR\_ATTRS\_ERROR**  
(2007, X'7D7') 문자 속성 문자열이 올바르지 않습니다.

**MQRC\_CICS\_WAIT\_FAILED**  
(2140, X'85C') CICS에서 대기 요청이 거부되었습니다.

**MQRC\_CONNECTION\_BROKEN**  
(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**MQRC\_CONNECTION\_NOT\_AUTHORIZED**  
(2217, X'8A9') 연결할 권한이 부여되지 않았습니다.

**MQRC\_CONNECTION\_STOPPING**  
(2203, X'89B') 연결이 종료됩니다.

**MQRC\_DB2\_NOT\_AVAILABLE**  
(2342, X'926') Db2 서브시스템이 사용 불가능합니다.

**MQRC\_HCONN\_ERROR**  
(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**MQRC\_HOBJ\_ERROR**  
(2019, X'7E3') 오브젝트 핸들이 올바르지 않습니다.

**MQRC\_INHIBIT\_VALUE\_ERROR**  
(2020, X'7E4') Get 금지(inhibit-get) 또는 Put 금지(inhibit-put) 큐 속성의 값이 올바르지 않습니다.

**MQRC\_INT\_ATTR\_COUNT\_ERROR**  
(2021, X'7E5') 정수 속성의 수가 유효하지 않습니다.

**MQRC\_INT\_ATTRS\_ARRAY\_ERROR**  
(2023, X'7E7') 정수 속성 배열이 유효하지 않습니다.

**MQRC\_NOT\_OPEN\_FOR\_SET**  
(2040, X'7F8') 설정을 위해 큐를 열지 못했습니다.

**MQRC\_OBJECT\_CHANGED**  
(2041, X'7F9') 오브젝트가 열린 후에 정의가 변경되었습니다.

**MQRC\_OBJECT\_DAMAGED**  
(2101, X'835') 오브젝트가 손상되었습니다.

**MQRC\_PAGESET\_ERROR**  
(2193, X'891') 페이지 세트 데이터 세트에 액세스하는 중에 오류가 발생했습니다.

**MQRC\_Q\_DELETED**  
(2052, X'804') 큐가 삭제되었습니다.

**MQRC\_Q\_MGR\_NAME\_ERROR**  
(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.

**MQRC\_Q\_MGR\_NOT\_AVAILABLE**  
(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.

**MQRC\_Q\_MGR\_STOPPING**  
(2162, X'872') 큐 관리자가 종료되었습니다.

**MQRC\_RESOURCE\_PROBLEM**  
(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**MQRC\_SELECTOR\_COUNT\_ERROR**  
(2065, X'811') 선택자의 수가 유효하지 않습니다.



**MQRC\_SELECTOR\_ERROR**

(2067, X'813') 속성 선택자가 올바르지 않습니다.

**MQRC\_SELECTOR\_LIMIT\_EXCEEDED**

(2066, X'812') 선택자 수가 너무 큼니다.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**MQRC\_SUPPRESSED\_BY\_EXIT**

(2109, X'83D') 엑시트 프로그램에서 호출이 억제되었습니다.

**MQRC\_TRIGGER\_CONTROL\_ERROR**

(2075, X'81B') trigger-control 속성의 값이 올바르지 않습니다.

**MQRC\_TRIGGER\_DEPTH\_ERROR**

(2076, X'81C') trigger-depth 속성의 값이 올바르지 않습니다.

**MQRC\_TRIGGER\_MSG\_PRIORITY\_ERR**

(2077, X'81D') trigger-message-priority 속성의 값이 올바르지 않습니다.

**MQRC\_TRIGGER\_TYPE\_ERROR**

(2078, X'81E') trigger-type 속성의 값이 올바르지 않습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

## 사용법 참고

1. 이 호출을 사용하여 애플리케이션이 정수 속성 배열, 문자 속성 문자열 컬렉션 또는 둘 다를 지정할 수 있습니다. 오류가 발생하지 않으면 지정된 속성이 모두 동시에 설정됩니다. 오류가 발생하면(예: 선택자가 올바르지 않거나 속성을 올바르지 않은 값으로 설정하려고 함), 호출이 실패하고 속성이 설정되지 않습니다.
2. 속성의 값은 MQINQ 호출을 사용하여 판별할 수 있습니다. 세부사항은 [680 페이지의 『MQINQ - 오브젝트 속성 조회』](#)의 내용을 참조하십시오.

**참고:** MQINQ 호출을 사용하여 조회할 수 있는 값을 가진 모든 속성이 MQSET 호출을 사용하여 값을 변경할 수 있는 것은 아닙니다. 예를 들어, process-object 또는 큐 관리자 속성은 이 호출을 사용하여 설정할 수 없습니다.

3. 속성 변경은 큐 관리자를 재시작해도 보존됩니다. (단, 임시 동적 큐에 대한 변경사항은 큐 관리자를 재시작하면 지속되지 않습니다.)
4. MQSET 호출을 사용하여 모델 큐의 속성을 변경할 수는 없습니다. 그러나, MQOO\_SET 옵션과 MQOPEN 호출을 사용하여 모델 큐를 여는 경우 MQSET 호출을 사용하여 MQOPEN 호출에 의해 작성된 동적 로컬 큐의 속성을 설정할 수 있습니다.
5. 설정되는 오브젝트가 클러스터 큐인 경우 열기가 성공하려면 클러스터 큐의 인스턴스가 있어야 합니다.

오브젝트 속성에 대한 자세한 정보는 다음을 참조하십시오.

- [804 페이지의 『큐의 속성』](#)
- [834 페이지의 『이름 목록에 대한 속성』](#)
- [836 페이지의 『프로세스 정의에 대한 속성』](#)
- [768 페이지의 『큐 관리자의 속성』](#)

## C 호출

```
MQSET (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs, CharAttrLength, CharAttrs, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN Hconn; /* Connection handle */
```

```

MQHOBJ  Hobj;           /* Object handle */
MQLONG  SelectorCount; /* Count of selectors */
MQLONG  Selectors[n];  /* Array of attribute selectors */
MQLONG  IntAttrCount;  /* Count of integer attributes */
MQLONG  IntAttrs[n];   /* Array of integer attributes */
MQLONG  CharAttrLength; /* Length of character attributes buffer */
MQCHAR  CharAttrs[n];  /* Character attributes */
MQLONG  CompCode;      /* Completion code */
MQLONG  Reason;        /* Reason code qualifying CompCode */

```

## COBOL 호출

```

CALL 'MQSET' USING HCONN, HOBJ, SELECTORCOUNT, SELECTORS-TABLE,
                  INTATTRCOUNT, INTATTRS-TABLE, CHARATTRLENGTH,
                  CHARATTRS, COMPCODE, REASON.

```

매개변수를 다음과 같이 선언합니다.

```

** Connection handle
01 HCONN          PIC S9(9) BINARY.
** Object handle
01 HOBJ          PIC S9(9) BINARY.
** Count of selectors
01 SELECTORCOUNT PIC S9(9) BINARY.
** Array of attribute selectors
01 SELECTORS-TABLE.
02 SELECTORS     PIC S9(9) BINARY OCCURS n TIMES.
** Count of integer attributes
01 INTATTRCOUNT PIC S9(9) BINARY.
** Array of integer attributes
01 INTATTRS-TABLE.
02 INTATTRS     PIC S9(9) BINARY OCCURS n TIMES.
** Length of character attributes buffer
01 CHARATTRLENGTH PIC S9(9) BINARY.
** Character attributes
01 CHARATTRS     PIC X(n).
** Completion code
01 COMPCODE      PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON        PIC S9(9) BINARY.

```

## PL/I 호출

```

call MQSET (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount,
            IntAttrs, CharAttrLength, CharAttrs, CompCode, Reason);

```

매개변수를 다음과 같이 선언합니다.

```

dcl Hconn          fixed bin(31); /* Connection handle */
dcl Hobj           fixed bin(31); /* Object handle */
dcl SelectorCount  fixed bin(31); /* Count of selectors */
dcl Selectors(n)   fixed bin(31); /* Array of attribute selectors */
dcl IntAttrCount   fixed bin(31); /* Count of integer attributes */
dcl IntAttrs(n)    fixed bin(31); /* Array of integer attributes */
dcl CharAttrLength fixed bin(31); /* Length of character attributes
buffer */
dcl CharAttrs      char(n);       /* Character attributes */
dcl CompCode       fixed bin(31); /* Completion code */
dcl Reason         fixed bin(31); /* Reason code qualifying
CompCode */

```

## 상위 레벨 어셈블러 호출

```

CALL MQSET, (HCONN, HOBJ, SELECTORCOUNT, SELECTORS, INTATTRCOUNT, X
            INTATTRS, CHARATTRLENGTH, CHARATTRS, COMPCODE, REASON)

```

매개변수를 다음과 같이 선언합니다.

HCONN	DS	F	Connection handle
HOBJ	DS	F	Object handle
SELECTORCOUNT	DS	F	Count of selectors
SELECTORS	DS	(n)F	Array of attribute selectors
INTATTRCOUNT	DS	F	Count of integer attributes
INTATTRS	DS	(n)F	Array of integer attributes
CHARATTRLENGTH	DS	F	Length of character attributes buffer
CHARATTRS	DS	CL(n)	Character attributes
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## Visual Basic 호출

```
MQSET Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs,  
CharAttrLength, CharAttrs, CompCode, Reason
```

매개변수를 다음과 같이 선언합니다.

```
Dim Hconn As Long 'Connection handle'  
Dim Hobj As Long 'Object handle'  
Dim SelectorCount As Long 'Count of selectors'  
Dim Selectors As Long 'Array of attribute selectors'  
Dim IntAttrCount As Long 'Count of integer attributes'  
Dim IntAttrs As Long 'Array of integer attributes'  
Dim CharAttrLength As Long 'Length of character attributes buffer'  
Dim CharAttrs As String 'Character attributes'  
Dim CompCode As Long 'Completion code'  
Dim Reason As Long 'Reason code qualifying CompCode'
```

## MQSETMP - 메시지 특성 설정

MQSETMP 호출을 사용하여 메시지 핸들의 특성을 설정하거나 수정하십시오.

### 구문

```
MQSETMP(Hconn, Hmsg, SetPropOpts, Name, PropDesc, Type, ValueLength, Value, Compcode, Reason)
```

### 매개변수

#### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다.

값이 **Hmsg** 매개변수에서 지정된 메시지 핸들을 작성하는 데 사용된 연결 핸들과 일치해야 합니다. 메시지 핸들이 MQHC\_UNASSOCIATED\_HCONN을 사용하여 작성된 경우 메시지 핸들의 특성을 설정하는 스레드에서 올바른 연결을 설정해야 합니다. 그렇지 않으면 호출이 이유 코드 MQRC\_CONNECTION\_BROKEN과 함께 실패합니다.

#### Hmsg

유형: MQHMSG - 입력

수정할 메시지 핸들입니다. 값은 이전 MQCRTMH 호출에서 리턴합니다.

#### SetPropOpts

유형: MQSMPO - 입력

메시지 특성의 설정 방법을 제어합니다.

이 구조에서는 애플리케이션이 메시지 특성의 설정 방법을 제어하는 옵션을 지정할 수 있습니다. 구조는 MQSETMP 호출의 입력 매개변수입니다. 자세한 정보는 [MQSMPO](#)의 내용을 참조하십시오.

## 이름

유형: MQCHARV - 입력

설정할 특성의 이름입니다.

특성 이름 사용에 대한 자세한 정보는 [특성 이름](#) 및 [특성 이름 제한사항](#)을 참조하십시오.

## PropDesc

유형: MQPD - 입출력(I/O)

이 구조는 다음과 같은 특성의 속성을 정의하는 데 사용됩니다.

- 특성이 지원되는 않으면 일어나는 현상
- 특성이 속한 메시지 컨텍스트
- 이동 시 특성이 복사되는 메시지

이 구조에 대한 자세한 정보는 [MQPD](#)를 참조하십시오.

## 유형

유형: MQLONG - 입력

설정하는 특성의 데이터 유형. 다음 중 하나가 될 수 있습니다.

### **MQTYPE\_BOOLEAN**

부울입니다. *ValueLength*는 4여야 합니다.

### **MQTYPE\_BYTE\_STRING**

바이트 문자열. *ValueLength*는 0 이상이어야 합니다.

### **MQTYPE\_INT8**

8비트 서명 정수입니다. *ValueLength*는 1이어야 합니다.

### **MQTYPE\_INT16**

16비트 서명 정수입니다. *ValueLength*는 2이어야 합니다.

### **MQTYPE\_INT32**

32비트 서명 정수입니다. *ValueLength*는 4여야 합니다.

### **MQTYPE\_INT64**

64비트 서명 정수입니다. *ValueLength*는 8이어야 합니다.

### **MQTYPE\_FLOAT32**

32비트 부동 소수점 숫자입니다. *ValueLength*는 4여야 합니다.

참고: 이 유형은 z/OS용 IBM COBOL을 사용하는 애플리케이션에서 지원되지 않습니다.

### **MQTYPE\_FLOAT64**

64비트 부동 소수점 숫자입니다. *ValueLength*는 8이어야 합니다.

참고: 이 유형은 z/OS용 IBM COBOL을 사용하는 애플리케이션에서 지원되지 않습니다.

### **MQTYPE\_STRING**

문자열. *ValueLength*는 0 이상이거나 특수 값 MQVL\_NULL\_TERMINATED이어야 합니다.

### **MQTYPE\_NULL**

특성이 존재하지만 값이 널입니다. *ValueLength*는 0이어야 합니다.

## ValueLength

유형: MQLONG - 입력

*Value* 매개변수에서 특성 값의 길이(바이트)입니다. 널 값이나 문자열 또는 바이트 문자열에 대해서만 0이 유효합니다. 0은 특성이 존재하지만 값에 문자 또는 바이트가 없음을 표시합니다.

*Type* 매개변수에 MQTYPE\_STRING 세트가 있는 경우 값은 0 이상이거나 다음 특수 값이어야 합니다.

### **MQVL\_NULL\_TERMINATED**

값이 문자열에 나타난 첫 번째 널로 구분됩니다. 널은 문자열의 일부로 포함되지 않습니다.

MQTYPE\_STRING도 설정되지 않는 경우 이 값은 올바르지 않습니다.

참고: MQVL\_NULL\_TERMINATED가 설정된 경우 문자열을 종료하는 데 사용되는 널 문자는 값의 문자 세트의 널입니다.

## 값

유형: MQBYTEValueLength - 입력

설정할 특성의 값. 버퍼는 값 데이터의 네이처에 적절한 경계에 맞춰야 합니다.

C 프로그래밍 언어에서 매개변수는 pointer-to-void로서 선언됩니다. 임의의 데이터 유형의 주소를 매개변수로 지정할 수 있습니다.

ValueLength가 0이면 Value가 참조되지 않습니다. 이 경우에 C 또는 System/390 어셈블러로 작성된 프로그램이 전달하는 매개변수 주소는 널(null)일 수 있습니다.

## CompCode

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

### **MQCC\_OK**

정상적으로 완료되었습니다.

### **MQCC\_FAILED**

호출에 실패했습니다.

## 원인

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드입니다.

CompCode가 MQCC\_OK인 경우:

### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_WARNING인 경우:

### **MQRC\_RFH\_FORMAT\_ERROR**

(2421, X'0975') 특성을 포함하는 MQRFH2 폴더를 구문 분석할 수 없습니다.

CompCode가 MQCC\_FAILED인 경우:

### **MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'089C') 어댑터를 사용할 수 없습니다.

### **MQRC\_ADAPTER\_SERV\_LOAD\_ERROR**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

### **MQRC\_ASID\_MISMATCH**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

### **MQRC\_BUFFER\_ERROR**

(2004, X'07D4') 값 매개변수가 올바르지 않습니다.

### **MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'07D5') 값 길이 매개변수가 올바르지 않습니다.

### **MQRC\_CALL\_IN\_PROGRESS**

(2219, X'08AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

### **MQRC\_HMSG\_ERROR**

(2460, X'099C') 메시지 핸들 포인터가 올바르지 않습니다.

### **MQRC\_MSG\_HANDLE\_IN\_USE**

(2499, X'09C3') 메시지 핸들이 이미 사용 중입니다.

### **MQRC\_OPTIONS\_ERROR**

(2046, X'07FE') 옵션이 올바르지 않거나 일치하지 않습니다.

### **MQRC\_PD\_ERROR**

(2482, X'09B2') 특성 디스크립터 구조가 올바르지 않습니다.

### **MQRC\_PROPERTY\_NAME\_ERROR**

(2442, X'098A') 올바르지 않은 특성 이름입니다.

**MQRC\_PROPERTY\_TYPE\_ERROR**

(2473, X'09A9') 특성 데이터 유형이 올바르지 않습니다.

**MQRC\_PROP\_NUMBER\_FORMAT\_ERROR**

(2472, X'09A8') 숫자 형식 오류가 값 데이터에서 발견되었습니다.

**MQRC\_SMPO\_ERROR**

(2463, X'099F') 메시지 특성 설정 옵션 구조가 올바르지 않습니다.

**MQRC\_SOURCE\_CCSID\_ERROR**

(2111, X'083F') 특성 이름 코드화 문자 세트 ID가 올바르지 않습니다.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드를 참조하십시오](#).

## C 호출

```
MQSETMP (Hconn, Hmsg, &SetPropOpts, &Name, &PropDesc, Type,
ValueLength, &Value, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN  Hconn;          /* Connection handle */
MQHMSG   Hmsg;          /* Message handle */
MQSMPO   SetPropOpts;   /* Options that control the action of MQSETMP */
MQCHARV  Name;         /* Property name */
MQPD     PropDesc;     /* Property descriptor */
MQLONG   Type;         /* Property data type */
MQLONG   ValueLength;  /* Length of property value in Value */
MQBYTE   Value[n];    /* Property value */
MQLONG   CompCode;    /* Completion code */
MQLONG   Reason;      /* Reason code qualifying CompCode */
```

## COBOL 호출

```
CALL 'MQSETMP' USING HCONN, HMSG, SETMSGOPTS, NAME, PROPDESC, TYPE,
VALUELENGTH, VALUE, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```
** Connection handle
01 HCONN    PIC S9(9) BINARY.
** Message handle
01 HMSG     PIC S9(18) BINARY.
** Options that control the action of MQSETMP
01 SETMSGOPTS
   COPY CMQSMPOV.
** Property name
01 NAME
   COPY CMQCHRNV.
** Property descriptor
01 PROPDESC
   COPY CMQPDV.
** Property data type
01 TYPE     PIC S9(9) BINARY.
** Length of property value in VALUE
01 VALUELENGTH PIC S9(9) BINARY.
** Property value
01 VALUE    PIC X(n).
** Completion code
01 COMPCODE PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON   PIC S9(9) BINARY.
```

## PL/I 호출

```
call MQSETMP (Hconn, Hmsg, SetPropOpts, Name, PropDesc, Type, ValueLength,  
             Value, CompCode, Reason);
```

매개변수를 다음과 같이 선언합니다.

```
dcl Hconn      fixed bin(31); /* Connection handle */  
dcl Hmsg      fixed bin(63); /* Message handle */  
dcl SetPropOpts like MQSMP0; /* Options that control the action of MQSETMP */  
dcl Name      like MQCHARV; /* Property name */  
dcl PropDesc  like MQPD; /* Property descriptor */  
dcl Type      fixed bin(31); /* Property data type */  
dcl ValueLength fixed bin(31); /* Length of property value in Value */  
dcl Value     char(n); /* Property value */  
dcl CompCode  fixed bin(31); /* Completion code */  
dcl Reason    fixed bin(31); /* Reason code qualifying CompCode */
```

## 상위 레벨 어셈블러 호출

```
CALL MQSETMP, (HCONN,HMSG,SETMSGHOPTS,NAME,PROPDSC,TYPE,VALUELENGTH,  
             VALUE,COMPCODE,REASON)
```

매개변수를 다음과 같이 선언합니다.

HCONN	DS	F	Connection handle
HMSG	DS	D	Message handle
SETMSGOPTS	CMQSMP0A	,	Options that control the action of MQSETMP
NAME	CMQCHRVA	,	Property name
PROPDSC	CMQPDA	,	Property descriptor
TYPE	DS	F	Property data type
VALUELENGTH	DS	F	Length of property value in VALUE
VALUE	DS	CL(n)	Property value
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## MQSTAT - 상태 정보 검색

상태 정보를 검색하려면 MQSTAT 호출을 사용하십시오. 리턴된 상태 정보 유형은 호출에서 지정된 유형 값으로 판별됩니다.

### 구문

MQSTAT(*Hconn, Type, Stat, Compcode, Reason*)

### 매개변수

#### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *Hconn*의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

z/OS for CICS 애플리케이션에서 MQCONN 호출을 생략할 수 있으며 *Hconn*에 다음 값이 지정됩니다.

#### MQHC\_DEF\_HCONN

기본 연결 핸들

#### 유형

유형: MQLONG - 입력

요청되는 상태 정보의 유형. > 올바른 값은 다음과 같습니다.

**MQSTAT\_TYPE\_ASYNC\_ERROR**

이전의 비동기 넣기 조작에 대한 정보를 리턴합니다.

**MQSTAT\_TYPE\_RECONNECTION**

연결에 대한 정보를 리턴합니다. 연결이 다시 연결 중이거나 다시 연결에 실패한 경우 정보는 연결이 다시 연결을 시작하게 한 실패를 설명합니다.

이 값은 클라이언트 연결에만 유효합니다. 다른 유형의 연결에서 이 호출은 실패하고 이유 코드는 **MQRC\_ENVIRONMENT\_ERROR**입니다.

**MQSTAT\_TYPE\_RECONNECTION\_ERROR**

다시 연결과 관련된 이전 실패에 대한 정보를 리턴합니다. 연결이 다시 연결에 실패한 경우 정보는 다시 연결을 실패하게 한 실패를 설명합니다.

이 값은 클라이언트 연결에만 유효합니다. 기타 연결 유형의 경우 이유 코드 **MQRC\_ENVIRONMENT\_ERROR**와 함께 호출이 실패합니다.

**Stat**

유형: MQSTS - 입출력(I/O)

상태 정보 구조. 자세한 내용은 [571 페이지의 『MQSTS - 상태 보고 구조』](#)의 내용을 참조하십시오.

**CompCode**

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_FAILED**

호출에 실패했습니다.

**원인**

유형: MQLONG - 출력

*CompCode*를 규정하는 이유 코드입니다.

*CompCode*이(가) MQCC\_OK인 경우:

**MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_FAILED인 경우:

**MQRC\_API\_EXIT\_ERROR**

(2374, X'946') API 엑시트가 실패했습니다.

**MQRC\_API\_EXIT\_LOAD\_ERROR**

(2183, X'887') API 엑시트를 로드할 수 없습니다.

**MQRC\_CALL\_IN\_PROGRESS**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**MQRC\_CONNECTION\_BROKEN**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**MQRC\_CONNECTION\_STOPPING**

(2203, X'89B') 연결이 종료됩니다.

**MQRC\_FUNCTION\_NOT\_SUPPORTED**

(2298, X'8FA') 요청된 함수는 현재 환경에서 사용할 수 없습니다.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**MQRC\_Q\_MGR\_STOPPING**

(2162, X'872') - 큐 관리자가 중지 중입니다.



**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**MQRC\_STAT\_TYPE\_ERROR**

(2430, X'97E') MQSTAT 유형에 대한 오류입니다.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**MQRC\_STS\_ERROR**

(2426, X'97A') MQSTS 구조에 대한 오류입니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

**사용법 참고**

1. MQSTAT\_TYPE\_ASYNC\_ERROR 유형을 지정하는 MQSTAT에 대한 호출은 이전 비동기 MQPUT 및 MQPUT1 조작에 대한 정보를 리턴합니다. MQSTAT 호출에서 리턴 시 다시 전달된 MQSTS 구조는 해당 연결에 대해 처음으로 기록된 비동기 경고 또는 오류 정보를 포함합니다. 이후 추가 오류 또는 경고가 이어지면 일반적으로 이 값을 대체하지 않습니다. 그러나 MQCC\_WARNING의 완료 코드와 함께 오류가 발생하는 경우 MQCC\_FAILED의 완료 코드와 함께 후속 실패가 대신 리턴됩니다.
2. 연결이 설정된 이후 또는 MQSTAT에 대한 마지막 호출 이후 오류가 발생하지 않은 경우 MQCC\_OK의 CompCode 및 MQRC\_NONE의 이유가 MQSTS 구조에서 리턴됩니다.
3. 연결 핸들 아래에서 처리된 비동기 호출 수의 개수는 PutSuccessCount, PutWarningCount, PutFailureCount의 세 카운터 필드로 리턴됩니다. 이러한 카운터는 비동기 조작이 성공적으로 처리되거나 경고가 있거나 실패할 때마다 큐 관리자에 의해 증분됩니다. 회계 목적으로 분배 목록에 넣기는 분배 목록당 한 번이 아닌 목적지 큐당 한 번으로 계수됨을 참고하십시오. 카운터는 최대 양의 값 AMQ\_LONG\_MAX를 넘어서 증분되지 않습니다.
4. MQSTAT에 대한 호출이 성공하면 이전 오류 정보 또는 수가 재설정됩니다.
5. MQSTAT의 작동은 사용자가 제공하는 **MQSTAT Type** 매개변수의 값에 따라 다릅니다.
6. **MQSTAT\_TYPE\_ASYNC\_ERROR**
  - a. MQSTAT\_TYPE\_ASYNC\_ERROR 유형을 지정하는 MQSTAT에 대한 호출은 이전 비동기 MQPUT 및 MQPUT1 조작에 대한 정보를 리턴합니다. MQSTAT 호출에서 리턴 시 다시 전달된 MQSTS 구조는 해당 연결에 대해 처음으로 기록된 비동기 경고 또는 오류 정보를 포함합니다. 이후 추가 오류 또는 경고가 이어지면 일반적으로 이 값을 대체하지 않습니다. 그러나 MQCC\_WARNING의 완료 코드와 함께 오류가 발생하는 경우 MQCC\_FAILED의 완료 코드와 함께 후속 실패가 대신 리턴됩니다.
  - b. 연결이 설정된 이후 또는 MQSTAT에 대한 마지막 호출 이후 오류가 발생하지 않은 경우 MQCC\_OK의 CompCode 및 MQRC\_NONE의 이유가 MQSTS 구조에서 리턴됩니다.
  - c. 연결 핸들 아래에서 처리된 비동기 호출 수의 개수는 PutSuccessCount, PutWarningCount, PutFailureCount의 세 카운터 필드로 리턴됩니다. 이러한 카운터는 비동기 조작이 성공적으로 처리되거나 경고가 있거나 실패할 때마다 큐 관리자에 의해 증분됩니다. 회계 목적으로 분배 목록에 넣기는 분배 목록당 한 번이 아닌 목적지 큐당 한 번으로 계수됨을 참고하십시오. 카운터는 최대 양의 값 AMQ\_LONG\_MAX를 넘어서 증분되지 않습니다.
  - d. MQSTAT에 대한 호출이 성공하면 이전 오류 정보 또는 수가 재설정됩니다.

**MQSTAT\_TYPE\_RECONNECTION**

다시 연결 동안 이벤트 핸들러 내부에서 Type을 MQSTAT\_TYPE\_RECONNECTION으로 설정한 MQSTAT를 호출한다고 가정합니다. 이러한 예를 고려하십시오.

**클라이언트가 다시 연결을 시도 중이거나 다시 연결에 실패했습니다.**

MQSTS 구조의 CompCode는 MQCC\_FAILED이며 Reason은 MQRC\_CONNECTION\_BROKEN 또는 MQRC\_Q\_MGR QUIESCING일 수 있습니다. ObjectType은 MQOT\_Q\_MGR이고 ObjectName은 큐 관리자의 이름이며 ObjectQMgrName은 공백입니다.

클라이언트가 다시 연결을 완료했거나 연결이 끊어지지 않았습니다.

MQSTS 구조의 CompCode는 MQCC\_OK이고 Reason은 MQRC\_NONE입니다.

MQSTAT에 대한 후속 호출이 동일한 결과를 리턴합니다.

### MQSTAT\_TYPE\_RECONNECTION\_ERROR

MQI 호출에 대해 MQRC\_RECONNECT\_FAILED를 수신하는 것에 응답하여 Type을 MQSTAT\_TYPE\_RECONNECTION\_ERROR로 설정한 MQSTAT를 호출한다고 가정합니다. 이러한 예를 고려하십시오.

다른 큐 관리자에 다시 연결하는 중에 큐를 다시 열 때 권한 부여 실패가 발생했습니다.

MQSTS 구조의 CompCode는 MQCC\_FAILED이며 Reason은 다시 연결이 실패한 이유입니다(예: MQRC\_NOT\_AUTHORIZED). ObjectType은 문제점을 발생시킨 오브젝트의 유형이고(예: MQOT\_QUEUE) ObjectName은 큐의 이름이며 ObjectQMgrName은 큐를 소유한 큐 관리자의 이름입니다.

다시 연결 중에 소켓 연결 오류가 발생했습니다.

MQSTS 구조의 CompCode는 MQCC\_FAILED이며 Reason은 다시 연결이 실패한 이유입니다(예: MQRC\_HOST\_NOT\_AVAILABLE). ObjectType은 MQOT\_Q\_MGR이고 ObjectName은 큐 관리자의 이름이며 ObjectQMgrName은 공백입니다.

MQSTAT에 대한 후속 호출이 동일한 결과를 리턴합니다.

## C 호출

```
MQSTAT (Hconn, StatType, &Stat, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN Hconn;          /* Connection Handle */
MQLONG StatType;        /* Status type */
MQSTS Stat;             /* Status information structure */
MQLONG CompCode;        /* Completion code */
MQLONG Reason;          /* Reason code qualifying CompCode */
```

## COBOL 호출

```
CALL 'MQSTAT' USING HCONN, STATTYPE, STAT, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```
**      Connection handle
01      HCONN      PIC S9(9)      BINARY.
**      Status type
01      STATTYPE  PIC S9(9)      BINARY.
**      Status information
01      STAT.
      COPY CMQSTSV.
**      Completion code
01      COMPCODE  PIC S9(9)      BINARY.
**      Reason code qualifying COMPCODE
01      REASON    PIC S9(9)      BINARY.
```

## PL/I 호출

```
call MQSTAT (Hconn, StatType, Stat, Compcode, Reason);
```

매개변수를 다음과 같이 선언합니다.

```

dcl Hconn      fixed bin(31); /* Connection handle */
dcl StatType   fixed bin(31); /* Status type */
dcl Stat       like MQSTS;    /* Status information structure */
dcl CompCode   fixed bin(31); /* Completion code */
dcl Reason     fixed bin(31); /* Reason code qualifying CompCode */

```

## System/390 Assembler 호출

```
CALL MQSTAT, (HCONN, STATTYPE, STAT, COMPCODE, REASON)
```

매개변수를 다음과 같이 선언합니다.

HCONN	DS	F	Connection handle
STATTYPE	DS	F	Status type
STAT	CMQSTSA,		Status information structure
COMPCODE	DS	F	Completion code
REASON	DS	F	Reason code qualifying COMPCODE

## MQSUB - 구독 등록

특정 토픽에 대한 애플리케이션 구독을 등록하려면 MQSUB 호출을 사용하십시오.

### 구문

```
MQSUB (Hconn, SubDesc, Hobj, Hsub, Compcode, Reason)
```

### 매개변수

#### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *Hconn*의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

z/OS for CICS 애플리케이션에서 MQCONN 호출을 생략할 수 있으며 *Hconn*에 다음 값이 지정됩니다.

#### MQHC\_DEF\_HCONN

기본 연결 핸들

#### SubDesc

유형: MQSD - 입력/출력

이는 애플리케이션에서 등록하고 있는, 사용 중인 오브젝트를 식별하는 구조입니다. 자세한 정보는 [549 페이지의 『MQSD - 구독 디스크립터』](#)의 내용을 참조하십시오.

#### Hobj

유형: MQHOBJ - 입출력(I/O)

이 핸들은 이 구독으로 송신된 메시지를 가져오기 위해 설정된 액세스 권한을 나타냅니다. 이러한 메시지는 특정 큐에 저장하거나, 큐 관리자가 특정 큐를 사용하지 않고 자신의 스토리지를 관리할 수 있습니다.

특정 큐를 사용하려면 구독이 작성될 때 해당 큐를 구독과 연관시켜야 합니다. 이는 두 가지 방법으로 수행할 수 있습니다.

- DEFINE SUB MQSC 명령을 사용하고 해당 명령에 큐 오브젝트의 이름을 제공합니다.
- MQSO\_CREATE를 사용한 MQSUB를 호출할 때 이 핸들을 제공합니다.

이 핸들이 호출의 입력 매개변수로 제공되는 경우, 이는 다음 옵션 중 하나 이상을 사용하여 큐의 이전 MQOPEN 호출로부터 리턴된 올바른 오브젝트 핸들이어야 합니다.

- MQOO\_INPUT\_\*
- MQOO\_BROWSE

- MQOO\_OUTPUT(큐가 리모트 큐인 경우)

그렇지 않은 경우에는 MQRC\_HOBJ\_ERROR가 발생하며 호출이 실패합니다. 이는 토픽 오브젝트로 해석되는 알리어스 큐에 대한 오브젝트 핸들일 수 없습니다. 그러한 경우에는 MQRC\_HOBJ\_ERROR가 발생하며 호출이 실패합니다.

큐 관리자가 이 구독에 전송된 메시지의 스토리지를 관리하는 경우, 이는 구독을 작성할 때 MQSO\_MANAGED 옵션을 사용하여 설정해야 합니다. 그러면 큐 관리자가 호출의 출력 매개변수로서 이 핸들을 리턴합니다. 리턴된 이 핸들을 관리 핸들이라고 합니다. MQHO\_NONE이 지정되었으나 MQSO\_MANAGED가 지정되지 않은 경우에는 MQRC\_HOBJ\_ERROR가 발생하며 호출이 실패합니다.

큐 관리자가 관리 핸들을 리턴한 경우에는 이를 찾아보기 옵션이 있거나 없는 MQGET 또는 MQCB 호출에서 사용하거나, MQINQ 호출에서 사용하거나, MQCLOSE에서 사용할 수 있습니다. 이를 MQPUT, MQSUB, MQSET에서는 사용할 수 없으며, 사용하려 시도하면 MQRC\_NOT\_OPEN\_FOR\_OUTPUT, MQRC\_HOBJ\_ERROR 또는 MQRC\_NOT\_OPEN\_FOR\_SET이 발생하며 실패합니다.

이 구독이 MQSD 구조의 MQSO\_RESUME 옵션을 사용하여 재개되는 경우에는 MQSO\_MANAGED를 MQHO\_NONE으로 설정하여 이 매개변수에서 이 핸들을 애플리케이션에 리턴할 수 있습니다. 이는 구독의 관리 핸들 사용 여부와 관계없이 수행할 수 있으며 핸들을 포함하는 DEFINE SUB를 사용하여 작성된 구독을 해당 명령에 정의된 구독 큐에 제공하는 데 유용합니다. 관리상으로 작성된 구독이 재개되는 경우, 큐는 MQOO\_INPUT\_AS\_Q\_DEF 및 MQOO\_BROWSE로 열립니다. 다른 옵션을 지정해야 하는 경우에는 애플리케이션이 명시적으로 구독 큐를 열고 오브젝트 핸들을 호출에 제공해야 합니다. 큐는 여는 데 문제점이 있는 경우에는 MQRC\_INVALID\_DESTINATION이 발생하며 호출이 실패합니다. *Hobj*가 제공되는 경우 이는 원래 MQSUB 호출의 *Hobj*에 해당해야 합니다. 이는 MQOPEN 호출에서 리턴된 오브젝트 핸들이 제공되는 경우, 이 핸들은 이전에 사용된 것과 동일한 큐여야 함을 의미합니다. 동일한 큐가 아닌 경우에는 MQRC\_HOBJ\_ERROR가 발생하며 호출이 실패합니다.

이 구독이 MQSD 구조의 MQSO\_ALTER 옵션을 사용하여 대체되는 경우에는 다른 *Hobj*를 제공할 수 있습니다. 큐에 전달되었으며 이 매개변수를 통해 이전에 식별된 모든 발행물은 해당 큐에서 유지되며, **Hobj** 매개변수가 이제 다른 큐를 나타내는 경우 이러한 메시지를 검색하는 것은 애플리케이션의 책임입니다.

표 555. <i>Hobj</i> 를 다양한 구독 옵션과 함께 사용		
옵션	<i>Hobj</i>	설명
MQSO_CREATE + MQSO_MANAGED	입력에서 무시됨	큐 관리자가 관리하는 메시지 스토리지를 사용하는 구독을 작성합니다.
MQSO_CREATE	올바른 오브젝트 핸들	특정 큐를 메시지 목적지로 제공하는 구독을 작성합니다.
MQSO_RESUME	MQHO_NONE	이전에 작성된 구독을 관리 여부에 상관없이, 그리고 큐 관리자가 애플리케이션이 사용할 오브젝트 핸들을 리턴했는지에 상관없이 재개합니다.
MQSO_RESUME	올바르며 일치하는 오브젝트 핸들	특정 큐를 메시지의 목적지로 사용하며 특정 열기 옵션이 있는 오브젝트 핸들을 사용하는, 이전에 작성된 구독을 재개합니다.
MQSO_ALTER + MQSO_MANAGED	MQHO_NONE	이전에 특정 큐를 사용한 기존 구독을 관리 구독으로 대체합니다. 목적지의 클래스(관리 또는 비관리)는 변경할 수 없습니다.
MQSO_ALTER	올바른 오브젝트 핸들	관리 또는 비관리 기존 구독을 특정 큐를 사용하도록 대체합니다. MQSO_MANAGED 옵션이 사용되지 않은 경우에는 제공된 큐를 변경할 수 있으나 목적지의 클래스(관리 또는 비관리)는 변경할 수 없습니다.

제공되거나 리턴된 경우에 관계없이, *Hobj*는 이 구독에 전송된 발행 메시지를 수신하려는 후속 MQGET 또는 MQCB 호출에 지정되어야 합니다.

*Hobj* 핸들은 이에 대해 MQCLOSE 호출이 실행되는 경우, 또는 핸들의 범위를 정의하는 처리 단위가 종료되는 경우에는 더 이상 올바르지 않게 됩니다(애플리케이션의 연결이 끊어질 때까지). 리턴되는 오브젝트 핸들의 범위는 호출에서 지정된 연결 핸들의 범위와 동일합니다. 핸들 범위에 대한 정보는 [Hconn\(MQHCONN\)](#) - 출력을 참조하십시오. *Hobj* 핸들의 MQCLOSE는 *Hsub* 핸들에 영향을 주지 않습니다.

### Hsub

유형: MQHOBJ - 출력

이 핸들은 작성된 구독을 나타냅니다. 다음 두 가지 추가 조작에 사용할 수 있습니다.

- 구독을 수행할 때 MQSO\_PUBLICATIONS\_ON\_REQUEST 옵션이 사용된 경우에는 발행물을 전송하도록 요청하는 후속 MQSUBRQ 호출에서 이를 사용할 수 있습니다.
- 작성된 구독을 제거하려면 후속 MQCLOSE 호출에 사용합니다. *Hsub* 핸들은 MQCLOSE 호출이 실행된 경우, 또는 핸들의 범위를 정의하는 처리 단위가 종료되는 경우 더 이상 올바르지 않게 됩니다. 리턴되는 오브젝트 핸들의 범위는 호출에서 지정된 연결 핸들의 범위와 동일합니다. *Hsub* 핸들의 MQCLOSE는 *Hobj* 핸들에 영향을 주지 않습니다.

이 핸들을 MQGET 또는 MQCB 호출로 전달할 수 없습니다. **Hobj** 매개변수를 사용해야 합니다. MQCLOSE 또는 MQSUBRQ가 아닌 IBM MQ 호출에서는 이 핸들을 사용할 수 없습니다. 이 핸들을 다른 IBM MQ 호출에 전달하면 MQRC\_HOBJ\_ERROR가 발생합니다.

### CompCode

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

#### MQCC\_OK

성공적인 완료

#### MQCC\_WARNING

경고(일부 완료)

#### MQCC\_FAILED

호출에 실패했습니다.

### 원인

유형: MQLONG - 출력

*CompCode*를 규정하는 이유 코드입니다.

*CompCode*가 MQCC\_OK인 경우 이유 코드는 다음과 같습니다.

#### MQRC\_NONE

(0, X'000') 보고할 이유가 없습니다.

*CompCode*가 MQCC\_FAILED인 경우 이유 코드는 다음 중 하나입니다.

#### MQRC\_CLUSTER\_RESOLUTION\_ERROR

(2189, X'88D') 클러스터 이름을 해석하지 못했습니다.

#### MQRC\_DURABILITY\_NOT\_ALLOWED

2436 (X'0984') MQSO\_DURABLE 옵션을 사용한 MQSUB 호출이 실패했습니다.

#### MQRC\_FUNCTION\_NOT\_SUPPORTED

2298 (X'08FA') 요청된 함수를 현재 환경에서 사용할 수 없습니다.

#### MQRC\_HOBJ\_ERROR

2019 (X'07E3') 오브젝트 핸들 *Hobj*가 올바르지 않습니다.

#### MQRC\_IDENTITY\_MISMATCH

2434 (X'0982') 구독 이름이 기존 구독과 일치합니다.

#### MQRC\_NOT\_AUTHORIZED

2035 (X'07F3') 조작을 수행할 수 있는 권한이 사용자에게 부여되지 않았습니다.

**MQRC\_NO\_SUBSCRIPTION**

2428 (X'097C') 식별된 구독 이름이 존재하지 않습니다.

**MQRC\_OBJECT\_STRING\_ERROR**

2441 (X'0989') ObjectString 필드가 올바르지 않습니다.

**MQRC\_OPTIONS\_ERROR**

2046 (X'07FE') 올바르지 않은 옵션을 포함하는 옵션 매개변수 또는 필드 또는 올바르지 않은 옵션의 결합입니다.

**MQRC\_Q\_MGR QUIESCING**

2161 (X'0871') 큐 관리자가 정지 중입니다.

**MQRC\_RECONNECT\_Q\_MGR REQD**

2555 (X'09FB'X) MQCNO\_RECONNECT\_Q\_MGR 옵션이 필요합니다.

**MQRC\_RETAINED\_MSG\_Q\_ERROR**

2525 (X'09DD') 구독된 토픽 문자열에 존재하는 보유된 발행을 검색할 수 없습니다.

**MQRC\_RETAINED\_NOT\_DELIVERED**

2526 (X'09DE') 구독된 토픽 문자열에 존재하는 보유된 발행을 구독 목적지 큐에 전달할 수 없으며 데드-레터 큐에 전달할 수 없습니다.

**MQRC\_SD\_ERROR**

2424 (X'0978') 구독 디스크립터(MQSD)가 올바르지 않습니다.

**MQRC\_SELECTION\_NOT\_AVAILABLE**

2551 (X'09F7') 선택 문자열이 IBM MQ 선택자 구문을 따르지 않으며 사용 가능한 확장 메시지 선택제 공자가 없습니다.

**MQRC\_SELECTION\_STRING\_ERROR**

2519 (X'09D7') 선택 문자열은 MQCHARV 구조 문서에 설명되어 있는 바와 같이 지정해야 합니다.

**MQRC\_SELECTOR\_SYNTAX\_ERROR**

2459 (X'099B') MQOPEN, MQPUT1 또는 MQSUB 호출이 발행되었으나 구문 오류를 포함하는 선택 문자열이 지정되었습니다.

**MQRC\_SUB\_USER\_DATA\_ERROR**

2431 (X'097F') SubUserData 필드가 올바르지 않습니다.

**MQRC\_SUB\_NAME\_ERROR**

2440 (X'0988') SubName 필드가 올바르지 않습니다.

**MQRC\_SUB\_ALREADY\_EXISTS**

2432 (X'0980') 구독이 이미 있습니다.

**MQRC\_SUB\_USER\_DATA\_ERROR**

2431 (X'097F') SubUserData 필드가 올바르지 않습니다.

**MQRC\_TOPIC\_STRING\_ERROR**

2425 (X'0979') 토픽 문자열이 올바르지 않습니다.

**MQRC\_UNKNOWN\_OBJECT\_NAME**

2085 (X'0825') MQSD ObjectName 필드에서 식별된 오브젝트를 찾을 수 없습니다.

**MQRC\_SUB\_JOIN\_NOT ALTERABLE**

29440 (X'7300') 구독 공유 모드가 기존 구독과 호환되지 않습니다. 이 오류는 비JMS 애플리케이션에서 JMS 2.0 공유 구독을 재개하려고 시도하는 경우 리턴될 수 있습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

**사용법 참고**

- 구독은 사전정의된 토픽 오브젝트의 간략한 이름, 토픽 문자열의 전체 이름, 또는 두 부분의 연결로 이름 지정된 토픽에 대해 수행됩니다. 549 페이지의 『MQSD - 구독 디스크립터』에 있는 *ObjectName* 및 *ObjectString*에 대한 설명을 참조하십시오.
- MQSUB 호출 발행 시 액세스 허용 전에 애플리케이션이 실행 중인 사용자 ID에 적절한 레벨의 권한이 있는지 확인하기 위해 큐 관리자가 보안 검사를 수행합니다. 적절한 토픽 오브젝트는 토픽 계층 구조에 있으며 구독을 위한 권한이 설정되었는지 확인하기 위해 이 토픽 오브젝트에 대한 권한 검사가 수행됩니다.

MQSO\_MANAGED 옵션이 사용되지 않은 경우에는 출력에 대한 권한이 설정되었는지 확인하기 위해 목적지 큐에 대해 권한 검사가 수행됩니다. MQSO\_MANAGED 옵션이 사용된 경우에는 출력에 대한 관리 큐에 대해 권한 검사가 수행되지 않으며 액세스가 조회되지 않습니다.

- Hobj를 입력으로 제공하지 않은 경우에는 MQSUB 호출이 오브젝트 핸들(Hobj)과 구독 핸들(Hsub)의 두 가지 핸들을 할당합니다.
- MQSO\_MANAGED 옵션이 사용된 MQSUB 호출에서 리턴된 Hobj는 백아웃 임계값 및 초과 백아웃 리큐 이름과 같은 속성을 찾기 위해 조회될 수 있습니다. 관리 큐의 이름도 조회할 수 있지만 이 큐를 직접 열려고 시도해서는 안 됩니다.
- 구독을 그룹화하면 둘 이상의 그룹이 발행물에 일치하더라도 하나의 구독만 구독 그룹에 전달되게 할 수 있습니다. 구독은 MQSO\_GROUP\_SUB 옵션을 사용하여 그룹화되며, 구독을 그룹화하려면 이러한 구독이 다음과 같아야 합니다.
  - 동일한 큐 관리자에서 동일한 이름을 가진 큐(MQSO\_MANAGED 옵션을 사용하지 않는)를 사용해야 합니다 (MQSUB 호출의 Hobj 매개변수로 표시됨).
  - SubCorrelId가 동일해야 합니다.
  - SubLevel이 동일해야 합니다.

이 속성은 그룹에 있는 것으로 간주되는 구독 세트를 정의하며, 구독이 그룹화된 경우 대체할 수 없는 속성이기도 합니다. SubLevel을 대체하면 MQRC\_SUBLEVEL\_NOT\_ALTERABLE이 발생하며, 그 외 항목(구독이 그룹화되지 않은 경우 변경할 수 있는)을 대체하면 MQRC\_GROUPING\_NOT\_ALTERABLE이 발생합니다.

- MQSUB 호출이 완료되었다고 해서 조치가 완료되었음을 의미하지는 않습니다. 이 호출이 완료되었는지 확인하려면 분산 네트워크에 대한 비동기 명령이 완료되었는지 확인의 DEFINE SUB 단계를 참조하십시오.
- MQSD의 필드는 MQSO\_RESUME 옵션을 사용하는 MQSUB 호출에서 리턴될 때 채워집니다. 리턴된 MQSD는 MQSD에 적용된 구독에서 변경해야 하는 사항과 함께, MQSO\_ALTER 옵션을 사용하는 MQSUB 호출에 직접 전달할 수 있습니다. 일부 필드는 표에 명시된 대로 특별한 고려사항이 적용됩니다.

표 556. MQSD의 필드에 대한 특별한 고려사항	
MQSD의 필드 이름	특별 고려사항
액세스 또는 작성 옵션	일부 옵션은 MQSUB 호출에서 리턴될 때 재설정할 수 있습니다. 그 후 해당 MQSD를 MQSUB 호출에서 재사용하는 경우에는 필요한 옵션을 명시적으로 설정해야 합니다.
지속성 옵션, 목적지 옵션, 등록 옵션 및 와일드카드 옵션	이러한 옵션은 적절하게 설정됩니다.
발행물 옵션	이러한 옵션은 MQSO_CREATE에만 적용 가능한 MQSO_NEW_PUBLICATIONS_ONLY를 제외하고 적절하게 설정됩니다.
기타 옵션	이 옵션은 MQSUB 호출의 리턴에서 변경되지 않습니다. 이는 API 호출이 실행되고 구독에서 저장되지 않는 방법을 제어합니다. MQSD를 재사용하는 후속 MQSUB 호출에서 필수로 설정해야 합니다.
ObjectName	이 입력 전용 필드는 MQSUB 호출의 리턴에서 변경되지 않습니다.
ObjectString	이 입력 전용 필드는 MQSUB 호출의 리턴에서 변경되지 않습니다. 버퍼가 제공되는 경우에는 ResObjectString 필드에 사용된 전체 토픽 이름이 리턴됩니다.
AlternateUserId 및 AlternateSecurityId	이 입력 전용 필드는 MQSUB 호출의 리턴에서 변경되지 않습니다. 이는 API 호출이 실행되고 구독에서 저장되지 않는 방법을 제어합니다. MQSD를 재사용하는 후속 MQSUB 호출에서 필수로 설정해야 합니다.



표 556. MQSD의 필드에 대한 특별한 고려사항 (계속)	
MQSD의 필드 이름	특별 고려사항
SubExpiry	MQSO_RESUME 옵션을 사용한 MQSUB 호출에서 리턴될 때, 이 필드는 남은 구독 만기 시간이 아니라 원래 구독 만기 시간으로 설정됩니다. 그 후 MQSO_ALTER 옵션을 사용한 MQSUB 호출에서 해당 MQSD를 재사용하면 카운트다운을 다시 시작하도록 구독 만기가 재설정됩니다.
SubName	이 필드는 MQSUB 호출의 입력 필드이고 출력에서 변경되지 않습니다.
SubUserData 및 SelectionString	이러한 변수 길이 필드는 버퍼가 제공되며 <i>VSBufSize</i> 에 양수 버퍼 길이가 있는 경우 MQSO_RESUME 옵션을 사용한 MQSUB 호출의 출력에서 리턴됩니다. 버퍼가 제공되지 않은 경우에는 MQCHARV의 <i>VSLength</i> 필드에 길이만 리턴됩니다. 제공된 버퍼가 필드를 리턴하는 데 필요한 공간보다 작은 경우에는 제공된 버퍼에 <i>VSBufSize</i> 바이트만 리턴됩니다.  그 후 MQSO_ALTER 옵션을 사용한 MQSUB 호출에서 해당 MQSD를 재사용할 때 버퍼는 제공되지 않았으나 0이 아닌 <i>VSLength</i> 가 제공되는 경우, 이 길이가 필드의 기존 길이와 일치하면 필드가 대체되지 않습니다.
SubCorrelId 및 PubAccountingToken	MQSO_SET_CORREL_ID를 사용하지 않으면 큐 관리자가 <i>SubCorrelId</i> 를 생성합니다. MQSO_SET_IDENTITY_CONTEXT를 사용하지 않으면 큐 관리자가 <i>PubAccountingToken</i> 을 생성합니다.  이러한 필드는 MQSO_RESUME 옵션을 사용한 MQSUB 호출의 MQSD에서 리턴됩니다. 큐 관리자가 이들을 생성하는 경우에는 MQSO_CREATE 또는 MQSO_ALTER 옵션을 사용한 MQSUB 호출에서 생성된 값이 리턴됩니다.
PubPriority, SubLevel & PubApplIdentityData	이러한 필드는 MQSD에서 리턴됩니다.
ResObjectString	이 출력 전용 필드는 버퍼가 제공되는 경우 MQSD에서 리턴됩니다.

## C 호출

```
MQSUB (Hconn, &SubDesc, &Hobj, &Hsub, &CompCode, &Reason)
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN Hconn; /* Connection handle */
MQSD SubDesc; /* Subscription descriptor */
MQHOBJ Hobj; /* Object handle */
MQHOBJ Hsub; /* Subscription handle */
MQLONG CompCode; /* Completion code */
MQLONG Reason; /* Reason code qualifying CompCode */
```

## COBOL 호출

```
CALL 'MQSUB' USING HCONN, SUBDESC, HOBJ, HSUB, COMPCODE, REASON.
```



매개변수를 다음과 같이 선언합니다.

```
** Connection handle
01 HCONN      PIC S9(9) BINARY.
** Subscription descriptor
01 SUBDESC.
   COPY CMQSDV.
** Object handle
01 HOBJ      PIC S9(9) BINARY.
** Subscription handle
01 HSUB      PIC S9(9) BINARY.
** Completion code
01 COMPCODE  PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON    PIC S9(9) BINARY.
```

## PL/I 호출

```
call MQSUB (Hconn, SubDesc, Hobj, Hsub, CompCode, Reason)
```

매개변수를 다음과 같이 선언합니다.

```
dcl Hconn      fixed bin(31); /* Connection handle */
dcl SubDesc    like MQSD;    /* Subscription descriptor */
dcl Hobj       fixed bin(31); /* Object handle */
dcl Hsub       fixed bin(31); /* Subscription handle */
dcl CompCode   fixed bin(31); /* Completion code */
dcl Reason     fixed bin(31); /* Reason code qualifying CompCode */
```

## 상위 레벨 어셈블러 호출

```
CALL MQSUB, (HCONN, SUBDESC, HOBJ, HSUB, COMPCODE, REASON)
```

매개변수를 다음과 같이 선언합니다.

```
HCONN      DS      F  Connection handle
SUBDESC    CMQSDA  ,  Subscription descriptor
HOBJ       DS      F  Object handle
HSUB       DS      F  Subscription handle
COMPCODE   DS      F  Completion code
REASON     DS      F  Reason code qualifying COMPCODE
```

## MQSUBRQ - 구독 요청

구독자가 MQSO\_PUBLICATIONS\_ON\_REQUEST에 등록되면 MQSUBRQ 호출을 사용하여 보유된 발행에 대해 요청을 작성하십시오.

### 구문

```
MQSUBRQ(Hconn, Hsub, Action, SubRqOpts, Compcode, Reason)
```

### 매개변수

#### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *Hconn*의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

z/OS for CICS 애플리케이션에서 MQCONN 호출을 생략할 수 있으며 *Hconn*에 다음 값이 지정됩니다.

## **MQHC\_DEF\_HCONN**

기본 연결 핸들

### **Hsub**

유형: MQHOBJ - 입력

이 핸들은 업데이트가 요청되는 구독을 나타냅니다. *Hsub*의 값이 이전 MQSUB 호출에서 리턴되었습니다.

### **Action**

유형: MQLONG - 입력

이 매개변수는 구독에 요청되는 특별한 조치를 제어합니다. 다음 값을 지정할 수 있습니다.

### **MQSR\_ACTION\_PUBLICATION**

이 조치는 지정된 토픽에 대해 업데이트 발행이 전송됨을 요청합니다. 이는 구독을 작성할 때 구독자가 MQSUB 호출에서 MQSO\_PUBLICATIONS\_ON\_REQUEST 옵션을 지정한 경우에만 사용할 수 있습니다. 큐 관리자에게 토픽에 대해 보유된 발행물이 있는 경우 이는 구독자에게 전송됩니다. 그렇지 않은 경우 호출은 실패합니다. 애플리케이션은 보유된 발행물이 전송되는 경우 해당 발행물의 MQIsRetained 메시지 특성으로 표시됩니다.

Hsub 매개변수로 표시된 기존 구독의 토픽이 와일드카드를 포함할 수 있으므로 구독자는 다중의 보유된 발행을 수신할 수 있습니다.

### **SubRqOpts**

유형: MQSRO - 입출력(I/O)

이러한 옵션은 MQSUBRQ의 조치를 제어하며 세부사항은 569 페이지의 『MQSRO - 구독 요청 옵션』의 내용을 참조하십시오.

옵션이 필요하지 않은 경우 C 또는 S/390 어셈블러로 작성된 프로그램은 MQSRO 구조의 주소를 지정하는 대신 널 매개변수 주소를 지정할 수 있습니다.

### **CompCode**

유형: MQLONG - 출력

완료 코드는 다음 중 하나입니다.

#### **MQCC\_OK**

성공적인 완료

#### **MQCC\_WARNING**

경고(일부 완료)

#### **MQCC\_FAILED**

호출에 실패했습니다.

### **원인**

유형: MQLONG - 출력

*CompCode*를 규정하는 이유 코드입니다.

*CompCode*이(가) MQCC\_OK인 경우:

#### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_FAILED인 경우:

#### **MQRC\_FUNCTION\_NOT\_SUPPORTED**

2298 (X'08FA') 요청된 함수를 현재 환경에서 사용할 수 없습니다.

#### **MQRC\_NO\_RETAINED\_MSG**

2437 (X'0985') 현재 이 토픽에 대해 저장된 보유된 발행물이 없습니다.

#### **MQRC\_OPTIONS\_ERROR**

2046 (X'07FE') 올바르지 않은 옵션을 포함하는 옵션 매개변수 또는 필드 또는 올바르지 않은 옵션의 결합입니다.

### **MQRC\_Q\_MGR QUIESCING**

2161 (X'0871') 큐 관리자가 정지 중입니다.

### **MQRC\_SRO\_ERROR**

2438 (X'0986') MQSUBRQ 호출에서 구독 요청 옵션 MQSRO는 올바르지 않습니다.

### **MQRC\_RETAINED\_MSG\_Q\_ERROR**

2525 (X'09DD') 구독된 토픽 문자열에 존재하는 보유된 발행을 검색할 수 없습니다.

### **MQRC\_RETAINED\_NOT\_DELIVERED**

2526 (X'09DE') 구독된 토픽 문자열에 존재하는 보유된 발행을 구독 목적지 큐에 전달할 수 없으며 데드-레터 큐에 전달할 수 없습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

## **사용법 참고**

다음 사용법 참고는 조치 코드 MQSR\_ACTION\_PUBLICATION의 사용에 적용됩니다.

1. 이 동사가 성공적으로 완료되면 지정된 구독과 일치하는 보유된 발행은 구독에 전송되고 구독을 작성한 원래 MQSUB 동사에서 리턴된 Hobj를 사용하는 MQCB 또는 MQGET을 사용하여 수신할 수 있습니다.
2. 구독을 작성한 원래 MQSUB 동사로 등록된 토픽이 와일드카드를 포함한 경우 둘 이상의 보유된 발행을 전송할 수 있습니다. 이 호출의 결과로 전송된 발행 수는 SubRqOpts 구조로 NumPubs 필드에 기록됩니다.
3. 이 동사가 MQRC\_NO\_RETAINED\_MSG의 이유 코드와 함께 완료되는 경우 지정된 토픽에 대해 현재 보유된 발행이 없습니다.#
4. 이 동사가 MQRC\_RETAINED\_MSG\_Q\_ERROR 또는 MQRC\_RETAINED\_NOT\_DELIVERED의 이유 코드와 함께 완료되는 경우 지정된 토픽에 대해 현재 보유된 발행이 있지만 전달할 수 없음을 의미하는 오류가 발생했습니다.
5. 이 호출을 작성하기 전에 애플리케이션에 토픽에 대한 현재 구독이 있어야 합니다. 이전 애플리케이션 인스턴스에서 구독이 작성되었고 구독에 대한 유효한 핸들을 사용할 수 없는 경우 애플리케이션은 먼저 MQSO\_RESUME 옵션으로 MQSUB를 호출하여 이 호출에 대해 사용할 핸들을 획득해야 합니다.
6. 발행물은 이 애플리케이션의 현재 구독과 함께 사용할 등록되는 목적지에 전송됩니다. 발행을 임의 위치로 전송해야 하는 경우 MQSO\_ALTER 옵션이 있는 MQSUB 호출을 사용하여 구독을 먼저 대체해야 합니다.

## **C 호출**

```
MQSUB (Hconn, Hsub, Action, &SubRqOpts, &CompCode, &Reason)
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN Hconn;      /* Connection handle */
MQHOBJ  Hsub;        /* Subscription handle */
MQLONG  Action;      /* Action requested by MQSUBRQ */
MQSRO   SubRqOpts;   /* Options that control the action of MQSUBRQ */
MQLONG  CompCode;    /* Completion code */
MQLONG  Reason;      /* Reason code qualifying CompCode */
```

## **COBOL 호출**

```
CALL 'MQSUBRQ' USING HCONN, HSUB, ACTION, SUBRQOPTS, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```
** Connection handle
01 HCONN PIC S9(9) BINARY.
** Subscription handle
01 HSUB PIC S9(9) BINARY.
** Action requested by MQSUBRQ
01 ACTION PIC S9(9) BINARY.
```

```

** Options that control the action of MQSUBRQ
01 SUBRQOPTS.
COPY CMQSROV.
** Completion code
01 COMPCODE PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON PIC S9(9) BINARY.

```

## PL/I 호출

```
call MQSUBRQ (Hconn, Hsub, Action, SubRqOpts, CompCode, Reason)
```

매개변수를 다음과 같이 선언합니다.

```

dcl Hconn fixed bin(31); /* Connection handle */
dcl Hsub fixed bin(31); /* Subscription handle */
dcl Action fixed bin(31); /* Action requested by MQSUBRQ */
dcl SubRqOpts like MQSR0; /* Options that control the action of MQSUBRQ */
dcl CompCode fixed bin(31); /* Completion code */
dcl Reason fixed bin(31); /* Reason code qualifying CompCode */

```

## 상위 레벨 어셈블러 호출

```
CALL MQSUBRQ,(HCONN, HSUB, ACTION, SUBRQOPTS,COMPCODE,REASON)
```

매개변수를 다음과 같이 선언합니다.

```

HCONN DS F Connection handle
HSUB DS F Subscription handle
ACTION DS F Action requested by MQSUBRQ
SUBRQOPTS CMQSROA , Options that control the action of MQSUBRQ
COMPCODE DS F Completion code
REASON DS F Reason code qualifying COMPCODE

```

## 오브젝트 속성

이 토픽의 컬렉션은 MQINQ 함수 호출의 제목이 될 수 있는 IBM MQ 오브젝트만 나열하고 조회할 수 있는 속성 및 사용될 선택자에 대한 세부사항을 제공합니다.

### 큐 관리자의 속성

일부 큐 관리자 속성은 특정 구현에 맞게 수정됩니다. 다른 큐 관리자 속성은 MQSC 명령 ALTER QMGR을 사용하여 변경할 수 있습니다.

DISPLAY QMGR 명령을 사용하는 방법으로도 속성을 표시할 수 있습니다. 대부분의 큐 관리자 속성은 특수 MQOT\_Q\_MGR 오브젝트를 열고 리턴된 핸들과 함께 MQINQ 호출을 사용하여 조회할 수 있습니다.

다음은 큐 관리자와 관련된 속성을 요약한 표입니다. 속성은 알파벳 순서로 설명합니다.

**참고:** 이 섹션에 표시된 속성 이름은 MQINQ 호출과 함께 사용되는 설명적인 이름이며 PCF 명령에 대한 이름과 동일합니다. 속성을 정의, 대체 또는 표시하기 위해 MQSC 명령이 사용되는 경우, 대체 짧은 이름이 사용됩니다. 자세한 정보는 [MQSC 명령](#)을 참조하십시오.

표 557. 큐 관리자의 속성	
속성	설명
<a href="#">AccountingConnOverride</a>	계정 설정을 대체합니다.
<a href="#">AccountingInterval</a>	중간 계정 레코드를 쓰는 빈도입니다.
<a href="#">ActivityConnOverride</a>	활동 설정을 대체합니다.
<a href="#">ActivityTrace</a>	IBM MQ MQI 애플리케이션 활동 추적의 컬렉션을 제어합니다.

표 557. 큐 관리자의 속성 (계속)	
속성	설명
<a href="#">AdoptNewMCACheck</a>	새 MCA 채택 여부를 판별하기 위해 검사하는 요소입니다.
 <a href="#">AdoptNewMCAType</a>	자동으로 특정 채널 유형의 MCA의 고아 인스턴스를 재시작할지 여부입니다.
<a href="#">AlterationDate</a>	정의가 마지막으로 변경된 날짜
<a href="#">AlterationTime</a>	정의가 마지막으로 변경된 시간
<a href="#">AuthorityEvent</a>	권한 부여(권한 부여되지 않음) 이벤트 생성 여부를 제어합니다.
 <a href="#">BridgeEvent</a>	브릿지 이벤트의 제어 속성입니다.
<a href="#">ChannelAutoDef</a>	자동 채널 정의의 허용 여부를 제어합니다.
<a href="#">ChannelAutoDefEvent</a>	채널 자동 정의 이벤트 생성 여부를 제어합니다.
<a href="#">ChannelAutoDefExit</a>	자동 채널 정의에 대한 사용자 엑시트의 이름입니다.
<a href="#">ChannelEvent</a>	채널 이벤트에 대한 제어 속성입니다.
<a href="#">ChannelInitiatorControl</a>	채널 시작기에 대한 제어 속성입니다.
<a href="#">ChannelMonitoring</a>	채널에 대한 온라인 모니터링 데이터입니다.
<a href="#">ChannelStatistics</a>	채널에 대한 통계 데이터의 콜렉션을 제어합니다.
 <a href="#">ChinitAdapters</a>	IBM MQ 호출 처리를 위한 어댑터 하위 태스크의 수입입니다.
 <a href="#">ChinitDispatchers</a>	채널 시작기에 사용할 디스패처의 수.
	IBM 사용을 위해 예약됩니다.
 <a href="#">ChinitTraceAutoStart</a>	채널 시작기 추적의 자동 시작 여부.
 <a href="#">ChinitTraceTableSize</a>	채널 시작기의 추적 데이터 공간의 크기입니다.
<a href="#">ClusterSenderMonitoringDefault</a>	클러스터 송신자 채널의 온라인 모니터링 데이터 기본값입니다.
<a href="#">ClusterSenderStatistics</a>	클러스터 송신자 채널에 대한 통계 모니터링 정보의 콜렉션을 제어합니다.
<a href="#">ClusterWorkloadData</a>	클러스터 워크로드 엑시트의 사용자 데이터입니다.
<a href="#">ClusterWorkloadExit</a>	클러스터 워크로드 관리에 대한 사용자 엑시트의 이름입니다.
<a href="#">ClusterWorkloadLength</a>	클러스터 워크로드 엑시트에 전달된 메시지 데이터의 최대 길이입니다.
<a href="#">CLWLMRUChannels</a>	클러스터 워크로드 밸런싱에 대해 가장 최근에 사용된 채널의 수입입니다.
<a href="#">CLWLUseQ</a>	클러스터 워크로드 사용 리모트 큐입니다.
<a href="#">CodedCharSetId</a>	코드화 문자 세트 ID
<a href="#">CommandEvent</a>	명령 이벤트에 대한 제어 속성입니다.
<a href="#">CommandInputQName 속성</a>	명령 입력 큐 이름입니다.
<a href="#">CommandLevel</a>	명령 레벨
<a href="#">CommandServerControl 속성</a>	명령 서버에 대한 제어 속성입니다.
<a href="#">구성 이벤트 속성</a>	구성 이벤트에 대한 제어 속성입니다.
<a href="#">DeadLetterQName</a>	데드-레터 큐의 이름입니다.
<a href="#">DefClusterXmitQueue유형</a>	기본 클러스터 전송 큐 유형
<a href="#">DefXmitQName</a>	기본 전송 큐 이름입니다.
<a href="#">DistLists</a>	분배 목록 지원
 <a href="#">DNSGroup</a>	워크로드 관리자 동적 도메인 이름 서비스 지원을 사용할 때 TCP 리스너에 대한 그룹 이름입니다.
 <a href="#">DNSWLM</a>	TCP 리스너가 동적 도메인 이름 서비스에 대해 워크로드 관리자를 사용하여 등록하는지 여부입니다.
<a href="#">ExpiryInterval</a>	만기된 메시지에 대한 스캔 간의 간격입니다.
<a href="#">IGQPutAuthority</a>	그룹 내 큐잉 넣기 권한입니다.

표 557. 큐 관리자의 속성 (계속)	
속성	설명
IGQUserId	그룹 내 큐잉 사용자 ID입니다.
InhibitEvent	금지(가져오기 금지 및 넣기 금지) 이벤트 생성 여부를 제어합니다.
InitialKey 1	비밀번호 보호 시스템의 초기 키입니다.
IPAddressVersion	인터넷 프로토콜 주소의 버전입니다.
 IntraGroup큐잉	그룹 내 큐잉 지원입니다.
 ListenerTimer	APPC 또는 TCP/IP 실패 후에 리스너 재시작 시도 간의 시간 간격입니다.
LocalEvent	로컬 오류 이벤트 생성 여부를 제어합니다.
LoggerEvent	로거 이벤트 생성 여부를 제어합니다.
 LUGroupName	큐 공유 그룹의 인바운드 전송을 핸들링하는 LU 6.2 리스너에 대한 일반 LU 이름입니다.
 LUName	아웃바운드 LU 6.2 전송에 사용할 LU의 이름입니다.
LU62ARMSuffix	이 채널 시작기에 대해 LUADD를 지정하는, SYS1.PARMLIB 멤버 APPCPMxx의 접미부입니다.
 LU62Channels	LU 6.2를 사용하는 현재 채널 또는 연결된 클라이언트의 최대 수입니다.
MaxActiveChannels	언제든지 활성화할 수 있는 최대 채널 수.
MaxChannels	현재 채널의 최대 수입니다.
MaxHandles	핸들의 최대 수입니다.
MaxMsgLength	최대 메시지 길이(바이트)
MaxPriority 속성	최대 우선순위입니다.
MaxPropertiesLength	특성 데이터의 최대 길이(바이트)입니다.
MaxUncommittedMsgs	작업 단위 내에서 커밋되지 않은 최대 메시지 수입니다.
MQIAccounting	MQI 데이터에 대한 회계 정보의 콜렉션을 제어합니다.
MQIStatistics	큐 관리자에 대한 통계 모니터링 정보의 콜렉션을 제어합니다.
MsgMarkBrowseInterval	큐 관리자가 찾아본 메시지에서 표시를 제거할 수 있는 간격입니다.
 OutboundPort최소	OutboundPortMin을 사용하여 발신 채널을 바인딩할 때 사용할 포트 번호의 범위를 정의합니다.
 OutboundPort최소	OutboundPortMax을 사용하여 발신 채널을 바인딩할 때 사용할 포트 번호의 범위를 정의합니다.
PerformanceEvent	성능 관련 이벤트 생성 여부를 제어합니다.
플랫폼	큐 관리자가 실행 중인 플랫폼입니다.
PubSubNPIInputMsg	미배달 입력 메시지를 제거할지(또는 유지할지) 여부입니다.
PubSubNPResponse	미배달된 작동을 제어합니다.
PubSubMaxMsgRetryCount	동기점 아래에서 실패한 명령 메시지를 처리할 때 재시도 수입니다.
PubSubSyncPoint	지속적 메시지만 또는 모든 메시지를 동기점 아래에서 처리해야 하는지 여부입니다.
PubSubMode	큐된 발행/구독 인터페이스가 실행 중인지 여부입니다.
QMgrDesc	큐 관리자 설명입니다.
QMgrIdentifier	큐 관리자에 대해 내부적으로 생성된 고유한 ID입니다.
QMgrName	큐 관리자 이름
QSGName	큐 공유 그룹의 이름
QueueAccounting	큐에 대한 회계 정보의 콜렉션을 제어합니다.
QueueMonitoring	큐에 대한 온라인 모니터링 데이터
QueueStatistics	큐에 대한 통계 데이터의 콜렉션을 제어합니다.
 ReceiveTimeout	비활성 상태로 돌아가기 전에 TCP/IP 채널이 데이터를 기다리는 시간입니다.

표 557. 큐 관리자의 속성 (계속)	
속성	설명
 <a href="#">ReceiveTimeout최소</a>	<i>ReceiveTimeout</i> 의 규정자입니다.
 <a href="#">ReceiveTimeout유형</a>	비활성 상태로 돌아가기 전에 TCP/IP 채널이 데이터를 기다리는 최소 시간입니다.
<a href="#">RemoteEvent</a>	리모트 오류 이벤트 생성 여부를 제어합니다.
<a href="#">RepositoryName</a>	이 큐 관리자가 저장소 서비스를 제공하는 클러스터의 이름입니다.
<a href="#">RepositoryNamelist</a>	이 큐 관리자가 저장소 서비스를 제공하는 클러스터의 이름을 포함하는 이름 목록 오브젝트의 이름입니다.
<a href="#">ScyCase</a>	보안 프로파일의 경우입니다.
<a href="#">SharedQMgrName</a>	공유 큐 관리자 이름
800 페이지의 『SPLCAP』	큐 관리자에 대한 IBM MQ 고급 메시지 보안 보호가 켜짐 또는 꺼짐입니다.
<a href="#">SSLCRLNameList 1</a>	인증 정보 오브젝트의 이름을 포함하는 이름 목록 오브젝트의 이름입니다.
<a href="#">SSLCryptoHardware 1</a>	암호화 하드웨어 구성 문자열입니다.
<a href="#">SSEvent</a>	TLS 이벤트에 대한 제어 속성입니다.
<a href="#">SSLFIPSRequired</a>	암호화에 FIPS 인증 알고리즘만 사용합니다.
<a href="#">SSLKeyRepository 1</a>	TLS 키 저장소의 위치입니다.
<a href="#">SSLKeyRepository비밀번호 1</a>	TLS키 저장소의 비밀번호입니다.
<a href="#">SSLKeyResetCount</a>	TLS 키 재설정 수입니다.
<a href="#">SSLTasks 1</a>	TLS 호출 처리를 위한 서버 하위 태스크 수입니다.
<a href="#">StatisticsInterval</a>	통계 모니터링 데이터 작성 빈도입니다.
<a href="#">StartStopEvent</a>	시작 및 중지 이벤트 생성 여부를 제어합니다.
<a href="#">SyncPoint</a>	동기점 가용성입니다.
 <a href="#">TCP 채널</a>	TCP/IP를 사용하는 현재 채널 또는 연결된 클라이언트의 최대 수입니다.
 <a href="#">TCPKeepAlive</a>	연결의 다른 끝을 검사하기 위한 TCP KEEPALIVE 사용 여부입니다.
 <a href="#">TCP 이름</a>	사용 중인 TCP/IP 시스템의 이름입니다.
 <a href="#">TCPStackType</a>	채널 시작기가 TCP/IP 주소를 사용할 수 있는 방법입니다.
<a href="#">TraceRouteRecording attribute</a>	추적 라우트 정보의 기록을 제어합니다.
<a href="#">TriggerInterval</a>	트리거 메시지 간격입니다.
<a href="#">Version</a>	버전
<a href="#">XrCapability</a>	텔레메트리 명령 지원 여부를 지정합니다.
<b>참고:</b>	
1. 이 속성은 MQINQ 호출을 사용하여 조회할 수 없으며 이 절에서 설명하지 않습니다. 이 속성에 대한 세부사항은 <a href="#">큐 관리자 변경을 참조하십시오</a> .	

### 관련 태스크

MQI 클라이언트에서 런타임 시 FIPS 인증 CipherSpec만 사용하도록 지정

### 관련 참조

[AIX, Linux, and Windows용 FIPS\(Federal Information Processing Standard\)](#)

## AccountingConnOverride(MQLONG)

이는 애플리케이션이 Qmgr 속성에서 ACCTMQI 및 ACCTQDATA 값의 설정을 대체하도록 허용합니다.

값은 다음 중 하나입니다.

### MQMON\_DISABLED


애플리케이션은 MQCONNX 호출 시 MQCNO 구조에서 옵션 필드를 사용하여 ACCTMQI 및 ACCTQ Qmgr 속성의 설정을 대체할 수 없습니다. 이는 기본값입니다.

## **MQMON\_ENABLED**

애플리케이션은 MQCNO 구조에서 옵션 필드를 사용하여 ACCTQ 및 ACCTMQI Qmgr 속성을 대체할 수 있습니다.

이 값에 대한 변경사항은 속성을 변경한 이후 큐 관리자에 대한 연결에만 유효합니다.

이 속성은 다음 플랫폼에서만 지원됩니다.

-  IBM i
-   AIX and Linux
-  Windows






이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_ACCOUNTING\_CONN\_OVERRIDE 선택자를 사용하십시오.

## **AccountingInterval(MQLONG)**

이는 중간 계정 레코드가 기록되기 전의 시간(초)을 지정합니다.

값은 0 - 604800 범위의 정수이며 기본값은 1800(30분)입니다. 중간 레코드를 끄려면 0을 지정하십시오.

이 속성은 다음 플랫폼에서만 지원됩니다.

-  IBM i
-   AIX and Linux
-  Linux
-  Windows

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_ACCOUNTING\_INTERVAL 선택자를 사용하십시오.

## **ActivityConnOverride (MQLONG)**

애플리케이션에서 큐 관리자 속성에 있는 ACTVTRC 값의 설정을 대체할 수 있습니다.

값은 다음 중 하나입니다.

## **MQMON\_DISABLED**

애플리케이션은 MQCONNX 호출에서 MQCNO 구조의 옵션 필드를 사용하여 ACTVTRC 큐 관리자 속성의 설정을 대체할 수 없습니다. 이는 기본값입니다.

## **MQMON\_ENABLED**

애플리케이션이 MQCNO 구조의 옵션 필드를 사용하여 ACTVTRC 큐 관리자 속성을 대체할 수 있습니다.

이 값에 대한 변경사항은 속성을 변경한 이후 큐 관리자에 대한 연결에만 유효합니다.

이 속성은 멀티플랫폼에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 CAPRON 선택자를 사용하십시오.

## **ActivityTrace (MQLONG)**

이 속성은 IBM MQ MQI 애플리케이션 추적의 수집을 제어합니다.

값은 다음 중 하나입니다.

## **MQMON\_ON**

IBM MQ MQI 애플리케이션 활성 추적을 수집합니다.

## **MQMON\_OFF**

IBM MQ MQI 애플리케이션 활동 추적을 수집하지 않습니다. 이는 기본값입니다.

큐 관리자 속성 ACTVCONO를 ENABLED로 설정하면 이 값은 개별 연결에서 MQCNO 구조의 옵션 필드를 사용하여 대체될 수 있습니다.

이 값에 대한 변경사항은 속성을 변경한 이후 큐 관리자에 대한 연결에만 유효합니다.



이 속성은 멀티플랫폼에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_ACTIVITY\_TRACE 선택자를 사용하십시오.

### **AdoptNewMCACheck(MQLONG)**

이는 이미 활성인 MCA와 동일한 이름이 있는 새 인바운드 채널이 감지된 경우 MCA를 채택할지 여부를 판별하기 위해 검사하는 요소를 정의합니다.

값은 다음 중 하나입니다.

#### **MQADOPT\_CHECK\_Q\_MGR\_NAME**

큐 관리자 이름을 점검합니다.

#### **MQADOPT\_CHECK\_NET\_ADDR**

네트워크 주소를 점검합니다.


#### **MQADOPT\_CHECK\_ALL**

큐 관리자 이름 및 네트워크 주소를 검사합니다. 가능한 경우 이 검사를 수행하여 채널이 부주의로 또는 악의적으로 종료되는 것을 막으십시오. 이는 기본값입니다.

#### **MQADOPT\_CHECK\_NONE**

요소를 검사하지 않습니다.

이 매개변수의 변경사항은 다음 번에 채널이 채널 채택을 시도할 때 적용됩니다.

 이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_ADOPTNEWMCA\_CHECK 선택자를 사용하십시오.

### **AdoptNewMCAType(MQLONG)**

이는 AdoptNewMCACheck 속성과 일치하는 새 인바운드 채널 요청이 감지되는 경우 특정 채널 유형의 MCA의 고아 인스턴스를 자동으로 다시 시작할지 여부를 지정합니다.

다음 값 중 하나입니다.

#### **MQADOPT\_TYPE\_NO**

고아 채널 인스턴스 채택이 필요하지 않습니다. 이는 기본값입니다.

#### **MQADOPT\_TYPE\_ALL**

모든 채널 유형을 채택합니다.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_ADOPTNEWMCA\_TYPE 선택자를 사용하십시오.

### **AlterationDate (MQCHAR12)**

정의를 마지막으로 변경된 날짜입니다. 날짜의 형식은 YYYY-MM-DD이며 길이를 12바이트로 만들기 위해 두 개의 후미 공백으로 채워집니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_ALTERATION\_DATE 선택자를 사용하십시오. 이 속성의 길이는 MQ\_DATE\_LENGTH에서 제공됩니다.

### **AlterationTime (MQCHAR8)**

정의를 마지막으로 변경된 시간입니다. 시간의 형식은 HH.MM.SS입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_ALTERATION\_TIME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_TIME\_LENGTH에서 제공됩니다.

### **AuthorityEvent(MQLONG)**

이는 권한 부여(권한 부여되지 않음) 이벤트 생성 여부를 제어합니다. 다음 값 중 하나입니다.

#### **MQEVR\_DISABLED**

이벤트 보고를 사용하지 않습니다.

## **MQEVR\_ENABLED**

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_AUTHORITY\_EVENT 선택자를 사용하십시오.

## **z/OS** **BridgeEvent(MQLONG)**

IMS 브릿지 이벤트의 생성 여부를 지정합니다.

값은 다음 중 하나입니다.

### **MQEVR\_ENABLED**

다음과 같이 IMS 브릿지 이벤트를 생성하십시오.

MQRC\_BRIDGE\_STARTED  
MQRC\_BRIDGE\_STOPPED

### **MQEVR\_DISABLED**

IMS 브릿지 이벤트를 생성하지 않습니다. 이는 기본값입니다.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_BRIDGE\_EVENT 선택자를 사용하십시오.

## **ChannelAutoDef(MQLONG)**

이 속성은 유형 MQCHT\_RECEIVER 및 MQCHT\_SVRCONN의 채널에 대한 자동 정의를 제어합니다. MQCHT\_CLUSSDR 채널에 대한 자동 정의는 항상 사용으로 설정됩니다. 값은 다음 중 하나입니다.

### **MQCHAD\_DISABLED**

채널 자동 정의 사용 안함.

### **MQCHAD\_ENABLED**

채널 자동 정의 사용.

**Multi** 이 속성은 [멀티플랫폼](#)에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_CHANNEL\_AUTO\_DEF 선택자를 사용하십시오.

## **ChannelAutoDefEvent(MQLONG)**

이는 채널 자동 정의 이벤트 생성 여부를 제어합니다. 이는 유형 MQCHT\_RECEIVER, MQCHT\_SVRCONN, MQCHT\_CLUSSDR의 채널에 적용됩니다. 값은 다음 중 하나입니다.

### **MQEVR\_DISABLED**

이벤트 보고를 사용하지 않습니다.

### **MQEVR\_ENABLED**

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

**Multi** 이 속성은 [멀티플랫폼](#)에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_CHANNEL\_AUTO\_DEF\_EVENT 선택자를 사용하십시오.

## **ChannelAutoDefExit(MQCHARn)**

이는 자동 채널 정의에 대한 사용자 엑시트의 이름입니다. 이 이름이 공백이고 *ChannelAutoDef*에 값 MQCHAD\_ENABLED가 있는 경우 큐 관리자가 채널 정의를 작성하려고 할 때마다 엑시트가 호출됩니다. 이는 유형 MQCHT\_RECEIVER, MQCHT\_SVRCONN, MQCHT\_CLUSSDR의 채널에 적용됩니다. 그러면 엑시트는 다음 중 하나를 수행할 수 있습니다.

- 변경 없이 채널 정의를 작성하십시오.
- 작성된 채널 정의의 속성을 수정합니다.

- 채널 작성을 완전히 차단합니다.

**참고:** 이 속성의 길이 및 값 모두 환경에 특정합니다. 다양한 환경에서의 이 속성 값에 대한 세부사항은 1409 페이지의 『MQCD - 채널 정의』에서 MQCD 구조에 대한 도입을 참조하십시오.

**z/OS** z/OS에서 이 속성은 클러스터-송신자 및 클러스터-수신자 채널에만 적용됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_CHANNEL\_AUTO\_DEF\_EXIT 선택자를 사용하십시오. 이 속성의 길이는 MQ\_EXIT\_NAME\_LENGTH에서 지정됩니다.

### **ChannelEvent(MQLONG)**

채널 이벤트 생성 여부를 지정합니다.

다음 값 중 하나입니다.

#### **MQEVR\_EXCEPTION**

다음 채널 이벤트만 생성합니다.

- MQRC\_CHANNEL\_ACTIVATED
- MQRC\_CHANNEL\_CONV\_ERROR
- MQRC\_CHANNEL\_NOT\_ACTIVATED
- 다음 ReasonQualifiers가 있는 MQRC\_CHANNEL\_STOPPED:
  - MQRQ\_CHANNEL\_STOPPED\_ERROR
  - MQRQ\_CHANNEL\_STOPPED\_RETRY
  - MQRQ\_CHANNEL\_STOPPED\_DISABLED
  - MQRC\_CHANNEL\_STOPPED\_BY\_USER

#### **MQEVR\_ENABLED**

모든 채널 이벤트를 생성합니다. 즉, EXCEPTION가 생성한 사항 외에도 다음 채널 이벤트를 생성합니다.

- MQRC\_CHANNEL\_STARTED
- 다음 ReasonQualifier가 있는 MQRC\_CHANNEL\_STOPPED:
  - MQRQ\_CHANNEL\_STOPPED\_OK

#### **MQEVR\_DISABLED**

채널 이벤트를 생성하지 않습니다. 이는 기본값입니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_CHANNEL\_EVENT 선택자를 사용하십시오.

### **ChannelInitiatorControl(MQLONG)**

큐 관리자가 시작될 때 채널 시작기가 시작되는지 지정합니다.

다음 값 중 하나입니다.

#### **MQSVC\_CONTROL\_MANUAL**

채널 시작기가 자동으로 시작되지 않습니다.

#### **MQSVC\_CONTROL\_Q\_MGR**

큐 관리자가 시작될 때 채널 시작기가 자동으로 시작됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_CHINIT\_CONTROL 선택자를 사용하십시오.

### **ChannelMonitoring (MQLONG)**

이 속성은 채널의 온라인 모니터링 데이터를 지정합니다.

값은 다음 중 하나입니다.

#### **MQMON\_NONE**

MONCHL 채널 속성의 설정에 관계없이 모든 채널의 채널 모니터링에 대한 데이터 콜렉션을 사용 안함으로 설정합니다. 이는 기본값입니다.

**MQMON\_OFF**

MONCHL 채널 속성에서 QMGR을 지정하는 채널에 대해 모니터링 데이터 콜렉션을 끕니다.

**MQMON\_LOW**

MONCHL 채널 속성에서 QMGR을 지정하는 채널의 데이터 콜렉션 비율이 낮은 모니터링 데이터 콜렉션을 켭니다.

**MQMON\_MEDIUM**

MONCHL 채널 속성에서 QMGR을 지정하는 채널의 데이터 콜렉션 비율이 중간인 모니터링 데이터 콜렉션을 켭니다.

**MQMON\_HIGH**

MONCHL 채널 속성에서 QMGR을 지정하는 채널의 데이터 콜렉션 비율이 높은 모니터링 데이터 콜렉션을 켭니다.

**z/OS** ~에 z/OS 시스템에서 이 매개변수를 활성화하면 선택한 값에 관계없이 통계 데이터 수집이 활성화됩니다. LOW, MEDIUM 또는 HIGH를 지정해도 결과에는 차이가 없습니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_MONITORING\_CHANNEL 선택자를 사용하십시오.

**ChannelStatistics (MQLONG)**

이는 채널에 대한 통계 데이터 콜렉션을 제어합니다.

값은 다음 중 하나입니다.

**MQMON\_NONE**

STATCHL 채널 속성의 설정에 관계없이 모든 채널의 채널 통계에 대한 데이터 콜렉션을 사용 안함으로 설정합니다. 이는 기본값입니다.

**MQMON\_OFF**

STATCHL 채널 속성에서 QMGR을 지정하는 채널에 대해 통계 데이터 콜렉션을 끕니다.

**MQMON\_LOW**

STATCHL 채널 속성에서 QMGR을 지정하는 채널의 데이터 콜렉션 비율이 낮은 통계 데이터 콜렉션을 켭니다.

**MQMON\_MEDIUM**

STATCHL 채널 속성에서 QMGR을 지정하는 채널의 데이터 콜렉션 비율이 중간인 통계 데이터 콜렉션을 켭니다.

**MQMON\_HIGH**

STATCHL 채널 속성에서 QMGR을 지정하는 채널의 데이터 콜렉션 비율이 높은 통계 데이터 콜렉션을 켭니다.

대부분의 시스템에 중간을 사용하도록 권장됩니다. 그러나 매초마다 높은 메시지 볼륨을 처리하는 채널의 경우 낮음을 선택하여 샘플링 레벨을 줄이려고 할 수 있습니다. 또한 일부 메시지만 처리하는 채널 및 해당 채널에 대한 최신 정보가 중요한 경우 높음을 선택하려고 할 수 있습니다.

**z/OS** ~에 z/OS 시스템에서 이 매개변수를 활성화하면 선택한 값에 관계없이 통계 데이터 수집이 활성화됩니다. LOW, MEDIUM 또는 HIGH를 지정해도 결과에는 차이가 없습니다. 채널 회계 레코드를 수집하려면 이 매개변수를 사용해야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_STATISTICS\_CHANNEL 선택자를 사용하십시오.

**z/OS ChinitAdapters(MQLONG)**

이는 IBM MQ 호출을 처리하는 데 사용할 어댑터 하위 태스크의 수입입니다. 값은 0 - 9999여야 하며 기본값은 8입니다.

디스패처에 대한 어댑터의 비율(ChinitDispatchers 속성)은 약 8 대 5여야 합니다. 그러나 채널 수가 적으면 기본값에서 이 매개변수의 값을 줄일 필요가 없습니다. 테스트 시스템의 경우 8(기본값), 프로덕션 시스템의 경우 20의 값을 사용할 수 있습니다. 이상적으로 20개의 어댑터가 있어야 하며 이는 IBM MQ 호출에 대해 더 큰 유사성을 제공합니다. 이는 지속 메시지의 경우에는 중요합니다. 비지속 메시지의 경우 더 적은 어댑터가 더 적합할 수 있습니다.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_CHINIT\_ADAPTERS 선택자를 사용하십시오.

### **z/OS ChinitDispatchers(MQLONG)**

이는 채널 시작기에 사용할 디스패처의 수입니다. 값은 0 - 9999여야 하며 기본값은 5입니다.

지침으로 50개의 현재 채널에 대해 하나의 디스패처를 허용합니다. 그러나 일부 채널만 있는 경우 기본값에서 이 속성의 값을 줄일 필요가 없습니다. TCP/IP를 사용 중인 경우 TCP/IP 채널에 사용되는 디스패처의 가장 큰 수는 더 큰 값을 지정하는 경우에도 100입니다. 다음 설정을 사용할 수 있습니다. 테스트 시스템: 5(기본값), 프로덕션 시스템: 20(최대 1000개의 활성 채널을 핸들링하려면 20개의 디스패처가 필요함).

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_CHINIT\_DISPATCHERS 선택자를 사용하십시오.

### **z/OS ChinitTraceAutoStart(MQLONG)**

이는 채널 시작기 추적을 자동으로 시작할지 여부를 지정합니다.

값은 다음 중 하나입니다.

#### **MQTRAXSTR\_YES**

채널 시작기 추적을 자동으로 시작합니다. 이는 기본값입니다.

#### **MQTRAXSTR\_NO**

채널 시작기 추적을 자동으로 시작하지 않습니다.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_CHINIT\_TRACE\_AUTO\_START 선택자를 사용하십시오.

### **z/OS ChinitTraceTableSize(MQLONG)**

이는 채널 시작기의 추적 데이터 공간의 크기(MB)입니다.

값은 0 - 2048의 범위에 있어야 하며 기본값은 2입니다.

**참고:** 대형 z/OS 데이터 공간을 사용할 때마다 관련 z/OS 페이징 활동을 지원하기 위해 시스템에 충분한 보조 스토리지가 있는지 확인하십시오. SYS1.DUMP 데이터 세트의 크기를 늘려야 할 수 있습니다.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_CHINIT\_TRACE\_TABLE\_SIZE 선택자를 사용하십시오.

### **ClusterSenderMonitoringDefault(MQLONG)**

자동으로 정의된 클러스터 송신자 채널의 ChannelMonitoring 속성을 대체할 값을 지정합니다.

값은 다음 중 하나입니다.

#### **MQMON\_Q\_MGR**

온라인 모니터링 데이터의 콜렉션이 큐 관리자 **ChannelMonitoring** 속성의 설정에서 상속됩니다. 이는 기본값입니다.

#### **MQMON\_OFF**

채널에 대한 모니터링이 사용 불가능합니다.

#### **MQMON\_LOW**

**ChannelMonitoring**이 MQMON\_NONE이 아닌 경우, 시스템 성능에 최소한의 영향을 미치는 낮은 비율의 데이터 콜렉션으로 모니터링이 사용 가능합니다. 수집된 데이터는 가장 최신의 것이 아닐 수도 있습니다.

#### **MQMON\_MEDIUM**

**ChannelMonitoring**이 MQMON\_NONE이 아닌 경우, 시스템 성능에 제한된 영향을 미치는 적당한 비율의 데이터 콜렉션으로 모니터링이 사용 가능합니다.

#### **MQMON\_HIGH**

**ChannelMonitoring**이 MQMON\_NONE이 아닌 경우, 시스템 성능에 영향을 미칠 수 있는 높은 비율의 데이터 콜렉션으로 모니터링이 사용 가능합니다. 수집된 데이터는 가장 최신의 것입니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_MONITORING\_AUTO\_CLUSSDR 선택자를 사용하십시오.

### **ClusterSenderStatistics(MQLONG)**

클러스터 송신자 채널은 저장소에 있는 CLUSRCVR의 정의에서 자동으로 정의될 수 있으므로 ALTER 채널을 사용하여 자동 정의된 클러스터 송신자 채널에 대해 STATCHL 속성의 설정을 대체할 수 없습니다. 이러한 채널을 위해 온라인 모니터링 데이터를 수집할지 여부에 대한 결정은 이 큐 관리자 속성의 설정을 기반으로 합니다.

값은 다음 중 하나입니다.

#### **MQMON\_Q\_MGR**

자동 정의된 클러스터 송신자 채널의 통계 데이터 수집은 큐 관리자 속성 STATCHL의 값을 기반으로 합니다. 이는 기본값입니다.

#### **MQMON\_OFF**

자동 정의된 클러스터 송신자 채널의 통계 데이터 수집을 끄십시오.

#### **MQMON\_LOW**

데이터 콜렉션 비율이 낮은 자동 정의된 클러스터 송신자 채널의 통계 데이터 콜렉션을 사용 가능하게 하십시오.


#### **MQMON\_MEDIUM**

데이터 콜렉션 비율이 중간인 자동 정의된 클러스터 송신자 채널의 통계 데이터 콜렉션을 사용 가능하게 하십시오.

#### **MQMON\_HIGH**

데이터 콜렉션 비율이 높은 자동 정의된 클러스터 송신자 채널의 통계 데이터 콜렉션을 사용 가능하게 하십시오.

대부분의 시스템에 중간이 권장됩니다. 그러나 매초마다 높은 메시지 볼륨을 처리하는 자동 정의된 클러스터 송신자 채널의 경우 낮음을 선택하여 샘플링 레벨을 줄이려고 할 수 있습니다. 또한 일부 메시지만 처리하는 채널 및 해당 채널에 대한 최신 정보가 중요한 경우 높음을 선택하려고 할 수 있습니다.

 ~에 z/OS 시스템에서 이 매개변수를 활성화하면 선택한 값에 관계없이 통계 데이터 수집이 활성화됩니다. LOW, MEDIUM 또는 HIGH를 지정해도 결과는 차이가 없습니다. 채널 회계 레코드를 수집하려면 이 매개변수를 사용해야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_STATISTICS\_AUTO\_CLUSSDR 선택자를 사용하십시오.

### **ClusterWorkloadData(MQCHAR32)**

호출 시 클러스터 워크로드 엑시트에 전달되는 사용자 정의 32바이트 문자열입니다. 엑시트에 전달할 데이터가 없으면 문자열이 공백입니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_CLUSTER\_WORKLOAD\_DATA 선택자를 사용하십시오.

### **ClusterWorkloadExit(MQCHARn)**

이는 클러스터 워크로드 관리를 위한 사용자 엑시트의 이름입니다. 이 이름이 공백이 아니면 메시지를 클러스터 큐에 넣거나 클러스터 송신자 큐 간에 이동할 때마다 엑시트가 호출됩니다. 그러면 엑시트는 큐 관리자에서 선택한 큐 인스턴스를 메시지의 목적지로 허용하거나 다른 큐 인스턴스를 선택할 수 있습니다.

**참고:** 이 속성의 길이 및 값 모두 환경에 특정합니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_CLUSTER\_WORKLOAD\_EXIT 선택자를 사용하십시오. 이 속성의 길이는 MQ\_EXIT\_NAME\_LENGTH에서 지정됩니다.

### **ClusterWorkloadLength(MQLONG)**

클러스터 워크로드 엑시트에 전달되는 메시지 데이터의 최대 길이입니다. 엑시트에 전달된 데이터의 실제 길이는 다음 중 가장 작은 값입니다.

- 메시지의 길이.
- 큐 관리자의 **MaxMsgLength** 속성.



- **ClusterWorkloadLength** 속성.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_CLUSTER\_WORKLOAD\_LENGTH 선택자를 사용하십시오.

### **CLWLMRUChannels(MQLONG)**

이는 클러스터 워크로드 선택 알고리즘에 의해 사용될 것으로 간주되는 가장 최근에 사용된 클러스터 채널의 최대 수를 지정합니다.

이 값의 범위는 1 - 999999999입니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_CLWL\_MRU\_CHANNELS 선택자를 사용하십시오.

### **CLWLUseQ(MQLONG)**

이는 클러스터 워크로드에 대해 리모트 큐를 사용할지 여부를 지정합니다.

값은 다음 중 하나입니다.

#### **MQCLWL\_USEQ\_ANY**

로컬 및 리모트 큐 모두를 사용하십시오.

#### **MQCLWL\_USEQ\_LOCAL**

리모트 큐를 사용하지 마십시오. 이는 기본값입니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_CLWL\_USEQ 선택자를 사용하십시오.

### **CodedCharSetId(MQLONG)**

오브젝트 이름, 큐 작성 날짜 및 시간 등 MQI에 정의된 모든 문자열 필드에 대해 큐 관리자가 사용하는 문자 세트를 정의합니다. 이 문자 세트는 오브젝트 이름에 유효한 문자의 1바이트 문자를 포함한 세트여야 합니다. 메시지로 전달되는 애플리케이션 데이터에는 적용되지 않습니다. 값은 환경에 따라 달라집니다.

- z/OS에서 값은 큐 관리자가 시작될 때 시스템 매개변수에서 설정됩니다. 기본값은 500입니다.
- Windows에서 값은 큐 관리자를 작성하는 사용자의 기본 CODEPAGE입니다.
- IBM i에서 값은 큐 관리자를 처음 작성할 때 환경에 설정된 항목입니다.
- AIX and Linux에서 값은 큐 관리자를 작성하는 사용자의 로케일에 대한 기본 CODESET입니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_CODED\_CHAR\_SET\_ID 선택자를 사용하십시오.

### **CommandEvent(MQLONG)**

이는 다음과 같이 명령 이벤트가 생성되는지 여부를 지정합니다.

#### **MQEVR\_DISABLED**

명령 이벤트를 생성하지 마십시오. 기본값입니다.

#### **MQEVR\_ENABLED**

명령 이벤트를 생성하십시오.

#### **MQEVR\_NO\_DISPLAY**

MQINQ가 아닌 모든 성공적인 명령에 대해 명령 이벤트가 생성됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_COMMAND\_EVENT 선택자를 사용하십시오.

### **CommandInputQName(MQCHAR48)**

이는 로컬 큐 관리자에서 정의된 명령 입력 큐의 이름입니다. 이는 수행할 권한이 부여된 경우 사용자가 명령을 전송할 수 있는 큐입니다. 큐의 이름은 환경에 따라 다릅니다.

- z/OS에서 큐의 이름은 SYSTEM.COMMAND.INPUT; MQSC 및 PCF 명령을 보낼 수 있습니다. MQSC 명령에 대한 세부사항은 [MQSC 명령을 참조](#)하고 PCF 명령에 대한 세부사항은 [프로그램 가능한 명령 형식의 정의를 참조](#) 하십시오.

- 다른 모든 환경에서 큐의 이름은 SYSTEM.ADMIN.COMMAND.QUEUE이고 PCF 명령만 전송할 수 있습니다. 그러나 MQSC 명령이 유형 MQCMD\_ESCAPE의 PCF 명령 내에 있는 경우 MQSC 명령을 이 큐에 전송할 수 있습니다. 이스케이프 명령에 대한 정보는 [이스케이프를 참조하십시오](#).

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_COMMAND\_INPUT\_Q\_NAME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다.

### **CommandLevel(MQLONG)**

**참고:** 서버 및 클라이언트를 포함하여 모든 IBM MQ 구성요소에 대한 HP-UX 운영 체제 지원이 IBM MQ 9.1에서 제거되었습니다.

이 매개변수는 큐 관리자가 지원하는 시스템 제어 명령의 레벨을 표시합니다. 다음 값 중 하나일 수 있습니다.

#### **MQCMDL\_LEVEL\_800**

시스템 제어 명령의 레벨 800입니다.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 8.0
- IBM MQ for IBM i 8.0
- IBM MQ for Linux 8.0
- IBM MQ for Windows 8.0
- IBM MQ for z/OS 8.0

#### **MQCMDL\_LEVEL\_801**

시스템 제어 명령의 레벨 801입니다.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 8.0.0 Fix Pack 2
- IBM MQ for HP-UX 8.0.0 Fix Pack 2
- IBM MQ for IBM i 8.0.0 Fix Pack 2
- IBM MQ for Linux 8.0.0 Fix Pack 2

#### **MQCMDL\_LEVEL\_802**

시스템 제어 명령의 레벨 802입니다.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 8.0.0 Fix Pack 3
- IBM MQ for IBM i 8.0.0 Fix Pack 3
- IBM MQ for Linux 8.0.0 Fix Pack 3
- IBM MQ for Windows 8.0.0 Fix Pack 3

#### **MQCMDL\_LEVEL\_900**

시스템 제어 명령의 레벨 900입니다.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.0
- IBM MQ for IBM i 9.0
- IBM MQ for Linux 9.0
- IBM MQ for Windows 9.0
- IBM MQ for z/OS 9.0

#### **MQCMDL\_LEVEL\_901**

시스템 제어 명령의 레벨 901입니다.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for Linux 9.0.1



- IBM MQ for Windows 9.0.1
- IBM MQ for z/OS 9.0.1

#### **MQCMDL\_LEVEL\_902**

시스템 제어 명령의 레벨 902입니다.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for Linux 9.0.2
- IBM MQ for Windows 9.0.2
- IBM MQ for z/OS 9.0.2

#### **MQCMDL\_LEVEL\_903**

시스템 제어 명령의 레벨 903

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for Linux 9.0.3
- IBM MQ for Windows 9.0.3
- IBM MQ for z/OS 9.0.3

#### **MQCMDL\_LEVEL\_904**

시스템 제어 명령의 레벨 904

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.0.4
- IBM MQ for Linux 9.0.4
- IBM MQ for Windows 9.0.4
- IBM MQ for z/OS 9.0.4

#### **MQCMDL\_LEVEL\_905**

시스템 제어 명령의 레벨 905입니다.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.0.5
- IBM MQ for Linux 9.0.5
- IBM MQ for Windows 9.0.5
- IBM MQ for z/OS 9.0.5

#### **MQCMDL\_LEVEL\_910**

시스템 제어 명령의 레벨 910입니다.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.1
- IBM MQ for IBM i 9.1
- IBM MQ for Linux 9.1
- IBM MQ for Windows 9.1
- IBM MQ for z/OS 9.1

#### **MQCMDL\_LEVEL\_911**

시스템 제어 명령의 레벨 911입니다.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.1.1
- IBM MQ for Linux 9.1.1
- IBM MQ for Windows 9.1.1
- IBM MQ for z/OS 9.1.1

**MQCMDL\_LEVEL\_912**

시스템 제어 명령의 레벨 912입니다.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.1.2
- IBM MQ for Linux 9.1.2
- IBM MQ for Windows 9.1.2
- IBM MQ for z/OS 9.1.2

**MQCMDL\_LEVEL\_913**

시스템 제어 명령의 레벨 913입니다.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.1.3
- IBM MQ for Linux 9.1.3
- IBM MQ for Windows 9.1.3
- IBM MQ for z/OS 9.1.3

**MQCMDL\_LEVEL\_914**

시스템 제어 명령들의 레벨 (914).

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.1.4
- IBM MQ for Linux 9.1.4
- IBM MQ for Windows 9.1.4
- IBM MQ for z/OS 9.1.4

**MQCMDL\_LEVEL\_915**

시스템 제어 명령의 레벨 915.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.1.5
- IBM MQ for Linux 9.1.5
- IBM MQ for Windows 9.1.5
- IBM MQ for z/OS 9.1.5

**MQCMDL\_LEVEL\_910**

시스템 제어 명령의 레벨 910입니다.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.1
- IBM MQ for IBM i 9.1
- IBM MQ for Linux 9.1
- IBM MQ for Windows 9.1
- IBM MQ for z/OS 9.1

**MQCMDL\_LEVEL\_920**

시스템 제어 명령의 레벨 920.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.2
- IBM MQ for IBM i 9.2
- IBM MQ for Linux 9.2
- IBM MQ for Windows 9.2

- IBM MQ for z/OS 9.2

#### **MQCMDL\_LEVEL\_921**

시스템 제어 명령의 레벨 921.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.2.1
- IBM MQ for Linux 9.2.1
- IBM MQ for Windows 9.2.1
- IBM MQ for z/OS 9.2.1

#### **MQCMDL\_LEVEL\_922**

시스템 제어 명령의 레벨 922.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.2.2
- IBM MQ for Linux 9.2.2
- IBM MQ for Windows 9.2.2
- IBM MQ for z/OS 9.2.2

#### **MQCMDL\_LEVEL\_923**

시스템 제어 명령의 레벨 923.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.2.3
- IBM MQ for Linux 9.2.3
- IBM MQ for Windows 9.2.3
- IBM MQ for z/OS 9.2.3

#### **MQCMDL\_LEVEL\_924**

시스템 제어 명령의 레벨 924.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.2.4
- IBM MQ for Linux 9.2.4
- IBM MQ for Windows 9.2.4
- IBM MQ for z/OS 9.2.4

#### **MQCMDL\_LEVEL\_925**

시스템 제어 명령의 레벨 925.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.2.5
- IBM MQ for Linux 9.2.5
- IBM MQ for Windows 9.2.5
- IBM MQ for z/OS 9.2.5

#### **MQCMDL\_LEVEL\_930**

시스템 제어 명령의 레벨 930.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.3
- IBM MQ for IBM i 9.3
- IBM MQ for Linux 9.3
- IBM MQ for Windows 9.3

- IBM MQ for z/OS 9.3

#### **MQCMDL\_LEVEL\_931**

시스템 제어 명령의 레벨 931.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.3.1
- IBM MQ for Linux 9.3.1
- IBM MQ for Windows 9.3.1
- IBM MQ for z/OS 9.3.1

#### **MQCMDL\_LEVEL\_932**

시스템 제어 명령의 레벨 932.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.3.2
- IBM MQ for Linux 9.3.2
- IBM MQ for Windows 9.3.2
- IBM MQ for z/OS 9.3.2

#### **MQCMDL\_LEVEL\_933**

시스템 제어 명령의 레벨 933.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.3.3
- IBM MQ for Linux 9.3.3
- IBM MQ for Windows 9.3.3
- IBM MQ for z/OS 9.3.3

#### **MQCMDL\_LEVEL\_934**

시스템 제어 명령의 레벨 934.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.3.4
- IBM MQ for Linux 9.3.4
- IBM MQ for Windows 9.3.4
- IBM MQ for z/OS 9.3.4

#### **MQCMDL\_LEVEL\_935**

시스템 제어 명령의 레벨 935.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.3.5
- IBM MQ for Linux 9.3.5
- IBM MQ for Windows 9.3.5
- IBM MQ for z/OS 9.3.5

#### **MQCMDL\_LEVEL\_940**

시스템 제어 명령의 레벨 940.

다음 버전에서 이 값을 리턴합니다.

- IBM MQ for AIX 9.4.0
- IBM MQ for Linux 9.4.0
- IBM MQ for Windows 9.4.0
- IBM MQ for z/OS 9.4.0

**CommandLevel** 속성의 특정 값에 해당하는 시스템 제어 명령 세트가 **Platform** 속성 값에 따라 다릅니다. 둘 다 지원되는 시스템 제어 명령을 결정하는 데 사용해야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_COMMAND\_LEVEL 선택자를 사용하십시오.

### **CommandServerControl(MQLONG)**

큐 관리자가 시작될 때 명령 서버가 시작되는지 여부를 지정합니다.

가능한 값은 다음 값 중 하나입니다.

#### **MQSVC\_CONTROL\_MANUAL**

명령 서버가 자동으로 시작되지 않습니다.

#### **MQSVC\_CONTROL\_Q\_MGR**

큐 관리자가 시작될 때 명령 서버가 자동으로 시작됩니다.

이 속성은 z/OS에서 지원되지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_CMD\_SERVER\_CONTROL 선택자를 사용하십시오.

### **ConfigurationEvent(MQLONG)**

구성 이벤트 생성 여부를 제어합니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_CONFIGURATION\_EVENT 선택자를 사용하십시오.

가능한 값은 다음 값 중 하나입니다.

#### **MQEVR\_DISABLED**

이벤트 보고를 사용하지 않습니다.

#### **MQEVR\_ENABLED**

이벤트 보고를 사용합니다.

### **Multi CurrentQFileSize(MQLONG)**

큐 파일의 현재 크기(메가바이트)가 가장 가까운 메가바이트로 반올림됩니다.

표 558. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

이 큐 상태 속성 값은 현재 큐가 어떤 크기이던 간에 가장 가까운 메가바이트로 증가될 수 있습니다. 기본 속성이 있는 새 큐의 경우, **CurrentQFileSize**의 값은 1입니다.

이 속성의 최대 값은 99,999,9999MB이고 이 속성에 대한 기본값은 없습니다.

### **Multi CurrentMaxQFileSize(MQLONG)**

큐에서 사용 중인 현재 블록 크기가 지정된 경우 가장 가까운 메가바이트로 증가할 수 있는 큐 파일의 현재 최대 크기를 표시합니다.

표 559. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

이 필드의 사용은 두 가지입니다.

- **MaxQFileSize**을(를) 현재 블록 크기의 기본값으로 설정하는 경우, **CurrentMaxQFileSize**은(는) 기본값이 동일한 실제 값을 표시합니다.
- **CurrentMaxQFileSize**이(가) **MaxQFileSize**과(와) 일치하지 않는 경우 더 큰 입도를 채택하려면 큐가 드레인되어야 합니다.

**참고:** 큐 파일의 크기 및 블록 크기와 단위 변경에 대한 자세한 정보는 [IBM MQ 큐 파일 수정](#)을 참조하십시오.

이 속성의 최대 값은 99,999,9999MB이고 기본값이 없습니다. 값은 현재 설정된 최대 값이 무엇이든 상관없습니다. 기본 속성을 가지는 새 큐의 경우, **CurrentMaxQFileSize**의 값은 2,088,960MB입니다.

### **DeadLetterQName(MQCHAR48)**

이는 데드 레터(미배달 메시지) 큐로 로컬 큐 관리자에서 정의된 큐의 이름입니다. 메시지를 올바른 목적지로 라우트할 수 없는 경우 메시지를 이 큐로 송신합니다.

예를 들어, 다음의 경우에 메시지가 이 큐에 넣어집니다.

- 큐 관리자에 아직 정의되지 않은 큐로 예정된 메시지가 이 큐 관리자에 도달한 경우
- 메시지가 큐 관리자에 도달했으나, 그 메시지를 수신하기로 예정된 큐가 다음과 같은 이유로 수신할 수 없는 경우
  - 큐가 가득 참
  - 넣기 요청이 금지됨
  - 송신 노드에는 메시지를 큐에 넣을 수 있는 권한이 없습니다.

애플리케이션도 메시지를 데드-레터 큐에 넣을 수 있습니다.

보고 메시지는 일반 메시지와 동일한 방법으로 처리됩니다. 보고 메시지를 해당 목적지 큐에 전달할 수 없는 경우 (일반적으로 원래 메시지의 메시지 디스크립터에서 *ReplyToQ* 필드가 지정하는 큐) 보고 메시지는 데드 레터(미배달 메시지) 큐에 배치됩니다.

**참고:** 만기 시간이 지난 메시지는(MQMD - 만기 필드 참조) 제거되면 이 큐에 전송되지 **않습니다**. 그러나 만기 보고 메시지(MQRO\_EXPIRATION)는 여전히 생성되며 전송 애플리케이션이 요청하면 *ReplyToQ* 큐에 전송됩니다.

넣기 요청을 발행한 애플리케이션에 MQPUT 또는 MQPUT1 호출로 리턴된 이유 코드를 사용하여 문제점을 동시에 알린 경우(예를 들어, 넣기 요청이 금지된 로컬 큐에서 메시지 넣기) 데드 레터(미배달 메시지) 큐에 메시지를 넣지 않습니다.

때에 따라 데드 레터(미배달 메시지) 큐의 메시지는 애플리케이션 메시지 데이터에 MQDLH 구조가 접두부로 붙습니다. 이 구조에는 메시지를 데드 레터(미배달 메시지) 큐에 넣은 이유를 표시하는 추가 정보가 있습니다. 이 구조의 자세한 정보는 346 페이지의 『MQDLH - 데드 레터 헤더』의 내용을 참조하십시오.

이 큐는 **Usage**속성이 MQUS\_NORMAL인 로컬 큐여야 합니다.

큐 관리자가 데드 레터(미배달 메시지) 큐를 지원하지 않거나 하나가 정의되지 않은 경우 이름이 모두 공백입니다. 모든 IBM MQ 큐 관리자는 데드 레터(미배달 메시지) 큐를 지원하지만 기본적으로 정의되지 않습니다.

데드 레터(미배달 메시지) 큐가 정의되지 않았거나 가득 찼거나 다른 이유로 인해 사용 불가능한 경우 메시지 채널 에이전트가 전송한 메시지를 전송 큐에서 대신 보유합니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_DEAD\_LETTER\_Q\_NAME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다.

### **DefClusterXmitQueueType(MQLONG)**

DefClusterXmitQueueType 속성은 클러스터-송신자 채널이 기본적으로 메시지를 가져오고 클러스터-수신자 채널로 메시지를 보내기 위해 선택하는 전송 큐를 제어합니다.

**DefClusterXmitQueueType**의 값은 MQCLXQ\_SCTQ 또는 MQCLXQ\_CHANNEL입니다.

#### **MQCLXQ\_SCTQ**

모든 클러스터 송신자 채널은 SYSTEM.CLUSTER.TRANSMIT.QUEUE에서 메시지를 전송합니다. 전송 큐에 있는 메시지의 *correlID*가 메시지의 목적지가 될 클러스터 송신자 채널을 식별합니다.

SCTQ 는 큐 관리자가 정의될 때 설정됩니다.

#### **MQCLXQ\_CHANNEL**

각 클러스터 송신자 채널이 다른 전송 큐에서 메시지를 보냅니다. 각 전송 큐는 모델 큐 SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE에서 영구 동적 큐로 작성됩니다.

큐 관리자 속성 DefClusterXmitQueueType이 CHANNEL로 설정된 경우, 기본 구성이 개별 클러스터 전송 큐와 연관된 클러스터 송신자 채널로 변경됩니다. 전송 큐는 모델 큐 SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE에서 작성된 영구적 동적 큐입니다. 각 전송 큐는 하나의 클러스터 송신자 채널과 연관됩니다. 하나의 클러스터 송신자 채널이 하나의 클러스터 전송 큐를 제공할 경우 전송 큐에는 한 클러스터의 한 큐 관리자에 대한 메시지만 포함됩니다. 클러스터에 있는 개별 큐 관리자가 하나의 클러스터 큐만 포함하도록 클러스터를 구성할 수 있습니다. 이 경우 큐 관리자에서 각 클러스터 큐로 전달되는 메시지 트래픽은 메시지에서 다른 큐로 개별적으로 송신됩니다.

값을 조회하려면 MQINQ를 호출하거나 MQIA\_DEF\_CLUSTER\_XMIT\_Q\_TYPE 선택자를 설정하여 큐 관리자 조회(MQCMD\_INQUIRE\_Q\_MGR) PCF 명령을 전송하십시오. 값을 변경하려면 MQIA\_DEF\_CLUSTER\_XMIT\_Q\_TYPE 선택자를 설정하여 큐 관리자 변경(MQCMD\_CHANGE\_Q\_MGR) PCF 명령을 전송하십시오.

#### 관련 참조

큐 관리자 변경

큐 관리자 조회

680 페이지의 『MQINQ - 오브젝트 속성 조회』

MQINQ 호출은 오브젝트의 속성이 포함된 문자열의 세트 및 정수의 배열을 리턴합니다.

### DefXmitQName(MQCHAR48)

사용하는 전송 큐에 대한 표시가 없는 경우, 리모트 큐 관리자에 메시지를 전송하는 데 사용되는 전송 큐의 이름입니다.

기본 전송 큐 이름이 없으면 이름이 완전히 공백입니다. 이 속성의 초기값은 공백입니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_DEF\_XMIT\_Q\_NAME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다.

### DistLists(MQLONG)

로컬 큐 관리자가 MQPUT 및 MQPUT1 호출에서 분배 목록을 지원하는지 여부를 표시합니다. 다음 값 중 하나입니다.

#### MQDL\_SUPPORTED

분배 목록이 지원됩니다.

#### MQDL\_NOT\_SUPPORTED

분배 목록이 지원되지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_DIST\_LISTS 선택자를 사용하십시오.

### z/OS DNSGroup(MQCHAR18)

이 매개변수는 더 이상 사용되지 않습니다. IBM MQ 8.0에서 변경된 사항을 참조하십시오.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_DNS\_GROUP 선택자를 사용하십시오. 이 속성의 길이는 MQ\_DNS\_GROUP\_NAME\_LENGTH로 지정됩니다.

### z/OS DNSWLM(MQLONG)

이 매개변수는 더 이상 사용되지 않습니다. IBM MQ 8.0에서 변경된 사항을 참조하십시오.

값은 다음 중 하나입니다.

#### MQDNSWLM\_YES

이 값은 이전 릴리스에서 마이그레이션된 큐 관리자에 표시될 수 있습니다. 값이 무시됩니다.

#### MQDNSWLM\_NO

큐 관리자에서 지원하는 유일한 값입니다.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_DNS\_WLM 선택자를 사용하십시오.


### **ExpiryInterval(MQLONG)**

이는 큐 관리자가 만기된 메시지를 찾기 위해 큐를 검색하는 빈도를 표시합니다. 이는 1 - 99,999,999 범위의 시간 간격(초) 또는 다음 특수 값입니다.

#### **MQEXPI\_OFF**

큐 관리자는 만기된 메시지를 찾는 큐를 스캔하지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_EXPIRY\_INTERVAL 선택자를 사용하십시오.

 이 속성은 z/OS에서만 지원됩니다.

### **IGQPutAuthority(MQLONG)**

이 속성은 로컬 큐 관리자가 큐 공유 그룹의 멤버인 경우에만 적용됩니다. 이는 로컬 그룹 내 큐잉 에이전트(IGQ 에이전트)가 공유 전송 큐에서 메시지를 제거하고 로컬 큐에서 메시지를 배치할 때 수행된 권한 검사의 유형을 표시합니다. 값은 다음 중 하나입니다.

#### **MQIGQPA\_DEFAULT**

권한에 대해 확인된 사용자 ID는 메시지가 공유 전송 큐에 있을 때 메시지와 연관된 별도의 MQMD의 *UserIdentifier* 필드 값입니다. 이는 공유 전송 큐에 메시지를 배치한 프로그램의 사용자 ID이며 일반적으로 리모트 큐 관리자가 실행 중인 사용자 ID와 동일합니다.

RESLEVEL 프로파일이 둘 이상의 사용자 ID가 검사됨을 나타내는 경우 로컬 IGQ 에이전트의 사용자 ID(*IGQUserId*)도 검사됩니다.

#### **MQIGQPA\_CONTEXT**

권한에 대해 확인된 사용자 ID는 메시지가 공유 전송 큐에 있을 때 메시지와 연관된 별도의 MQMD의 *UserIdentifier* 필드 값입니다. 이는 공유 전송 큐에 메시지를 배치한 프로그램의 사용자 ID이며 일반적으로 리모트 큐 관리자가 실행 중인 사용자 ID와 동일합니다.

RESLEVEL 프로파일이 둘 이상의 사용자 ID가 검사됨을 나타내는 경우 로컬 IGQ 에이전트(*IGQUserId*)의 사용자 ID 및 임베드된 MQMD의 *UserIdentifier* 필드 값도 검사됩니다. 후자의 사용자 ID는 일반적으로 메시지를 생성한 애플리케이션의 사용자 ID입니다.

#### **MQIGQPA\_ONLY\_IGQ**

권한에 대해 검사된 사용자 ID는 로컬 IGQ 에이전트(*IGQUserId*)의 사용자 ID입니다.


RESLEVEL 프로파일이 둘 이상의 사용자 ID가 검사됨을 나타내는 경우 이 사용자 ID는 모든 검사에 사용됩니다.

#### **MQIGQPA\_ALTERNATE\_OR\_IGQ**

권한에 대해 검사된 사용자 ID는 로컬 IGQ 에이전트(*IGQUserId*)의 사용자 ID입니다.

RESLEVEL 프로파일이 둘 이상의 사용자 ID가 검사됨을 나타내는 경우 임베드된 MQMD에 있는 *UserIdentifier* 필드의 값도 검사됩니다. 이 사용자 ID는 일반적으로 메시지를 생성한 애플리케이션의 사용자 ID입니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_IGQ\_PUT\_AUTHORITY 선택자를 사용하십시오.

 이 속성은 z/OS에서만 지원됩니다.

### **IGQUserId(MQLONG)**

이 속성은 로컬 큐 관리자가 큐 공유 그룹의 멤버인 경우에만 적용 가능합니다. 이는 로컬 그룹 내 큐잉 에이전트(IGQ 에이전트)와 연관된 사용자 ID를 지정합니다. 이 ID는 IGQ 에이전트가 로컬 큐에 메시지를 넣을 때 권한에 대해 검사할 수 있는 사용자 ID 중 하나입니다. 확인된 실제 사용자 ID는 **IGQPutAuthority** 속성의 설정 및 외부 보안 옵션에 따라 다릅니다.

*IGQUserId*가 공백인 경우 권한에 대해 기타 사용자 ID를 검사할 수 있어도 IGQ 에이전트와 연관된 사용자 ID가 없으며 해당 권한 검사가 수행되지 않습니다.



이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_IGQ\_USER\_ID 선택자를 사용하십시오. 이 속성의 길이는 MQ\_USER\_ID\_LENGTH로 지정됩니다.

**z/OS** 이 속성은 z/OS에서만 지원됩니다.

### **InhibitEvent(MQLONG)**

이는 금지(가져오기 금지 및 넣기 금지) 이벤트 생성 여부를 제어합니다. 값은 다음 중 하나입니다.

#### **MQEVR\_DISABLED**

이벤트 보고를 사용하지 않습니다.

#### **MQEVR\_ENABLED**

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_INHIBIT\_EVENT 선택자를 사용하십시오.

z/OS에서 이 속성 값을 판별하기 위해 MQINQ 호출을 사용할 수 없습니다.

### **z/OS IntraGroupQueuing(MQLONG)**

이 속성은 로컬 큐 관리자가 큐 공유 그룹의 멤버인 경우에만 적용됩니다. 이는 그룹 내 큐잉이 큐 공유 그룹에 사용 가능한지 여부를 표시합니다.

값은 다음 중 하나입니다.

#### **MQIGQ\_DISABLED**

큐 공유 그룹에 있는 다른 큐 관리자로 보낼 모든 메시지는 기존 채널을 사용하여 전송됩니다.

#### **MQIGQ\_ENABLED**

큐 공유 그룹의 기타 큐 관리자에 발신되는 메시지는 다음 조건이 충족되면 공유 전송 큐를 사용하여 전송됩니다.

- 메시지 데이터 및 전송 헤더의 길이는 63KB(64,512바이트)를 초과하지 않습니다.

전송 헤더에 MQXQH의 크기보다 좀 더 큰 공간을 할당하는 것이 좋습니다. 이러한 목적으로 상수 MQ\_MSG\_HEADER\_LENGTH가 제공됩니다.

이 조건이 충족되지 않으면 메시지는 기존 채널을 사용하여 전송됩니다.

**참고:** 그룹 내 큐잉을 사용하는 경우 공유 전송 큐를 사용하여 전송된 메시지 순서는 기존 채널을 사용하여 전송된 순서에 관하여 보존되지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_INTRA\_GROUP\_QUEUING 선택자를 사용하십시오.

### **IPAddressVersion(MQLONG)**

사용되는 IP 주소 버전(IPv4 또는 IPv6)을 지정합니다.

이 속성은 IPv4 및 IPv6 둘 다 실행하는 시스템과만 관련되고 다음 조건 중 하나가 true일 때 MQXPY\_TCP의 *TransportType*을 가지도록 정의된 채널에만 영향을 줍니다.

- 채널의 *ConnectionName*은 IPv4 및 IPv6 주소 모두에 대해 해석하는 호스트 이름이며 해당 **LocalAddress** 매개변수는 지정되지 않습니다.
- 채널의 *ConnectionName* 및 *LocalAddress*는 모두 IPv4 및 IPv6 주소 모두에 대해 해석하는 호스트 이름입니다.

가능한 값은 다음 값 중 하나입니다.

#### **MQIPADDR\_IPV4**

IPv4가 사용됩니다.

#### **MQIPADDR\_IPV6**

IPv6가 사용됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_IP\_ADDRESS\_VERSION 선택자를 사용하십시오.

## **z/OS** **ListenerTimer(MQLONG)**

이는 APPC 또는 TCP/IP 실패가 있는 경우 리스너를 다시 시작하기 위한 IBM MQ 시도 간의 시간 간격(초)입니다. 값은 5 - 9999 사이에 있어야 하며 기본값은 60입니다.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_LISTENER\_TIMER 선택자를 사용하십시오.

## **LocalEvent(MQLONG)**

이는 로컬 오류 이벤트가 생성되는지 여부를 제어합니다. 값은 다음 중 하나입니다.

### **MQEVR\_DISABLED**

이벤트 보고를 사용하지 않습니다.

### **MQEVR\_ENABLED**

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_LOCAL\_EVENT 선택자를 사용하십시오.

z/OS에서 이 속성 값을 판별하기 위해 MQINQ 호출을 사용할 수 없습니다.

## **LoggerEvent(MQLONG)**

복구 로그 이벤트가 생성되는지 여부를 제어합니다. 값은 다음 중 하나입니다.

### **MQEVR\_DISABLED**

이벤트 보고를 사용하지 않습니다.

### **MQEVR\_ENABLED**

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_LOGGER\_EVENT 선택자를 사용하십시오.

**Multi** 이 속성은 [멀티플랫폼](#)에서만 지원됩니다.

## **z/OS** **LUGroupName (MQCHAR8)**

이는 큐 공유 그룹의 인바운드 전송을 핸들링하는 LU 6.2 리스너의 일반 LU 이름입니다. 이 이름을 공백으로 두면 이 리스너를 사용할 수 없습니다.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_LU\_GROUP\_NAME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_LU\_NAME\_LENGTH로 지정됩니다.

## **z/OS** **LUName(MQCHAR8)**

이는 아웃바운드 LU 6.2 전송에 사용할 LU의 이름입니다. 이를 리스너가 인바운드 전송에 사용하는 동일한 LU로 설정합니다. 이 이름을 공백으로 두는 경우 APPC/MVS 기본 LU가 사용됩니다. 이는 변수이므로 LU6.2를 사용 중인 경우 항상 LUName을 설정합니다.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_LU\_NAME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_LU\_NAME\_LENGTH로 지정됩니다.

## **LU62ARMSuffix (MQCHAR2)**

이는 이 채널 시작기의 LUADD를 지정하는 SYS1.PARMLIB 멤버 APPCPMxx의 접미부입니다. ARM이 채널 시작기를 재시작할 때 z/OS 명령 SET APPC=xx가 발행됩니다. 이 이름을 공백으로 두는 경우 SET APPC=xx가 발행되지 않습니다.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_LU62\_ARM\_SUFFIX 선택자를 사용하십시오. 이 속성의 길이는 MQ\_ARM\_SUFFIX\_LENGTH에서 지정됩니다.

### **z/OS LU62Channels(MQLONG)**

이는 LU 6.2 전송 프로토콜을 사용하는 현재 실행할 수 있는 채널 또는 연결 가능한 클라이언트의 최대 수입니다.

이 값은 0 - 9999 범위에 있어야 하며 기본값은 200입니다. 0으로 설정하면 LU 6.2 전송 프로토콜이 사용되지 않습니다.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_LU62\_CHANNELS 선택자를 사용하십시오.

### **MaxActiveChannels(MQLONG)**

이 속성은 언제든지 활성화할 수 있는 최대 채널 수입니다.

기본값은 MaxChannels 속성에 지정된 값입니다.

z/OS의 경우 값의 범위는 1 - 9,999여야 합니다.

다른 모든 플랫폼에서 기본값은 999 999 999이며 이는 채널의 수가 무제한임을 의미하며 이 값을 한계값을 초과하는 실제 숫자로 설정할 수도 있습니다.

**MQ Appliance** IBM MQ Appliance에서 **MaxActiveChannels** 값을 변경해서는 안됩니다. 최대 클라이언트 채널 수를 제한하려면 SVRCONN 채널 정의에서 채널당 MAXINST 및 MAXINSTC 속성을 사용하여 각 SVRCONN 채널에 대한 한계를 정의하십시오. IBM MQ Appliance 문서에서 [IBM MQ Appliance](#) 을 참조하십시오.

**MaxActiveChannels** 매개변수는 z/OS 전용의 큐 관리자 속성입니다. 기타 플랫폼에서 **MaxActiveChannels**는 qm.ini 파일의 속성입니다. 다른 플랫폼에서 **MaxActiveChannels** 속성을 설정하는 방법에 대한 정보는 분산 큐잉을 위한 구성 파일 스탠자를 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_ACTIVE\_CHANNELS 선택자를 사용하십시오.

#### **관련 개념**

[채널 상태](#)

### **MaxChannels(MQLONG)**

이 속성은 연결된 클라이언트가 있는 서버 연결 채널을 포함하여 현재일 수 있는 채널의 최대 수입니다.

z/OS의 경우 값의 범위는 1 - 9,999여야 하며 기본값은 200입니다.

**MQ Appliance** IBM MQ Appliance의 경우 기본값은 999 999 999이며 이는 변경하지 않아야 합니다. 최대 클라이언트 채널 수를 제한하려면 SVRCONN 채널 정의에서 채널당 MAXINST 및 MAXINSTC 속성을 사용하여 각 SVRCONN 채널에 대한 한계를 정의하십시오. IBM MQ Appliance 문서에서 [IBM MQ Appliance](#) 을 참조하십시오.

네트워크로부터의 연결을 제공하는 시스템에는 기본 설정값보다 더 높은 값이 필요할 수 있습니다. 테스트 중에 시스템의 동작을 이상적으로 관찰하여 사용자의 환경에 적절한 값을 판별하십시오.

다른 모든 플랫폼에서 기본값은 100입니다. 필요한 경우 최대 현재 채널 수를 제한하기 위해 **MaxChannels**를 다른 값으로 설정할 수 있습니다.

**MaxChannels** 매개변수는 z/OS 전용의 큐 관리자 속성입니다. 기타 플랫폼에서 **MaxChannels**는 qm.ini 파일의 속성입니다. 다른 플랫폼에서 **MaxChannels** 속성을 설정하는 방법에 대한 정보는 분산 큐잉을 위한 구성 파일 스탠자를 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_MAX\_CHANNELS 선택자를 사용하십시오.

#### **관련 개념**

[채널 상태](#)

### **MaxHandles(MQLONG)**

하나의 태스크가 동시에 사용할 수 있는 열기 핸들의 최대 수입입니다. 단일 큐(또는 큐가 아닌 오브젝트)에 대해 성공한 각 MQOPEN 호출에 한 개의 핸들이 사용됩니다. 오브젝트를 닫으면 해당 핸들을 다시 사용할 수 있게 됩니다. 그러나, 분배 목록을 연 경우에는 분배 목록에 있는 각 큐에 별도의 핸들이 할당되므로 MQOPEN 호출이 분배 목록에 있는 큐의 수만큼 많은 핸들을 사용합니다. *MaxHandles*에 적절한 값을 결정할 때 이러한 점을 고려해야 합니다.

MQPUT1 호출이 처리의 일부로 MQOPEN 호출을 수행합니다. 따라서 MQPUT1은 MQOPEN만큼 많은 핸들을 사용하지만, 해당 핸들은 MQPUT1 호출의 지속 기간 동안에만 사용됩니다.

z/OS에서 태스크는 CICS 태스크, MVS 태스크 또는 IMS 종속 영역을 의미합니다.

값의 범위는 1 - 999 999 999입니다. 기본값은 환경에 따라 달라집니다.

- z/OS에서 기본값은 100입니다.
- 다른 모든 환경에서 기본값은 256입니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_MAX\_HANDLES 선택자를 사용하십시오.

### **MaxMsgLength(MQLONG)**

이는 큐 관리자가 핸들링할 수 있는 가장 긴 실제 메시지의 길이입니다. 그러나, **MaxMsgLength** 큐 관리자 속성을 **MaxMsgLength** 큐 속성에 상관없이 설정할 수 있기 때문에 큐에 넣는 가장 긴 물리적 메시지는 이 두 값 중 작은 값입니다.

큐 관리자가 세그먼트화를 지원하는 경우 애플리케이션은 두 **MaxMsgLength** 속성 중 더 작은 속성보다 긴 논리 메시지를 넣을 수 있지만 애플리케이션이 MQMD에서 MQMF\_SEGMENTATION\_ALLOWED 플래그를 지정하는 경우에만 가능합니다. 이 플래그가 지정되는 경우 논리 메시지의 길이에 대한 상한은 999 999 999바이트이지만 일반적으로 운영 체제 또는 애플리케이션이 실행 중인 환경에 의해 부과된 제한조건으로 인해 제한이 낮아집니다.

**MaxMsgLength** 속성에 대한 하한은 32KB(32 768바이트)입니다. 상한은 100MB(104 857 600바이트)입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_MAX\_MSG\_LENGTH 선택자를 사용하십시오.

### **MaxPriority(MQLONG)**

큐 관리자에 지원되는 최대 메시지 우선순위입니다. 우선순위의 범위는 0(최저) - *MaxPriority* (최고)입니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_MAX\_PRIORITY 선택자를 사용하십시오.

### **MaxPropertiesLength(MQLONG)**

이는 메시지로 플로우할 수 있는 특성의 크기를 제어하는 데 사용됩니다. 여기에는 특성 이름(바이트) 및 특성 크기 값(바이트)도 모두 포함됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_MAX\_PROPERTIES\_LENGTH 선택자를 사용하십시오.

### **Multi MaxQFileSize(MQLONG)**

큐 파일이 커질 수 있는 최대 크기입니다(메가바이트).

표 560. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

현재 큐 파일 크기보다 낮은 값으로 구성된 경우 큐 파일이 최대 크기를 초과할 수 있습니다. 그런 경우, 큐 파일이 더 이상 새로운 메시지를 승인하지 않지만, 기존 메시지가 이용될 수 있습니다. 큐 파일 크기가 구성된 값 아래로 떨어지면 새 메시지를 큐에 넣을 수 있습니다.

**참고:** 내부적으로 큐 관리자가 선택된 크기에 도달하기 위해 더 큰 블록 크기를 사용해야 하기 때문에 이 숫자는 큐에 구성된 속성의 값과 다를 수 있습니다. 큐 파일의 크기 및 블록 크기와 단위 변경에 대한 자세한 정보는 [IBM MQ 큐 파일 수정을 참조하십시오.](#)

이 속성이 증가하여 세분성이 변경되어야 하는 경우 경고 메시지 AMQ7493W 변경된 세분성이 AMQERR 로그에 기록됩니다. 이는 IBM MQ가 새 단위를 채택하기 위해 큐에 대한 계획을 비워야 한다는 표시를 제공합니다.

이 속성의 최대값은 267,386,880MB이고 기본값 및 마이그레이션된 값은 2,088,960MB이며, 이는 세분성이 512인 큐의 현재 최대값입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_MAX\_Q\_FILE\_SIZE 선택자를 사용하십시오.

### **MaxUncommittedMsgs(MQLONG)**

작업 단위 내에 있을 수 있는 커밋되지 않은 메시지의 최대 수입니다. 커밋되지 않은 메시지의 수는 현재 작업 단위가 시작한 이후 다음 항목의 합계입니다.

- MQPMO\_SYNCPOINT 옵션으로 애플리케이션이 넣는 메시지
- MQGMO\_SYNCPOINT 옵션으로 애플리케이션이 검색하는 메시지
- MQPMO\_SYNCPOINT 옵션으로 넣는 메시지에 대해 큐 관리자가 생성한 트리거 메시지 및 COA 보고 메시지
- MQGMO\_SYNCPOINT 옵션으로 검색되는 메시지에 대해 큐 관리자가 생성한 COD 보고 메시지

다음 메시지는 커밋되지 않음으로 계수되지 않습니다.

- 작업 단위 외부에서 애플리케이션이 넣거나 검색한 메시지
- 작업 단위 외부에서 메시지 넣기 또는 검색의 결과로 큐 메시지가 생성한 트리거 메시지 또는 COA/COD 보고 메시지
- 큐 관리자가 생성한 만기 보고 메시지(만기 보고 메시지를 야기하는 호출이 MQGMO\_SYNCPOINT를 지정했을 경우라도)
- 큐 관리자가 생성한 이벤트 메시지(이벤트 메시지를 야기하는 호출이 MQPMO\_SYNCPOINT 또는 MQGMO\_SYNCPOINT를 지정했을 경우라도)

#### **참고:**

1. 예외 보고서 메시지는 메시지 채널 에이전트(MCA) 또는 애플리케이션에서 생성하고 애플리케이션에서 넣거나 검색한 원래 메시지와 동일한 방법으로 처리됩니다.
2. 메시지 또는 세그먼트를 MQPMO\_SYNCPOINT 옵션을 사용하여 넣으면 미확약 메시지의 수는 실제 넣은 실제 메시지 수와 관계없이 하나씩 증가됩니다. (큐 관리자가 메시지 또는 세그먼트를 세분화해야 하는 경우 둘 이상의 실제 메시지가 발생할 수 있습니다.)
3. 분배 목록을 MQPMO\_SYNCPOINT 옵션을 사용하여 넣으면 미확약 메시지의 수는 생성된 하나의 각 실제 메시지에 대해 하나씩 증가됩니다. 이 수는 분배 목록에 있는 목적지의 수만큼 적거나 많습니다.

이 속성에 대한 하한은 1이고 상한은 999 999 999입니다. 기본값은 10000입니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_MAX\_UNCOMMITTED\_MSGS 선택자를 사용하십시오.

### **MQIAccounting(MQLONG)**

이는 MQI 데이터에 대한 계정 정보 콜렉션을 제어합니다.

값은 다음 중 하나입니다.

#### **MQMON\_ON**

API 계정 데이터를 수집하십시오.

#### **MQMON\_OFF**

API 계정 데이터를 수집하지 마십시오. 이는 기본값입니다.

큐 관리자가 ACCTCONO 속성을 ENABLED로 설정한 경우 이 값은 MQCNO 구조에서 옵션 필드를 사용하는 개별 연결에 대해 대체될 수 있습니다. 이 값에 대한 변경사항은 속성을 변경한 이후 발생하는 큐 관리자에 대한 연결에만 유효합니다.

이 속성은 멀티플랫폼에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_ACCOUNTING\_MQI 선택자를 사용하십시오.

### **MQIStatistics(MQLONG)**



이는 큐 관리자에 대한 통계 모니터링 정보 콜렉션을 제어합니다.

값은 다음 중 하나입니다.

#### **MQMON\_ON**

MQI 통계를 수집하십시오.

#### **MQMON\_OFF**

MQI 통계를 수집하지 마십시오. 이는 기본값입니다.

이 속성은 멀티플랫폼에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MMQIA\_STATISTICS\_MQI 선택자를 사용하십시오.

### **MsgMarkBrowseInterval(MQLONG)**

큐 관리자가 찾아보기 메시지에서 표시를 자동으로 제거할 수 있을 때까지의 시간 간격(밀리초)

이는 큐 관리자가 찾아보기 메시지에서 표시를 자동으로 제거할 수 있을 때 이후의 시간 간격(밀리초)입니다.

이 속성은 메시지 가져오기 옵션 MQGMO\_MARK\_BROWSE\_CO\_OP를 사용하여 MQGET에 대한 호출을 통해 찾아봄으로 표시된 메시지가 찾아봄 표시 상태를 유지할 것으로 예상되는 시간 간격에 대해 설명합니다.

큐 관리자는 공동 핸들링 세트에 대해 찾아봄으로 표시된 메시지가 이 예상 간격을 초과하여 찾아봄으로 계속 표시되면 이 표시를 자동으로 지울 수 있습니다.

이는 메시지 가져오기 옵션 MQGMO\_MARK\_BROWSE\_HANDLE를 사용하여 MQGET에 대한 호출을 통해 찾아봄으로 표시된 메시지의 상태에는 영향을 주지 않습니다.

최대값은 999 999 999이고 기본값은 5000입니다. *MsgMarkBrowseInterval*에 대한 특수 값 -1은 무제한 시간 간격을 나타냅니다.



**주의:** 이 값은 기본값 5000 미만일 수 없습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_MSG\_MARK\_BROWSE\_INTERVAL 선택자를 사용하십시오.

### **z/OS OutboundPortMax(MQLONG)**

이는 보내는 채널을 바인딩하는 데 사용될 포트 번호 중 OutboundPortMin 및 OutboundPortMax에 의해 정의된 범위에서 가장 높은 포트 번호입니다.

이 값은 범위 0 - 6553 범위의 정수이고 OutboundPortMin 값과 같거나 커야 합니다. 기본값은 0입니다.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_OUTBOUND\_PORT\_MAX 선택자를 사용하십시오.

### **z/OS OutboundPortMin(MQLONG)**

이는 보내는 채널을 바인딩하는 데 사용될 포트 번호 중 OutboundPortMin 및 OutboundPortMax에 의해 정의된 범위에서 가장 낮은 포트 번호입니다.

이 값은 범위 0 - 6553 범위의 정수이고 OutboundPortMax 값과 같거나 작아야 합니다. 기본값은 0입니다.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_OUTBOUND\_PORT\_MIN 선택자를 사용하십시오.

### **PerformanceEvent(MQLONG)**

이는 성능 관련 이벤트가 생성되는지 여부를 제어합니다. 다음 값 중 하나입니다.

#### **MQEVR\_DISABLED**

이벤트 보고를 사용하지 않습니다.

#### **MQEVR\_ENABLED**

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_PERFORMANCE\_EVENT 선택자를 사용하십시오.

### **Platform(MQLONG)**

이는 큐 관리자가 실행 중인 운영 체제를 표시합니다.

#### **MQPL\_AIX**

AIX(MQPL\_UNIX와 동일한 값).

#### **MQPL\_APPLIANCE**

IBM MQ Appliance

#### **MQPL\_MVS**

z/OS(MQPL\_ZOS와 동일한 값).

#### **MQPL\_OS390**

z/OS(MQPL\_ZOS와 동일한 값).

#### **MQPL\_OS400**

IBM i.

#### **MQPL\_UNIX**

UNIX.

#### **MQPL\_WINDOWS\_NT**

Windows 시스템.

#### **MQPL\_ZOS**

z/OS.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_PLATFORM 선택자를 사용하십시오.

### **PubSubNPInputMsg(MQLONG)**

미발송 입력 메시지를 제거 또는 보관할지 여부

값은 다음 중 하나입니다.

#### **MQUNDELIVERED\_DISCARD**

비지속 입력 메시지가 처리 불가능한 경우 제거될 수 있습니다.

이는 기본값입니다.

#### **MQUNDELIVERED\_KEEP**

비지속 입력 메시지가 처리 불가능한 경우 제거되지 않습니다. 이 경우 큐잉된 발행/구독 인터페이스를 처리하기 위해 적절한 간격으로 계속 재시도하고 후속 메시지는 처리하지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_PUBSUB\_NP\_MSG 선택자를 사용하십시오.

### **PubSubNPResponse(MQLONG)**

미발송 응답 메시지의 작동을 제어합니다.

값은 다음 중 하나입니다.

#### **MQUNDELIVERED\_NORMAL**

응답 큐에 배치할 수 없는 비지속적 응답을 데드-레터 큐에 넣으며 DLQ에 배치할 수 없는 경우에는 제거됩니다.

#### **MQUNDELIVERED\_SAFE**

응답 큐에 놓을 수 없는 비지속 응답을 데드-레터 큐에 넣습니다. 응답을 설정할 수 없고 DLQ에 배치할 수 없으면 큐잉된 발행/구독 인터페이스가 현재 작업을 롤백한 후 적절한 간격으로 재시도하고 후속 메시지는 처리하지 않습니다.

#### **MQUNDELIVERED\_DISCARD**

응답 큐에 배치되지 않은 비지속 응답은 제거됩니다.

이는 새 큐 관리자에 대한 기본값입니다.

#### **MQUNDELIVERED\_KEEP**

비지속 응답이 데드-레터 큐에 배치하지 않았거나 제거됩니다. 대신 큐잉된 발행/구독 인터페이스가 현재 작업을 백아웃한 다음 적당한 간격으로 재시도합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_PUBSUB\_NP\_RESP 선택자를 사용하십시오.

### 마이그레이션된 큐 관리자의 기본값.

큐 관리자가 IBM MQ V6.0에서 마이그레이션된 경우 이 속성의 초기값은 다음 표에 표시된 대로 마이그레이션 전의 *DiscardNonPersistentResponse* 및 *DLQNonPersistentResponse*의 값에 따라 달라집니다.

		DLQNonPersistentResponse		
		예	아니오	설정되지 않음
DiscardNonPersistentResponse	예	MQUNDELIVERED_NORMAL	MQUNDELIVERED_DISCARD	MQUNDELIVERED_NORMAL
	아니오	MQUNDELIVERED_SAFE	MQUNDELIVERED_KEEP	MQUNDELIVERED_SAFE
	설정되지 않음	If SyncPointPersistent = No, MQUNDELIVERED_SAFE else MQUNDELIVERED_NORMAL	If SyncPointPersistent = No, MQUNDELIVERED_KEEP else MQUNDELIVERED_DISCARD	If SyncPointPersistent = No, MQUNDELIVERED_SAFE else MQUNDELIVERED_NORMAL

### PubSubMaxMsgRetryCount(MQLONG)

동기점 아래의 실패한 명령 메시지를 처리할 때의 재시도의 횟수입니다.

값은 다음 중 하나입니다.

#### 0 - 999 999 999

기본값은 5입니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_PUBSUB\_MAXMSG\_RETRY\_COUNT 선택자를 사용하십시오.

### PubSubSyncPoint(MQLONG)

동기점 아래의 지속 메시지만 처리할지 또는 모든 메시지를 처리할지 여부입니다.

값은 다음 중 하나입니다.

#### MQSYNCPPOINT\_IFPER

이 값을 사용하면 큐잉된 발행/구독 인터페이스가 동기점 외부에서 비지속 메시지를 수신할 수 있습니다. 디먼이 동기점 외부의 발행물을 수신하는 경우 디먼이 동기점 외부로 알려진 발행물을 구독자에게 전달합니다.

이는 기본값입니다.

#### MQSYNCPPOINT\_YES

이 값을 사용하면 큐잉된 발행/구독 인터페이스가 동기점 아래에서 모든 메시지를 수신합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_PUBSUB\_SYNC\_PT 선택자를 사용하십시오.

### PubSubMode(MQLONG)

발행/구독 엔진 및 큐에 있는 발행/구독 인터페이스가 실행 중이므로 애플리케이션이 API(Application Programming Interface) 및 큐에 있는 발행/구독 인터페이스로 모니터링되는 큐를 사용하여 발행 또는 구독할 수 있는지 여부입니다.

값은 다음 중 하나입니다.

#### MQPSM\_COMPAT

발행/구독 엔진이 실행 중입니다. 따라서 API(Application Programming Interface)를 사용하여 발행/구독할 수 있습니다. 큐에 있는 발행/구독 인터페이스가 실행 중이 아니므로 큐에 있는 발행/구독 인터페이스로 모니터링하는 큐에 넣은 메시지가 적용되지 않습니다. 이 설정은 이 큐 관리자를 사용하는 WebSphere Message Broker V6 또는 이전 버전과의 호환성을 위해 사용하는데, 그 이유는 큐에 보관된 발행/구독 인터페이스가 일반적으로 읽는 동일한 큐를 읽어야 하기 때문입니다.

#### MQPSM\_DISABLED

발행/구독 엔진 및 큐 발행/구독 인터페이스가 실행 중이지 않습니다. 따라서 API(Application Programming Interface)를 사용하여 발행/구독할 수 없습니다. 큐된 발행/구독 인터페이스에서 모니터링하는 큐에 넣은 발행/구독 메시지가 처리되지 않습니다.



## MQPSM\_ENABLED

발행/구독 엔진 및 큐에 있는 발행/구독 인터페이스가 실행 중입니다. 따라서 API(Application Programming Interface) 및 큐에 있는 발행/구독 인터페이스에서 모니터링하는 큐를 사용하여 발행/구독할 수 있습니다. 이것이 큐 관리자의 초기 기본값입니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_PUBSUB\_MODE 선택자를 사용하십시오.

## QMGrDesc (MQCHAR64)

큐 관리자를 설명하는 주석에 이 필드를 사용하십시오. 이 필드의 내용은 큐 관리자에 중요하지 않습니다. 그러나 큐 관리자에서 필드에 표시될 수 있는 문자만 포함되어야 할 수도 있습니다. 널(null) 문자는 포함할 수 없지만, 필요한 경우 오른쪽으로 공백을 채워넣을 수 있습니다. DBCS 설치 시 이 필드는 DBCS 문자(최대 필드 길이 64비트에 따라)를 포함할 수 있습니다.

**참고:** 이 필드에 큐 관리자의 문자 세트에 없는 문자가 포함된 경우(**CodedCharSetId** 큐 관리자 속성에 정의된 대로), 이 필드가 다른 큐 관리자로 송신되면 해당 문자가 올바르게 않게 변환될 수 있습니다.

- z/OS에서 기본값은 제품 이름 및 버전 번호입니다.
- 다른 모든 환경에서 기본값은 공백입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_Q\_MGR\_DESC 선택자를 사용하십시오. 이 속성의 길이는 MQ\_Q\_MGR\_DESC\_LENGTH로 제공됩니다.

## QMGrIdentifier (MQCHAR48)

이 속성은 큐 관리자의 내부적으로 생성된 고유 이름입니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_Q\_MGR\_IDENTIFIER 선택자를 사용하십시오. 이 속성의 길이는 MQ\_Q\_MGR\_IDENTIFIER\_LENGTH로 제공됩니다.

이 속성은 다음 환경에서 지원됩니다.

-  AIX
-  IBM i
-  Linux
-  Windows
-  z/OS

및 해당 시스템에 연결된 IBM MQ 클라이언트.

## QMGrName (MQCHAR48)

로컬 큐 관리자의 이름, 즉 애플리케이션이 연결되는 큐 관리자의 이름입니다.

이름의 첫 12자는 고유 메시지 ID를 구성하는 데 사용됩니다(MQMD - MsgId 필드 참조). 따라서 메시지 ID가 큐 관리자 네트워크에서 고유하기 위해 상호 통신할 수 있는 큐 관리자 이름의 처음 12자가 서로 달라야 합니다.


z/OS에서 이름은 서브시스템 이름과 동일하며 공백이 아닌 4자로 제한됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_Q\_MGR\_NAME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_Q\_MGR\_NAME\_LENGTH에서 제공됩니다.

## QSGName(MQCHAR4)

이는 큐 관리자가 속한 큐 공유 그룹의 이름입니다. 로컬 큐 관리자가 큐 공유 그룹에 속하지 않은 경우 이름은 비어 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_QSG\_NAME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_QSG\_NAME\_LENGTH로 제공됩니다.

-  이 속성은 z/OS에서만 지원됩니다.

## **QueueAccounting(MQLONG)**

이는 큐에 대한 계정 정보 콜렉션을 제어합니다.

값은 다음 중 하나입니다.

### **MQMON\_NONE**

계정 속성 ACCTQ 큐의 설정에 관계없이 큐에 대한 계정 데이터를 수집하지 마십시오. 이는 기본값입니다.

### **MQMON\_OFF**

ACCTQ 큐 속성에 QMGR을 지정하는 큐에 대한 계정 데이터를 수집하지 마십시오.

### **MQMON\_ON**

ACCTQ 큐 속성에 QMGR을 지정하는 큐에 대한 계정 데이터를 수집하십시오.

이 값에 대한 변경사항은 속성을 변경한 이후 발생하는 큐 관리자에 대한 연결에만 유효합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_ACCOUNTING\_Q 선택자를 사용하십시오.

## **QueueMonitoring(MQLONG)**

큐의 온라인 모니터링에 대한 기본 설정을 지정합니다.

**QueueMonitoring** 큐 속성을 MQMON\_Q\_MGR로 설정하는 경우 이 속성은 채널에서 가정한 값을 지정합니다. 가능한 값은 다음과 같습니다.

### **MQMON\_OFF**

온라인 모니터링 데이터 콜렉션이 꺼집니다. 이것이 큐 관리자의 초기 기본값입니다.

### **MQMON\_NONE**

**QueueMonitoring** 속성의 설정에 관계없이 큐에 대한 온라인 모니터링 데이터 콜렉션이 꺼집니다.

### **MQMON\_LOW**

온라인 모니터링 데이터 콜렉션이 낮은 데이터 콜렉션 비율로 켜집니다.

### **MQMON\_MEDIUM**

온라인 모니터링 데이터 콜렉션이 보통의 데이터 콜렉션 비율로 켜집니다.

### **MQMON\_HIGH**

온라인 모니터링 데이터 콜렉션이 높은 데이터 콜렉션 비율로 켜집니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_MONITORING\_Q 선택자를 사용하십시오.

## **QueueStatistics(MQLONG)**

이는 큐에 대한 통계 데이터 콜렉션을 제어합니다.

다음 값 중 하나입니다.

### **MQMON\_NONE**

**QueueStatistics** 큐 속성의 설정에 관계없이 큐에 대한 큐 통계를 수집하지 마십시오. 이는 기본값입니다.

### **MQMON\_OFF**

**QueueStatistics** 큐 속성에서 큐 관리자를 지정하는 큐에 대한 통계 데이터를 수집하지 마십시오.

### **MQMON\_ON**

**QueueStatistics** 큐 속성에서 큐 관리자를 지정하는 큐에 대한 통계 데이터를 수집하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_STATISTICS\_Q 선택자를 사용하십시오.

## **z/OS ReceiveTimeout(MQLONG)**

이는 비활성 상태로 리턴하기 전에 TCP/IP 채널이 파트너로부터 데이터(하트비트 포함)를 수신하기 위해 대기하는 시간을 지정합니다. 이는 MQI 채널이 아닌 메시지 채널에만 적용됩니다.

ReceiveTimeout의 정확한 의미는 ReceiveTimeoutType에 지정된 값으로 대체됩니다. ReceiveTimeoutType은 다음 중 하나로 설정할 수 있습니다.

- MQRCVTIME\_EQUAL - 이 값은 채널이 대기하는 시간(초)입니다. 0 - 999999 범위의 값을 지정하십시오.
- MQRCVTIME\_ADD - 이 값은 조정된 HBINT에 추가할 시간(초)이며 채널이 대기하는 시간을 판별합니다. 1 - 999999 범위의 값을 지정하십시오.
- MQRCVTIME\_MULTIPLY - 이 값은 조정된 HBINT에 적용할 배수입니다. 0 또는 2 - 99 범위의 값을 지정하십시오.

기본값은 0입니다.

채널이 파트너로부터 데이터를 수신하기 위한 대기 시간을 초과하지 못하도록 ReceiveTimeoutType을 MQRCVTIME\_MULTIPLY 또는 MQRCVTIME\_EQUAL로 설정하십시오.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_RECEIVE\_TIMEOUT 선택자를 사용하십시오.

### **z/OS ReceiveTimeoutMin(MQLONG)**

이는 비활성 상태로 리턴하기 전에 TCP/IP 채널이 해당 파트너로부터 데이터(하트비트 포함)를 수신하기 위해 대기하는 최소 시간(초)입니다.

이는 MQI 채널이 아닌 메시지 채널에만 적용됩니다. 값은 0 - 999999 범위에 있어야 하며 기본값은 0입니다.

ReceiveTimeoutType을 사용하여 HBINT의 조정된 값에 상대적인 TCP/IP 채널 대기 시간을 계산하도록 지정하며 결과 값이 이 매개변수의 값보다 작으면 이 값이 대신 사용됩니다.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_RECEIVE\_TIMEOUT\_MIN 선택자를 사용하십시오.

### **z/OS ReceiveTimeoutType(MQLONG)**

이는 비활성 상태로 리턴하기 전에 TCP/IP 채널이 해당 파트너로부터 데이터(하트비트 포함)를 수신하기 위해 대기하는 시간을 정의하기 위해 ReceiveTimeout에 적용된 규정자입니다. 이는 MQI 채널이 아닌 메시지 채널에만 적용됩니다.

값은 다음 중 하나입니다.

#### **MQRCVTIME\_MULTIPLY**

ReceiveTimeout은 채널이 대기하는 시간을 판별하기 위해 조정된 HBINT 값에 적용할 배수입니다. 이는 기본값입니다.

#### **MQRCVTIME\_ADD**

ReceiveTimeout은 채널이 대기하는 시간을 판별하기 위해 조정된 HBINT 값에 추가할 값(초)입니다.

#### **MQRCVTIME\_EQUAL**

ReceiveTimeout은 채널이 대기하는 값(초)입니다.

채널이 파트너로부터 데이터를 수신하기 위한 대기 시간을 초과하지 못하도록 ReceiveTimeoutType을 MQRCVTIME\_MULTIPLY 또는 MQRCVTIME\_EQUAL로 설정하고 ReceiveTimeout을 0으로 설정하십시오.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_RECEIVE\_TIMEOUT\_TYPE 선택자를 사용하십시오.

### **RemoteEvent(MQLONG)**

이는 원격 오류 이벤트 생성 여부를 지정합니다. 다음 값 중 하나입니다.

#### **MQEVR\_DISABLED**

이벤트 보고를 사용하지 않습니다.

#### **MQEVR\_ENABLED**

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_REMOTE\_EVENT 선택자를 사용하십시오.

## **RepositoryName (MQCHAR48)**

이 큐 관리자가 저장소 관리자 서비스를 제공하는 클러스터의 이름입니다. 큐 관리자가 둘 이상의 클러스터에 이 서비스를 제공하면 *RepositoryNameList*는 클러스터를 식별하는 이름 목록 오브젝트의 이름을 지정하고, *RepositoryName*은 공백입니다. *RepositoryName* 및 *RepositoryNameList* 중 하나 이상은 공백이어야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_REPOSITORY\_NAME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_Q\_MGR\_NAME\_LENGTH에서 제공됩니다.

## **RepositoryNameList (MQCHAR48)**

이 큐 관리자가 저장소 관리자 서비스를 제공하는 클러스터의 이름을 포함하는 이름 목록 오브젝트의 이름입니다. 큐 관리자가 하나의 클러스터에만 이 서비스를 제공하는 경우, 이름 목록 오브젝트에 하나의 이름만 있습니다. 또는, *RepositoryName*을 사용하여 클러스터의 이름을 지정할 수 있는데, 이 경우 *RepositoryNameList*는 공백입니다. *RepositoryName* 및 *RepositoryNameList* 중 하나 이상은 공백이어야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_REPOSITORY\_NAMELIST 선택자를 사용하십시오. 이 속성의 길이는 MQ\_NAMELIST\_NAME\_LENGTH로 제공됩니다.

## **ScyCase(MQCHAR8)**

큐 관리자가 대소문자를 함께 사용하거나 대문자만 사용한 보안 프로파일 이름을 지원하는지 여부를 지정합니다.

값은 다음 중 하나입니다.


### **MQSCYC\_UPPER**

보안 프로파일 이름은 대문자여야 합니다.

### **MQSCYC\_MIXED**

보안 프로파일 이름에 대문자만 사용하거나 대소문자를 함께 사용할 수 있습니다.

보안 새로 고치기 명령이 지정된 *SecurityType* (MQSECTYPE\_CLASSES)과 함께 실행되는 경우 이 속성에 대한 변경이 적용됩니다.

 이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_SECURITY\_CASE 선택자를 사용하십시오.

## **SharedQMgrName(MQLONG)**

이는 *ObjectQmgrName*이 큐 공유 그룹에 있는 다른 큐 관리자의 속성일 때 *ObjectQmgrName*이 사용되거나 공유 큐를 위해 MQOPEN 호출에 로컬 큐 관리자로 처리되어야 할지 여부를 지정합니다.

가능한 값은 다음 값 중 하나입니다.

### **MQSQQM\_USE**

*ObjectQmgrName*이(가) 사용되고 적절한 전송 큐가 열립니다.

### **MQSQQM\_IGNORE**

대상 큐가 공유되는 경우 *ObjectQmgrName*은 동일한 큐 공유 그룹에 있는 큐 관리자의 속성이며 열기는 로컬로 수행됩니다.

이 속성은 z/OS에서만 유효합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_SHARED\_Q\_Q\_MGR\_NAME 선택자를 사용하십시오.

## **SPLCAP**

Advanced Message Security의 보안 기능이 큐 관리자에 사용 가능한지 여부를 표시합니다.

### **MQCAP\_SUPPORTED**

이는 AMS 컴포넌트가 큐 관리자가 실행되고 있는 설치를 위해 설치되는 경우 기본값입니다.

### **MQCAP\_NOT\_SUPPORTED**

## **SSLEvent(MQLONG)**

TLS 이벤트가 생성되는지 지정합니다.

다음 값 중 하나입니다.

### **MQEVR\_ENABLED**

다음과 같이 TLS 이벤트를 생성하십시오.

MQRC\_CHANNEL\_SSL\_ERROR

### **MQEVR\_DISABLED**

TLS 이벤트를 생성하지 마십시오. 이는 기본값입니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_SSL\_EVENT 선택자를 사용하십시오.

## **SSLFIPSRequired(MQLONG)**

**참고:** AIX, Linux, and Windows에서 IBM MQ 는 IBM Crypto for C (ICC) 암호화 모듈을 통해 FIPS 140-2준수를 제공합니다. 이 모듈의 인증서가 히스토리 상태로 이동되었습니다. 고객은 IBM Crypto for C (ICC) 인증서를 보고 NIST에서 제공하는 조언을 알고 있어야 합니다. 대체 FIPS 140-2모듈이 현재 진행 중이며 해당 상태는 프로세스 목록의 NIST CMVP 모듈에서 검색하여 볼 수 있습니다.

IBM MQ Operator 3.2.0 및 큐 관리자 컨테이너 이미지 9.4.0.0 이상은 UBI 9를 기반으로 합니다. FIPS 140-2 준수가 현재 보류 중이며 해당 상태는 프로세스 목록의 NIST CMVP 모듈에서 "Red Hat Enterprise Linux 9-OpenSSL FIPS 제공자" 를 검색하여 볼 수 있습니다.

이렇게 하면 암호화 하드웨어가 아니라 IBM MQ에서 암호화가 실행되는 경우에 FIPS 인증 알고리즘만 사용할지 여부를 지정할 수 있습니다. 암호화 하드웨어가 구성된 경우 사용된 암호화 모듈은 하드웨어 제품에서 제공된 모듈이며 사용 중인 하드웨어 제품에 따라 특정 레벨로 FIPS 인증된 것이거나 인증되지 않은 것일 수 있습니다.

값은 다음 중 하나입니다.

### **MQSSL\_FIPS\_NO**

사용 중인 플랫폼에서 지원되는 CipherSpec을 사용하십시오. 이 값이 기본값입니다.

### **MQSSL\_FIPS\_YES**

이 큐 관리자로부터(로)의 모든 SSL 연결에 허용된 CipherSpecs에서 FIPS 인증 암호화 알고리즘만 사용하십시오.

이 매개변수는 z/OS, AIX, Linux, and Windows 플랫폼에서만 유효합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_SSL\_FIPS\_REQUIRED 선택자를 사용하십시오.

### **관련 태스크**

[MQI 클라이언트에서 런타임 시 FIPS 인증 CipherSpec만 사용하도록 지정](#)

### **관련 참조**

[AIX, Linux, and Windows용 FIPS\(Federal Information Processing Standard\)](#)

## **SSLKeyResetCount(MQLONG)**

이는 통신을 시작하는 TLS 채널 메시지 채널 에이전트(MCA)가 채널의 암호화에 사용된 비밀 키를 재설정할 시기를 지정합니다.

값은 비밀 키가 재조정되기 전에 채널에서 송신하거나 수신한 암호화되지 않은 총 바이트 수를 나타냅니다. 바이트 수에는 MCA에서 전송된 제어 정보가 포함됩니다.

값은 0 - 999 999 999 범위의 숫자이며, 기본값은 0입니다. TLS 비밀 키 재설정 계수를 1B - 32KB 범위로 지정하는 경우 TLS 채널은 32KB의 비밀 키 재설정 계수를 사용합니다. 이것은 TLS 비밀 키 재설정 값이 작은 경우에 발생하는 초과 키 재설정의 처리 비용을 피하기 위함입니다.

이 비밀 키는 시작 채널 MCA가 송신 및 수신하는 암호화되지 않은 바이트의 총 수가 지정된 값을 초과하는 경우 재조정됩니다. 채널 하트비트가 사용으로 설정되는 경우 이 비밀 키는 데이터가 채널 하트비트 다음에 송신 또는 수신되기 전 또는 암호화되지 않은 바이트의 총 수가 지정된 값을 초과할 때(어는 것이든지 우선시함) 재조정됩니다.

재조정을 위해 송신 및 수신되는 바이트 수는 채널 MCA에서 송신 및 수신된 제어 정보를 포함하고 재조정이 발생할 때마다 재설정됩니다.

비밀 키가 재조정되지 않음을 표시하기 위해 0의 값을 사용하십시오.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_SSL\_RESET\_COUNT 선택자를 사용하십시오.

### **StartStopEvent(MQLONG)**

이는 시작 및 중지 이벤트 생성 여부를 지정합니다. 값은 다음 중 하나입니다.

#### **MQEVR\_DISABLED**

이벤트 보고를 사용하지 않습니다.

#### **MQEVR\_ENABLED**

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_START\_STOP\_EVENT 선택자를 사용하십시오.

### **StatisticsInterval(MQLONG)**

이는 통계 모니터링 데이터가 모니터링 큐에 기록되는 빈도(초)입니다.

값은 0 - 604800 범위의 정수이며 기본값은 1800(30분)입니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_STATISTICS\_INTERVAL 선택자를 사용하십시오.

### **SyncPoint(MQLONG)**

로컬 큐 관리자가 작업 단위 및 MQGET, MQPUT, MQPUT1 호출과의 동기점 조정을 지원하는지 여부입니다.

#### **MQSP\_AVAILABLE**

작업 단위 및 동기점 조정을 사용할 수 있습니다.

#### **MQSP\_NOT\_AVAILABLE**

작업 단위 및 동기점 조정이 사용 불가능합니다.

- z/OS에서 이 값은 리턴되지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_SYNCPOINT 선택자를 사용하십시오.

### **z/OS TCPChannels(MQLONG)**

이는 TCP/IP 전송 프로토콜을 사용하는 현재일 수 있는 채널 또는 연결할 수 있는 클라이언트의 최대 수입니다.

이 값은 0 - 9999 범위에 있어야 하며 기본값은 200입니다. 0을 지정하는 경우 TCP/IP는 사용되지 않습니다.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_TCP\_CHANNELS 선택자를 사용하십시오.

### **z/OS TCPKeepAlive(MQLONG)**

이는 연결의 다른 측 끝이 여전히 사용 가능한지 확인하기 위해 TCP KEEPALIVE를 사용할지 여부를 지정합니다. 사용 불가능한 경우 채널이 닫힙니다.

값은 다음 중 하나입니다.

#### **MQTCPKEEP\_YES**

TCP 프로파일 구성 데이터 세트에 지정된 대로 TCP KEEPALIVE를 사용하십시오. 채널 속성 KeepAliveInterval(KAINT)을 지정하는 경우 설정된 값이 사용됩니다.

#### **MQTCPKEEP\_NO**

TCP KEEPALIVE를 사용하지 마십시오. 이는 기본값입니다.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_TCP\_KEEP\_ALIVE 선택자를 사용하십시오.

### **z/OS TCPName(MQCHAR8)**



이는 TCPStackType의 값에 따라 사용할 유일한 또는 선호되는 TCP/IP 스택의 이름입니다. 이 매개변수는 CINET 다중 스택 환경에서만 적용 가능합니다. 기본값은 TCPIP입니다.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_TCP\_NAME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_TCP\_NAME\_LENGTH로 제공됩니다.

### **z/OS TCPStackType(MQLONG)**

이는 채널 시작기가 TCPName에 지정된 TCP/IP 스택만 사용할지 또는 선택된 TCP/IP 스택에 선택적으로 바인딩할지 여부를 지정합니다. 이 매개변수는 CINET 다중 스택 환경에서만 적용 가능합니다.

값은 다음 중 하나입니다.

#### **MQTCPSTACK\_SINGLE**

채널 시작기는 TCPName에 이름 지정된 TCP/IP 주소 공간만 사용할 수 있습니다. 이는 기본값입니다.

#### **MQTCPSTACK\_MULTIPLE**

채널 시작기는 사용 가능한 모든 TCP/IP 주소 공간을 사용할 수 있습니다. 채널 또는 리스너에 달리 지정되지 않은 경우 TCPName에 지정된 값이 기본값입니다.

이 속성은 z/OS에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_TCP\_STACK\_TYPE 선택자를 사용하십시오.

### **TraceRouteRecording(MQLONG)**

이는 추적 라우트 정보의 레코딩을 제어합니다.

값은 다음 중 하나입니다.

#### **MQRECORDING\_DISABLED**

라우트 추적 메시지에 대한 추가가 허용되지 않습니다.

#### **MQRECORDING\_Q**

고정된 이름 지정된 큐에 추적 라우트 메시지를 넣으십시오.

#### **MQRECORDING\_MSG**

메시지 자체를 사용하여 판별된 큐에 추적 라우트 메시지를 넣으십시오. (기본값)

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_TRACE\_ROUTE\_RECORDING 선택자를 사용하십시오.

### **TriggerInterval(MQLONG)**

트리거 메시지의 수를 제한하는 데 사용하는 시간 간격(밀리초)입니다. 이는 *TriggerType*이 MQTT\_FIRST일 때만 해당됩니다. 이 경우 적절한 메시지가 큐에 도착하고 이전에 큐가 비어 있었을 때에만 트리거 메시지가 일반적으로 생성됩니다. 하지만 특정 상황에서는 큐가 비어 있지 않은 경우에도 MQTT\_FIRST 트리거를 사용하여 추가 트리거 메시지를 생성할 수 있습니다. 이러한 추가 트리거 메시지는 각 *TriggerInterval* 밀리초보다 더 자주 생성되지 않습니다.

트리거링에 대한 자세한 정보는 트리거링 채널을 참조하십시오.

이 값은 0보다 작지 않고 999 999 999보다 크지 않습니다. 기본값은 999,999,999입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_TRIGGER\_INTERVAL 선택자를 사용하십시오.

### **TriggerInterval(MQLONG)**

트리거 메시지의 수를 제한하는 데 사용하는 시간 간격(밀리초)입니다. 이는 *TriggerType*이 MQTT\_FIRST일 때만 해당됩니다. 이 경우 적절한 메시지가 큐에 도착하고 이전에 큐가 비어 있었을 때에만 트리거 메시지가 일반적으로 생성됩니다. 하지만 특정 상황에서는 큐가 비어 있지 않은 경우에도 MQTT\_FIRST 트리거를 사용하여 추가 트리거 메시지를 생성할 수 있습니다. 이러한 추가 트리거 메시지는 각 *TriggerInterval* 밀리초보다 더 자주 생성되지 않습니다.

트리거링에 대한 자세한 정보는 트리거링 채널을 참조하십시오.

이 값은 0보다 작지 않고 999 999 999보다 크지 않습니다. 기본값은 999,999,999입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_TRIGGER\_INTERVAL 선택자를 사용하십시오.

### Version(MQCFST)

이는 VVRRMMFF로 IBM MQ 코드의 버전입니다. 여기서,

VV - 버전

RR - 릴리스

MM- 유지보수 레벨

FF- 수정사항 레벨

### XrCapability(MQLONG)

이 속성은 MQ Telemetry 명령이 큐 관리자에서 지원되는지 여부를 제어합니다.

값은 다음 중 하나입니다.

#### MQCAP\_SUPPORTED

MQ Telemetry 컴포넌트가 설치되고 텔레메트리 명령이 지원됩니다.

#### MQCAP\_NOT\_SUPPORTED

MQ Telemetry 컴포넌트가 설치되지 않습니다.

이 속성은 멀티플랫폼에서만 지원됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_XR\_CAPABILITY 선택자를 사용하십시오.

## 큐의 속성


큐 정의 유형에는 5가지가 있습니다. 일부 큐 속성은 모든 유형의 큐에 적용되고, 그 외 큐 속성은 특정 유형의 큐에만 적용됩니다.

## 큐 유형

큐 관리자는 다음 유형의 큐 정의를 지원합니다.

### 로컬 큐

로컬 큐에 메시지를 저장할 수 있습니다.

 z/OS에서 공유 또는 개인 큐를 작성할 수 있습니다.

프로그램이 연결된 큐 관리자가 큐를 소유하고 있는 경우 큐는 프로그램에 로컬로 표시됩니다. 메시지를 로컬 큐에서 가져오거나 로컬 큐에 넣을 수 있습니다.

큐 정의 오브젝트는 큐에 넣은 실제 메시지뿐만 아니라 큐 정의 정보도 포함하고 있습니다.

### 로컬 큐 관리자 큐

큐는 로컬 큐 관리자에 존재합니다.

 큐는 z/OS에서 개인 큐로 알려집니다.

### 공유 큐(z/OS 전용)

큐는 공유 저장소를 소유한 큐 공유 그룹에 속한 모든 큐 관리자에 액세스할 수 있는 공유 저장소에 존재합니다.

큐 공유 그룹 내의 큐 관리자에 연결된 애플리케이션이 이 유형의 큐에서 메시지를 배치하거나 큐로부터 메시지를 제거할 수 있습니다. 해당 큐는 사실상 로컬 큐와 동일합니다. **QType** 큐 속성의 값은

MQQT\_LOCAL입니다.

로컬 큐 관리자에 연결된 애플리케이션이 이 유형의 큐에 메시지를 넣거나 큐로부터 메시지를 제거할 수 있습니다. **QType** 큐 속성의 값은 MQQT\_LOCAL입니다.



## 클러스터 큐

정의된 큐 관리자의 클러스터 큐에 메시지를 저장할 수 있습니다. 클러스터 큐는 클러스터 큐 관리자에 의해 호스팅되며 클러스터의 다른 큐 관리자가 사용할 수 있는 큐입니다. **QType** 큐 속성의 값은 `MQQT_CLUSTER`입니다.

클러스터 큐 정의는 클러스터의 다른 큐 관리자에 통지됩니다. 다른 큐 관리자는 해당하는 리모트 큐 정의 없이도 클러스터 큐에 메시지를 넣을 수 있습니다. 클러스터 큐는 클러스터 이름 목록을 사용하여 둘 이상의 클러스터에 통지될 수 있습니다.

큐가 통지되면 클러스터의 큐 관리자가 해당 큐에 메시지를 넣을 수 있습니다. 메시지를 넣으려면 큐 관리자가 전체 저장소에서 큐가 호스팅되고 있는 위치를 찾아야 합니다. 그런 다음 메시지에 몇 가지 라우팅 정보를 추가하고 클러스터 전송 큐에 메시지를 넣습니다.

큐 관리자는 클러스터의 다른 큐 관리자에 대한 메시지를 다중 전송 큐에 저장할 수 있습니다. 서로 다른 두 가지 방법으로 다중 클러스터 전송 큐에서 메시지를 저장하도록 큐 관리자를 구성할 수 있습니다. 큐 관리자 속성 **DEFCLXQ**을(를) `CHANNEL`로 설정하면, 각 클러스터 송신자 채널에 대해 `SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE`에서 다른 클러스터 전송 큐가 자동으로 작성됩니다. 하나 이상의 클러스터 송신자 채널과 일치하도록 `CLCHNAME` 전송 큐 옵션을 설정할 경우 큐 관리자가 일치하는 채널에 대한 메시지를 해당 전송 큐에 저장할 수 있습니다.



**주의:** IBM WebSphere MQ 7.5보다 이전 버전의 제품에서 업그레이드된 큐 관리자와 함께 전용 `SYSTEM.CLUSTER.TRANSMIT.QUEUES` 을(를) 사용하는 경우, `SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE`에 **SHARE/NOSHARE** 옵션이 **SHARE**로 설정되어 있는지 확인하십시오.

클러스터 큐는 IBM MQ for z/OS에서 큐 공유 그룹의 멤버가 공유하는 큐일 수 있습니다.

## 리모트 큐

리모트 큐는 실제 큐가 아니며 이는 리모트 큐 관리자에 존재하는 큐의 로컬 정의입니다. 리모트 큐의 로컬 정의는 리모트 큐 관리자에 메시지를 라우팅하는 방법을 로컬 큐 관리자에 알려주는 정보를 포함합니다.

로컬 큐 관리자에 연결된 애플리케이션은 이 유형의 큐에 메시지를 배치할 수 있습니다. 메시지를 리모트 큐 관리자에 라우팅하기 위해 사용되는 로컬 전송 큐에 메시지가 배치됩니다. 애플리케이션이 리모트 큐로부터 메시지를 제거할 수 없습니다. **QType** 큐 속성의 값은 `MQQT_REMOTE`입니다.

또한 다음에 리모트 큐 정의를 사용할 수 있습니다.

### • 응답 큐 알리어스

이 경우 정의 이름은 응답 대상 큐의 이름입니다. 자세한 정보는 [응답 대상 큐 알리어스 및 클러스터를 참조](#)하십시오.

### • 큐 관리자 알리어스

이 경우 정의 이름은 큐의 이름이 아니라 큐 관리자의 알리어스입니다. 자세한 정보는 [큐 관리자 알리어스 및 클러스터](#)를 참조하십시오.

## 알리어스 큐

이는 실제 큐가 아닙니다. 로컬 큐, 공유 큐, 클러스터 큐 또는 리모트 큐의 대체 이름입니다. 알리어스가 해석되는 큐의 이름은 알리어스 큐의 정의 중 일부입니다.

로컬 큐 관리자에 연결된 애플리케이션은 이 유형의 큐에 메시지를 배치할 수 있습니다. 메시지는 알리어스가 해석하는 큐에 배치됩니다. 알리어스가 로컬 인스턴스가 있는 로컬 큐, 공유 큐 또는 클러스터 큐에 해석하는 경우 이 유형의 큐에서 메시지를 제거할 수 있습니다. **QType** 큐 속성의 값은 `MQQT_ALIAS`입니다.

## 모델 큐

이는 실제 큐가 아닙니다. 로컬 큐가 작성될 수 있는 큐 속성 세트입니다.

이 유형의 큐에는 메시지를 저장할 수 없습니다.

## 큐 한계

IBM MQ의 이전 릴리스에서 사용된 2TB이상의 기본 한계를 실질적으로 지원하는 큐를 구성하고 모니터링할 수 있습니다. 큐 파일이 커질 수 있는 크기를 줄이는 옵션도 있습니다.

큐를 구성할 수 있도록 하기 위해 로컬 및 모델 큐에서 **MAXFSIZE** 속성을 사용하고 큐를 모니터링하기 위해 큐 상태 속성 **CURFSIZE** 및 **CURMAXFS**을(를) 사용할 수 있습니다.

자세한 정보는 [IBM MQ 큐 파일 수정을 참조하십시오](#).

## 큐 속성

일부 큐 속성은 모든 유형의 큐에 적용되고, 그 외 큐 속성은 특정 유형의 큐에만 적용됩니다. 속성이 적용되는 큐 유형은 [806 페이지의 표 561](#) 및 후속 표에 표시됩니다.

[806 페이지의 표 561](#)에서는 큐에 특정한 속성을 요약합니다. 속성은 알파벳 순서로 설명합니다.

**참고:** 이 절에 표시된 속성 이름은 MQINQ 및 MQSET 호출과 함께 사용되는 설명 이름입니다. 이 이름은 PCF 명령과 동일합니다. 속성을 정의, 대체 또는 표시하기 위해 MQSC 명령이 사용되는 경우, 대체 짧은 이름이 사용됩니다. 자세한 정보는 [MQSC 명령의 내용을 참조하십시오](#).

다음 표에서, 열은 다음과 같이 적용됩니다.

- 로컬 큐에 대한 열은 공유 큐에도 적용됩니다.
- 모델 큐에 대한 열은 모델 큐로부터 작성된 로컬 큐에 상속되는 속성을 나타냅니다.
- 클러스터 큐에 대한 열은 조회 용도로만 또는 조회와 출력 용도로 클러스터 큐를 열 때 조회할 수 있는 속성을 나타냅니다. 기타 속성이 조회되면 호출은 완료 코드 MQCC\_WARNING 및 이유 코드 MQRC\_SELECTOR\_NOT\_FOR\_TYPE(2068)을 리턴합니다.

조회는 물론 입력, 찾아보기 또는 설정을 한 번 이상 수행하기 위해 클러스터 큐를 연 경우, 로컬 큐에 대한 열이 대신 적용됩니다.

조회 단독 또는 조회 및 출력을 위해 클러스터 큐가 열리고 기본 큐 관리자 이름을 지정하는 경우 로컬 큐의 열이 대신 적용됩니다.

표 561. 큐의 속성						
속성	설명	로컬	모델	알리어스	원격	클러스터
<a href="#">AlterationDate</a>	정의가 마지막으로 변경된 날짜	X		X	X	
<a href="#">AlterationTime</a>	정의가 마지막으로 변경된 시간	X		X	X	
<a href="#">BackoutRequeueQName</a>	초과 백아웃 큐 이름	X	X			
<a href="#">BackoutThreshold</a>	백아웃 임계값	X	X			
<a href="#">BaseQName</a>	알리어스가 해석되는 큐 이름			X		
<a href="#">CFStrucName</a>	커플링 기능 구조 이름	X	X			
<a href="#">CLCHNAME</a>	클러스터 송신자 채널 이름	✓	✓			
<a href="#">ClusterName</a>	큐가 속한 클러스터의 이름	X		X	X	X
<a href="#">ClusterNamelist</a>	큐가 속한 클러스터의 이름을 포함하는 이름 목록 오브젝트의 이름	X		X	X	
<a href="#">CLWLQueuePriority</a>	클러스터 워크로드 큐 우선순위	X		X	X	X
<a href="#">CLWLQueueRank</a>	클러스터 워크로드 큐 순위	X		X	X	X
<a href="#">CLWLUseQ</a>	리모트 큐 사용	X				
<a href="#">CreationDate</a>	큐가 작성된 날짜	X				
<a href="#">CreationTime</a>	큐가 작성된 시간	X				
<a href="#">CurrentQDepth</a>	현재 큐 용량	X				
<a href="#">DefaultPutResponse</a>	기본 넣기 응답	✓	✓	✓	✓	
<a href="#">DefBind</a>	기본 바인딩	X		X	X	X
<a href="#">DefinitionType attribute</a>	큐 정의 유형	X	X			
<a href="#">DefInputOpenOption</a>	기본 입력 열기 옵션	X	X			

표 561. 큐의 속성 (계속)						
속성	설명	로컬	모델	알리어스	원격	클러스터
<u>DefPersistence</u>	기본 메시지 지속성	X	X	X	X	X
<u>DefPriority</u>	기본 메시지 우선순위	✓	✓	✓	✓	✓
<u>DefReadAhead</u>	기본값 미리 읽기	X	X	X		
<u>DistLists</u>	분배 목록 지원	X	X			
<u>HardenGetBackout</u>	정확한 백아웃 수의 유지보수 여부	X	X			
<u>IndexType</u>	색인 유형	X	X			
<u>InhibitGet</u>	큐에 대한 가져오기 조작 허용 여부	X	X	X		
<u>InhibitPut</u>	큐에 대한 넣기 조작 허용 여부	X	X	X	X	X
<u>InitiationQName</u>	이니시에이션 큐의 이름	X	X			
<u>MaxMsgLength</u>	최대 메시지 길이(바이트)	X	X			
<u>MaxQDepth</u>	최대 큐 용량	X	X			
<u>MsgDeliverySequence attribute</u>	메시지 전달 순서	X	X			
<u>NonPersistentMessage Class</u>	비지속 메시지에 대한 신뢰성 목표	X	X			
<u>OpenInputCount</u>	입력하기 위한 열기 수	X				
<u>OpenOutputCount</u>	출력하기 위한 열기 수	X				
<u>PropertyControl</u>	특성 제어	✓	✓	✓		
<u>ProcessName</u>	프로세스 이름	X	X			
<u>QDepthHighEvent attribute</u>	큐 용량 상한 이벤트 생성 여부	X	X			
<u>QDepthHighLimit</u>	큐 용량에 대한 상한	X	X			
<u>QDepthLowEvent attribute</u>	큐 용량 하한 이벤트 생성 여부	X	X			
<u>QDepthLowLimit attribute</u>	큐 용량에 대한 하한	X	X			
<u>QDepthMaxEvent</u>	큐 가득 참 이벤트 생성 여부	X	X			
<u>QDesc</u>	큐 설명	X	X	X	X	X
<u>QName</u>	큐 이름	X		X	X	X
<u>QServiceInterval</u>	큐 서비스 간격 대상	X	X			
<u>QServiceIntervalEvent attribute</u>	서비스 간격 높음 또는 서비스 간격 확인 이벤트 생성 여부	X	X			
<u>QSGDisp attribute</u>	큐 공유 그룹 속성 지정	X		X	X	
<u>QueueAccounting</u>	큐 계정 데이터 콜렉션	X	X	X	X	X
<u>QueueMonitoring</u>	큐에 대한 온라인 모니터링 데이터	X	✓			
<u>QueueStatistics</u>	큐 통계 데이터 콜렉션	X	X	X	X	X
<u>QType</u>	큐 유형	X		X	X	X
<u>RemoteQMgrName</u>	리모트 큐 관리자의 이름				X	
<u>RemoteQName</u>	리모트 큐의 이름				X	
<u>RetentionInterval</u>	보유 간격	X	X			
<u>Scope</u>	큐에 대한 항목이 셀 디렉토리에 존재하는지 여부	X		X	X	
<u>Shareability</u>	큐 공유 가용성	X	X			
<u>StorageClass</u>	큐에 대한 스토리지 클래스	X	X			
<u>TriggerControl</u>	트리거 제어	X	X			
<u>TriggerData</u>	트리거 데이터	X	X			

표 561. 큐의 속성 (계속)						
속성	설명	로컬	모델	알리어스	원격	클러스터
<u>TriggerDepth</u>	트리거 용량	X	X			
<u>TriggerMsgPriority</u>	트리거에 대한 임계값 메시지 우선순위	X	X			
<u>TriggerType</u>	트리거 유형	X	X			
<u>Usage attribute</u>	큐 용도	X	X			
<u>XmitQName</u>	전송 큐 이름				X	

### 관련 개념

[클러스터 큐](#)

[로컬 큐](#)

[사용할 클러스터 전송 큐의 유형 선택 방법](#)

### **AlterationDate (MQCHAR12)**

정의가 마지막으로 변경된 날짜.

표 562. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	

정의가 마지막으로 변경된 날짜입니다. 날짜의 형식은 YYYY-MM-DD이며, 12바이트 길이로 만들기 위해 뒤에 두 개의 후미 공백으로 채워집니다(예: 1992-09-23--이며, 여기서 --은 두 개의 공백 문자를 나타냄).

큐 관리자가 작동하면 특정 속성 값(예: *CurrentQDepth*)이 변경됩니다. 이러한 속성에 대한 변경사항은 *AlterationDate*에 영향을 주지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_ALTERATION\_DATE 선택자를 사용하십시오. 이 속성의 길이는 MQ\_DATE\_LENGTH에서 제공됩니다.

### **AlterationTime (MQCHAR8)**

정의가 마지막으로 변경된 시간.

표 563. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	

정의가 마지막으로 변경된 시간입니다. 시간 형식은 24시간 클럭을 사용하는 HH.MM.SS이며, 시간이 10 미만이면 앞에 0이 있습니다(예: 09.10.20).

- z/OS에서 시간은 GMT로 정확하게 설정되는 시스템 시계에 따라 그리니치 표준시(GMT)입니다.
- 다른 환경에서 시간은 로컬 시간입니다.

큐 관리자가 작동하면 특정 속성 값(예: *CurrentQDepth*)이 변경됩니다. 이러한 속성에 대한 변경사항은 *AlterationTime*에 영향을 주지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_ALTERATION\_TIME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_TIME\_LENGTH에서 제공됩니다.

### **BackoutQueueQName(MQCHAR48)**

이는 초과 백아웃 큐 이름입니다. 값을 조회할 수 있는 것과는 별도로 큐 관리자는 이 속성의 값에 기반한 조치를 수행하지 않습니다.

표 564. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

WebSphere Application Server 내부에서 실행되는 애플리케이션과 IBM MQ Application Server Facilities를 사용하는 애플리케이션은 이 속성을 사용하여 백아웃된 메시지가 이동해야 하는 위치를 판별합니다. 기타 모든 애플리케이션의 경우 큐 관리자는 속성 값을 기반으로 조치를 취하지 않습니다.

IBM MQ classes for JMS에서는 이 속성을 사용하여 *BackoutThreshold* 속성에서 지정한 최대 횟수만큼 이 미 백아웃된 메시지를 전송할 위치를 판별합니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_BACKOUT\_REQ\_Q\_NAME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다.

### **BackoutThreshold(MQLONG)**

이는 백아웃 임계값입니다. 값을 조회할 수 있는 것과는 별도로 큐 관리자는 이 속성의 값에 기반한 조치를 수행하지 않습니다.

표 565. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

WebSphere Application Server에서 실행 중인 애플리케이션과 IBM MQ Application Server Facilities를 사용하는 애플리케이션에서는 이 속성을 사용하여 메시지를 백아웃해야 하는지 판별합니다. 기타 모든 애플리케이션의 경우 큐 관리자는 속성 값을 기반으로 조치를 취하지 않습니다.

IBM MQ classes for JMS에서는 이 속성을 사용하여 *BackoutQueueQName* 에서 지정한 큐에 메시지를 전송하기 전에 메시지가 백아웃되도록 허용할 시간을 판별합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_BACKOUT\_THRESHOLD 선택자를 사용하십시오.

### **BaseQName (MQCHAR48)**

로컬 큐 관리자에 정의된 큐의 이름입니다.

표 566. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
		X		

(큐 이름에 대한 자세한 정보는 [MQOD - ObjectName 필드](#)를 참조하십시오.) 큐는 다음 유형 중 하나입니다.

#### **MQQT\_LOCAL**

로컬 큐.

#### **MQQT\_REMOTE**

리모트 큐의 로컬 정의입니다.

#### **MQQT\_CLUSTER**

클러스터 큐.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_BASE\_Q\_NAME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다.

## BaseType(MQCFIN)

알리아스가 해석되는 오브젝트의 유형.

표 567. 이 속성이 적용되는 큐 유형				
로컬	모델	알리아스	원격	클러스터
		X		

다음 값 중 하나입니다.

### MQOT\_Q

기본 오브젝트 유형이 큐입니다.

### MQOT\_TOPIC

기본 오브젝트 유형이 토픽입니다.

## CFStrucName (MQCHAR12)


이것은 큐의 메시지가 저장되는 커플링 기능 구조의 이름입니다. 이름의 첫 번째 문자는 A-Z 범위에 있으며 나머지 문자는 A-Z, 0-9 또는 공백입니다.

표 568. 이 속성이 적용되는 큐 유형				
로컬	모델	알리아스	원격	클러스터
X	X			

커플링 기능에서 구조의 전체 이름을 가져오려면 **CFStrucName** 큐 속성의 값이 포함된 **QSGName** 큐 관리자 속성의 값을 끝에 붙이십시오.

이 속성은 공유 큐에만 적용됩니다. *QSGDisp*에 값 **MQQSGD\_SHARED**가 없는 경우 무시됩니다.

이 속성의 값을 판별하려면 **MQINQ** 호출에서 **MQCA\_CF\_STRUC\_NAME** 선택자를 사용하십시오. 이 속성의 길이는 **MQ\_CF\_STRUC\_NAME\_LENGTH**로 제공됩니다.

 이 속성은 z/OS에서만 지원됩니다.

## ClusterChannelName(MQCHAR20)

**ClusterChannelName** 이름은 이 큐를 전송 큐로 사용하는 클러스터-송신자 채널의 일반 이름입니다. 이 속성은 해당 클러스터 전송 큐에서 클러스터 수신자 채널로 메시지를 송신할 클러스터 송신자 채널을 지정합니다.

표 569. 이 속성이 적용되는 큐 유형				
로컬	모델	알리아스	원격	클러스터
X	X			

기본 큐 관리자 구성은 모든 클러스터 송신자 채널이 단일 전송 큐 **SYSTEM.CLUSTER.TRANSMIT.QUEUE**에서 메시지를 전송하기 위한 것입니다. 기본 구성은 큐 관리자 속성 (**DefClusterXmitQueueType**)을 변경하여 수정하여 변경할 수 있습니다. 이 속성의 기본값은 **SCTQ**이며, 값을 **CHANNEL**로 변경할 수 있습니다.

**DefClusterXmitQueueType** 속성을 **CHANNEL**로 설정하면 각 클러스터 송신자 채널의 기본값은 특정 클러스터 전송 큐(**SYSTEM.CLUSTER.TRANSMIT.ChannelName**)를 사용합니다.

수동으로 전송 큐 속성 **ClusterChannelName**을 클러스터 송신자 채널로 설정할 수도 있습니다. 클러스터 송신자 채널을 통해 연결된 큐 관리자를 목적지로 하는 메시지는 클러스터 송신자 채널을 식별하는 전송 큐에 저장되고, 메시지는 기본 클러스터 전송 큐에 저장되지 않습니다. **ClusterChannelName** 속성을 비워 둘 경우 채널이 재시작 시 기본 클러스터 전송 큐로 전환됩니다. 기본 큐는 큐 관리자 **DefClusterXmitQueueType** 속성의 값에 따라 **SYSTEM.CLUSTER.TRANSMIT.ChannelName** 또는 **SYSTEM.CLUSTER.TRANSMIT.QUEUE**입니다.

**ClusterChannelName**에 별표("\*")를 지정하면 전송 큐를 클러스터 송신자 채널 세트와 연관시킬 수 있습니다. 별표는 채널 이름 문자열의 시작 부분이나 끝에 지정하거나 채널 이름 문자열 중간의 원하는 위치에 지정할 수 있습니다. **ClusterChannelName**은(는) 길이가 20자로 제한됩니다. MQ\_CHANNEL\_NAME\_LENGTH.

### ClusterName (MQCHAR48)

큐가 속한 클러스터의 이름입니다.

표 570. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	X

큐가 둘 이상의 클러스터에 속하는 경우, **ClusterNameList**은(는) 클러스터를 식별하는 이름 목록 오브젝트의 이름을 지정하고 **ClusterName**은(는) 공백입니다. **ClusterName** 및 **ClusterNameList** 중 하나 이상은 공백이어야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_CLUSTER\_NAME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_CLUSTER\_NAME\_LENGTH로 제공됩니다.

### ClusterNameList (MQCHAR48)

이 이름은 이 큐가 속한 클러스터의 이름을 포함하는 이름 목록 오브젝트의 이름입니다.

표 571. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	

큐가 하나의 클러스터에만 속하는 경우, 이름 목록 오브젝트에는 하나의 이름만 있습니다. 또는 **ClusterName**을 사용하여 클러스터의 이름을 지정할 수 있는데, 이 경우 **ClusterNameList**는 공백입니다. **ClusterName** 및 **ClusterNameList** 중 하나 이상은 공백이어야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_CLUSTER\_NAMELIST 선택자를 사용하십시오. 이 속성의 길이는 MQ\_NAMELIST\_NAME\_LENGTH로 제공됩니다.

### CLWLQueuePriority(MQLONG)

이는 큐의 우선순위를 나타내는 클러스터 워크로드 큐 우선순위(0 - 9 범위의 값)입니다.

표 572. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	X

자세한 정보는 [클러스터 큐](#)를 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_CLWL\_Q\_PRIORITY 선택자를 사용하십시오.

### CLWLQueueRank(MQLONG)

이는 큐의 순위를 나타내는 클러스터 워크로드 큐 순위(0 - 9 범위의 값)입니다.

표 573. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	X

자세한 정보는 [클러스터 큐](#)를 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_CLWL\_Q\_RANK 선택자를 사용하십시오.



### CLWLUseQ(MQLONG)

대상 큐에 로컬 인스턴스 및 하나 이상의 원격 클러스터 인스턴스가 있으면 MQPUT의 작동을 정의합니다. 넣기가 클러스터 채널에서 시작되는 경우 이 속성이 적용되지 않습니다.

표 574. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X				

값은 다음 중 하나입니다.

#### MQCLWL\_USEQ\_ANY

리모트 큐 및 로컬 큐를 사용합니다.

#### MQCLWL\_USEQ\_LOCAL

리모트 큐를 사용하지 마십시오.

#### MQCLWL\_USEQ\_AS\_Q\_MGR

큐 관리자의 MQIA\_CLWL\_USEQ에서 정의를 상속하십시오.

자세한 정보는 [클러스터 큐](#)를 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_CLWL\_USEQ 선택자를 사용하십시오. 이 속성의 길이는 MQ\_CLWL\_USEQ\_LENGTH로 제공됩니다.

### CreationDate (MQCHAR12)

큐가 작성된 날짜입니다.

표 575. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X				

날짜의 형식은 YYYY-MM-DD이며 길이를 12바이트로 만들기 위해 두 개의 후미 문자 공백으로 채워집니다(예를 들어, 2013-09-23-- 여기서, --는 두 개의 공백 문자를 나타냄).

- IBM i에서 큐의 작성 날짜는 큐를 나타내는 기본 운영 체제 엔티티(파일 또는 사용자 공간)의 날짜와 다를 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_CREATION\_DATE 선택자를 사용하십시오. 이 속성의 길이는 MQ\_CREATION\_DATE\_LENGTH로 제공됩니다.

### CreationTime (MQCHAR8)

큐가 작성된 시간입니다.

표 576. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X				

시간 형식은 24시간 클럭을 사용하는 HH.MM.SS이며, 시간이 10 미만이면 앞에 0이 있습니다(예: 09.10.20).

- z/OS에서 시간은 GMT로 정확하게 설정되는 시스템 시계에 따라 그리니치 표준시(GMT)입니다.
- 다른 환경에서 시간은 로컬 시간입니다.
- IBM i에서 큐의 작성 시간은 큐를 나타내는 기본 운영 체제 엔티티(파일 또는 사용자 공간)의 시간과 다를 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_CREATION\_TIME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_CREATION\_TIME\_LENGTH로 제공됩니다.



### CurrentQDepth(MQLONG)

현재 큐에 있는 메시지 수입니다.

표 577. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X				

MQPUT 호출 중과 MQGET 호출의 백아웃 중에 증가합니다. 비열람 MQGET 호출 중과 MQPUT 호출의 백아웃 중에 감소합니다. 이의 효과는 MQGET 호출에 의해 검색될 수는 없지만 작업 단위 내에서 큐에 넣었으나 아직 커미트되지 않은 메시지를 포함합니다. 마찬가지로 MQGET 호출을 사용하여 작업 단위 내에 검색되었지만 아직 커미트되지 않은 메시지는 제외합니다.

또한 이 개수에는 메시지를 검색할 수는 없지만 만기 시간이 지났으나 아직 제거되지 않은 메시지도 포함됩니다. 자세한 내용은 [MQMD - 만기 필드를 참조하십시오](#).

메시지의 작업 단위 처리 및 세그먼트화로 인해 *CurrentQDepth*가 *MaxQDepth*를 초과할 수 있습니다. 그러나 이는 메시지의 검색 능력에 영향을 주지 않습니다. 큐에 있는 모든 메시지는 정상적인 방법으로 MQGET 호출을 사용하여 검색될 수 있습니다.

이 속성의 값은 큐 관리자가 조작함에 따라 변동됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_CURRENT\_Q\_DEPTH 선택자를 사용하십시오.

### DefaultPutResponse(MQLONG)

애플리케이션이 MQPMO\_RESPONSE\_AS\_Q\_DEF를 지정할 때 큐에 대한 넣기 조작에 사용할 응답 유형을 지정합니다.

표 578. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X	X	X	

다음 값 중 하나입니다.

#### MQPRT\_SYNC\_RESPONSE

Put 조작은 동기식으로 실행되며 응답을 리턴합니다.

#### MQPRT\_ASYNC\_RESPONSE

Put 조작이 비동기식으로 실행되며 MQMD 필드의 서브세트를 리턴합니다.

### DefBind(MQLONG)

MQOPEN 호출에 MQOO\_BIND\_AS\_Q\_DEF가 지정되고 큐가 클러스터 큐인 경우에 사용되는 기본 바인딩입니다.

표 579. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	X

값은 다음 중 하나입니다.

#### MQBND\_BIND\_ON\_OPEN

MQOPEN 호출로 고정된 바인딩.

#### MQBND\_BIND\_NOT\_FIXED

바인딩이 고정되지 않습니다.

#### MQBND\_BIND\_ON\_GROUP

애플리케이션을 통해 메시지 그룹이 모두 동일한 목적지 인스턴스에 할당되도록 요청할 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_DEF\_BINDE 선택자를 사용하십시오.

## DefinitionType(MQLONG)

큐가 정의된 방법을 표시합니다.

표 580. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

값은 다음 중 하나입니다.

### MQQDT\_PREDEFINED

이 큐는 시스템 관리자가 작성한 영구적 큐입니다. 시스템 관리자만 삭제할 수 있습니다.

사전정의된 큐는 DEFINE MQSC 명령을 사용하여 작성하고 DELETE MQSC 명령을 사용하여 삭제할 수 있습니다. 사전정의된 큐는 모델 큐에서 작성할 수 없습니다.

명령은 명령 메시지를 명령 입력큐에 전송하는 운영자 또는 권한 부여된 사용자에게 의해 실행할 수 있습니다 (자세한 정보는 [CommandInputQName](#) 속성 참조).

### MQQDT\_PERMANENT\_DYNAMIC

이 큐는 오브젝트 디스크립터 MQOD에 지정된 모델 큐의 이름으로 MQOPEN 호출을 발행하는 애플리케이션이 작성한 영구적 큐입니다. 모델 큐 정의에는 **DefinitionType** 속성에 대한 MQQDT\_PERMANENT\_DYNAMIC 값이 포함됩니다.

이 유형의 큐는 MQCLOSE 호출을 사용하여 삭제할 수 있습니다. 자세한 정보는 627 페이지의 『MQCLOSE - 오브젝트 닫기』의 내용을 참조하십시오.

영구적 동적 큐에 대한 **QSGDisp** 속성의 값은 MQQSGD\_Q\_MGR입니다.

### MQQDT\_TEMPORARY\_DYNAMIC

이 큐는 오브젝트 디스크립터 MQOD에 지정된 모델 큐의 이름으로 MQOPEN 호출을 발행하는 애플리케이션이 작성한 임시 큐입니다. 모델 큐 정의에는 **DefinitionType** 속성에 대한 MQQDT\_TEMPORARY\_DYNAMIC 값이 포함됩니다.

이 유형의 큐는 작성한 애플리케이션에 의해 닫히면 MQCLOSE 호출을 통해 자동으로 삭제됩니다.

임시 동적 큐에 대한 **QSGDisp** 속성의 값은 MQQSGD\_Q\_MGR입니다.

### MQQDT\_SHARED\_DYNAMIC

이 큐는 오브젝트 디스크립터 MQOD에 지정된 모델 큐의 이름으로 MQOPEN 호출을 발행하는 애플리케이션이 작성한 공유 영구적 큐입니다. 모델 큐 정의에는 **DefinitionType** 속성에 대한 MQQDT\_SHARED\_DYNAMIC 값이 포함됩니다.

이 유형의 큐는 MQCLOSE 호출을 사용하여 삭제할 수 있습니다. 자세한 정보는 627 페이지의 『MQCLOSE - 오브젝트 닫기』의 내용을 참조하십시오.

공유 동적 큐에 대한 **QSGDisp** 속성의 값은 MQQSGD\_SHARED입니다.

모델 큐는 항상 사전정의되기 때문에 모델 큐 정의에서 이 속성이 모델 큐가 어떻게 정의되었는지 표시하지 않습니다. 대신, 모델 큐의 이 속성 값은 MQOPEN 호출을 사용하여 모델 큐 정의에서 작성한 각 동적 큐의 **DefinitionType**을 판별하는 데 사용됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_DEFINITION\_TYPE 선택자를 사용하십시오.

## DefInputOpenOption(MQLONG)

이 방법은 입력을 위해 큐를 여는 기본 방법입니다.

표 581. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

큐가 열릴 때 MQOO\_INPUT\_AS\_Q\_DEF 옵션이 MQOPEN 호출에 지정되는 경우 적용됩니다. 값은 다음 중 하나입니다.

### **MQOO\_INPUT\_EXCLUSIVE**

배타적 액세스로 메시지를 가져오기 위해 큐를 엽니다.

해당 큐가 후속 MQSET 호출에 사용하도록 열립니다. 임의의 유형(MQOO\_INPUT\_SHARED 또는 MQOO\_INPUT\_EXCLUSIVE)의 입력에 대해 현재 이 애플리케이션 또는 다른 애플리케이션에서 큐가 열려 있는 경우 이유 코드 MQRC\_OBJECT\_IN\_USE와 함께 호출에 실패합니다.

### **MQOO\_INPUT\_SHARED**

공유 액세스로 메시지를 가져오기 위해 큐를 엽니다.

해당 큐가 후속 MQSET 호출에 사용하도록 열립니다. 현재 이 애플리케이션 또는 다른 애플리케이션에서 MQOO\_INPUT\_SHARED를 사용하여 큐가 열려 있는 경우 호출에 성공하지만 큐가 현재 MQOO\_INPUT\_EXCLUSIVE를 사용하여 열려 있으면 이유 코드 MQRC\_OBJECT\_IN\_USE와 함께 호출에 실패합니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_DEF\_INPUT\_OPEN\_OPTION 선택자를 사용하십시오.

### **DefPersistence(MQLONG)**

이는 큐에 있는 메시지의 기본 지속성입니다. 메시지를 넣을 때 메시지 디스크립터에 MQPER\_PERSISTENCE\_AS\_Q\_DEF가 지정된 경우에 적용됩니다.

표 582. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X	X	X	X

큐 이름 해석 경로에 정의가 둘 이상인 경우, 기본 지속성은 MQPUT 또는 MQPUT1 호출 시 경로의 첫 번째 정의에 있는 이 속성의 값에서 가져옵니다. 값은 다음이 될 수 있습니다.

- 알리어스 큐
- 로컬 큐
- 리모트 큐의 로컬 정의
- 큐 관리자 알리어스
- 전송 큐(예: *DefXmitQName* 큐)

값은 다음 중 하나입니다.

### **MQPER\_PERSISTENT**

메시지는 시스템 실패 및 큐 관리자 재시작에서 지속됩니다. 다음 큐에는 지속 메시지를 넣을 수 없습니다.

- 임시 동적 큐
- CFLEVEL(2) 이하의 CFSTRUCT 오브젝트에 맵핑되는 공유 큐 또는 CFSTRUCT 오브젝트가 RECOVER(NO)로 정의되는 공유 큐.

지속 메시지는 영구적 동적 큐 및 사전정의된 큐에 배치할 수 있습니다.

### **MQPER\_NOT\_PERSISTENT**

메시지는 시스템 실패 또는 큐 관리자 재시작에서 정상적으로 지속되지 않습니다. 큐 관리자가 재시작되는 동안 메시지의 원래 사본이 보조 기억장치에서 발견되는 경우에도 적용됩니다.

공유 큐의 경우 큐 공유 그룹에서 큐 관리자 재시작 시 비지속 메시지가 지속되나 공유 큐에 메시지를 저장하기 위해 사용되는 커플링 기능의 실패 시에는 지속되지 않습니다.

지속 및 비지속 메시지 둘 다 동일한 큐에 존재할 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_DEF\_PERSISTENCE 선택자를 사용하십시오.

### **DefPriority(MQLONG)**

큐에 있는 메시지의 기본 우선순위입니다. 메시지가 큐에 놓이면 메시지 디스크립터에 MQPRI\_PRIORITY\_AS\_Q\_DEF가 지정된 경우 적용됩니다.

표 583. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X	X	X	X

큐 이름 해석 경로에 정의가 둘 이상인 경우, 메시지의 기본 우선순위는 Put 조작 시 경로의 첫 번째 정의에 있는 이 속성의 값에서 가져옵니다. 값은 다음이 될 수 있습니다.

- 알리어스 큐
- 로컬 큐
- 리모트 큐의 로컬 정의
- 큐 관리자 알리어스
- 전송 큐(예: *DefXmitQName* 큐)

메시지를 큐에 넣는 방법은 큐의 **MsgDeliverySequence** 속성 값에 따라 다릅니다.

- **MsgDeliverySequence** 속성이 MQMDS\_PRIORITY인 경우 메시지를 큐에 배치 시 논리적인 위치는 메시지 디스크립터의 *Priority* 필드 값에 따라 다릅니다.
- **MsgDeliverySequence** 속성이 MQMDS\_FIFO인 경우 메시지는 메시지 디스크립터의 *Priority* 필드 값에 관계없이 해석된 큐의 *DefPriority*와 동일한 우선순위를 갖는 것처럼 큐에 배치됩니다. 그러나 *Priority* 필드는 메시지를 넣은 애플리케이션에서 지정된 값을 보유합니다. 자세한 정보는 **MsgDeliverySequence** 속성을 참조하십시오.

우선순위는 0(최하위) - *MaxPriority* (최상위) 범위 내에 있습니다. **MaxPriority** 속성을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_DEF\_PRIORITY 선택자를 사용하십시오.

### DefReadAhead(MQLONG)

클라이언트에 전달된 비지속 메시지의 디폴트 미리 읽기 작동을 지정합니다.

표 584. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X	X		

DefReadAhead는 다음 값 중 하나로 설정할 수 있습니다.

#### MQREADA\_NO

애플리케이션이 요청하기 전에 비지속 메시지를 클라이언트에 미리 송신하지 않습니다. 클라이언트가 비정상적으로 종료되면 하나의 비지속 메시지의 최대값을 잃을 수 있습니다.

#### MQREADA\_YES

애플리케이션이 요청하기 전에 비지속 메시지를 클라이언트에 미리 송신합니다. 클라이언트가 비정상적으로 종료된 경우나 클라이언트가 송신된 메시지를 모두 이용하지 않는 경우 비지속 메시지가 손실될 수 있습니다.

#### MQREADA\_DISABLED

비지속 메시지의 미리 읽기는 이 큐에 사용되지 않습니다. 클라이언트 애플리케이션이 미리 읽기를 요청했는지 여부와 무관하게 메시지가 클라이언트에 미리 송신되지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_DEF\_READ\_AHEAD 선택자를 사용하십시오.

### DefPResp(MQLONG)

기본 넣기 응답 유형(DEFPRESP) 속성은 MQPMO 내의 PutResponseType이 MQPMO\_RESPONSE\_AS\_Q\_DEF로 설정되었을 때 애플리케이션에서 사용된 값을 정의합니다. 이 속성은 모든 큐 유형에 대해 유효합니다.

표 585. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X	X	X	X

값은 다음 중 하나입니다.

**SYNC**

Put 조작은 동기식으로 발행되며 응답을 리턴합니다.

**ASYNQ**

Put 조작이 비동기식으로 실행되며 MQMD 필드의 서브세트를 리턴합니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_DEF\_PUT\_RESPONSE\_TYPE 선택자를 사용하십시오.

**DistLists(MQLONG)**

분배 목록 메시지를 큐에 배치할 수 있는지 표시합니다.

표 586. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

메시지 채널 에이전트(MCA)는 채널의 다른 끝에 있는 큐 관리자가 분배 목록을 지원할지 여부를 로컬 큐 관리자에게 알리기 위해 속성을 설정합니다. 송신 MCA에 의해 로컬 전송 큐에서 제거된 후 이 후자 큐 관리자(파트너 큐 관리자라고 함)는 다음에 메시지를 수신하는 큐 관리자입니다.

파트너 큐 관리자에서 수신 MCA에 연결할 때마다 송신 MCA는 해당 속성을 설정합니다. 이런 방법으로 송신 MCA는 로컬 큐 관리자가 파트너 큐 관리자가 올바르게 처리할 수 있는 메시지만 전송 큐에 배치할 수 있게 할 수 있습니다.

이 속성은 주로 전송 큐와 함께 사용되지만 설명된 처리는 큐에 대해 정의된 사용법에 관계없이 수행됩니다(사용법 속성 참조).

값은 다음 중 하나입니다.

**MQDL\_SUPPORTED**

분배 목록은 메시지는 큐에 저장될 수 있고 해당 양식으로 파트너 큐 관리자에 전송됩니다. 이렇게 하면 메시지를 여러 목적으로 송신하는 데 필요한 처리량이 줄어듭니다.

**MQDL\_NOT\_SUPPORTED**

파트너 큐 관리자가 분배 목록을 지원하지 않으므로 분배 목록 메시지는 큐에 저장될 수 없습니다. 애플리케이션이 분배 목록 메시지를 넣고 해당 메시지가 이 큐에 놓이는 경우, 큐 관리자는 대신 분배 목록을 분할하여 개별 메시지를 큐에 놓습니다. 이로 인해 메시지를 여러 대상으로 보내는 데 필요한 처리량이 증가하지만 메시지가 파트너 큐 관리자에서 제대로 처리되도록 합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_DIST\_LISTS 선택자를 사용하십시오. 이 속성의 값을 변경하려면 MQSET 호출을 사용하십시오.

이 속성은 z/OS에서 지원되지 않습니다.

**HardenGetBackout(MQLONG)**

메시지마다 작업 단위 내에서 MQGET 호출을 통해 메시지가 검색된 횟수와 해당 작업 단위가 이후에 백아웃된 횟수가 유지됩니다.

표 587. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

이 수는 MQGET 호출이 완료된 후 메시지 디스크립터의 *BackoutCount* 필드에서 사용할 수 있습니다.

메시지 백아웃 수는 큐 관리자의 재시작에서 지속됩니다. 그러나 이 수가 정확한지 확인하려면 MQGET 호출이 이 큐의 작업 단위 내에서 메시지를 검색할 때마다 정보는 정보를 기록해야 합니다(디스크 또는 다른 영구적 스토리지 디바이스에 기록됨). 이 작업이 완료되지 않으면 큐 관리자가 실패하고 MQGET 호출이 백아웃되며 이 수가 증가하거나 증가하지 않을 수 있습니다.

작업 단위 내에 있는 각 MQGET 호출에 대한 정보를 기록하면 추가 처리 비용이 부과되므로 이 수가 정확한 경우에만 **HardenGetBackout** 속성을 MQQA\_BACKOUT\_HARDENED로 설정하십시오.

멀티플랫폼에서는 이 속성 설정에 상관없이 메시지 백아웃 수가 항상 기록됩니다.

가능한 값은 다음과 같습니다.

#### **MQQA\_BACKOUT\_HARDENED**

기록은 이 큐에 있는 메시지의 백아웃 수가 정확한지 확인하는 데 사용됩니다.

#### **MQQA\_BACKOUT\_NOT\_HARDENED**

이 큐에 있는 메시지의 백아웃 수가 정확한지 확인하기 위해 기록을 사용하지 않습니다. 따라서 해당 수는 예상보다 적을 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_HARDEN\_GET\_BACKOUT 선택자를 사용하십시오.

### **IndexType(MQLONG)**

큐 관리자가 큐의 메시지에 대해 유지보수하는 인덱스의 유형을 지정합니다.

표 588. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

필요한 색인 유형은 애플리케이션이 메시지를 검색하는 방법 및 큐가 공유 큐인지 비공유 큐인지 여부에 따라 다릅니다(QSGDisp 속성 참조). 다음 값은 *IndexType*에 대해 가능합니다.

#### **MQIT\_NONE**

이 큐에 대한 큐 관리자에서 유지보수되는 색인이 없습니다. MQGET 호출에서 선택 기준을 사용하지 않고 일반적으로 순차적으로 처리되는 큐에 이 값을 사용하십시오.

#### **MQIT\_MSG\_ID**

큐 관리자는 큐에 있는 메시지의 메시지 ID를 사용하는 색인을 유지보수합니다. 애플리케이션에서 MQGET 호출의 선택기준으로 메시지 ID를 사용하여 일반적으로 메시지를 검색하는 이 값 큐를 사용하십시오.

#### **MQIT\_CORREL\_ID**

큐 관리자는 큐에 있는 메시지의 상관 ID를 사용하는 색인을 유지보수합니다. 애플리케이션에서 MQGET 호출의 선택기준으로 상관 ID를 사용하여 일반적으로 메시지를 검색하는 이 큐 값을 사용하십시오.

#### **MQIT\_MSG\_TOKEN**

**중요사항:** 이 색인 유형은 IBM MQ Workflow for z/OS 제품과 함께 사용되는 큐에만 사용해야 합니다.

큐 관리자는 z/OS의 워크로드 관리자(WLM) 기능과 함께 사용할 큐에 있는 메시지의 메시지 토큰을 사용하는 색인을 유지보수합니다.

WLM 관리 큐에 이 옵션을 지정해야 합니다. 큐의 기타 유형에는 지정하지 마십시오. 또한 애플리케이션이 z/OS 워크로드 관리자 기능을 사용하고 있지 않지만 MQGET 호출 시 선택기준으로 메시지 토큰을 사용하여 메시지를 검색하고 있는 큐에 이 값을 사용하지 마십시오.

#### **MQIT\_GROUP\_ID**

큐 관리자는 큐에 있는 메시지의 그룹 ID를 사용하는 색인을 유지보수합니다. 이 값은 애플리케이션이 MQGET 호출 시 MQGMO\_LOGICAL\_ORDER 옵션을 사용하여 메시지를 검색하는 큐에 사용해야 합니다.

이 인덱스 유형의 큐는 전송 큐일 수 없습니다. 이 인덱스 유형의 공유 큐는 CFLEVEL(3) 이상에서 CFSTRUCT 오브젝트에 맵핑되도록 정의해야 합니다.

#### **참고:**

1. 큐가 MQGET 호출 시 MQGMO\_LOGICAL\_ORDER 옵션을 사용하여 메시지의 효율적 검색 유형에 대해 최적화된 것처럼 인덱스 유형 MQIT\_GROUP\_ID의 큐에 있는 실제 순서는 정의되지 않습니다. 이는 메시지의 실제 순서가 일반적으로 메시지가 큐에 도착한 순서가 아님을 의미합니다.

2. MQIT\_GROUP\_ID 큐에 MQMDS\_PRIORITY의 *MsgDeliverySequence* 이 있는 경우 큐 관리자는 메시지 우선순위 0 및 1을 사용하여 논리적 순서로 메시지 검색을 최적화합니다. 따라서 그룹에서 첫 번째 메시지는 0 또는 하나의 우선순위를 가지고 있지 않아야 합니다. 해당 경우 2의 우선순위를 가지고 있는 것처럼 메시지가 처리됩니다. MQMD 구조의 *Priority* 필드는 변경되지 않습니다.

메시지 그룹에 대한 자세한 정보는 MQGMO - Options field에서 그룹 및 세그먼트 옵션에 대한 설명을 참조하십시오.

다양한 경우에서 사용해야 하는 색인 유형이 819 페이지의 표 589 및 820 페이지의 표 590에 표시됩니다.

표 589. MQGMO_LOGICAL_ORDER가 지정되지 않은 경우의 큐 색인 유형에 대한 제안 또는 필수 값		
MQGET 호출 시 선택 기준	비공유 큐의 색인 유형	공유 큐의 색인 유형
없음	임의	임의
<b>하나의 ID를 사용하여 선택:</b>		
메시지 ID	MQIT_MSG_ID가 제안됩니다.	MQIT_NONE 또는 MQIT_MSG_ID는 필수이며 MQIT_MSG_ID가 제안됩니다.
상관 ID	MQIT_CORREL_ID가 제안됩니다.	MQIT_CORREL_ID는 필수입니다.
그룹 ID	MQIT_GROUP_ID가 제안됩니다.	MQIT_GROUP_ID는 필수입니다.
<b>두 개의 ID를 사용하여 선택:</b>		
메시지 ID와 상관 ID	MQIT_MSG_ID 또는 MQIT_CORREL_ID가 제안됩니다.	MQIT_NONE, MQIT_MSG_ID 또는 MQIT_CORREL_ID는 필수입니다. (효율성을 위해 색인 유형이 가장 다른 키가 포함된 MQMD 필드와 일치하도록 선택할 것을 제안)
메시지 ID와 그룹 ID	MQIT_MSG_ID 또는 MQIT_GROUP_ID가 제안됩니다.	지원되지 않음
상관 ID와 그룹 ID	MQIT_CORREL_ID 또는 MQIT_GROUP_ID가 제안됩니다.	지원되지 않음
<b>세 개의 ID를 사용하여 선택:</b>		
메시지 ID, 상관 ID와 그룹 ID	MQIT_MSG_ID, MQIT_CORREL_ID 또는 MQIT_GROUP_ID가 제안됩니다.	지원되지 않음
<b>그룹 관련 기준을 사용하여 선택:</b>		
그룹 ID와 메시지 순서 번호	MQIT_GROUP_ID는 필수입니다.	MQIT_GROUP_ID는 필수입니다.
메시지 순서 번호(1이어야함)	MQIT_GROUP_ID는 필수입니다.	MQIT_GROUP_ID는 필수입니다.
<b>메시지 토큰을 사용하여 선택:</b>		
애플리케이션 사용을 위한 메시지 토큰	MQIT_MSG_TOKEN 사용 안함	
WLM 사용을 위한 메시지 토큰	MQIT_MSG_TOKEN은 필수입니다.	지원되지 않음



표 590. MQGMO_LOGICAL_ORDER가 지정된 경우의 큐 색인 유형에 대한 제안 또는 필수 값		
MQGET 호출 시 선택 기준	비공유 큐의 색인 유형	공유 큐의 색인 유형
없음	MQIT_GROUP_ID는 필수입니다.	MQIT_GROUP_ID는 필수입니다.
<b>하나의 ID를 사용하여 선택:</b>		
메시지 ID	MQIT_GROUP_ID는 필수입니다.	지원되지 않음
상관 ID	MQIT_GROUP_ID는 필수입니다.	지원되지 않음
그룹 ID	MQIT_GROUP_ID는 필수입니다.	MQIT_GROUP_ID는 필수입니다.
<b>두 개의 ID를 사용하여 선택:</b>		
메시지 ID와 상관 ID	MQIT_GROUP_ID는 필수입니다.	지원되지 않음
메시지 ID와 그룹 ID	MQIT_GROUP_ID는 필수입니다.	지원되지 않음
상관 ID와 그룹 ID	MQIT_GROUP_ID는 필수입니다.	지원되지 않음
<b>세 개의 ID를 사용하여 선택:</b>		
메시지 ID, 상관 ID와 그룹 ID	MQIT_GROUP_ID는 필수입니다.	지원되지 않음

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_INDEX\_TYPE 선택자를 사용하십시오.

**z/OS** 이 속성은 z/OS에서만 지원됩니다.

### **InhibitGet(MQLONG)**

이 큐에 대한 가져오기 조작이 허용되는지 제어합니다.

표 591. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X	X		

알리어스 큐인 경우 MQGET 호출이 성공하려면 가져오기 조작 시에 알리어스 큐 및 기본 큐 둘 다에 대해 가져오기 조작이 허용되어야 합니다. 값은 다음 중 하나입니다.

#### **MQQA\_GET\_INHIBITED**

Get 조작이 금지됩니다.

이유 코드 MQRC\_GET\_INHIBITED와 함께 MQGET 호출에 실패합니다. 여기에는 MQGMO\_BROWSE\_FIRST 또는 MQGMO\_BROWSE\_NEXT를 지정하는 MQGET 호출이 포함됩니다.

**참고:** 작업 단위 내에서 MQGET 호출 조작이 성공적으로 완료되면 후속적으로 MQQA\_GET\_INHIBITED에 대한 **InhibitGet** 속성 값 변경이 작업 단위 커미트를 금지하지 않습니다.

#### **MQQA\_GET\_ALLOWED**

Get 조작이 허용됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_INHIBIT\_GET 선택자를 사용하십시오. 이 속성의 값을 변경하려면 MQSET 호출을 사용하십시오.

### **InhibitPut(MQLONG)**

이 큐에 대한 넣기 조작이 허용되는지 제어합니다.

표 592. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X	X	X	X



큐 이름 해석 경로에 정의가 둘 이상이면 MQPUT 또는 MQPUT1 호출이 성공하기 위해 넣기 조작 시에 경로 내의 모든 정의(모든 큐 관리자 알리어스 정의 포함)에 대해 넣기 조작이 허용되어야 합니다. 값은 다음 중 하나입니다.

**MQQA\_PUT\_INHIBITED**

Put 조작이 금지됩니다.

MQPUT 및 MQPUT1 호출이 이유 코드 MQRC\_PUT\_INHIBITED와 함께 실패합니다.

**참고:** 작업 단위 내에서 MQPUT 호출 조작이 성공적으로 완료되면 후속적으로 MQQA\_PUT\_INHIBITED에 대한 **InhibitPut** 속성 값 변경이 작업 단위 커미트를 금지하지 않습니다.

**MQQA\_PUT\_ALLOWED**

Put 조작이 허용됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_INHIBIT\_PUT 선택자를 사용하십시오. 이 속성의 값을 변경하려면 MQSET 호출을 사용하십시오.

**InitiationQName (MQCHAR48)**

이 이름은 로컬 큐 관리자가 정의한 큐 이름이며, 큐는 MQQT\_LOCAL 유형이어야 합니다.

표 593. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X				

큐 관리자는 이 속성이 속한 큐에 메시지가 도착하여 애플리케이션 시작이 필요할 때 이니시에이션 큐에 트리거 메시지를 보냅니다. 이니시에이션 큐는 트리거 모니터 수신 후 적절한 애플리케이션을 시작하는 트리거 모니터 애플리케이션에서 모니터링해야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_INITIATION\_Q\_NAME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다.

**MaxMsgLength(MQLONG)**

큐에 둘 수 있는 가장 긴 물리적 메시지 길이의 상한입니다.

표 594. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

그러나, **MaxMsgLength** 큐 속성을 **MaxMsgLength** 큐 관리자 속성에 상관없이 설정할 수 있기 때문에 큐에 넣을 수 있는 가장 긴 물리적 메시지 길이의 실제 상한은 이 두 값 중 작은 값입니다.

큐 관리자가 세그먼트화를 지원하는 경우 애플리케이션은 두 **MaxMsgLength** 속성 중 더 작은 속성보다 긴 논리 메시지를 넣을 수 있지만 애플리케이션이 MQMD에서 MQMF\_SEGMENTATION\_ALLOWED 플래그를 지정하는 경우에만 가능합니다. 이 플래그가 지정되는 경우 논리 메시지의 길이에 대한 상한은 999 999 999바이트이지만 일반적으로 운영 체제 또는 애플리케이션이 실행 중인 환경에 의해 부과된 제한조건으로 인해 제한이 낮아집니다.

너무 오래 큐 메시지에 배치하면 다음 이유 코드 중 하나로 실패합니다.

- MQRC\_MSG\_TOO\_BIG\_FOR\_Q 메시지가 큐에 대해 너무 큰 경우
- MQRC\_MSG\_TOO\_BIG\_FOR\_Q\_MGR 메시지가 큐 관리자에 대해 너무 크지만 큐에 대해 너무 크지 않은 경우

**MaxMsgLength** 속성의 하한은 0입니다. 상한은 100MB(104 857 600 바이트)입니다.

자세한 정보는 [MQPUT - BufferLength](#) 매개변수를 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_MAX\_MSG\_LENGTH 선택자를 사용하십시오.

**MaxQDepth(MQLONG)**

한 번에 큐에 있을 수 있는 물리적 메시지 수에 정의된 상한입니다.


표 595. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			


이미 **MaxQDepth** 메시지를 포함한 큐에 메시지를 넣으면 이유 코드 MQRC\_Q\_FULL과 함께 실패합니다.

메시지의 작업 단위 처리 및 세그먼트화로 인해 큐에 있는 물리적 메시지의 실제 수가 **MaxQDepth**를 초과할 수 있습니다. 그러나 큐의 모든 메시지는 MQGET 호출을 사용하여 검색할 수 있으므로 이는 메시지의 검색 가능성에는 영향을 주지 않습니다.

이 속성의 값은 0 이상입니다. 상한은 환경에 따라 판별됩니다.

- 다음 플랫폼에서 값은 999 999 999를 초과할 수 없습니다.

-  AIX
-  Linux
-  Windows
-  z/OS

-  IBM i에서 값은 640,000을 초과할 수 없습니다.

**참고:** 큐에 있는 **MaxQDepth** 메시지보다 더 적은 경우에도 큐에 사용 가능한 스토리지 공간이 사용될 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_MAX\_Q\_DEPTH 선택자를 사용하십시오.

### MsgDeliverySequence(MQLONG)

표 596. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

이는 MQGET 호출이 메시지를 애플리케이션에 리턴하는 순서를 판별합니다.

#### MQMDS\_FIFO

메시지는 선입선출(FIFO) 순서로 리턴됩니다.

MQGET 호출은 메시지의 우선순위와 관계없이 호출에 지정된 선택 기준을 충족하는 첫 번째 메시지를 리턴합니다.

#### MQMDS\_PRIORITY

메시지가 우선순위 순서로 리턴됩니다.

MQGET 호출은 호출에 지정된 선택 기준을 충족하는 최상위 우선순위 메시지를 리턴합니다. 각 우선순위 레벨 내에서 메시지는 선입선출(FIFO) 순서로 리턴됩니다.

- z/OS에서 큐에 MQIT\_GROUP\_ID의 *IndexType*이 있는 경우 **MsgDeliverySequence** 속성은 메시지 그룹이 애플리케이션으로 리턴되는 순서를 지정합니다. 그룹이 리턴되는 특정 순서는 각 그룹에서 첫 번째 메시지의 위치 또는 우선순위에 의해 판별됩니다. 큐가 MQGET 호출 시 MQGMO\_LOGICAL\_ORDER 옵션을 사용하여 메시지의 효율적 검색 유형에 대해 최적화된 것처럼 큐에 있는 메시지의 실제 순서는 정의되지 않습니다.
- z/OS에서 *IndexType*이 MQIT\_GROUP\_ID이고 **MsgDeliverySequence**가 MQMDS\_PRIORITY인 경우 큐 관리자는 논리적인 순서로 메시지의 검색을 최적화하기 위해 메시지 우선순위 0과 1을 사용합니다. 따라서 그룹에서 첫 번째 메시지는 0 또는 하나의 우선순위를 가지고 있지 않아야 합니다. 해당 경우 2의 우선순위를 가지고 있는 것처럼 메시지가 처리됩니다. MQMD 구조의 *Priority* 필드는 변경되지 않습니다.

큐에 메시지가 있는 동안 관련 속성이 변경되면 전달 순서는 다음과 같습니다.

- 메시지가 MQGET 호출에 의해 리턴되는 순서는 메시지가 큐에 도착할 때 큐에 강제 실행되는 **MsgDeliverySequence** 및 **DefPriority** 속성 값에 의해 판별됩니다.
  - 메시지가 도착할 때 **MsgDeliverySequence**가 MQMDS\_FIFO인 경우 해당 우선순위가 **DefPriority** 인 것처럼 메시지는 큐에 배치됩니다. 이는 메시지에 대한 메시지 디스크립터의 **Priority** 필드의 값에 영향을 주지 않습니다. 해당 필드는 메시지를 처음 넣었을 때의 값을 유지합니다.
  - 메시지가 도착할 때 **MsgDeliverySequence**가 MQMDS\_PRIORITY인 경우 메시지는 메시지 디스크립터의 **Priority** 필드에서 지정한 우선순위로 적절한 장소의 큐에 배치됩니다.

큐에 메시지가 있는 동안 **MsgDeliverySequence** 속성 값이 변경되어도 큐에 있는 메시지의 순서는 변경되지 않습니다.

큐에 메시지가 있는 동안 **DefPriority** 속성 값이 변경되는 경우 **MsgDeliverySequence** 속성이 MQMDS\_FIFO로 설정된 경우라도 메시지가 반드시 FIFO 순서로 전달되지는 않습니다. 우선순위가 더 높은 큐에 배치된 메시지가 먼저 전달됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_MSG\_DELIVERY\_SEQUENCE 선택자를 사용하십시오.

### **NonPersistentMessageClass(MQLONG)**

비지속적 메시지에 대한 신뢰도 목표입니다.

표 597. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

이는 이 큐에 넣은 비지속 메시지가 제거되는 환경을 지정합니다.

#### **MQNPM\_CLASS\_NORMAL**

비지속 메시지는 큐 관리자 세션의 수명으로 제한됩니다. 메시지는 큐 관리자 재시작의 경우에 버려집니다. 이는 비공 큐에만 유효하며 기본값입니다.

#### **MQNPM\_CLASS\_HIGH**

큐 관리자가 큐 지속 시간 동안 비지속 메시지를 보유하려 합니다. 비지속 메시지가 실패 시 손실됩니다. 이 값은 공유 큐에 적용됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_NPM\_CLASS 선택자를 사용하십시오.

### **OpenInputCount(MQLONG)**

MQGET 호출을 통해 큐에서 메시지를 제거하는 데 현재 유효한 핸들 수입니다.

표 598. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X				

로컬 큐 관리자에 인식되는 해당 핸들의 합계입니다. 큐가 공유 큐이면 로컬 큐 관리자가 속한 큐 공유 그룹의 다른 큐 관리자에서 큐에 수행된 입력을 위한 열기가 해당 수에 포함되지 않습니다.

이 수는 이 큐로 해석되는 알리어스 큐가 입력에 대해 열린 핸들을 포함합니다. 이 수는 큐가 입력을 포함하지 않는 조치에 대해 열린 핸들을 포함하지 않습니다(예: 찾아보기 전용으로 열린 큐).

이 속성의 값은 큐 관리자가 조작함에 따라 변동됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_OPEN\_INPUT\_COUNT 선택자를 사용하십시오.

### **OpenOutputCount(MQLONG)**

MQPUT 호출을 통해 큐에 메시지를 추가하는 데 현재 유효한 핸들 수입니다.

표 599. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X				

로컬 큐 관리자에 인식되는 해당 핸들의 합계입니다. 리모트 큐 관리자에서 이 큐에 수행된 출력을 위한 열기는 포함되지 않습니다. 큐가 공유 큐이면 로컬 큐 관리자가 속한 큐 공유 그룹의 다른 큐 관리자에서 큐에 수행된 출력을 위한 열기가 해당 수에 포함되지 않습니다.

이 수는 이 큐로 해석되는 알리어스 큐가 출력에 대해 열린 핸들을 포함합니다. 이 수는 큐가 출력을 포함하지 않는 조치에 대해 열린 핸들을 포함하지 않습니다(예: 조회 전용으로 열린 큐).

이 속성의 값은 큐 관리자가 조작함에 따라 변동됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_OPEN\_OUTPUT\_COUNT 선택자를 사용하십시오.

### **ProcessName (MQCHAR48)**

로컬 큐 관리자에 정의된 프로세스 오브젝트의 이름입니다. 프로세스 오브젝트는 큐를 서비스할 수 있는 프로그램을 식별합니다.

표 600. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_PROCESS\_NAME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_PROCESS\_NAME\_LENGTH로 제공됩니다.

### **PropertyControl (MQLONG)**

MQGET 호출을 MQGMO\_PROPERTIES\_AS\_Q\_DEF 옵션과 함께 사용하여 메시지가 큐에서 검색되는 메시지 특성이 핸들링되는 방법을 지정합니다.

표 601. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X	X		

값은 다음 중 하나입니다.

#### **MQPROP\_ALL**

메시지가 애플리케이션에게 전달될 때 메시지의 모든 특성이 메시지에 포함됩니다. 메시지 디스크립터(또는 확장자)의 특성을 제외하고는, 메시지 데이터의 하나 이상의 MQRFH2 헤더에 특성이 배치됩니다. 메시지 핸들이 제공된 경우 작동은 메시지 핸들의 특성을 리턴하는 것입니다.

#### **MQPROP\_COMPATIBILITY**

메시지에 mcd라는 접두부가 있는 특성이 포함된 경우 jms., usr. 또는 mqext., 모든 메시지 특성은 MQRFH2 헤더의 애플리케이션으로 전달됩니다. 그렇지 않은 경우에는 메시지 디스크립터(또는 확장자)의 특성을 제외한 메시지의 모든 특성이 제거되며 애플리케이션에 더 이상 액세스할 수 없습니다. 이 값은 기본값이며 JMS 관련 특성이 메시지 데이터의 MQRFH2 헤더에 있을 것으로 예상하는 애플리케이션이 수정되지 않은 채로 계속해서 작업할 수 있게 합니다. 메시지 핸들이 제공되면 이 작동은 메시지 핸들의 특성을 리턴하는 것입니다.

#### **MQPROP\_FORCE\_MQRFH2**

애플리케이션이 메시지 핸들을 지정하는지 여부와 관계없이 특성은 MQRFH2 헤더의 메시지 데이터에 항상 리턴됩니다. MQGET 호출에서 MQGMO 구조의 MsgHandle 필드에 제공된 유효한 메시지 핸들이 무시됩니다. 메시지의 특성은 메시지 핸들을 통해 액세스할 수 없습니다.

### MQPROP\_NONE

애플리케이션에 메시지가 전달되기 전에 메시지 디스크립터(또는 확장자)의 특성을 제외한 메시지의 모든 특성이 메시지에서 제거됩니다. 메시지 핸들이 제공된 경우 작동은 메시지 핸들의 특성을 리턴하는 것입니다.

이 매개변수는 로컬, 알리아스 및 모델 큐에 적용 가능합니다. 해당 값을 판별하려면 MQINQ 호출에서 MQIA\_PROPERTY\_CONTROL 선택자를 사용하십시오.

### QDepthHighEvent(MQLONG)

큐 항목 수 많음 이벤트 생성 여부를 제어합니다.

표 602. 이 속성이 적용되는 큐 유형				
로컬	모델	알리아스	원격	클러스터
X	X			

큐 용량 상한 이벤트는 애플리케이션이 큐에 메시지를 넣었으며 이로 인해 큐에 있는 메시지 수가 큐 용량 상한 임계값 이상이 되었음을 나타냅니다(QDepthHighLimit 속성 참조).

**참고:** 이 속성의 값은 동적으로 변경할 수 있습니다.

값은 다음 중 하나입니다.

### MQEVR\_DISABLED

이벤트 보고를 사용하지 않습니다.

### MQEVR\_ENABLED

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_Q\_DEPTH\_HIGH\_EVENT 선택자를 사용하십시오.

이 속성은 z/OS에서 지원되지만 MQINQ 호출을 사용하여 해당 값을 판별할 수 없습니다.

### QDepthHighLimit(MQLONG)

큐 항목 수 많음 이벤트를 생성하기 위해 큐 항목 수를 비교하는 데 기준이 되는 임계값입니다.

표 603. 이 속성이 적용되는 큐 유형				
로컬	모델	알리아스	원격	클러스터
X	X			

이 이벤트는 애플리케이션이 큐에 메시지를 넣었음을 나타내며 이로 인해 큐에 있는 메시지 수가 큐 용량 상한 임계값 이상이 됩니다. QDepthHighEvent 속성을 참조하십시오.

값은 최대 큐 용량(MaxQDepth 속성)의 백분율로 표시되며 0 이상이고 100 이하입니다. 기본값은 80입니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_Q\_DEPTH\_HIGH\_LIMIT 선택자를 사용하십시오.

이 속성은 z/OS에서 지원되지만 MQINQ 호출을 사용하여 해당 값을 판별할 수 없습니다.

### QDepthLowEvent(MQLONG)

큐 항목 수 적음 이벤트 생성 여부를 제어합니다.

표 604. 이 속성이 적용되는 큐 유형				
로컬	모델	알리아스	원격	클러스터
X	X			

큐 용량 하한 이벤트는 애플리케이션이 큐에서 메시지를 검색했으며 이로 인해 큐에 있는 메시지 수가 큐 용량 하한 임계값 이하가 되었음을 나타냅니다(QDepthLowLimit attribute 속성 참조).

**참고:** 이 속성의 값은 동적으로 변경할 수 있습니다.

값은 다음 중 하나입니다.

**MQEVR\_DISABLED**

이벤트 보고를 사용하지 않습니다.

**MQEVR\_ENABLED**

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_Q\_DEPTH\_LOW\_EVENT 선택자를 사용하십시오.

이 속성은 z/OS에서 지원되지만 MQINQ 호출을 사용하여 해당 값을 판별할 수 없습니다.

**QDepthLowLimit(MQLONG)**

큐 항목 수 적음 이벤트를 생성하기 위해 큐 항목 수를 비교하는 데 기준이 되는 임계값입니다.

표 605. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

이 이벤트는 애플리케이션이 큐에서 메시지를 검색했으며 이로 인해 큐에 있는 메시지 수가 큐 용량 하한 임계값 이하가 되었음을 나타냅니다. QDepthLowEvent 속성을 참조하십시오.

값은 최대 큐 용량(MaxQDepth 속성)의 백분율로 표시되며 0 이상이고 100 이하입니다. 기본값은 20입니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_Q\_DEPTH\_LOW\_LIMIT 선택자를 사용하십시오.

이 속성은 z/OS에서 지원되지만 MQINQ 호출을 사용하여 해당 값을 판별할 수 없습니다.

**QDepthMaxEvent(MQLONG)**

이는 큐 가득 참 이벤트의 생성 여부를 제어합니다. 큐 가득 참 이벤트는 큐가 가득 차서(즉, 큐 항목 수가 이미 최대 값에 도달함) 큐에 넣기가 거부되었음을 표시합니다.

표 606. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

**참고:** 이 속성의 값은 동적으로 변경할 수 있습니다.

값은 다음 중 하나입니다.

**MQEVR\_DISABLED**

이벤트 보고를 사용하지 않습니다.

**MQEVR\_ENABLED**

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_Q\_DEPTH\_MAX\_EVENT 선택자를 사용하십시오.

이 속성은 z/OS에서 지원되지만 MQINQ 호출을 사용하여 해당 값을 판별할 수 없습니다.

**QDesc(MQCHAR64)**

자세하게 설명하는 주석을 붙이려면 이 필드를 사용하십시오.

표 607. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X	X	X	X

이 필드의 내용은 큐 관리자에 중요하지 않습니다. 그러나 큐 관리자에서 필드에 표시될 수 있는 문자만 포함되어야 할 수도 있습니다. 널(null) 문자는 포함할 수 없지만, 필요한 경우 오른쪽으로 공백을 채워넣을 수 있습니다. DBCS 설치시 필드는 DBCS 문자(최대 필드 길이 64비트에 따라)를 포함할 수 있습니다.

**참고:** 이 필드에 큐 관리자의 문자 세트에 없는 문자가 포함된 경우(**CodedCharSetId** 큐 관리자 속성에 정의된 대로), 이 필드가 다른 큐 관리자로 송신되면 해당 문자가 올바르게 않게 변환될 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_Q\_DESC 선택자를 사용하십시오. 이 속성의 길이는 MQ\_Q\_DESC\_LENGTH로 제공됩니다.

### QName(MQCHAR48)

로컬 큐 관리자에 정의된 큐의 이름입니다.

표 608. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	X

큐 관리자에 정의된 모든 큐가 같은 큐 네임스페이스를 공유합니다. 따라서 MQQT\_LOCAL 큐 및 MQQT\_ALIAS 큐에는 동일한 이름이 있을 수 없습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_Q\_NAME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다.

### QServiceInterval(MQLONG)

Service Interval High 및 Service Interval OK 이벤트를 생성하기 위한 비교에 사용되는 서비스 간격입니다.

표 609. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

[QServiceIntervalEvent](#) 속성을 참조하십시오.

값은 밀리초 단위이며 0보다 크거나 같고 999 999 999보다 작거나 같습니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_Q\_SERVICE\_INTERVAL 선택자를 사용하십시오.

이 속성은 z/OS에서 지원되지만 MQINQ 호출을 사용하여 해당 값을 판별할 수 없습니다.

### QServiceIntervalEvent(MQLONG)

서비스 간격 상위 또는 서비스 간격 확인 이벤트가 생성되는지 여부를 제어합니다.

표 610. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

- 검사 결과 최소한 **QServiceInterval** 속성에 의해 지정된 시간 동안 큐에서 검색된 메시지가 없는 것으로 표시되면 서비스 간격 높음 이벤트가 생성됩니다.
- 검사 결과 **QServiceInterval** 속성에 의해 지정된 시간 내에 큐에서 검색된 메시지가 있는 것으로 표시되면 서비스 간격 확인 이벤트가 생성됩니다.

**참고:** 이 속성의 값은 동적으로 변경할 수 있습니다.

값은 다음 중 하나입니다.

#### MQQSIE\_HIGH

큐 서비스 간격 높음 이벤트가 사용 가능합니다.

- 큐 서비스 간격 높음 이벤트를 **사용**하고
- 큐 서비스 간격 확인 이벤트를 **사용 안함**으로 설정합니다.



## MQQSIE\_OK

큐 서비스 간격 확인 이벤트가 사용 가능합니다.

- 큐 서비스 간격 높음 이벤트를 **사용**하고
- 큐 서비스 간격 확인 이벤트를 **사용**합니다.

## MQQSIE\_NONE

사용 가능한 큐 서비스 간격 이벤트가 없습니다.

- 큐 서비스 간격 높음 이벤트를 **사용**하고
- 큐 서비스 간격 확인 이벤트를 **사용 안함**으로 설정합니다.

공유 큐인 경우 이 속성 값이 무시됩니다. MQQSIE\_NONE 값이 가정됩니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_Q\_SERVICE\_INTERVAL\_EVENT 선택자를 사용하십시오.

z/OS에서 이 속성 값을 판별하기 위해 MQINQ 호출을 사용할 수 없습니다.

## QSGDisp (MQLONG)

큐의 처리를 지정합니다.

표 611. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	

값은 다음 중 하나입니다.

### MQQSGD\_Q\_MGR

오브젝트에 큐 관리자 속성 지정 값이 있습니다. 즉, 오브젝트 정의가 로컬 큐 관리자에만 인식되고 큐 공유 그룹의 다른 큐 관리자에는 인식되지 않습니다.

큐 공유 그룹의 각 큐 관리자는 현재 오브젝트와 이름 및 유형이 동일한 오브젝트를 보유할 수 있지만, 이는 개별 오브젝트이며 이들 간에는 상관 관계가 없습니다. 속성이 서로 동일해야 한다는 제한 사항은 없습니다.


### MQQSGD\_COPY

오브젝트는 공유 저장소에 존재하는 마스터 오브젝트 정의의 로컬 사본입니다. 큐 공유 그룹의 각 큐 관리자는 오브젝트의 자체 사본을 가질 수 있습니다. 처음에는 모든 사본이 동일한 속성을 가지지만 MQSC 명령을 사용하여 사본의 속성을 기타 사본의 속성과 다르게 설정할 수 있습니다. 사본의 속성은 공유 저장소 내의 마스터 정의가 대체될 때 다시 동기화됩니다.

### MQQSGD\_SHARED

오브젝트에 공유 속성 지정 값이 있습니다. 즉, 큐 공유 그룹의 모든 큐 관리자에 인식되는 오브젝트의 단일 인스턴스가 공유 저장소에 있습니다. 그룹의 큐 관리자가 이 오브젝트에 액세스할 때에는 오브젝트의 단일 공유 인스턴스에 액세스하게 됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_QSG\_DISP 선택자를 사용하십시오.

 이 속성은 z/OS에서만 지원됩니다.

## QueueAccounting(MQLONG)

표 612. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X	X	X	

이는 큐에 대한 계정 데이터 콜렉션을 제어합니다. 이 큐에 대해 수집할 계정 데이터의 경우 이 연결의 계정 데이터가 MQCONNX 호출에서 MQCNO 구조의 옵션 필드 또는 QMGR 속성 ACCTQ를 사용하여 사용 가능해야 합니다.



이 속성의 값은 다음 중 하나입니다.

**MQMON\_Q\_MGR**

이 큐에 대한 계정 데이터는 QMGR 속성 ACCTQ의 설정을 기반으로 수집됩니다. 디폴트 설정입니다.

**MQMON\_OFF**

이 큐에 대한 계정 데이터를 수집하지 마십시오.

**MQMON\_ON**

이 큐에 대한 계정 데이터를 수집하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_ACCOUNTING\_Q 선택자를 사용하십시오.

**QueueMonitoring(MQLONG)**

큐에 대한 온라인 모니터링 데이터의 콜렉션을 제어합니다.

표 613. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

값은 다음 중 하나입니다.

**MQMON\_Q\_MGR**

**QueueMonitoring** 큐 관리자 속성의 설정에 따라 모니터링 데이터를 수집하십시오. 이는 기본값입니다.

**MQMON\_OFF**

해당 큐에 대한 온라인 모니터링 데이터 콜렉션이 꺼집니다.

**MQMON\_LOW**

**QueueMonitoring** 큐 관리자의 속성 값이 MQMON\_NONE이 아닌 경우 이 큐에 대해 낮은 비율의 데이터 콜렉션으로 온라인 모니터링 데이터 콜렉션이 켜집니다.

**MQMON\_MEDIUM**

**QueueMonitoring** 큐 관리자의 속성 값이 MQMON\_NONE이 아닌 경우 이 큐에 대해 중간 비율의 데이터 콜렉션으로 온라인 모니터링 데이터 콜렉션이 켜집니다.

**MQMON\_HIGH**

**QueueMonitoring** 큐 관리자의 속성 값이 MQMON\_NONE이 아닌 경우 이 큐에 대해 높은 비율의 데이터 콜렉션으로 온라인 모니터링 데이터 콜렉션이 켜집니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_MONITORING\_Q 선택자를 사용하십시오.

**QueueStatistics (MQCHAR12)**

표 614. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X	X	X	

이는 큐에 대한 통계 데이터 콜렉션을 제어합니다.

이 속성의 값은 다음 중 하나입니다.

**MQMON\_Q\_MGR**

이 큐에 대한 계정 데이터는 QMGR 속성 STATQ의 설정을 기반으로 수집됩니다. 디폴트 설정입니다.

**MQMON\_OFF**

이 큐에 대한 통계 데이터 콜렉션을 끄십시오.

**MQMON\_ON**

이 큐에 대한 통계 데이터 콜렉션을 사용 가능하게 하십시오.

**QType(MQLONG)**

표 615. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	X

이는 큐의 유형입니다. 다음 값 중 하나를 가지고 있습니다.

**MQQT\_ALIAS**

알리어스 큐 정의입니다.

**MQQT\_CLUSTER**

클러스터 큐.

**MQQT\_LOCAL**

로컬 큐.

**MQQT\_REMOTE**

리모트 큐의 로컬 정의입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_Q\_TYPE 선택자를 사용하십시오.

**RemoteQMgrName(MQCHAR48)**

표 616. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
			X	

큐 **RemoteQName**이 정의된 리모트 큐 관리자의 이름입니다. **RemoteQName** 큐에 MQQSGD\_COPY 또는 MQQSGD\_SHARED의 **QSGDisp** 값이 있는 경우 **RemoteQMgrName**은 **RemoteQName**을 소유하는 큐 공유 그룹의 이름일 수 있습니다.

애플리케이션이 리모트 큐의 로컬 정의를 열면 **RemoteQMgrName**이 공백이 아니거나 로컬 큐 관리자의 이름이 아니어야 합니다. **XmitQName**이 공백이면 **RemoteQMgrName**과 이름이 같은 로컬 큐가 전송 큐로 사용됩니다. **RemoteQMgrName** 이름의 큐가 없으면 **DefXmitQName** 큐 관리자 속성으로 식별된 큐가 사용됩니다.

이 정의가 큐 관리자 알리어스에 사용되는 경우, **RemoteQMgrName**은(는) 별명이 지정된 큐 관리자의 이름입니다. 로컬 큐 관리자의 이름일 수 있습니다. 그렇지 않은 경우 열기가 발생할 때 **XmitQName**이 비어 있다면 **RemoteQMgrName**과 같은 이름을 가진 로컬 큐가 있어야 합니다. 이 큐는 전송 큐로 사용됩니다.

이 정의가 응답 대상 알리어스 사용되는 경우 이 이름은 **ReplyToQMgr**이 될 큐 관리자의 이름입니다.

**참고:** 큐 정의가 작성되거나 수정된 경우, 이 속성에 대해 지정된 값에 대해 유효성 검증이 수행되지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_REMOTE\_Q\_MGR\_NAME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_Q\_MGR\_NAME\_LENGTH에서 제공됩니다.

**RemoteQName (MQCHAR48)**

표 617. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
			X	

이 이름은 리모트 큐 관리자 **RemoteQMgrName**에 알려진 큐의 이름입니다.

애플리케이션이 리모트 큐의 로컬 정의를 열면 열기가 발생할 때 **RemoteQName**이(가) 공백이 아니어야 합니다.

큐 관리자 알리어스 정의에 이 정의가 사용되는 경우, 열기가 발생할 때 **RemoteQName**이(가) 공백이어야 합니다.

정의가 응답 대상 알리어스에 사용되는 경우, 이 이름은 **ReplyToQ**이(가) 될 큐의 이름입니다.

**참고:** 큐 정의가 작성되거나 수정된 경우, 이 속성에 대해 지정된 값에 대해 유효성 검증이 수행되지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_REMOTE\_Q\_NAME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다.

### RetentionInterval(MQLONG)

이는 큐를 유지하는 시간의 기간입니다. 이 시간이 경과하면 큐를 삭제할 수 있습니다.

표 618. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

시간은 큐를 작성한 날짜 및 시간부터 시간 단위로 측정됩니다. 큐의 작성일 및 시간은 **CreationDate** 및 **CreationTime** 속성에서 기록됩니다.

이 정보는 보조관리 애플리케이션 또는 운영자가 더 이상 필요하지 않은 큐를 식별하고 삭제할 수 있도록 제공됩니다.

**참고:** 큐 관리자는 이 속성을 기반으로 큐를 삭제하거나 만료되지 않은 유지 간격을 가진 큐의 삭제를 방지하기 위한 조치를 취하지 않습니다. 필요한 조치를 취하는 것은 사용자의 책임입니다.

영구적 동적 큐의 누적을 방지하기 위해 현실적 유지 기간을 사용하십시오(DefinitionType 속성 참조). 그러나, 사전정의된 큐에도 이 속성을 사용할 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_RETENTION\_INTERVAL 선택자를 사용하십시오.

### Scope(MQLONG)

이 큐의 항목이 셀 디렉토리에 있는지 제어합니다.

표 619. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	

셀 디렉토리가 설치 가능한 이름 서비스에 의해 제공됩니다. 값은 다음 중 하나입니다.

#### MQSCO\_Q\_MGR

큐 정의에는 큐 관리자 범위가 있습니다. 큐 정의가 큐를 소유하는 큐 관리자의 범위를 넘어서 확장되지 않습니다. 일부 다른 큐 관리자로부터 출력을 위해 큐를 열려면, 소유하는 큐 관리자의 이름을 지정해야 하고, 다른 큐 관리자에 큐의 로컬 정의가 있어야 합니다.

#### MQSCO\_CELL

큐 정의에는 셀 범위가 있습니다. 큐 정의는 또한 셀에서 모든 큐 관리자에 사용 가능한 셀 디렉토리에 위치합니다. 큐는 큐의 이름을 지정하여 셀에서 큐 관리자의 출력에 대해 열 수 있습니다. 큐를 소유하는 큐 관리자의 이름은 지정할 필요가 없습니다. 그러나 로컬 정의가 우선하므로 또한 해당 이름의 큐에 대한 로컬 정의가 있는 셀의 큐 관리자는 큐 정의를 사용할 수 없습니다.

셀 디렉토리가 설치 가능한 이름 서비스에 의해 제공됩니다.

모델과 동적 큐에는 셀 범위를 포함할 수 없습니다.

이 값은 셀 디렉토리를 지원하는 이름 서비스가 구성된 경우에만 유효합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_SCOPE 선택자를 사용하십시오.

이 속성의 지원에는 다음 제한이 적용됩니다.

- IBM i에서 속성은 지원되지만 MQSCO\_Q\_MGR만 유효합니다.
- z/OS에서 속성은 지원되지 않습니다.

### Shareability(MQLONG)

동시에 여러 번 입력하기 위해 큐를 열 수 있는지 나타냅니다.

표 620. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

값은 다음 중 하나입니다.

**MQQA\_SHAREABLE**

큐를 공유할 수 있습니다.

MQOO\_INPUT\_SHARED 옵션이 있는 다중 열기가 허용됩니다.

**MQQA\_NOT\_SHAREABLE**

큐를 공유할 수 없습니다.

MQOO\_INPUT\_SHARED 옵션이 있는 MQOPEN 호출은 MQOO\_INPUT\_EXCLUSIVE로 처리됩니다.


이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_SHAREABILITY 선택자를 사용하십시오.

**StorageClass (MQCHAR8)**

이는 큐를 소유하는 데 사용되는 물리적 스토리지를 정의하는 사용자 정의된 이름입니다. 실제로 메모리 버퍼에서 페이지 아웃해야 하는 경우에만 메시지를 디스크에 씁니다.

표 621. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_STORAGE\_CLASS 선택자를 사용하십시오. 이 속성의 길이는 MQ\_STORAGE\_CLASS\_LENGTH로 제공됩니다.

 이 속성은 z/OS에서만 지원됩니다.

**TriggerControl(MQLONG)**

큐를 서비스하는 애플리케이션을 시작하기 위해 트리거 메시지를 시작 큐에 쓰는지 제어합니다.

표 622. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

다음 중 하나입니다.

**MQTC\_OFF**

이 큐에는 트리거 메시지가 기록되지 않습니다. 이 경우, *TriggerType* 값은 관계가 없습니다.

**MQTC\_ON**

트리거 메시지는 적절한 트리거 이벤트가 발생할 때 이 큐에 대해 작성됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_TRIGGER\_CONTROL 선택자를 사용하십시오. 이 속성의 값을 변경하려면 MQSET 호출을 사용하십시오.

**TriggerData (MQCHAR64)**

메시지가 이 큐에 도착하여 트리거 메시지가 이니시에이션 큐에 기록될 때 큐 관리자가 트리거 메시지에 삽입하는 자유 형식 데이터입니다.

표 623. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

이 데이터의 콘텐츠는 큐 관리자에 중요하지 않습니다. 이니시에이션 큐를 처리하는 트리거 모니터 애플리케이션 또는 트리거 모니터를 시작하는 애플리케이션에 의미가 있습니다.

문자열에 널이 없어야 합니다. 필요한 경우, 오른쪽에 공백을 채워 넣어야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_TRIGGER\_DATA 선택자를 사용하십시오. 이 속성의 값을 변경하려면 MQSET 호출을 사용하십시오. 이 속성의 길이는 MQ\_TRIGGER\_DATA\_LENGTH로 제공됩니다.

### TriggerDepth(MQLONG)

표 624. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

우선순위가 *TriggerMsgPriority* 이상이고 트리거 메시지가 기록되기 전에 큐에 있어야 하는 메시지의 수입니다. 이는 *TriggerType*이 MQTT\_DEPTH로 설정될 때 적용됩니다. *TriggerDepth*의 값은 1 이상입니다. 그 밖의 경우에는 이 속성이 사용되지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_TRIGGER\_DEPTH 선택자를 사용하십시오. 이 속성의 값을 변경하려면 MQSET 호출을 사용하십시오.

### TriggerMsgPriority(MQLONG)

메시지가 트리거 메시지 생성에 기여하지 않는 메시지 우선순위입니다(즉, 트리거 메시지를 생성할지 여부를 결정할 때 큐 관리자가 이 메시지를 무시함).

표 625. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

*TriggerMsgPriority*는 0(최하위) - *MaxPriority* (최상위, *MaxPriority* 속성 참조) 범위일 수 있고 0값은 모든 메시지가 트리거 메시지의 생성에 기여하도록 합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_TRIGGER\_MSG\_PRIORITY 선택자를 사용하십시오. 이 속성의 값을 변경하려면 MQSET 호출을 사용하십시오.

### TriggerType(MQLONG)

이 큐에 도착하는 메시지의 결과로 트리거 메시지가 기록되는 조건을 제어합니다.

표 626. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

값은 다음 중 하나입니다.

#### MQTT\_NONE

이 큐에 있는 메시지의 결과로 트리거 메시지가 기록되지 않습니다. 이는 *TriggerControl*을 MQTC\_OFF로 설정하는 것과 동일한 효과를 가집니다.

#### MQTT\_FIRST

큐에서 우선순위가 *TriggerMsgPriority* 이상인 메시지의 수가 0에서 1로 변경될 때마다 트리거 메시지가 기록됩니다.

#### MQTT EVERY

우선순위가 *TriggerMsgPriority* 이상인 메시지가 큐에 도착할 때마다 트리거 메시지가 기록됩니다.

## MQTT\_DEPTH

큐에서 우선순위가 *TriggerMsgPriority* 이상인 메시지의 수가 *TriggerDepth* 이상일 때마다 트리거 메시지가 기록됩니다. 트리거 메시지가 작성된 후 *TriggerControl*은 명시적으로 다시 켜질 때까지 트리거를 방지하기 위해 MQTC\_OFF로 설정됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_TRIGGER\_TYPE 선택자를 사용하십시오. 이 속성의 값을 변경하려면 MQSET 호출을 사용하십시오.

## Usage(MQLONG)

큐가 사용되는 대상을 표시합니다.

표 627. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

값은 다음 중 하나입니다.

## MQUS\_NORMAL

이는 메시지를 넣고 가져올 때 애플리케이션이 사용하는 큐입니다. 이 큐는 전송 큐가 아닙니다.

## MQUS\_TRANSMISSION

리모트 큐 관리자로 이동하는 메시지를 보관하는 데 사용되는 큐입니다. 애플리케이션이 메시지를 리모트 큐에 보낼 때 로컬 큐 관리자는 특정 형식으로 메시지를 적절한 전송 큐에 임시로 저장합니다. 그러면 메시지 채널 에이전트가 전송 큐에서 메시지를 읽어오고, 이 메시지를 리모트 큐 관리자로 전송합니다. 원격 관리 구성에 대한 자세한 정보는 원격 관리를 위한 큐 관리자 구성을 참조하십시오.

권한이 있는 애플리케이션만 직접적으로 메시지를 넣기 위해 MQOO\_OUTPUT에 대한 전송 큐를 열 수 있습니다. 일반적으로 유틸리티 애플리케이션만 이를 수행합니다. 메시지 데이터 형식이 올바른지(597 페이지의 『MQXQH - 전송 큐 헤더』 참조) 또는 전송 처리 중 오류가 발생할 수 있는지 확인하십시오.

MQPMO\_\*\_CONTEXT 컨텍스트 옵션 중 하나가 지정되지 않는 한 컨텍스트는 전달되거나 설정되지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_USAGE 선택자를 사용하십시오.

## XmitQName (MQCHAR48)

이는 전송 큐 이름입니다. 리모트 큐 또는 큐 관리자 알리어스 정의에 대해 열릴 때 이 속성이 공백이 아니면 메시지를 전달하는 데 사용할 로컬 전송 큐의 이름을 지정합니다.

표 628. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
			X	

**XmitQName**이 공백인 경우 **RemoteQMgrName**과 동일한 이름을 가진 로컬 큐를 전송 큐로 사용합니다.

**RemoteQMgrName** 이름의 큐가 없으면 **DefXmitQName** 큐 관리자 속성으로 식별된 큐가 사용됩니다.

정의를 큐 관리자 알리어스로 사용 중이고 **RemoteQMgrName**이 로컬 큐 관리자의 이름인 경우 이 속성이 무시됩니다. 또한 정의가 응답 대상 큐 알리어스 정의로 사용되는 경우에도 무시됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_XMIT\_Q\_NAME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다.

## 이름 목록에 대한 속성

다음 표는 이름 목록에 특정한 속성을 요약합니다. 속성은 알파벳 순서로 설명합니다.

이름 목록은 모든 IBM MQ 시스템 및 이러한 시스템에 연결된 IBM MQ MQI clients 에서 지원됩니다.

**참고:** 이 절에 표시된 속성 이름은 MQINQ 및 MQSET 호출과 함께 사용되는 설명 이름입니다. 이 이름은 PCF 명령과 동일합니다. 속성을 정의, 대체 또는 표시하기 위해 MQSC 명령이 사용되는 경우, 대체 짧은 이름이 사용됩니다. 자세한 정보는 MQSC 명령을 참조하십시오.

표 629. 이름 목록에 대한 속성	
속성	설명
AlterationDate	정의가 마지막으로 변경된 날짜
AlterationTime	정의가 마지막으로 변경된 시간
NameCount	이름 목록의 이름 수
NamelistDesc	이름 목록 설명
NamelistName	이름 목록 이름
Names	NameCount 이름 목록
NamelistType	이름 목록 유형
QSGDisp	큐 공유 그룹 속성 지정

### **AlterationDate (MQCHAR12)**

정의가 마지막으로 변경된 날짜입니다. 날짜의 형식은 YYYY-MM-DD이며 길이를 12바이트로 만들기 위해 두 개의 후미 공백으로 채워집니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_ALTERATION\_DATE 선택자를 사용하십시오. 이 속성의 길이는 MQ\_DATE\_LENGTH에서 제공됩니다.

### **AlterationTime (MQCHAR8)**

정의가 마지막으로 변경된 시간입니다. 시간의 형식은 HH.MM.SS입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_ALTERATION\_TIME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_TIME\_LENGTH에서 제공됩니다.

### **NameCount(MQLONG)**

이는 이름 목록의 이름 수입니다. 이는 0보다 크거나 같습니다. 다음 값이 정의됩니다.

#### **MQNC\_MAX\_NAMELIST\_NAME\_COUNT**

이름 목록에 있는 최대 이름 수.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_NAME\_COUNT 선택자를 사용하십시오.

### **NamelistDesc (MQCHAR64)**

설명 주석에 이 필드를 사용하십시오. 해당 값은 정의 프로세스에 의해 설정됩니다. 이 필드의 내용은 큐 관리자에 중요하지 않습니다. 그러나 큐 관리자에서 필드에 표시될 수 있는 문자만 포함되어야 할 수도 있습니다. 널 (null) 문자는 포함할 수 없지만, 필요한 경우 오른쪽으로 공백을 채워넣을 수 있습니다. DBCS 설치 시 이 필드는 DBCS 문자(최대 필드 길이 64비트에 따라)를 포함할 수 있습니다.

**참고:** 이 필드에 큐 관리자의 문자 세트에 없는 문자가 포함된 경우(CodedCharSetId 큐 관리자 속성에 정의된 대로), 이 필드가 다른 큐 관리자로 송신되면 해당 문자가 올바르게 않게 변환될 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_NAMELIST\_DESC 선택자를 사용하십시오.

이 속성의 길이는 MQ\_NAMELIST\_DESC\_LENGTH로 제공됩니다.

### **NamelistName (MQCHAR48)**

이는 로컬 큐 관리자에서 정의된 이름 목록의 이름입니다. 이름 목록 이름에 대한 자세한 정보는 [기타 오브젝트 이름 절](#)을 참조하십시오.

각 이름 목록에 큐 관리자에 속하는 기타 이름 목록의 이름과 다른 이름이 있지만, 여기에서 큐와 같이 다른 유형의 기타 큐 관리자 오브젝트의 이름을 복제할 수도 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_NAMELIST\_NAME 선택자를 사용하십시오.

이 속성의 길이는 MQ\_NAMELIST\_NAME\_LENGTH로 제공됩니다.

### **NamelistType(MQLONG)**



이는 이름 목록에서 이름의 네이처를 지정하고 이름 목록이 사용되는 방법을 표시합니다. 다음 값 중 하나입니다.

**MQNT\_NONE**

지정된 유형의 이름 목록이 없습니다.

**MQNT\_Q**

큐의 이름을 포함하는 이름 목록입니다.


**MQNT\_CLUSTER**

클러스터의 이름을 포함하는 이름 목록입니다.

**MQNT\_AUTH\_INFO**

인증 정보 오브젝트의 이름을 포함하는 이름 목록입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQIA\_NAMELIST\_TYPE 선택자를 사용하십시오.

 이 속성은 z/OS에서만 지원됩니다.

**Names(MQCHAR48xNameCount)**

이는 각 이름이 로컬 큐 관리자에 정의된 오브젝트의 이름인 *NameCount* 이름의 목록입니다. 오브젝트 이름에 대한 자세한 정보는 [IBM MQ 오브젝트 이름 지정 규칙](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_NAMES 선택자를 사용하십시오.

목록에 있는 각 이름의 길이는 MQ\_OBJECT\_NAME\_LENGTH로 지정됩니다.

**QSGDisp (MQLONG)**

이름 목록의 속성 지정 값을 지정합니다. 값은 다음 중 하나입니다.

**MQQSGD\_Q\_MGR**


오브젝트에 큐 관리자 속성 지정 값이 있고 오브젝트 정의는 로컬 큐 관리자에만 알려지며 이 정의는 큐 공유 그룹 내의 기타 큐 관리자에는 알려지지 않습니다.

큐 공유 그룹의 각 큐 관리자는 현재 오브젝트와 이름 및 유형이 동일한 오브젝트를 보유할 수 있지만, 이는 개별 오브젝트이며 이들 간에는 상관 관계가 없습니다. 속성이 서로 동일해야 한다는 제한 사항은 없습니다.

**MQQSGD\_COPY**

오브젝트는 공유 저장소에 존재하는 마스터 오브젝트 정의의 로컬 사본입니다. 큐 공유 그룹의 각 큐 관리자는 오브젝트의 자체 사본을 가질 수 있습니다. 처음에는 모든 사본에 동일한 속성이 있지만 MQSC 명령을 사용하여 사본의 속성을 기타 사본의 속성과 다르게 대체할 수 있습니다. 사본의 속성은 공유 저장소 내의 마스터 정의가 대체될 때 다시 동기화됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_QSG\_DISP 선택자를 사용하십시오.

 이 속성은 z/OS에서만 지원됩니다.

**프로세스 정의에 대한 속성**

다음 표는 프로세스 정의에 특정한 속성을 요약합니다. 속성은 알파벳순으로 설명합니다.

**참고:** 이 절에 표시된 속성 이름은 MQINQ 및 MQSET 호출과 함께 사용되는 설명 이름입니다. 이 이름은 PCF 명령과 동일합니다. 속성을 정의, 대체 또는 표시하기 위해 MQSC 명령이 사용되는 경우, 대체 짧은 이름이 사용됩니다. 자세한 정보는 [MQSC 명령](#)을 참조하십시오.

표 630. 프로세스 정의에 대한 속성	
속성	설명
AlterationDate	정의가 마지막으로 변경된 날짜
AlterationTime	정의가 마지막으로 변경된 시간
AppId	애플리케이션 ID
AppType	애플리케이션 유형
EnvData	환경 데이터.



표 630. 프로세스 정의에 대한 속성 (계속)	
속성	설명
ProcessDesc	프로세스 설명
ProcessName	프로세스 이름
QSGDisp	큐 공유 그룹 속성 지정
UserData	사용자 데이터

### **AlterationDate (MQCHAR12)**

정의가 마지막으로 변경된 날짜입니다. 날짜의 형식은 YYYY-MM-DD이며 길이를 12바이트로 만들기 위해 두 개의 후미 공백으로 채워집니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_ALTERATION\_DATE 선택자를 사용하십시오. 이 속성의 길이는 MQ\_DATE\_LENGTH에서 제공됩니다.

### **AlterationTime (MQCHAR8)**

정의가 마지막으로 변경된 시간입니다. 시간의 형식은 HH.MM.SS입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_ALTERATION\_TIME 선택자를 사용하십시오. 이 속성의 길이는 MQ\_TIME\_LENGTH에서 제공됩니다.

### **ApplId (MQCHAR256)**

시작할 애플리케이션을 식별하는 문자열입니다. 이 정보는 이니시에이션 큐에서 메시지를 처리하는 트리거 모니터 애플리케이션에서 사용됩니다. 정보가 트리거 메시지의 일부로서 이니시에이션 큐에 송신됩니다.

*ApplId*의 의미는 트리거 모니터 애플리케이션에 의해 판별됩니다. IBM MQ에서 제공되는 트리거 모니터는 *ApplId*를 실행 가능 프로그램의 이름으로 설정해야 합니다. 다음 참고사항은 표시된 환경에서 적용됩니다.

- z/OS에서 *ApplId*는 다음이어야 합니다.
  - CICS 트랜잭션 ID, CICS 트리거 모니터 트랜잭션 CKTI를 사용하여 시작된 애플리케이션의 경우
  - IMS 트랜잭션 ID, IMS 트리거 모니터 CSQQTRMN을 사용하여 시작된 애플리케이션의 경우
- Windows에서 프로그램 이름 앞에 드라이브 및 디렉토리 경로가 접두부로 올 수 있습니다.
- AIX and Linux에서 프로그램 이름에 디렉토리 경로를 접두부로 지정할 수 있습니다.

널을 포함할 수 없는 문자열입니다. 필요한 경우, 오른쪽에 공백을 채워 넣어야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_APPL\_ID 선택자를 사용하십시오. 이 속성의 길이는 MQ\_PROCESS\_APPL\_ID\_LENGTH로 제공됩니다.

### **ApplType(MQLONG)**

트리거 메시지 수신에 대한 응답으로 시작할 프로그램의 네이처를 식별합니다. 이 정보는 이니시에이션 큐에서 메시지를 처리하는 트리거 모니터 애플리케이션에서 사용됩니다. 정보가 트리거 메시지의 일부로서 이니시에이션 큐에 송신됩니다.

*ApplType*에 값이 있지만 다음 값이 표준 유형에 권장됩니다. MQAT\_USER\_FIRST - MQAT\_USER\_LAST 범위의 값에 사용자 정의 애플리케이션 유형을 제한하십시오.

#### **MQAT\_AIX**

AIX 애플리케이션(MQAT\_UNIX와 동일한 값).

#### **MQAT\_BATCH**

배치 애플리케이션입니다.

#### **MQAT\_CICS**

CICS 트랜잭션.

**MQAT\_IMS**

IMS 애플리케이션입니다.

**MQAT\_IMS\_BRIDGE**

IMS 브릿지 애플리케이션입니다.

**MQAT\_JAVA**

Java 애플리케이션입니다.

**MQAT\_MVS**

MVS 또는 TSO 애플리케이션(MQAT\_ZOS와 동일한 값)입니다.

**MQAT\_OS390**

OS/390 애플리케이션(MQAT\_ZOS와 동일한 값)입니다.

**MQAT\_OS400**

IBM i 애플리케이션입니다.

**MQAT\_UNIX**

UNIX 애플리케이션입니다.

**MQAT\_UNKNOWN**

알 수 없는 유형의 애플리케이션입니다.

**MQAT\_USER**

사용자 애플리케이션입니다.

**MQAT\_WINDOWS**

64비트 Windows 애플리케이션.

**MQAT\_WINDOWS\_NT**

32비트 Windows 애플리케이션입니다.

**MQAT\_WLM**

z/OS 워크로드 관리자 애플리케이션입니다.

**MQAT\_ZOS**

z/OS 애플리케이션입니다.

**MQAT\_USER\_FIRST**

사용자 정의된 애플리케이션 유형의 최저값입니다.

**MQAT\_USER\_LAST**

사용자 정의된 애플리케이션 유형의 최고값입니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_APPL\_TYPE 선택자를 사용하십시오.

**EnvData (MQCHAR128)**

시작할 애플리케이션에 적용되는 환경 관련 정보를 포함하는 문자열입니다. 이 정보는 이니시에이션 큐에서 메시지를 처리하는 트리거 모니터 애플리케이션에서 사용됩니다. 정보가 트리거 메시지의 일부로서 이니시에이션 큐에 송신됩니다.

*EnvData*의 의미는 트리거 모니터 애플리케이션에 의해 판별됩니다. IBM MQ이(가) 제공하는 트리거 모니터는 시작된 애플리케이션에 전달된 매개변수 목록에 *EnvData*을(를) 추가합니다. 매개변수 목록은 MQTMC2 구조로 구성되며 뒤에 공백이 있고 뒤에 공백이 있는 *EnvData*이(가) 제거됩니다. 다음 참고사항은 표시된 환경에서 적용됩니다.

- z/OS의 경우:
  - *EnvData*는 IBM MQ에서 제공한 트리거 모니터 애플리케이션에서 사용되지 않습니다.
  - ApplType이 MQAT\_WLM인 경우 작업 정보 헤더(MQWIH)에서 ServiceNam 및 ServiceStep 필드에 *EnvData*의 기본값을 제공할 수 있습니다.
- AIX and Linux에서 *EnvData*를 & 문자로 설정하여 백그라운드에서 시작된 애플리케이션을 실행할 수 있습니다.

널을 포함할 수 없는 문자열입니다. 필요한 경우, 오른쪽에 공백을 채워 넣어야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_ENV\_DATA 선택자를 사용하십시오. 이 속성의 길이는 MQ\_PROCESS\_ENV\_DATA\_LENGTH로 제공됩니다.

### **ProcessDesc (MQCHAR64)**

설명 주석에 이 필드를 사용하십시오. 이 필드의 내용은 큐 관리자에 중요하지 않습니다. 그러나 큐 관리자에서 필드에 표시될 수 있는 문자만 포함되어야 할 수도 있습니다. 널(null) 문자는 포함할 수 없지만, 필요한 경우 오른쪽으로 공백을 채워넣을 수 있습니다. DBCS 설치시 필드는 DBCS 문자(최대 필드 길이 64비트에 따라)를 포함할 수 있습니다.

**참고:** 이 필드에 큐 관리자의 문자 세트에 없는 문자가 포함된 경우(CodedCharSetId 큐 관리자 속성에 정의된 대로), 이 필드가 다른 큐 관리자로 송신되면 해당 문자가 올바르게 않게 변환될 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 MQCA\_PROCESS\_DESC 선택자를 사용하십시오.

이 속성의 길이는 MQ\_PROCESS\_DESC\_LENGTH로 제공됩니다.

### **ProcessName (MQCHAR48)**

로컬 큐 관리자에 정의된 프로세스 정의의 이름입니다.

각 프로세스 정의에는 큐 관리자에 속하는 기타 프로세스 정의의 이름과 다른 이름이 있습니다. 그러나 프로세스 정의의 이름은 큐와 같이 다른 유형의 기타 큐 관리자 오브젝트의 이름과 동일할 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_PROCESS\_NAME 선택자를 사용하십시오.

이 속성의 길이는 MQ\_PROCESS\_NAME\_LENGTH로 제공됩니다.

### **QSGDisp (MQLONG)**

이는 프로세스 정의의 속성 지정 값을 지정합니다. 값은 다음 중 하나입니다.

#### **MQQSGD\_Q\_MGR**


오브젝트에 큐 관리자 속성 지정 값이 있고 오브젝트 정의는 로컬 큐 관리자에만 알려지며 이 정의는 큐 공유 그룹 내의 기타 큐 관리자에는 알려지지 않습니다.

큐 공유 그룹의 각 큐 관리자는 현재 오브젝트와 이름 및 유형이 동일한 오브젝트를 보유할 수 있지만, 이는 개별 오브젝트이며 이들 간에는 상관 관계가 없습니다. 속성이 서로 동일해야 한다는 제한 사항은 없습니다.

#### **MQQSGD\_COPY**

오브젝트는 공유 저장소에 존재하는 마스터 오브젝트 정의의 로컬 사본입니다. 큐 공유 그룹의 각 큐 관리자는 오브젝트의 자체 사본을 가질 수 있습니다. 처음에는 모든 사본에 동일한 속성이 있지만 MQSC 명령을 사용하여 사본의 속성을 기타 사본의 속성과 다르게 대체할 수 있습니다. 사본의 속성은 공유 저장소 내의 마스터 정의가 대체될 때 다시 동기화됩니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQIA\_QSG\_DISP 선택자를 사용하십시오.

 이 속성은 z/OS에서만 지원됩니다.

### **UserData (MQCHAR128)**

사용자 데이터는 시작할 애플리케이션에 적용되는 사용자 정보를 포함하는 문자열입니다. 이 정보는 이니시에이션 큐에서 메시지를 처리하는 트리거 모니터 애플리케이션 또는 트리거 모니터에 의해 시작된 애플리케이션에 의해 사용됩니다. 정보가 트리거 메시지의 일부로 이니시에이션 큐에 송신됩니다.

*UserData*의 의미는 트리거 모니터 애플리케이션에 의해 판별됩니다. IBM MQ에서 제공되는 트리거 모니터는 매개변수 목록의 일부로 *UserData*를 시작된 애플리케이션으로 전달합니다. 매개변수 목록은 MQTMC2 구조 (*UserData* 포함)뒤에 하나의 공백이 있고 뒤에 오는 공백이 있는 *EnvData*이(가) 제거됩니다.

널을 포함할 수 없는 문자열입니다. 필요한 경우, 오른쪽에 공백을 채워 넣어야 합니다. Microsoft Windows에 대해 프로세스 정의가 *runmqtrm*에 전달되는 경우 문자 문자열은 큰따옴표 표시를 포함할 수 없습니다.

이 속성의 값을 판별하려면 MQINQ 호출과 함께 MQCA\_USER\_DATA 선택자를 사용하십시오. 이 속성의 길이는 MQ\_PROCESS\_USER\_DATA\_LENGTH로 제공됩니다.

## 리턴 코드

각 MQI(IBM MQ Message Queue Interface) 및 MQAI(IBM MQ Administration Interface) 호출의 경우 **완료 코드** 및 **이유 코드**는 호출의 성공 또는 실패를 나타내기 위해 큐 관리자 또는 엑시트 루틴으로 리턴됩니다.

특히 명시된 경우를 제외하고, 오류를 확인하는 특정 순서에 따라 애플리케이션이 달라지지 않아야 합니다. 둘 이상의 완료 코드 또는 이유 코드가 호출에서 발생할 수 있는 경우, 보고되는 특정 오류는 구현에 따라 다릅니다.

IBM MQ API 호출에 따라 성공적으로 완료되었는지 확인하는 애플리케이션은 항상 완료 코드를 확인해야 합니다. 이유 코드 값을 기반으로 완료 코드 값을 가정하지 마십시오.

### 완료 코드

완료 코드 매개변수(*CompCode*)를 사용하여 호출자가 호출이 성공적으로 완료되었는지 부분적으로 완료되었는지 또는 실패되었는지를 빨리 확인할 수 있습니다. 다음은 호출 설명에 제공된 것보다 자세한 정보로 완료 코드 목록을 나열합니다.

#### MQCC\_OK

호출이 완전히 완료되었습니다. 모든 출력 매개변수가 설정되었습니다. 이 경우 **Reason** 매개변수에는 항상 MQRC\_NONE 값이 있습니다.

#### MQCC\_WARNING

호출이 부분적으로 완료되었습니다. *CompCode* 및 *Reason* 출력 매개변수 외에 일부 출력 매개변수가 설정될 수 있습니다. **Reason** 매개변수는 부분 완료에 관한 추가 정보를 제공합니다.

#### MQCC\_FAILED

호출 처리가 완료되지 않았습니다. 특별히 기록된 곳을 제외하고는 큐 관리자의 상태가 변경되지 않습니다. *CompCode* 및 *Reason* 출력 매개변수가 설정되었습니다. 기록된 곳을 제외하고는 다른 매개변수는 변경되지 않습니다.

이유가 애플리케이션 프로그램에서의 결함이거나 프로그램에 대한 외부적인 일부 상황의 결과(예: 사용자의 권한이 해지될 수 있음)일 수 있습니다. **Reason** 매개변수는 오류에 관한 추가 정보를 제공합니다.

## 이유 코드

이유 코드 매개변수 (*Reason*)는 완료 코드 매개변수(*CompCode*)를 제한합니다.

보고할 특정 이유가 없는 경우 MQRC\_NONE이 리턴됩니다. 성공적인 호출은 MQCC\_OK 및 MQRC\_NONE을 리턴합니다.

완료 코드가 MQCC\_WARNING 또는 MQCC\_FAILED인 경우 큐 관리자는 항상 규정 이유를 보고합니다. 각 호출 설명 아래 자세한 내용이 제공되어 있습니다.

사용자 엑시트 루틴이 완료 코드와 이유를 설정한 곳에서는 이 규칙을 준수해야 합니다. 또한 사용자 엑시트에서 정의된 특별한 이유 값은 0(영)보다 작아야 큐 관리자에서 정의한 값과 충돌되지 않습니다. 해당되는 경우 엑시트는 큐 관리자에서 이미 정의한 이유를 설정할 수 있습니다.

이유 코드가 다음 위치에도 있을 수 있습니다.

- MQDLH 구조의 *Reason* 필드
- MQMD 구조의 *Feedback* 필드

이유 코드에 대한 전체 설명은 [메시지 및 이유 코드](#)를 참조하십시오.

## MQI 옵션의 유효성 검증을 위한 규칙

이 섹션에는 MQOPEN, MQPUT, MQPUT1, MQGET, MQCLOSE 또는 MQSUB 호출에서 MQRC\_OPTIONS\_ERROR 이유 코드를 생성하는 상황이 나열되어 있습니다.

### MQOPEN 호출

MQOPEN 호출의 옵션:

- 다음 옵션 중 하나 이상을 지정해야 합니다.

- MQOO\_BROWSE
- MQOO\_INPUT\_EXCLUSIVE<sup>1</sup>
- MQOO\_INPUT\_SHARED<sup>1</sup>
- MQOO\_INPUT\_AS\_Q\_DEF<sup>1</sup>
- MQOO\_INQUIRE
- MQOO\_OUTPUT
- MQOO\_SET
- MQOO\_BIND\_ON\_OPEN<sup>2</sup>
- MQOO\_BIND\_NOT\_FIXED<sup>2</sup>
- MQOO\_BIND\_ON\_GROUP<sup>2</sup>
- MQOO\_BIND\_AS\_Q\_DEF<sup>2</sup>
- 다음 중 하나만 허용됩니다.
  - MQOO\_READ\_AHEAD
  - MQOO\_NO\_READ\_AHEAD
  - MQOO\_READ\_AHEAD\_AS\_Q\_DEF

1. 다음 중 하나만 허용됩니다.

- MQOO\_INPUT\_EXCLUSIVE
- MQOO\_INPUT\_SHARED
- MQOO\_INPUT\_AS\_Q\_DEF

2. 다음 중 하나만 허용됩니다.

- MQOO\_BIND\_ON\_OPEN
- MQOO\_BIND\_NOT\_FIXED
- MQOO\_BIND\_ON\_GROUP
- MQOO\_BIND\_AS\_Q\_DEF

**참고:** 이전에 나열된 옵션은 상호 배타적입니다. 그러나 MQOO\_BIND\_AS\_Q\_DEF의 값이 0이기 때문에 다른 두 바인드 옵션 중 하나로 지정하면 이유 코드 MQRC\_OPTIONS\_ERROR가 나타나지 않습니다. MQOO\_BIND\_AS\_Q\_DEF는 프로그램 문서화를 지원하기 위해 제공됩니다.

- MQOO\_SAVE\_ALL\_CONTEXT가 지정된 경우 MQOO\_INPUT\_\* 옵션 중 하나도 지정해야 합니다.
- MQOO\_SET\*\_CONTEXT 또는 MQOO\_PASS\*\_CONTEXT 옵션 중 하나가 지정된 경우 MQOO\_OUTPUT도 지정해야 합니다.
- MQOO\_CO\_OP가 지정된 경우 MQOO\_BROWSE도 지정해야 합니다.
- MQOO\_NO\_MULTICAST가 지정된 경우 MQOO\_OUTPUT도 지정해야 합니다.

## MQPUT 호출

메시지 넣기 옵션:

- MQPMO\_SYNCPOINT와 MQPMO\_NO\_SYNCPOINT의 결합은 허용되지 않습니다.
- 다음 중 하나만 허용됩니다.
  - MQPMO\_DEFAULT\_CONTEXT
  - MQPMO\_NO\_CONTEXT
  - MQPMO\_PASS\_ALL\_CONTEXT
  - MQPMO\_PASS\_IDENTITY\_CONTEXT
  - MQPMO\_SET\_ALL\_CONTEXT
  - MQPMO\_SET\_IDENTITY\_CONTEXT

- 다음 중 하나만 허용됩니다.
  - MQPMO\_ASYNC\_RESPONSE
  - MQPMO\_SYNC\_RESPONSE
  - MQPMO\_RESPONSE\_AS\_TOPIC\_DEF
  - MQPMO\_RESPONSE\_AS\_Q\_DEF
- MQPMO\_ALTERNATE\_USER\_AUTHORITY가 허용되지 않습니다(MQPUT1 호출에서만 유효함).

## MQPUT1 호출

넣기 메시지 옵션의 경우 규칙은 다음을 제외하고는 MQPUT 호출과 동일합니다.

- MQPMO\_ALTERNATE\_USER\_AUTHORITY가 허용됩니다.
- MQPMO\_LOGICAL\_ORDER가 허용되지 않습니다.

## MQGET 호출

메시지 가져오기 옵션:

- 다음 중 하나만 허용됩니다.
  - MQGMO\_NO\_SYNCPOINT
  - MQGMO\_SYNCPOINT
  - MQGMO\_SYNCPOINT\_IF\_PERSISTENT
- 다음 중 하나만 허용됩니다.
  - MQGMO\_BROWSE\_FIRST
  - MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
  - MQGMO\_BROWSE\_NEXT
  - MQGMO\_MSG\_UNDER\_CURSOR
- MQGMO\_SYNCPOINT는 다음 중 하나와 허용되지 않습니다.
  - MQGMO\_BROWSE\_FIRST
  - MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
  - MQGMO\_BROWSE\_NEXT
  - MQGMO\_LOCK
  - MQGMO\_UNLOCK
- MQGMO\_SYNCPOINT\_IF\_PERSISTENT는 다음 중 하나와 허용되지 않습니다.
  - MQGMO\_BROWSE\_FIRST
  - MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
  - MQGMO\_BROWSE\_NEXT
  - MQGMO\_COMPLETE\_MSG
  - MQGMO\_UNLOCK
- MQGMO\_MARK\_SKIP\_BACKOUT을 사용하려면 MQGMO\_SYNCPOINT를 지정해야 합니다.
- MQGMO\_WAIT 및 MQGMO\_SET\_SIGNAL의 결합은 허용되지 않습니다.
- MQGMO\_LOCK이 지정된 경우 다음 중 하나도 지정해야 합니다.
  - MQGMO\_BROWSE\_FIRST
  - MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
  - MQGMO\_BROWSE\_NEXT
- MQGMO\_UNLOCK이 지정되는 경우, 다음 값만 허용됩니다.

- MQGMO\_NO\_SYNCPOINT
- MQGMO\_NO\_WAIT

## MQCLOSE 호출

MQCLOSE 호출 옵션용:

- MQCO\_DELETE 및 MQCO\_DELETE\_PURGE의 결합은 허용되지 않습니다.
- 다음 중 하나만 허용됩니다.
  - MQCO\_KEEP\_SUB
  - MQCO\_REMOVE\_SUB

## MQSUB 호출

MQSUB 호출의 옵션:

- 다음 옵션 중 하나 이상을 지정해야 합니다.
  - MQSO\_ALTER
  - MQSO\_RESUME
  - MQSO\_CREATE
- 다음 중 하나만 허용됩니다.
  - MQSO\_DURABLE
  - MQSO\_NON\_DURABLE

**참고:** 이전에 나열된 옵션은 상호 배타적입니다. 그러나 MQSO\_NON\_DURABLE의 값이 0이기 때문에 MQSO\_DURABLE로 지정되어도 이유 코드 MQRC\_OPTIONS\_ERROR가 나타나지 않습니다. MQSO\_NON\_DURABLE은 프로그램 문서화를 지원하기 위해 제공됩니다.

- MQSO\_GROUP\_SUB 및 MQSO\_MANAGED의 결합은 허용되지 않습니다.
- MQSO\_GROUP\_SUB을 사용하려면 MQSO\_SET\_CORREL\_ID를 지정해야 합니다.
- 다음 중 하나만 허용됩니다.
  - MQSO\_ANY\_USERID
  - MQSO\_FIXED\_USERID
- MQSO\_NEW\_PUBLICATIONS\_ONLY는 다음과의 결합이 허용됩니다.
  - MQSO\_CREATE
  - MQSO\_NEW\_PUBLICATIONS\_ONLY가 원래 구독에 설정된 경우 MQSO\_ALTER
- 1보다 큰 MQSO\_PUBLICATIONS\_ON\_REQUEST 및 SubLevel의 결합은 허용되지 않습니다.
- 다음 중 하나만 허용됩니다.
  - MQSO\_WILDCARD\_CHAR
  - MQSO\_WILDCARD\_TOPIC
- MQSO\_NO\_MULTICAST를 사용하려면 MQSO\_MANAGED를 지정해야 합니다.

## 큐잉된 발행/구독 명령 메시지

애플리케이션은 큐잉된 발행/구독 애플리케이션을 제어하기 위해 MQRFH2 명령 메시지를 사용할 수 있습니다.

발행/구독을 위해 MQRFH2을 사용 중인 애플리케이션은 다음 명령 메시지를 SYSTEM.BROKER.CONTROL.QUEUE에 전송할 수 있습니다.

- [844 페이지의 『Delete Publication 메시지』](#)
- [845 페이지의 『구독자 등록 취소 메시지』](#)

- [848 페이지의 『발행 메시지』](#)
- [850 페이지의 『구독자 등록 메시지』](#)
- [855 페이지의 『업데이트 요청 메시지』](#)

큐잉된 발행/구독 애플리케이션을 작성하는 경우 해당 메시지, 큐 관리자 응답 메시지 및 메시지 디스크립터 (MQMD)를 이해해야 합니다. 다음 정보를 참조하십시오.

- [856 페이지의 『큐 관리자 응답 메시지』](#)
- [861 페이지의 『큐 관리자가 전달한 발행에 대한 MQMD 설정』](#)
- [862 페이지의 『큐 관리자 응답 메시지에서 MQMD 설정』](#)
- [858 페이지의 『발행/구독 이유 코드』](#)

명령은 MQRFH22 헤더의 **NameValueData** 필드에 있는 psc 폴더에 포함되어 있습니다. 명령 메시지에 대한 응답으로 브로커가 전송할 수 있는 메시지는 pscr 폴더에 포함되어 있습니다.

각 명령에 대한 설명에는 폴더에 포함될 수 있는 특성이 나열됩니다. 별도로 지정하지 않은 경우 특성은 선택적이고 한 번만 발생할 수 있습니다.

특성 이름은 <Command>(으)로 표시됩니다.

값은 Publish와 같은 문자열 형식이어야 합니다.

특성 값을 나타내는 문자열 상수는 괄호로 표시됩니다(예: (MQPSC\_PUBLISH)).

문자열 상수는 큐 관리자와 함께 제공되는 헤더 파일 cmqpsc.h에 정의됩니다.

## Delete Publication 메시지

**Delete Publication** 명령 메시지는 지정된 토픽에 대해 보유된 발행을 삭제하도록 큐 관리자에게 알리기 위해 발행자 또는 다른 큐 관리자에서 큐 관리자에게 전송됩니다.

이 메시지는 큐 관리자의 큐잉된 발행/구독 인터페이스에 의해 모니터링된 큐에 전송됩니다.

입력 큐는 원래 발행이 송신된 큐입니다.

**Delete Publication** 명령 메시지에 지정된 토픽의 전체가 아닌 일부에 대한 권한이 있는 경우 해당 토픽만 삭제됩니다. **Broker Response** 메시지는 삭제되지 않은 토픽을 표시합니다.

이와 유사하게 **Publish** 명령에 둘 이상의 토픽이 있고 **Delete Publication** 명령이 해당 토픽 전체가 아닌 일부와 일치하는 경우 **Delete Publication** 명령에 지정된 토픽에 대한 발행만 삭제됩니다.

큐 관리자에 명령 메시지를 보낼 때 필요한 메시지 디스크립터(MQMD)에 대한 세부사항은 [861 페이지의 『큐 관리자가 전달한 발행에 대한 MQMD 설정』](#)의 내용을 참조하십시오.

## 특성

### Command(MQPSC\_COMMAND)

이 값은 DeletePub(MQPSC\_DELETE\_PUBLICATION)입니다.

이 특성은 반드시 지정해야 합니다.

### Topic> (MQPSC\_TOPIC)

이 값은 보유된 발행을 삭제할 토픽이 들어 있는 문자열입니다. 문자열에 포함된 와일드카드는 둘 이상의 토픽에 대한 발행을 삭제할 수 있습니다.

이 특성은 지정해야 하며 필요에 따라 여러 토픽에 반복할 수 있습니다.

### DelOpt(MQPSC\_DELETE\_OPTION)

삭제 옵션 특성은 다음 값 중 하나를 사용할 수 있습니다.

### Local(MQPSC\_LOCAL)

지정된 토픽에 대해 보유된 모든 발행은 Local 옵션으로 발행했는지 여부에 관계없이 로컬 큐 관리자(즉, 이 메시지를 전송할 큐 관리자)에서 삭제됩니다.



다른 큐 관리자의 발행은 영향을 받지 않습니다.

### None(MQPSC\_NONE)

모든 옵션은 기본값을 갖습니다. 이 값은 De10pt 특성을 생략하는 것과 같은 효과를 가집니다. 다른 옵션이 동시에 지정된 경우 None은 무시됩니다.

이 특성을 생략하면 기본값은 Local 옵션을 사용하여 발행했는지 여부와 관계없이 네트워크의 모든 큐 관리자에서 지정된 항목에 대한 모든 보유된 발행이 삭제되는 것입니다.

## 예

다음은 **Delete Publication** 명령 메시지의 NameValueData의 예입니다. 이는 샘플 애플리케이션이 로컬 큐 관리자에서 팀1 및 팀2 간의 일치 사항에 최종 스코어가 포함된 보유된 발행을 삭제하는 데 사용됩니다.

```
<psc>
  <Command>DeletePub</Command>
  <Topic>Sport/Soccer/State/LatestScore/Team1 Team2</Topic>
  <De10pt>Local</De10pt>
</psc>
```

## 구독자 등록 취소 메시지

**Deregister Subscriber** 명령 메시지는 구독자 또는 구독자 대신 다른 애플리케이션에서 큐 관리자로 보내 지정된 매개변수와 일치하는 메시지를 더 이상 수신하지 않음을 나타냅니다.

이 메시지가 SYSTEM.BROKER.CONTROL.QUEUE에 전송되며 이는 큐 관리자의 제어 큐입니다. 사용자는 메시지를 이 큐에 넣기 위해 필요한 권한을 가지고 있어야 합니다.

큐 관리자에 명령 메시지를 보낼 때 필요한 메시지 디스크립터(MQMD) 매개변수에 대한 세부사항은 [큐 관리자가 전달한 발행의 MQMD 설정을 참조하십시오](#).

해당 토픽, 구독 지점 및 원래 구독의 필터 값을 지정하면 개별 구독을 등록 취소할 수 있습니다. 원래 구독에 이 값 중 하나라도 지정되지 않은 경우(즉, 기본값을 사용하는 경우) 구독이 등록 취소될 때 해당 값은 생략되어야 합니다.

구독자에 대한 모든 구독 또는 구독자 그룹은 **DeregAll** 옵션을 사용하여 등록 취소할 수 있습니다. 예를 들어, **DeregAll**이 구독 지점과 함께 지정된 경우(토픽이나 필터 없이) 토픽 및 필터와 관계없이 지정된 구독 지점에서 해당 구독자에 대한 모든 구독이 등록 취소됩니다. 토픽, 필터, 구독 지점의 모든 결합이 허용됩니다. 세 가지 모두 지정하면 하나의 구독 지점만 일치하게 되고 **DeregAll** 옵션은 무시됩니다.

메시지는 구독을 등록한 구독자가 송신해야 합니다. 이에 대해서는 구독자의 사용자 ID를 점검하여 확인됩니다.

구독은 또한 MQSC 또는 PCF 명령을 사용하여 시스템 관리자에 의해 등록 취소될 수 있습니다. 그러나 임시 동적 큐에 등록된 구독은 큐 이름만이 아닌 큐와 연관됩니다. 큐가 명시적으로 삭제되거나 큐 관리자에서 연결을 끊은 애플리케이션에 의해 삭제된 경우 더 이상 **Deregister Subscriber** 명령을 사용하여 해당 큐의 구독을 등록 취소할 수 없습니다. 구독은 개발자 워크벤치를 사용하여 등록 취소될 수 있으며 다음에 발행이 구독과 일치하거나 큐 관리자가 재시작될 때 큐 관리자에 의해 자동으로 제거됩니다. 일반적인 상황에서 애플리케이션은 큐를 삭제하거나 큐 관리자에서 연결을 끊기 전에 해당 구독을 등록 취소해야 합니다.

구독자가 구독을 등록 취소하기 위해 메시지를 보내고 이 메시지가 처리되었다는 응답 메시지를 받는 경우 구독의 등록 취소와 동시에 큐 관리자에 의해 처리되고 있으면 일부 발행은 여전히 구독자 큐에 도착합니다. 큐에서 메시지가 제거되지 않은 경우 구독자 큐에서 처리되지 않은 메시지가 작성되고 있을 수 있습니다. 애플리케이션이 잠시 휴면 상태를 취한 후 적절한 CorrelId가 있는 MQGET 호출이 포함된 루프를 실행할 경우 해당 메시지는 큐에서 지워집니다.

마침가지로 구독자가 영구적 동적 큐를 사용하며 MQCLOSE 호출에 MQCO\_DELETE\_PURGE 옵션을 설정하여 큐를 등록 취소한 다음 닫을 경우 큐는 비어 있지 않을 수 있습니다. 큐가 삭제될 때 큐 관리자의 발행물이 아직 커미트되지 않은 경우 MQRC\_Q\_NOT\_EMPTY 리턴 코드는 MQCLOSE 호출에 의해 발행됩니다. 애플리케이션은 때때로 MQCLOSE 호출을 중지했다가 재실행하여 이 문제를 방지할 수 있습니다.

## 특성

### Command(MQPSC\_COMMAND)

이 값은 DeregSub(MQPSC\_DEREGISTER\_SUBSCRIBER)입니다.

이 특성은 반드시 지정해야 합니다.

### Topic(MQPSC\_TOPIC)

이 값은 등록 취소할 주제가 포함된 문자열입니다.

여러 토픽을 등록 취소하려는 경우 선택적으로 이 특성을 반복할 수 있습니다. DeregAll이 <RegOpt>에 지정된 경우 생략할 수 있습니다.

구독자가 다른 토픽에 대한 구독을 보유하려는 경우 지정된 토픽은 등록된 토픽의 서브세트일 수 있습니다. 와일드 카드 문자는 허용되지만 와일드 카드 문자를 포함하는 토픽 문자열이 **Deregister Subscriber** 명령 메시지에 지정된 해당 문자열과 정확하게 일치해야 합니다.

### SubPoint(MQPSC\_SUBSCRIPTION\_POINT)

이 값은 구독이 분리될 구독 지점을 지정하는 문자열입니다.

이 특성은 반드시 반복할 수 없습니다. <Topic>이(가) 지정되었거나 DeregAll이 <RegOpt>에 지정된 경우 생략할 수 있습니다. 이 특성을 생략하면 다음이 수행됩니다.

- DeregAll을 지정하지 **않으면**, <Topic> 특성(및 <Filter> 특성(있는 경우))과 일치하는 구독이 기본 구독 지점에서 등록 해제됩니다.
- DeregAll을 지정하면 모든 구독(<Topic> 및 <Filter> 특성(있는 경우)과 일치)이 모든 구독 지점에서 등록 해제됩니다.

기본 구독 지점을 명시적으로 지정할 수 없음을 참고하십시오. 따라서 이 구독 지점에서만 모든 구독을 등록 취소할 방법은 없으며 토픽을 지정해야 합니다.

### SubIdentity(MQPSC\_SUBSCRIPTION\_IDENTITY)

이는 최대 길이가 64자인 가변 길이 문자열입니다. 구독에 관심 있는 애플리케이션을 나타내는 데 사용됩니다. 큐 관리자는 각 구독에 대한 일련의 구독자 ID를 유지보수합니다. 각 구독으로 해당 ID가 단일 ID만 보유하도록 설정하거나 무제한의 ID 수를 보유하도록 설정할 수 있습니다.

SubIdentity가 구독의 ID 세트에 있는 경우 세트에서 제거됩니다. ID 세트가 이 결과로 비어 있게 되는 경우 LeaveOnly가 RegOpt 특성의 값으로 지정되지 않는 한 구독은 큐 관리자에서 제거됩니다. ID 세트가 여전히 다른 ID를 포함하면 구독은 큐 관리자에서 제거되지 않고 발행 플로우가 중단되지 않습니다.

SubIdentity가 지정되었지만 SubIdentity가 구독의 ID 세트에 없으면 리턴 코드 **MQRCCF\_SUB\_IDENTITY\_ERROR**로 **Deregister Subscriber** 명령에 실패합니다.

### Filter(MQPSC\_FILTER)

이 값은 등록 취소할 필터를 지정하는 문자열입니다. 대소문자 및 공백을 포함하여 이전에 등록된 구독 필터와 정확히 일치해야 합니다.

둘 이상의 필터를 등록 취소하려는 경우 선택적으로 이 특성을 반복할 수 있습니다. <Topic>이(가) 지정된 경우 또는 DeregAll이 <RegOpt>에 지정된 경우 생략할 수 있습니다.

구독자가 다른 필터에 대한 구독을 보유하려는 경우 지정된 필터는 등록된 필터의 서브세트일 수 있습니다.

### RegOpt(MQPSC\_REGISTRATION\_OPTION)

등록 옵션 특성은 다음 값을 사용할 수 있습니다.

#### DeregAll

(MQPSC\_DEREGISTER\_ALL)

이 구독자에 대해 등록된 일치하는 모든 구독이 등록 취소됩니다.

DeregAll을 지정하는 경우:

- <Topic>, <SubPoint> 및 <Filter>을(를) 생략할 수 있습니다.
- 필요한 경우 <Topic> 및 <Filter>을(를) 반복할 수 있습니다.
- <SubPoint>은(는) 반복하지 않아야 합니다.

DeregAll을 지정하지 않은 경우:

- <Topic>을(를) 지정해야 하며 필요한 경우 반복할 수 있습니다.
- <SubPoint> 및 <Filter>을(를) 생략할 수 있습니다.
- <SubPoint>은(는) 반복하지 않아야 합니다.
- 필요한 경우 <Filter>을(를) 반복할 수 있습니다.

토픽 및 필터 모두 반복되는 경우 두 결합 모두와 일치하는 모든 구독이 제거됩니다. 예를 들어, 3가지 주제 및 3개의 필터를 지정하는 **Deregister Subscriber** 명령은 9개의 구독을 제거하려고 시도합니다.

### **CorrelAsId**

(MQPSC\_CORREL\_ID\_AS\_IDENTITY)

메시지 디스크립터(MQMD)의 CorrelId는 0이 아니어야 하며 구독자를 식별하는 데 사용됩니다. 원래 구독에 사용된 CorrelId와 일치해야 합니다.

### **FullResp**

(MQPSC\_FULL\_RESPONSE)

FullResp가 지정된 경우 명령이 실패하지 않으면 구독의 모든 속성이 응답 메시지에 리턴됩니다.

FullResp가 지정된 경우 **Deregister Subscriber** 명령에 DeregAll을 사용할 수 없습니다. 여러 토픽을 지정할 수도 없습니다. 두 경우 모두 리턴 코드 *MQRCCF\_REG\_OPTIONS\_ERROR*로 명령에 실패합니다.

### **LeaveOnly**

(MQPSC\_LEAVE\_ONLY)

구독의 ID 세트에 SubIdentity를 사용하여 이를 지정하면 SubIdentity가 구독의 ID 세트에서 제거됩니다. 결과 ID 세트가 비어 있는 경우라도 구독은 큐 관리자에서 제거되지 않습니다. SubIdentity 값이 ID 세트에 없는 경우 리턴 코드 *MQRCCF\_SUB\_IDENTITY\_ERROR*로 명령에 실패합니다.

LeaveOnly가 SubIdentity 없이 지정되면 리턴 코드 *MQRCCF\_REG\_OPTIONS\_ERROR*로 명령에 실패합니다.

LeaveOnly나 SubIdentity 중 아무것도 지정되지 않은 경우 구독의 ID 세트 콘텐츠에 관계없이 구독이 제거됩니다.

### **없음**

(MQPSC\_NONE)

모든 옵션은 기본값을 갖습니다. 이 값은 등록 옵션 특성을 생략하는 것과 동일한 효과가 있습니다. 다른 옵션이 동시에 지정된 경우 None은 무시됩니다.

### **VariableUserId**

(MQPSC\_VARIABLE\_USER\_ID)

지정된 경우 구독자의 ID(큐, 큐 관리자, correlid)는 단일 사용자 ID로 제한되지 않습니다. 이는 원래 등록 메시지의 사용자 ID를 구독자의 ID와 연관시키는 큐 관리자의 기존 작동과 다르고 이후부터 해당 ID를 사용하는 다른 사용자를 방지합니다. 새 구독자가 동일한 ID를 사용할 경우 리턴 코드 *MQRCCF\_DUPLICATE\_SUBSCRIPTION*이 리턴됩니다.

모든 사용자는 적절한 권한이 있는 경우 구독을 수정하거나 등록 취소할 수 있어 사용자 ID가 원래의 구독자 ID와 일치해야 하는지 여부에 대해 기존에 수행하던 점검을 수행할 필요가 없습니다.

이 옵션을 기존 구독에 추가하려면 명령은 원래의 구독 자체와 동일한 사용자 ID로 실행해야 합니다.

등록 취소할 구독에 VariableUserId가 설정되어 있는 경우 등록 취소 시 이 옵션을 설정하여 등록 취소 중인 구독을 표시해야 합니다. 그렇지 않은 경우 **Deregister Subscriber** 명령의 사용자 ID가 구독을 식별하는 데 사용됩니다. 구독 이름이 제공된 경우 이는 기타 구독자 ID와 함께 대체됩니다.

이 특성이 생략된 경우 기본값은 등록 옵션이 설정되지 않은 상태입니다.

### QMgrName(MQPSC\_Q\_MGR\_NAME)

이 값은 구독자 큐의 큐 관리자 이름입니다. 원래 구독에 사용된 QMgrName과 일치해야 합니다.

이 특성을 생략할 경우, 기본값은 메시지 디스크립터(MQMD)의 ReplyToQMgr 이름입니다. 결과 이름이 비어 있는 경우 큐 관리자의 이름으로 기본 설정됩니다.

### QName(MQPSC\_Q\_NAME)

이 값은 구독 큐의 이름입니다. 원래 구독에 사용된 QName과 일치해야 합니다.

이 특성이 생략된 경우 기본값은 메시지 디스크립터(MQMD)의 ReplyToQ 이름이며 비어 있으면 안됩니다.

### SubName(MQPSC\_SUBSCRIPTION\_NAME)

**Deregister Subscriber** 명령에 SubName이 지정된 경우 SubName 값은 구독 자체에 VariableUserId가 설정되지 않는 한 사용자 ID를 제외하고 다른 모든 ID 필드보다 우선합니다. VariableUserId가 설정되지 않은 경우 **Deregister Subscriber** 명령은 명령 메시지의 사용자 ID가 구독의 사용자 ID와 일치하는 경우에만 성공하며 그렇지 않은 경우 리턴 코드 MQRCCF\_DUPLICATE\_IDENTITY로 명령에 실패합니다.

이 명령의 기존 ID와 일치하는 구독이 존재하지만 SubName이 없는 경우 리턴 코드 MQRCCF\_SUB\_NAME\_ERROR로 **Deregister Subscriber** 명령에 실패합니다. 기존의 ID와 일치하는 명령 메시지를 사용하여 SubName을 갖는 구독에 대해 등록을 시도했지만 SubName이 지정되지 않은 경우 명령에 성공합니다.

### SubUserData(MQPSC\_SUBSCRIPTION\_USER\_DATA)

이는 가변 길이 텍스트 문자열입니다. 이 값은 구독이 있는 큐 관리자에서 저장되지만 구독자에 대한 발행물 전달에 영향이 없습니다. 새 값으로 동일한 구독에 재등록하여 값을 변경할 수 있습니다. 이 속성은 애플리케이션용입니다.

SubUserData가 존재하면 SubUserData는 구독을 위한 Metatopic 정보(MQCACF\_REG\_SUB\_USER\_DATA)에서 리턴됩니다.

## 예

다음은 **Deregister Subscriber** 명령 메시지의 NameValueData 예입니다. 이 예에서는 샘플 애플리케이션이 모든 일치 항목의 최신 스코어가 포함된 토픽에서 구독을 등록 취소합니다. CorrelId를 포함한 구독자 ID는 MQMD의 기본값에서 가져옵니다.

```
<psc>
  <Command>DeregSub</Command>
  <RegOpt>CorrelAsId</RegOpt>
  <Topic>Sport/Soccer/State/LatestScore/#</Topic>
</psc>
```

## 발행 메시지

**Publish** 명령 메시지를 큐에 넣거나 큐 관리자에서 구독자로 넣어 지정된 토픽에 대한 정보를 발행합니다.

지정된 토픽에 대한 정보를 발행하려면 큐 및 권한에 메시지를 넣을 수 있는 권한이 필요합니다.

사용자에게 토픽의 전체가 아닌 일부에 대한 발행 정보에 대한 권한이 있는 경우 해당 토픽만 발행에 사용됩니다. 경고 응답은 토픽이 발행에 사용되지 않음을 나타냅니다.

구독자에게 일치하는 구독이 있는 경우 큐 관리자는 해당 **Register Subscriber** 명령 메시지에 정의된 구독자 큐로 **Publish** 메시지를 보냅니다.

명령 메시지를 큐 관리자에 전송할 때 필요하며 큐 관리자가 구독자로 발행을 전달할 때 사용되는 메시지 디스크립터(MQMD) 매개변수에 대한 세부사항은 [큐 관리자 응답 메시지](#)를 참조하십시오.

로컬 발행이 아닌 한 큐 관리자는 일치하는 구독이 포함된 네트워크에서 **Publish** 메시지를 다른 큐 관리자로 전달합니다.

발행 데이터가 있는 경우 메시지 본문에 포함됩니다. 데이터는 MQRFH2 헤더의 NameValueData 필드에 있는 <mcd> 폴더에 설명될 수 있습니다.

## 특성

### Command(MQPSC\_COMMAND)

이 값은 Publish(MQPSC\_PUBLISH)입니다.

이 특성은 반드시 지정해야 합니다.

### Topic(MQPSC\_TOPIC)

이 값은 이 발행을 범주화하는 토픽이 포함된 문자열입니다. 와일드카드 문자는 허용되지 않습니다.

이름 목록 SYSTEM.QPUBSUB.QUEUE.NAMELIST, 이 태스크를 완료하는 방법에 대한 지시사항은 [스트림 추가](#)를 참조하십시오.

이 특성은 지정해야 하며 필요에 따라 여러 토픽에 선택적으로 반복할 수 있습니다.

### SubPoint(MQPSC\_SUBSCRIPTION\_POINT)

발행물이 발행되는 구독 지점입니다.

WebSphere Event Broker 6.0에서 <SubPoint> 특성의 값은 공개를 처리하는 공개 노드의 구독 포인트 속성 값입니다.

IBM WebSphere MQ 7.0.1에서 <SubPoint> 특성의 값은 구독 포인트의 이름과 일치해야 합니다. [구독 포인트 추가](#)를 참조하십시오.

### PubOpt(MQPSC\_PUBLICATION\_OPTION)

발행 옵션 특성은 다음 값을 사용할 수 있습니다.

#### RetainPub

(MQPSC\_RETAIN\_PUB)

큐 관리자는 발행 사본을 유지합니다. 이 옵션이 설정되지 않은 경우 큐 관리자가 모든 현재 구독자에게 발행을 보내는 즉시 발행이 삭제됩니다.

#### IsRetainedPub

(MQPSC\_IS\_RETAINED\_PUB)

(큐 관리자만 설정할 수 있습니다.) 이 발행은 큐 관리자에 의해 유지됩니다. 큐 관리자는 구독이 InformIfRetained 옵션에 등록되면 구독자에게 이 발행이 이전에 발행되었으며 유지되었음을 알리도록 이 옵션을 설정합니다. Register Subscriber 또는 Request Update 명령 메시지에 대한 응답으로만 설정됩니다. 구독자에게 직접 송신된 보유된 발행에는 이 옵션 설정이 없습니다.

#### Local

(MQPSC\_LOCAL)

이 옵션은 이 발행이 다른 큐 관리자에 전송되지 않아야 한다고 큐 관리자에게 알려줍니다. 일치하는 구독이 있는 경우 이 큐 관리자에 등록된 모든 구독자는 이 발행을 수신합니다.

#### OtherSubsOnly

(MQPSC\_OTHER\_SUBS\_ONLY)

이 옵션을 사용하면 구독자가 동일한 토픽에 대한 구독자이기도 한 컨퍼런스 유형 애플리케이션을 간단하게 처리할 수 있습니다. 일치하는 구독이 있는 경우라도 큐 관리자가 발행자의 구독자 큐로 발행을 전송하지 않음을 알립니다. 다음 목록에 설명된 대로 발행자의 구독자 큐는 해당 QMgrName, QName 및 선택적 CorrelId로 구성됩니다.

#### CorrelAsId

(MQPSC\_CORREL\_ID\_AS\_IDENTITY)

MQMD(0이어서는 안됨)의 CorrelId는 발행자가 구독자이기도 한 애플리케이션에서 발행자의 구독자 큐의 일부입니다.

#### 없음

(MQPSC\_NONE)

모든 옵션은 기본값을 갖습니다. 이 값은 발행 옵션 특성을 생략하는 것과 동일한 효과가 있습니다. 다른 옵션이 동시에 지정된 경우 None은 무시됩니다.

추가 <PubOpt> 요소를 도입하여 둘 이상의 공개 옵션을 사용할 수 있습니다.

이 특성이 생략된 경우 기본값은 발행 옵션이 설정되지 않은 상태입니다.

#### **PubTime(MQPSC\_PUBLISH\_TIMESTAMP)**

값은 발행자가 설정하는 선택적 발행 시간소인입니다. 다음과 같은 형식을 가진 16자리 문자입니다.

```
YYYYMMDDHHMSSSTH
```

유니버설 시간을 사용합니다. 이 정보는 구독자로 전송되기 전에 큐 관리자에 의해 확인되지 않습니다.

#### **SeqNum(MQPSC\_SEQUENCE\_NUMBER)**

이 값은 발행자가 설정한 선택적 순서 번호입니다.

각 발행으로 1까지 증분시켜야 합니다. 그러나 이는 큐 관리자에 의해 확인되지 않으며 단순히 구독자에게 이 정보를 전송합니다.

동일한 토픽에 대한 발행물이 다른 상호 연결된 큐 관리자에게 발행되는 경우 순서 번호가 (사용되는 경우) 의미가 있음을 보장하는 것은 발행자의 책임입니다.

#### **QMgrName(MQPSC\_Q\_MGR\_NAME)**

이 값은 발행자가 구독자이기도 한 애플리케이션에서 발행자의 구독자 큐를 위한 큐 관리자의 이름을 포함하는 문자열입니다(OtherSubsOnly 참조).

이 특성을 생략할 경우, 기본값은 메시지 디스크립터(MQMD)의 ReplyToQMgr 이름입니다. 결과 이름이 비어 있는 경우 큐 관리자의 이름으로 기본 설정됩니다.

#### **QName(MQPSC\_Q\_NAME)**

이 값은 발행자가 구독자이기도 한 애플리케이션에서 발행자의 구독자 큐의 이름을 포함하는 문자열입니다(OtherSubsOnly 참조).

이 특성이 생략된 경우 기본값은 메시지 디스크립터(MQMD)의 ReplyToQ 이름이며 OtherSubsOnly가 설정된 경우 비어 있으면 안됩니다.

## 예

다음은 **Publish** 명령 메시지에 대한 *NameValueData*의 일부 예입니다.

첫 번째 예는 일치가 시작되었음을 나타내기 위해 샘플 애플리케이션의 일치 시뮬레이터에서 보낸 발행에 대한 것입니다.

```
<psc>
  <Command>Publish</Command>
  <Topic>Sport/Soccer/Event/MatchStarted</Topic>
</psc>
```

두 번째 예는 보유된 발행에 대한 것입니다. 팀1과 팀2 간 일치 항목의 최종 스코어가 발행됩니다.

```
<psc>
  <Command>Publish</Command>
  <PubOpt>RetainPub</PubOpt>
  <Topic>Sport/Soccer/State/LatestScore/Team1 Team2</Topic>
</psc>
```

## 구독자 등록 메시지

**Register Subscriber** 명령 메시지는 구독 지점에서 하나 이상의 토픽을 구독하려 함을 표시하기 위해 구독자 또는 구독자 대신 다른 애플리케이션에 의해 큐 관리자로 송신됩니다. 메시지 콘텐츠 필터 또한 지정할 수 있습니다.

발행/구독 필터 표현식에서 중첩 괄호로 인해 성능이 기하급수적으로 감소합니다. 6개 이상으로 괄호를 중첩하지 마십시오.

메시지가 SYSTEM.BROKER.CONTROL.QUEUE에 전송되며 이는 큐 관리자의 제어 큐입니다. 구독에서 토픽에 대한 액세스 권한(큐 관리자 시스템 관리자가 설정) 외에도 메시지를 이 큐에 넣기 위한 권한이 필요합니다.

사용자가 토픽의 전체가 아닌 일부에 대한 권한을 가지고 있는 경우 권한이 있는 토픽에만 등록됩니다. 경고 응답은 등록되지 않은 토픽을 나타냅니다.

큐 관리자에 명령 메시지를 보낼 때 필요한 메시지 디스크립터(MQMD)에 대한 세부사항은 [860 페이지의 『큐 관리자에 대한 명령 메시지에서 MQMD 설정』](#)의 내용을 참조하십시오.

큐에 대한 응답이 임시 동적 큐인 경우 큐가 처리 완료되면 구독은 큐 관리자에서 자동으로 등록 취소됩니다.

## 특성

### Command(MQPSC\_COMMAND)

이 값은 RegSub(MQPSC\_REGISTER\_SUBSCRIBER)입니다. 이 특성은 반드시 지정해야 합니다.

### Topic(MQPSC\_TOPIC)

구독자가 발행물을 수신할 토픽입니다. 와일드카드 문자는 토픽의 일부로 지정할 수 있습니다.

MQSC 명령 **display sub** 를 사용하여 이 방법으로 작성된 구독을 검사하는 경우 < Topic> 태그의 값이 구독의 TOPICSTR 특성으로 표시됩니다.

이 특성은 필수이며 필요에 따라 여러 토픽에 선택적으로 반복할 수 있습니다.

### SubPoint(MQPSC\_SUBSCRIPTION\_POINT)

이 값은 구독이 첨부된 구독 지점입니다.

이 특성을 생략한 경우 기본 구독 지점이 사용됩니다.

WebSphere Event Broker 6.0에서 <SubPoint> 특성의 값은 구독한 발행 노드의 구독 포인트 속성 값과 일치해야 합니다.

IBM WebSphere MQ 7.0.1에서 <SubPoint> 특성의 값은 구독 포인트의 이름과 일치해야 합니다. [구독 포인트 추가](#)를 참조하십시오.

### Filter(MQPSC\_FILTER)

이 값은 발행 메시지의 콘텐츠에서 필터로 사용되는 SQL 표현식입니다. 지정된 토픽의 발행이 필터와 일치하는 경우 구독자로 송신됩니다. 이 특성은 MQSUB 및 MQOPEN 호출에서 사용되는 선택 문자열에 해당됩니다. 자세한 정보는 메시지의 콘텐츠에서 [선택](#)을 참조하십시오.

이 특성이 생략된 경우 콘텐츠 필터링이 발생하지 않습니다.

### RegOpt(MQPSC\_REGISTRATION\_OPTION)

등록 옵션 특성은 다음 값을 사용할 수 있습니다.

#### AddName

(MQPSC\_ADD\_NAME)

이 등록 구독 명령의 일반 ID와 일치하는 기존 구독에 지정했지만 현재 SubName 값이 없는 경우 이 명령에 지정된 SubName이 구독에 추가됩니다.

AddName이 지정되면 SubName 필드는 필수이며 그렇지 않은 경우 MQRCCF\_REG\_OPTIONS\_ERROR가 리턴됩니다.

#### CorrelAsId

(MQPSC\_CORREL\_ID\_AS\_IDENTITY)

메시지 디스크립터(MQMD)의 CorrelId는 구독자 큐와 일치하는 발행을 송신할 때 사용됩니다. CorrelId는 0이어야 합니다.

#### FullResp

(MQPSC\_FULL\_RESPONSE)

지정된 경우 명령이 실패하지 않으면 구독의 모든 속성이 응답 메시지에 리턴됩니다.

FullResp는 명령 메시지가 단일 구독을 참조할 때만 유효합니다. 따라서 하나의 토픽만 명령에 허용됩니다. 그렇지 않은 경우 리턴 코드 MQRCCF\_REG\_OPTIONS\_ERROR로 명령에 실패합니다.



## **InformIfRet**

(MQPSC\_INFORM\_IF\_RETAINED)

큐 관리자는 **Register Subscriber** 또는 **Request Update** 명령 메시지에 응답하여 발행 메시지를 보낼 때 발행이 유지되는지 여부를 구독자에게 알립니다. 큐 관리자는 메시지에 **IsRetainedPub** 발행 옵션을 포함하여 이를 수행합니다.

## **JoinExcl**

(MQPSC\_JOIN\_EXCLUSIVE)

이 옵션은 지정된 SubIdentity를 구독 ID 세트의 독점 구성원으로 추가해야 하며 다른 ID는 세트에 추가할 수 없음을 나타냅니다.

ID가 이미 결합된 '공유' 상태이며 세트의 유일한 항목인 경우 세트는 이 ID가 보유한 독점 잠금으로 변경됩니다. 그렇지 않은 경우 구독이 현재 ID 세트(공유 액세스 포함)에 다른 ID를 포함하고 있는 경우 리턴 코드 **MQRCCF\_SUBSCRIPTION\_IN\_USE**로 명령에 실패합니다.

## **JoinShared**

(MQPSC\_JOIN\_SHARED)

이 옵션은 지정된 SubIdentity를 구독의 ID 세트에 추가해야 함을 나타냅니다.

구독이 현재 독점적으로 잠금 상태인 경우(JoinExcl 옵션 사용) 구독이 잠긴 ID가 이 명령 메시지의 ID와 동일하지 않는 한 **MQRCCF\_SUBSCRIPTION\_LOCKED** 리턴 코드로 명령에 실패합니다. 이 경우 잠금이 자동으로 공유 잠금으로 수정됩니다.

## **Local**

(MQPSC\_LOCAL)

구독은 로컬이고 네트워크에서 다른 큐 관리자로 분배되지 않습니다. 해당하는 글로벌 구독이 없는 한 다른 큐 관리자에서 작성된 발행은 이 구독자에게 전달되지 않습니다.

## **NewPubsOnly**

(MQPSC\_NEW\_PUBS\_ONLY)

구독이 등록될 때 존재하는 보유된 발행물은 구독자에게 송신되지 않습니다. 새 발행문만 송신됩니다.

구독자가 재등록하고 이 옵션을 더 이상 설정하지 않도록 변경하는 경우 이미 송신된 발행문이 다시 송신될 수 있습니다.

## **NoAlter**

(MQPSC\_NO\_ALTER)

기존의 일치 구독의 속성은 변경되지 않습니다.

구독이 작성 중인 경우 이 옵션은 무시됩니다. 지정된 다른 모든 옵션이 새 구독에 적용됩니다.

SubIdentity가 또한 지정된 결합 옵션(JoinExcl 또는 JoinShared) 중 하나를 포함하는 경우 ID는 NoAlter가 지정되는지 여부에 관계없이 ID 세트에 추가됩니다.

## **없음**

(MQPSC\_NONE)

모든 등록 옵션은 기본값을 가집니다.

구독자가 이미 등록되어 있는 경우, 해당 옵션은 기본값으로 재설정되고(이는 등록 옵션 특성을 생략하는 것과 동일한 효과를 가지지 않음) 등록 만기는 **Register Subscriber** 메시지의 MQMD에서 업데이트됩니다.

다른 등록 옵션이 동시에 지정된 경우 None은 무시됩니다.

## **NonPers**

(MQPSC\_NON\_PERSISTENT)

이 구독과 일치하는 발행물이 비지속 메시지로 구독자에게 전달됩니다.

## **Pers**

(MQPSC\_PERSISTENT)



이 구독과 일치하는 발행물이 지속 메시지로 구독자에게 전달됩니다.

### **PersAsPub**

(MQPSC\_PERSISTENT\_AS\_PUBLISH)

이 구독과 일치하는 발행물이 발행자가 지정한 지속성을 가지고 구독자에게 전달됩니다. 이는 기본 작동입니다.

### **PersAsQueue**

(MQPSC\_PERSISTENT\_AS\_Q)

이 구독과 일치하는 발행물이 구독자 큐에서 지정된 지속성을 가지고 구독자에게 전달됩니다.

### **PubOnReqOnly**

(MQPSC\_PUB\_ON\_REQUEST\_ONLY)

큐 관리자는 **Request Update** 명령 메시지에 대한 응답을 제외하고 발행물을 구독자에게 보내지 않습니다.

### **VariableUserId**

(MQPSC\_VARIABLE\_USER\_ID)

지정된 경우 구독자의 ID(큐, 큐 관리자, correlid)는 단일 사용자 ID로 제한되지 않습니다. 이는 원래 등록 메시지의 사용자 ID를 구독자의 ID와 연관시키는 큐 관리자의 기존 작동과 다르고 이후부터 해당 ID를 사용하는 다른 사용자를 방지합니다. 새 구독자가 동일한 ID를 사용하려고 하면 *MQRCCF\_DUPLICATE\_SUBSCRIPTION*이 리턴됩니다.

그렇기 때문에 사용자에게 적절한 권한이 있는 경우 사용자는 구독을 수정하거나 등록 취소할 수 있습니다. 따라서 사용자 ID가 원래의 구독자 ID와 일치하는지 확인할 필요가 없습니다.

이 옵션을 기존 구독에 추가하려면 명령은 원래의 구독 자체와 동일한 사용자 ID로 실행해야 합니다.

**Request Update** 명령의 구독에 *VariableUserId*가 설정된 경우 이를 업데이트 요청 시 설정하여 참조되는 구독을 나타내야 합니다. 그렇지 않은 경우 **Request Update** 명령의 사용자 ID가 구독을 식별하는 데 사용됩니다. 구독 이름이 제공된 경우 이는 기타 구독자 ID와 함께 대체됩니다.

이 옵션이 설정되지 않은 **Register Subscriber** 명령 메시지가 이 옵션이 설정된 기존 구독을 참조하는 경우 이 구독에서 옵션이 제거되고 구독의 사용자 ID는 이제 고정됩니다. 동일한 ID(큐, 큐 관리자 및 상관 ID)가 있는 구독자가 이미 존재하지만 연관된 다른 사용자 ID가 있는 경우 구독자 ID와 연관된 하나의 사용자 ID만 있을 수 있기 때문에 *MQRCCF\_DUPLICATE\_IDENTITY* 리턴 코드로 명령에 실패합니다.

등록 옵션 특성이 생략되며 구독자가 이미 등록된 경우 해당 등록 옵션은 변경되지 않으며 구독 만기는 **Register Subscriber** 메시지의 MQMD에서 업데이트됩니다.

구독자가 아직 등록되지 않은 경우 기본값을 사용하여 모든 등록 옵션으로 새 구독이 작성됩니다.

기본값은 *PersAsPub*이며 다른 옵션은 설정되지 않습니다.

### **QMgrName(MQPSC\_Q\_MGR\_NAME)**

이 값은 구독자 큐의 큐 관리자 이름이며 일치하는 발행물이 큐 관리자에게 전송됩니다.

이 특성을 생략할 경우, 기본값은 메시지 디스크립터(MQMD)의 ReplyToQMGR 이름입니다. 결과 이름이 비어 있는 경우 큐 관리자의 QMgrName으로 기본 설정됩니다.

### **QName(MQPSC\_Q\_NAME)**

이 값은 구독자 큐의 이름이며 일치하는 발행물이 큐 관리자에게 전송됩니다.

이 특성이 생략된 경우 기본값은 메시지 디스크립터(MQMD)의 ReplyToQ 이름이며 이 경우 비어 있으면 안 됩니다.

큐가 임시 동적 큐이면 발행의 비지속적 전달(NonPers)을 <RegOpt> 특성에 지정해야 합니다.

큐가 임시 동적 큐인 경우 큐가 처리 완료되면 구독은 큐 관리자에서 자동으로 등록 취소됩니다.

### SubName(MQPSC\_SUBSCRIPTION\_NAME)

이는 특정 구독에 지정된 이름입니다. 큐 관리자, 큐 및 선택적 `correlId` 대신 이 이름을 사용하여 구독을 나타낼 수 있습니다.

구독이 이 **SubName** 으로 이미 존재하는 경우 지정되면 구독(Topic, QMgrName, QName, CorrelId, UserId, RegOpts, UserSubData 및 Expiry)의 기타 속성은 해당 속성으로 대체되고 새 Register Subscriber 명령 메시지에 전달됩니다. 그러나 **SubName** 이 QName 필드가 지정되지 않는 채로 사용되고 ReplyToQ가 MQMD 헤더에 지정되는 경우 구독자 큐는 ReplyToQ로 변경됩니다.

이 명령의 기존 ID와 일치하는 구독이 이미 존재하지만 **SubName** 이 없는 경우 **AddName** 옵션이 지정되지 않는 한 `MQRCCF_DUPLICATE_SUBSCRIPTION` 리턴 코드로 등록 명령에 실패합니다.

동일한 **SubName** 을 지정하는 다른 Register Subscriber 명령을 사용하여 기존의 이름 지정된 구독을 대체하려고 시도하고 새 명령의 토픽, QMgrName, QName 및 CorrelId 의 값이 SubName 이 정의되어 있거나 정의되어 있지 않은 다른 기존 구독과 일치하는 경우, 명령은 리턴 코드 `MQRCCF_DUPLICATE_SUBSCRIPTION`으로 실패합니다. 이로 인해 두 구독 이름이 동일한 구독을 참조하지 않게 됩니다.

### SubIdentity(MQPSC\_SUBSCRIPTION\_IDENTITY)

이 문자열은 구독에 관심 있는 애플리케이션을 나타내는 데 사용됩니다. 이는 최대 길이가 64자인 변수 길이 문자이며 선택사항입니다. 큐 관리자는 각 구독에 대한 일련의 구독자 ID를 유지보수합니다. 각 구독으로 해당 ID 세트가 하나의 ID만 또는 무제한 ID를 포함할 수 있도록 설정할 수 있습니다(**JoinShared** 및 **JoinExcl** 옵션 참조).

**JoinShared** 또는 **JoinExcl** 옵션이 지정하는 구독 명령은 이미 존재하지 않고 기존 ID 세트가 이러한 조치(즉, 기타 구독자에게 독점적인 결합이 없거나 ID 세트가 비어 있음)를 허용하는 경우 **SubIdentity** 를 구독의 ID 세트에 추가합니다.

**SubIdentity** 가 지정된 Register Subscriber 명령의 결과로서 구독의 속성 변경은 이 구독에 대한 ID 세트의 유일한 멤버인 경우에만 성공합니다. 그렇지 않은 경우 `MQRCCF_SUBSCRIPTIO_IN_USE` 리턴 코드로 명령에 실패합니다. 다른 관련 구독자에게 알리지 않고 구독 속성이 변경되지 않게 됩니다.

64자를 초과하는 문자열을 지정하는 경우 리턴 코드 `MQRCCF_SUB_IDENTITY_ERROR`로 명령에 실패합니다.

### SubUserData(MQPSC\_SUBSCRIPTION\_USER\_DATA)

이는 가변 길이 텍스트 문자열입니다. 이 값은 구독이 있는 큐 관리자에서 저장되지만 구독자에 대한 발행물 전달에 영향이 없습니다. 새 값으로 동일한 구독에 재등록하여 값을 변경할 수 있습니다. 이 속성은 애플리케이션 사용을 위한 속성입니다.

**SubUserData** 는 구독에 대한 메타토픽 정보(`MQCACF_REG_SUB_USER_DATA`)에서 리턴됩니다(존재하는 경우).

등록 옵션 값 `NonPers`, `PersAsPub`, `PersAsQueue`, and `Pers` 중 둘 이상을 지정하면 마지막 값만 사용됩니다. 개별 구독에 이 옵션을 결합할 수 없습니다.

## 예

다음은 **Register Subscriber** 명령 메시지의 NameValueData의 예입니다. 샘플 애플리케이션에서 결과 서비스는 이 메시지를 사용하여 '지속적 발행' 옵션을 설정하여 모든 일치 항목의 최근 스코어를 포함하는 토픽에 대한 구독을 등록합니다. `CorrelId`를 포함한 구독자 ID는 MQMD의 기본값에서 가져옵니다.

```
<psc>
  <Command>RegSub</Command>
  <RegOpt>PersAsPub</RegOpt>
  <RegOpt>CorrelAsId</RegOpt>
  <Topic>Sport/Soccer/State/LatestScore/#</Topic>
</psc>
```

## 업데이트 요청 메시지

**Request Update** 명령 메시지는 지정된 토픽 및 지정된 필터(선택사항)와 일치하는 구독 지점에 대해 현재 보유된 발행물을 요청하기 위해 구독자에서 큐 관리자로 전송됩니다.

이 메시지가 `SYSTEM.BROKER.CONTROL.QUEUE`에 전송되며 이는 큐 관리자의 제어 큐입니다. 메시지는 업데이트 요청에서 토픽에 대한 액세스 권한 이외에 이 큐에 메시지를 넣기 위한 권한이 필요합니다. 이는 큐 관리자의 시스템 관리자에 의해 설정됩니다.

일반적으로 이 명령은 구독자가 등록 시 `PubOnReqOnly` 옵션을 지정한 경우 사용됩니다. 큐 관리자에게 일치하는 보유된 발행이 있는 경우 구독자에게 전송됩니다. 큐 관리자에게 일치하는 보유된 발행이 없는 경우 리턴 코드 `MQRCCF_NO_RETAINED_MSG`로 요청에 실패합니다. 요청차는 동일한 Topic, SubPoint 및 Filter 값이 포함된 구독에 이전에 등록했어야 합니다.

## 특성

### Command(`MQPSC_COMMAND`)

이 값은 `ReqUpdate(MQPSC_REQUEST_UPDATE)`입니다. 이 특성은 반드시 지정해야 합니다.

### Topic(`MQPSC_TOPIC`)

이 값은 구독자가 요청하는 토픽입니다. 와일드카드 문자가 허용됩니다.

이 특성은 지정되어야 하지만 이 메시지에 한 번만 발생할 수 있습니다.

### SubPoint(`MQPSC_SUBSCRIPTION_POINT`)

이 값은 구독이 첨부된 구독 지점입니다.

이 특성을 생략한 경우 기본 구독 지점이 사용됩니다.

### Filter(`MQPSC_FILTER`)

이 값은 발행 메시지의 콘텐츠에서 필터로 사용되는 ESQL 표현식입니다. 지정된 토픽의 발행이 필터와 일치하는 경우 구독자로 송신됩니다.

<Filter> 특성은 현재 업데이트를 요청하는 원래 구독에 지정된 것과 값이 동일해야 합니다.

이 특성이 생략된 경우 콘텐츠 필터링이 발생하지 않습니다.

### RegOpt(`MQPSC_REGISTRATION_OPTION`)

등록 옵션 특성 값은 다음 값을 사용할 수 있습니다.

#### CorrelAsId

(`MQPSC_CORREL_ID_AS_IDENTITY`)

메시지 디스크립터(MQMD)의 `CorrelId`는 0이 아니어야 하며 구독자 큐와 일치하는 발행을 송신하는 데 사용됩니다.

#### 없음

(`MQPSC_NONE`)

모든 옵션은 기본값을 갖습니다. <RegOpt> 특성을 생략할 때와 같은 효과가 있습니다. 다른 옵션이 동시에 지정된 경우 None은 무시됩니다.

#### VariableUserId

(`MQPSC_VARIABLE_USER_ID`)

지정된 경우 구독자의 ID(큐, 큐 관리자 및 `correlid`)는 단일 사용자 ID로 제한되지 않습니다. 이는 원래 등록 메시지의 사용자 ID를 구독자의 ID와 연관시키는 큐 관리자의 기존 작동과 다르고 이후부터 해당 ID를 사용하는 다른 사용자를 방지합니다. 새 구독자가 동일한 ID를 사용하려고 하면 리턴 코드 `MQRCCF_DUPLICATE_SUBSCRIPTIO`로 명령에 실패합니다.

그렇기 때문에 적절한 권한이 있는 경우 사용자는 구독을 수정하거나 등록 취소할 수 있습니다. 따라서 사용자 ID가 원래의 구독자 ID와 일치하는지 확인할 필요가 없습니다.

이 옵션을 기존 구독에 추가하려면 명령은 원래의 구독 자체와 동일한 사용자 ID로 실행해야 합니다.

**Request Update** 명령의 구독에 `VariableUserId`가 설정된 경우 이를 업데이트 요청 시 설정하여 참조되는 구독을 나타내야 합니다. 그렇지 않은 경우 **Request Update** 명령의 사용자 ID가 구독을 식별하는 데 사용됩니다. 구독 이름이 제공된 경우 이는 기타 구독자 ID와 함께 대체됩니다.

이 특성이 생략된 경우 기본값은 등록 옵션이 설정되지 않은 상태입니다.

#### **QMgrName(MQPSC\_Q\_MGR\_NAME)**

이 값은 구독자 큐의 큐 관리자 이름이며 일치하는 발행물이 큐 관리자에게 전송됩니다.

이 특성을 생략할 경우, 기본값은 메시지 디스크립터(MQMD)의 `ReplyToQMgr` 이름입니다. 결과 이름이 비어 있는 경우 큐 관리자의 `QMgrName`으로 기본 설정됩니다.

#### **QName(MQPSC\_Q\_NAME)**

이 값은 구독자 큐의 이름이며 일치하는 발행물이 큐 관리자에게 전송됩니다.

이 특성이 생략된 경우 기본값은 메시지 디스크립터(MQMD)의 `ReplyToQ` 이름이며 이 경우 비어 있으면 안 됩니다.

#### **SubName(MQPSC\_SUBSCRIPTION\_NAME)**

이는 특정 구독에 지정된 이름입니다. 이 옵션을 **Request Update** 명령에서 지정하는 경우 구독 자체에 `VariableUserId`를 설정하지 않으면 `SubName` 값은 사용자 ID를 제외한 다른 모든 값에 우선합니다. `VariableUserId`가 설정되지 않은 경우 **Request Update** 명령은 명령 메시지의 사용자 ID가 구독의 사용자 ID와 일치하는 경우에만 성공합니다. 명령 메시지의 사용자 ID가 구독의 사용자 ID와 일치하지 않는 경우 리턴 코드 `MQRCCF_DUPLICATE_IDENTITY`로 명령에 실패합니다.

`VariableUserId`가 설정되고 사용자 ID가 구독의 사용자 ID와 다른 경우 명령은 새 명령 메시지의 사용자 ID가 스트림 큐를 찾아보고 구독의 구독자 큐에 넣을 권한이 있다면 성공합니다. 그렇지 않은 경우 `MQRCCF_NOT_AUTHORIZED` 리턴 코드로 명령에 실패합니다.

이 명령의 일반 ID와 일치하는 구독이 있지만 `SubName`이 없는 경우, **Request Update** 명령이 실패하고 리턴 코드는 `MQRCCF_SUB_NAME_ERROR`입니다.

기존의 ID와 일치하는 명령 메시지를 사용하여 `SubName`이 있는 구독에 업데이트하려고 시도했지만 `SubName`이 지정되지 않은 경우 명령이 성공합니다.

## 예

다음은 **Request Update** 명령 메시지의 `NameValueData`의 예입니다. 샘플 애플리케이션에서 결과 서비스는 이 메시지를 사용하여 모든 팀의 최신 스코어가 포함된 보유된 발행을 요청합니다. `CorrelId`를 포함한 구독자 ID는 MQMD의 기본값에서 가져옵니다.

```
<psc>
  <Command>ReqUpdate</Command>
  <RegOpt>CorrelAsId</RegOpt>
  <Topic>Sport/Soccer/State/LatestScore/#</Topic>
</psc>
```

## 큐 관리자 응답 메시지

**Queue Manager Response** 메시지는 명령 메시지 디스크립터가 응답이 필요함을 지정한 경우 큐 관리자에 의해 수신된 명령 메시지의 성공 또는 실패를 나타내기 위해 큐 관리자에서 발행자 또는 구독자의 `ReplyToQ`로 전송됩니다.

응답 메시지는 `<pscr>` 폴더에서 `MQRFH2` 헤더의 `NameValueData` 필드 내에 포함됩니다.

경고 또는 오류의 경우 응답 메시지는 명령 메시지의 `<psc>` 폴더와 `<pscr>` 폴더를 포함합니다. 메시지 데이터는 (있는 경우) 큐 관리자 응답 메시지에 포함되지 않습니다. 오류가 발생하면 오류를 발생시킨 메시지는 처리되지 않지만 경고가 발생하면 일부 메시지는 처리될 수 있습니다.

응답 송신에 실패하는 경우:

- 발행 메시지의 경우 큐 관리자는 `MQPUT`이 실패하면 응답을 IBM MQ 데드-레터 큐에 보내려고 합니다. 그러면 응답을 발행자로 다시 송신할 수 없는 경우에도 발행이 구독자로 송신될 수 있습니다.

- 다른 메시지의 경우 또는 발행 응답이 데드-레터 큐로 송신될 수 없는 경우 오류가 기록되고 일반적으로 명령 메시지가 롤백됩니다. 이러한 상황이 발생하는지 여부는 MQInput 노드의 구성 방식에 따라 다릅니다.

## 특성

### Completion(MQPSCR\_COMPLETION)

다음 세 가지 값 중 하나를 취하는 완료 코드:

#### 확인

명령이 성공적으로 완료되었습니다.

#### 경고

명령이 완료되었지만 경고가 발생했습니다.

#### Error

명령이 실패했습니다.

### Response(MQPSCR\_RESPONSE)

해당 명령이 완료 코드 경고 또는 오류를 생성한 경우 명령 메시지에 대한 응답입니다. <Reason> 특성이 포함되어 있으며 경고 또는 오류의 원인을 나타내는 기타 특성을 포함할 수 있습니다.

하나 이상의 오류가 발생한 경우 첫 번째 오류만을 나타내는 단 하나의 응답 폴더만 존재합니다. 하나 이상의 경고가 발생한 경우 각 경고마다 응답 폴더가 있습니다.

### Reason(MQPSCR\_REASON)

완료 코드가 경고 또는 오류인 경우 완료 코드를 규정하는 이유 코드입니다. 다음 예제에 나열된 오류 코드 중 하나로 설정됩니다. <Reason> 특성은 <Response> 폴더에 포함되어 있습니다. 이유 코드 다음에는 오류 또는 경고의 원인을 표시하는 <psc> 폴더의 유효한 특성(예: 주제 이름)이 올 수 있습니다. 이유 코드 가? ?????인 경우, 데이터가 올바른지 확인하십시오 (예: 일치하는 꺾쇠괄호 (< >)).

## 예

여기에 **Queue Manager Response** 메시지의 일부 NameValueData 예가 있습니다. 성공 응답은 다음과 같습니다.

```
<pscr>
  <Completion>ok</Completion>
</pscr>
```

다음은 실패 응답의 예입니다. 실패는 필터 오류입니다. 첫 번째 NameValueData 문자열에는 응답이 포함되고 두 번째에는 원래 명령이 포함됩니다.

```
<pscr>
  <Completion>error</Completion>
  <Response>
    <Reason>3150</Reason>
  </Reponse>
</pscr>

<psc>
  ...
  command message (to which
  the queue manager is responding)
  ...
</psc>
```

다음은 경고 응답(권한 없는 토픽으로 인한)의 예입니다. 첫 번째 NameValueData 문자열에는 응답이 포함되고 두 번째 NameValueData 문자열에는 원래 명령이 포함됩니다.

```
<pscr>
  <Completion>warning</Completion>
  <Response>
    <Reason>3081</Reason>
    <Topic>topic1</Topic>
  </Reponse>
  <Response>
    <Reason>3081</Reason>
```

```

<Topic>topic2</Topic>
</Reponse>
</pscr>

<psc>
...
command message (to which
the queue manager is responding)
...
</psc>

```

## 발행/구독 이유 코드

이러한 이유 코드는 발행/구독 응답 <pscr> 폴더의 Reason 필드에 보유할 수 있습니다. C 또는 C++ 프로그래밍 언어에서 이들 코드를 표시하는 데 사용할 수 있는 상수도 나열됩니다.

MQRC\_ 상수에는 IBM MQ cmqc.h 헤더 파일이 필요합니다. MQRCCF\_ 상수에는 IBM MQ cmqcf.h 헤더 파일이 필요합니다(cmqpssc.h 헤더 파일이 필요한 MQRCCF\_FILTER\_ERROR 및 MQRCCF\_WRONG\_USER 제외).

이유 코드 및 텍스트	설명	발행자
2336 MQRC_RFH_COMMAND_ERROR	<psc> 폴더의 < Command> 필드에 대한 올바른 값은 RegSub, DeregSub, Publish, DeletePub 및 ReqUpdate입니다. 다른 값을 지정하면 이 오류 코드가 발행됩니다.	모든 명령
2337 MQRC_RFH_PARM_ERROR	<psc> 및 <mcd> 폴더에는 둘 다에 지정할 수 있는 유효한 매개변수 세트가 있습니다. 이러한 폴더에 대한 설명을 확인하고 잘못된 매개변수를 지정하지 않았는지 확인하십시오.	모든 명령
2338 MQRC_RFH_DUPLICATE_PARM	<psc> 폴더 내의 일부 매개변수(예: 토픽)를 반복할 수 있지만 다른 매개변수(예: 명령)는 반복할 수 없습니다. 반복 가능하지 않은 매개변수를 중복하지 않았는지 검사하십시오.	모든 명령
2339 MQRC_RFH_PARM_MISSING	<psc> 또는 <mcd> 폴더 내의 일부 매개변수는 선택적이며 생략할 수 있습니다. 일부는 필수이며 생략해서는 안 됩니다. <psc> 및 <mcd> 폴더에 모든 필수 매개변수를 포함했는지 확인하십시오.	모든 명령
2551 MQRC_SELECTION_NOT_AVAILABLE	확장 메시지 선택 제공자가 발행을 수신해야 하는 지정된 필터가 있는 구독자를 판별할 수 없습니다.	발행, 구독자 등록 및 업데이트 요청
	확장 메시지 선택 제공자가 지정된 구독자의 필터를 핸들링할 수 없었습니다.	구독자 등록 및 업데이트 요청
2554 MQRC_CONTENT_ERROR	확장 메시지 선택 제공자가 현재 또는 보유했던 발행에서 오류를 찾았습니다.	발행 및 업데이트 요청
3008 MQRCCF_COMMAND_FAILED	내부 오류가 발생하여 명령이 올바르게 실행되지 않았습니다. 명령을 재실행하면 오류가 발생할 수 있습니다. 큐 관리자를 위한 시스템 이벤트 로그는 IBM에 문제점을 보고할 때 사용되어야 하는 정보를 포함합니다.	모든 명령

이유 코드 및 텍스트	설명	발행자
3072 MQRCCF_TOPIC_ERROR	Topic 매개변수에 대해 지정한 하나 이상의 값이 올바르지 않습니다. Topic 값이 지정된 제한을 따르는지 확인하십시오.	모든 명령
3073 MQRCCF_NOT_REGISTERED	DeregSub 또는 ReqUpdate 명령에 지정한 SubPoint, Topic 및 Filter의 결합이 이전에 등록한 결합이 아니거나 DeregAll 옵션이 지정될 때의 DeregSub 명령의 경우 SubPoint, Topic 또는 Filter 특성 중 하나가 구독을 등록 취소하는데 사용되지 않았습니다.	구독자 등록 취소 및 업데이트 요청 명령
3074 MQRCCF_Q_MGR_NAME_ERROR	지정된 큐 관리자가 올바르지 않거나 큐 관리자가 사용 불가능하거나 존재하지 않았습니다.	구독자 등록 취소, 발행, 구독자 등록 및 업데이트 요청 명령
3076 MQRCCF_Q_NAME_ERROR	지정된 큐 이름이 올바르지 않거나 지정된 큐 관리자에 큐가 존재하지 않습니다.	구독자 등록 취소, 발행, 구독자 등록 및 업데이트 요청 명령
3077 MQRCCF_NO_RETAINED_MSG	지정된 토픽에 대한 보유 메시지가 없습니다. 이는 애플리케이션 프로그램의 설계에 따라 오류이거나 오류가 아닐 수 있습니다.	업데이트 요청 명령
3079 MQRCCF_INCORRECT_Q	RegSub, DeregSub 및 ReqUpdate 명령은 항상 의도된 큐 관리자의 SYSTEM.BROKER.CONTROL.QUEUE 큐로 송신됩니다. 발행 및 발행 삭제 명령은 의도된 특정 발행/구독 메시지 플로우에 대한 입력 큐로 송신됩니다. 이는 메시지 플로우를 설계할 때 판별됩니다. 명령이 잘못된 큐로 송신될 경우 이 오류 코드가 리턴됩니다.	모든 명령
3080 MQRCCF_CORREL_ID_ERROR	CorrelAsId를 RegOpt 매개변수 중 하나로 지정했습니다. 그러나 MQMD의 CorrelId 필드에 올바른 상관 ID가 없습니다(즉, MQCI_NONE으로 설정됨).	구독자 등록 취소 및 구독자 등록 명령
3081 MQRCCF_NOT_AUTHORIZED	요청된 조치를 수행할 수 있는 권한이 없습니다. 큐 관리자에 대한 권한 부여 설정은 시스템 관리자가 토픽 계층 편집기를 사용하여 핸들링합니다.	발행 및 구독자 등록 명령
3083 MQRCCF_REG_OPTIONS_ERROR	RegSub 또는 DeregSub 명령을 포함하는 <psc> 폴더에 인식되지 않은 RegOpt 매개변수를 지정했습니다.	구독자 등록 취소 및 구독자 등록 명령
3084 MQRCCF_PUB_OPTIONS_ERROR	Publish 명령을 포함하는 <psc> 폴더에 인식되지 않은 PubOpt 매개변수를 지정했습니다.	발행 명령
3087 MQRCCF_DEL_OPTIONS_ERROR	DeletePub 명령을 포함하는 <psc> 폴더에 인식되지 않은 DelOpt 매개변수를 지정했습니다.	발행 삭제 명령

이유 코드 및 텍스트	설명	발행자
3150 MQRCCF_FILTER_ERROR	Filter 매개변수에 대해 지정된 값이 올바르지 않습니다. 필터 표현식의 올바른 구문을 설명하는 절을 점검하고 표현식이 올바른 구문을 따르는지 확인하십시오.	구독자 등록 취소, 구독자 등록 및 업데이트 요청 명령
3151 MQRCCF_WRONG_USER	지정된 것과 일치하는 구독이 이미 존재하지만 다른 사용자가 등록했습니다. 구독은 원래 구독을 등록한 사용자만 변경 또는 등록 취소할 수 있습니다.	구독자 등록 취소, 구독자 등록 및 업데이트 요청 명령
3152 MQRCCF_DUPLICATE_SUBSCRIPTION	일치하는 구독이 이미 다른 구독 이름으로 존재합니다.	
3153 MQRCCF_SUB_NAME_ERROR	구독 이름 형식이 올바르지 않거나 구독 이름 없이 일치하는 구독이 이미 존재합니다.	
3154 MQRCCF_SUB_IDENTITY_ERROR	구독 ID 매개변수에 오류가 있습니다. 제공된 값이 허용되는 최대 길이를 초과하거나 구독 ID가 현재 구독 ID 세트의 구성원이 아니며 등록 조인 옵션이 지정되지 않았습니다.	
3155 MQRCCF_SUBSCRIPTION_IN_USE	ID 세트 구성원이 구독을 수정하거나 등록 취소하려고 시도했으며 해당 구성원은 이 세트의 유일한 구성원이 아닙니다.	
3156 MQRCCF_SUBSCRIPTION_LOCKED	구독이 현재 다른 ID에 의해 독점적으로 잠겨있습니다.	
3157 MQRCCF_ALREADY_JOINED	등록 조인 옵션이 지정되었지만 구독자 ID가 이미 구독자의 ID 세트 구성원입니다.	

## 큐 관리자에 대한 명령 메시지에서 MQMD 설정

명령 메시지를 큐 관리자에 보내는 애플리케이션은 메시지 디스크립터(MQMD)에서 필드의 다음 설정을 사용합니다. 기본값으로 남아 있는 필드 또는 일반적인 방법으로 올바른 값으로 설정할 수 있는 필드는 여기에 나열되지 않았습니다.

### 보고서

MsgType 및 CorrelId를 참조하십시오.

### MsgType

MsgType은 각각 *MQMT\_REQUEST* 또는 *MQMT\_DATAGRAM*으로 설정되어야 합니다. MsgType이 이러한 값 중 하나로 설정되지 않은 경우 *MQRC\_MSG\_TYPE\_ERROR*는 리턴됩니다.

응답이 항상 필수적인 경우 명령 메시지에 대해 MsgType을 *MQMT\_REQUEST*로 설정해야 합니다. Report 필드의 *MQRO\_PAN* 및 *MQRO\_NAN* 필드는 이 경우 중요하지 않습니다.

MsgType이 *MQMT\_DATAGRAM*으로 설정되는 경우 응답은 Report 필드의 *MQRO\_PAN* 및 *MQRO\_NAN* 플래그 설정에 따라 다릅니다.

- *MQRO\_PAN*만으로 명령이 성공하는 경우에만 큐 관리자가 응답을 송신함을 의미합니다.
- *MQRO\_NAN*만으로 명령이 실패하는 경우에만 큐 관리자가 응답을 송신함을 의미합니다.
- 명령이 경고 상태로 완료되면 *MQRO\_PAN* 또는 *MQRO\_NAN* 중 하나만 설명되면 응답이 송신됩니다.
- *MQRO\_PAN*으로 명령이 성공하는지 실패하는지 여부를 큐 관리자가 응답을 송신함을 의미합니다. 이는 큐 관리자의 관점에서 MsgType을 *MQMT\_REQUEST*로 설정하는 것과 같이 효과가 있습니다.



- MQRO\_PAN이나 MQRO\_NAN을 모두 설정하지 않으면 아무 응답도 송신되지 않습니다.

**Format**

MQFMT\_RF\_HEADER\_2로 설정

**MsgId**

일반적으로 이 필드는 MQMI\_NONE로 설정되어 큐 관리자가 고유 값을 생성하도록 합니다.

**CorrelId**

이 필드는 모든 값으로 설정할 수 있습니다. 송신자의 ID에 CorrelId가 포함되는 경우 큐 관리자에 의해 전송된 모든 응답 메시지의 이 값이 송신자로 설정되는지 확인하려면 Report 필드에서 MQRO\_PASS\_CORREL\_ID와 함께 이 값을 지정하십시오.

**ReplyToQ**

이 필드는 응답(있는 경우)이 송신될 큐를 정의합니다. 이 큐는 송신자 큐일 수 있습니다. 이때 QName 매개변수를 메시지에서 생략할 수 있다는 이점이 있습니다. 그러나 응답이 다른 큐로 송신되려면 QName 매개변수가 필요합니다.

**ReplyToQMgr**

이 필드는 응답의 큐 관리자를 정의합니다. 이 필드를 비워둘 경우(기본값) 로컬 큐 관리자가 자신의 이름을 이 필드에 넣습니다.

## 큐 관리자가 전달한 발행에 대한 MQMD 설정

발행물을 구독자에게 보낼 때 큐 관리자는 메시지 디스크립터(MQMD)의 필드에 이러한 설정을 사용합니다. MQMD의 다른 모든 필드는 기본값으로 설정됩니다.

**보고서**

Report는 MQRO\_NONE으로 설정됩니다.

**MsgType**

MsgType은 MQMT\_DATAGRAM으로 설정됩니다.

**Expiry**

Expiry는 발행자로부터 수신한 Publish 메시지의 값으로 설정됩니다. 보유된 메시지의 경우 미해결 시간은 메시지가 큐 관리자에 있었던 대략적인 시간만큼 줄어듭니다.

**Format**

Format은 MQFMT\_RF\_HEADER\_2로 설정됩니다.

**MsgId**

MsgId는 고유 값으로 설정됩니다.

**CorrelId**

CorrelId가 구독자 ID의 일부인 경우 이는 등록시 구독자가 지정한 값입니다 그렇지 않은 경우 큐 관리자에 의해 선택된 0이 아닌 값입니다.

**Priority**

Priority는 발행자가 설정한 값을 가져오거나 발행자가 MQPRI\_PRIORITY\_AS\_Q\_DEF를 지정한 경우 해석된 값을 가져옵니다.

**persistence**

Persistence는 발행자가 설정한 값을 가져오거나 발행자 MQPRI\_PRIORITY\_AS\_Q\_DEF를 지정한 경우 해석된 값을 가져옵니다. 단, 이 발행물이 송신될 구독자의 Register Subscriber 메시지에 달리 지정된 경우에는 그 지정을 따릅니다.

**ReplyToQ**

ReplyToQ는 공백으로 설정됩니다.

**ReplyToQMgr**

ReplyToQMgr은 큐 관리자 이름으로 설정됩니다.

**UserIdentifier**

UserIdentifier는 구독자가 등록할 때 설정한 구독자 사용자 ID입니다.

**AccountingToken**

AccountingToken은 구독자가 처음 등록될 때 설정된 구독자 계정 토큰입니다.

**ApplIdentityData**

ApplIdentityData는 구독자가 처음 등록될 때 설정된 구독자 애플리케이션 ID 데이터입니다.

**PutApplType**

PutApplType은 MQAT\_BROKER로 설정됩니다.

**PutAppName**

PutAppName은 큐 관리자 이름의 처음 28자로 설정됩니다.

**PutDate**

PutDate는 메시지를 넣은 날짜입니다.

**PutTime**

PutTime은 메시지를 넣은 시간입니다.

**ApplOriginData**

ApplOriginData는 공백으로 설정됩니다.

**큐 관리자 응답 메시지에서 MQMD 설정**

응답을 발행 메시지에 보낼 때 큐 관리자는 메시지 디스크립터(MQMD)의 필드에 이러한 설정을 사용합니다. MQMD의 다른 모든 필드는 기본값으로 설정됩니다.

**보고서**

Report는 모두 0으로 설정됩니다.

**MsgType**

MsgType은 MQMT\_REPLY로 설정됩니다.

**Format**

Format은 MQFMT\_RF\_HEADER\_2로 설정됩니다.

**MsgId**

MsgId 설정은 원래 명령 메시지의 Report 옵션에 따라 다릅니다. 기본적으로 이는 MQMI\_NONE으로 설정되어 큐 관리자가 고유 값을 생성하도록 합니다.

**CorrelId**

CorrelId 설정은 원래 명령 메시지의 Report 옵션에 따라 다릅니다. 기본적으로 이는 CorrelId가 명령 메시지의 MsgId와 같은 값으로 설정됨을 의미합니다. 이는 응답으로 상관 명령에 사용될 수 있습니다.

**Priority**

Priority는 원래 명령 메시지와 같은 값으로 설정됩니다.

**persistence**

Persistence는 원래 명령 메시지의 값 세트로 설정됩니다.

**Expiry**

Expiry는 큐 관리자가 수신한 원래 명령 메시지와 같은 값으로 설정됩니다.

**PutApplType**

PutApplType은 MQAT\_BROKER로 설정됩니다.

**PutAppName**

PutAppName은 큐 관리자 이름의 처음 28자로 설정됩니다.

다른 컨텍스트 필드는 MQPMO\_PASS\_IDENTITY\_CONTEXT로 생성된 것처럼 설정됩니다.

**시스템 인코딩**

이 절에서는 메시지 디스크립터의 *Encoding* 필드 구조에 대해 설명합니다.

이 구조의 필드 요약은 412 페이지의 『MQMD - 메시지 디스크립터』의 내용을 참조하십시오.

*Encoding* 필드는 4개의 별도 하위 필드로 나뉘는 32비트 정수입니다. 이 하위 필드는 다음을 식별합니다.

- 2진 정수에 사용된 인코딩
- 팩형 10진수 정수에 사용된 인코딩
- 부동 소수점 숫자에 사용된 인코딩

- 예약 비트

각 하위 필드는 하위 필드에 해당되는 위치에 1비트가 있고 다른 곳에는 0비트가 있는 비트 마스크로 식별됩니다. 비트는 숫자로 지정되며, 비트 0은 가장 중요한 비트이고 비트 31은 가장 중요하지 않은 비트입니다. 다음 마스크가 정의됩니다.

#### **MQENC\_INTEGER\_MASK**

2진 정수 인코딩을 위한 마스크.

이 하위 필드는 *Encoding* 필드 내에서 28에서 31까지의 비트 위치를 차지합니다.

#### **MQENC\_DECIMAL\_MASK**

팩형 10진수 정수 인코딩을 위한 마스크.

이 하위 필드는 *Encoding* 필드 내에서 24에서 27까지의 비트 위치를 차지합니다.

#### **MQENC\_FLOAT\_MASK**

부동 소수점 인코딩을 위한 마스크.

이 하위 필드는 *Encoding* 필드 내에서 20에서 23까지의 비트 위치를 차지합니다.

#### **MQENC\_RESERVED\_MASK**

예약 비트를 위한 마스크.

이 하위 필드는 *Encoding* 필드 내에서 0에서 19까지의 비트 위치를 차지합니다.

## 2진 정수 인코딩

2진 정수 인코딩에 대해 올바른 값은 다음과 같습니다.

#### **MQENC\_INTEGER\_UNDEFINED**

2진 정수가 정의되지 않은 인코딩을 사용하여 표시됩니다.

#### **MQENC\_INTEGER\_NORMAL**

2진 정수가 전통적인 방법으로 표시됩니다.

- 숫자의 LSB(Least Significant Byte)에는 숫자의 임의 바이트의 최상위 주소가 있으며, MSB(Most Significant Byte)에는 최하위 주소가 있습니다.
- 각 바이트의 최하위 비트는 다음으로 높은 주소의 바이트와 인접하고 각 바이트의 최상위 비트는 다음으로 낮은 주소의 바이트와 인접합니다.

#### **MQENC\_INTEGER\_REVERSED**

2진 정수는 MQENC\_INTEGER\_NORMAL과 동일한 방법으로 나타내지만 바이트는 역순으로 배열됩니다. 각 바이트 내의 비트는 MQENC\_INTEGER\_NORMAL과 동일한 방법으로 배열됩니다.

## 팩형 10진수 정수 인코딩

팩형 10진수 정수 인코딩에 대해 올바른 값은 다음과 같습니다.

#### **MQENC\_DECIMAL\_UNDEFINED**

팩형 10진수 정수가 정의되지 않은 인코딩을 사용하여 표시됩니다.

#### **MQENC\_DECIMAL\_NORMAL**

팩형 10진수 정수가 전통적인 방법으로 표시됩니다.

- 인쇄 가능한 숫자 형식의 각 10진수는 'X'0' - 'X'9' 범위의 단일 16진수 숫자를 통해 팩형 10진수로 표시됩니다. 각 16진 숫자는 4비트를 차지하므로 팩형 10진수 숫자의 각 바이트는 인쇄 가능한 형식의 두 10진수 숫자를 나타냅니다.
- 팩형 10진수 숫자의 최하위 바이트는 최하위 10진수 숫자를 포함하는 바이트입니다. 해당 바이트 내에서 최상위 4비트는 최하위 10진수 숫자를 포함하고 최하위 4비트는 부호를 포함합니다. 부호는 'X'C'(양수), 'X'D'(음수) 또는 'X'F'(부호 없음)입니다.
- 숫자의 LSB(Least Significant Byte)에는 숫자의 임의 바이트의 최상위 주소가 있으며, MSB(Most Significant Byte)에는 최하위 주소가 있습니다.

- 각 바이트의 최하위 비트는 다음으로 높은 주소의 바이트와 인접하고 각 바이트의 최상위 비트는 다음으로 낮은 주소의 바이트와 인접합니다.

### **MQENC\_DECIMAL\_REVERSED**

팩형 10진수 정수는 MQENC\_DECIMAL\_NORMAL과 동일한 방법으로 나타내지만 바이트는 역순으로 배열됩니다. 각 바이트 내의 비트는 MQENC\_DECIMAL\_NORMAL과 동일한 방법으로 배열됩니다.

## **부동 소수점 인코딩**

부동 소수점 인코딩에 대해 올바른 값은 다음과 같습니다.

### **MQENC\_FLOAT\_UNDEFINED**

부동 소수점 숫자가 정의되지 않은 인코딩을 사용하여 표시됩니다.

### **MQENC\_FLOAT\_IEEE\_NORMAL**

부동 소수점 숫자는 표준 IEEE를 사용하여 표시합니다.<sup>4</sup> 부동 소수점 형식이며 다음과 같이 배열된 바이트

- 가수(mantissa)의 LSB(Least Significant Byte)에는 숫자의 임의 바이트의 최상위 주소가 있으며, 지수를 포함한 바이트에는 최하위 주소가 있습니다.
- 각 바이트의 최하위 비트는 다음으로 높은 주소의 바이트와 인접하고 각 바이트의 최상위 비트는 다음으로 낮은 주소의 바이트와 인접합니다.

IEEE float 인코딩에 대한 세부사항은 IEEE Standard 754에서 찾을 수 있습니다.

### **MQENC\_FLOAT\_IEEE\_REVERSED**

부동 소수점 수는 MQENC\_FLOAT\_IEEE\_NORMAL과 동일한 방법으로 나타내지만 바이트는 역순으로 배열됩니다. 각 바이트 내의 비트는 MQENC\_FLOAT\_IEEE\_NORMAL과 동일한 방법으로 배열됩니다.

### **MQENC\_FLOAT\_S390**

부동 소수점 숫자가 표준 System/390 부동 소수점 형식을 사용하여 표시됩니다. System/370에도 사용됩니다.

## **인코딩 구성**

MQMD에서 *Encoding* 필드의 값을 구성하기 위해서 필요한 인코딩을 설명하는 관련 상수가 함께 추가되거나 (두 번 이상 동일한 상수를 추가하지 않음) 또는 비트 단위의 OR 조작을 사용하여 결합될 수 있습니다(프로그래밍 언어가 비트 단위의 조작을 지원하는 경우).

어떤 방법을 사용하든 MQENC\_INTEGER\_\* 인코딩 중 하나와 MQENC\_DECIMAL\_\* encodings 및 MQENC\_FLOAT\_\* 인코딩 중 하나만 결합하십시오.

## **인코딩 분석**

*Encoding* 필드는 서브필드를 포함합니다. 이 때문에 정수, 팩형 10진수 또는 float 인코딩을 검사해야 하는 애플리케이션은 설명된 기술 중 하나를 사용해야 합니다.

## **비트 연산 사용**

프로그래밍 언어가 비트 연산을 지원하는 경우 다음 단계를 수행하십시오.

1. 필요한 인코딩 유형에 따라 다음 값 중 하나를 선택하십시오.
  - 2진 정수 인코딩에 대한 MQENC\_INTEGER\_MASK
  - 팩형 10진수 정수 인코딩에 대한 MQENC\_DECIMAL\_MASK
  - 부동 소수점 인코딩에 대한 MQENC\_FLOAT\_MASK
 값 A를 호출하십시오.
2. 비트 단위의 AND 연산을 사용하여 *Encoding* 필드를 A와 결합하고 결과 B를 호출하십시오.
3. B는 필요한 인코딩이며 인코딩의 해당 유형에 유효한 각 값과 동일한지 테스트할 수 있습니다.

<sup>4</sup> IEEE(Institute of Electrical and Electronics Engineers)

## 산술 사용

프로그래밍 언어가 비트 연산을 지원하지 않는 경우 정수 산술을 사용하여 다음 단계를 수행하십시오.

1. 필요한 인코딩 유형에 따라 다음 값 중 하나를 선택하십시오.

- 2진 정수 인코딩의 경우 1
- 팩형 10진수 정수 인코딩의 경우 16
- 부동 소수점 인코딩의 경우 256

값 A를 호출하십시오.

2. *Encoding* 필드의 값을 A로 나누고 결과 B를 호출하십시오.

3. B를 16으로 나누십시오. 결과 C를 호출합니다.

4. C에 16을 곱하고 B에서 빼십시오. 결과 D를 호출합니다.

5. D에 A를 곱합니다. 결과 E를 호출합니다.

6. E는 필요한 인코딩이며 인코딩의 해당 유형에 유효한 각 값과 동일한지 테스트할 수 있습니다.

## 시스템 아키텍처 인코딩의 요약

시스템 아키텍처 인코딩이 865 페이지의 표 631에 나와 있습니다.

표 631. 시스템 아키텍처 인코딩 요약			
시스템 아키텍처	2진 정수 인코딩	팩형 10진수 정수 인코딩	부동 소수점 인코딩
IBM i	정상	정상	IEEE 정상
Intel x86	역방향	역방향	IEEE 역방향
PowerPC®	정상	정상	IEEE 정상
System/390	정상	정상	System/390

## 보고 옵션 및 메시지 플래그

이 절에서는 MQGET, MQPUT 및 MQPUT1 호출에서 지정된 메시지 디스크립터 MQMD의 일부인 *Report* 및 *MsgFlags* 필드를 설명합니다.

이 절의 주제에서는 다음을 설명합니다.

- 보고 필드의 구조 및 큐 관리자의 처리 방법
- 애플리케이션의 보고서 필드 분석 방법
- 메시지 플래그 필드의 구조

MQMD 메시지 디스크립터에 대한 자세한 정보는 412 페이지의 『MQMD - 메시지 디스크립터』의 내용을 참조하십시오.

## 보고 필드의 구조

이 정보는 보고서 필드의 구조에 대해 설명합니다.

*Report* 필드는 3개의 별도 하위 필드로 나뉘는 32비트 정수입니다. 이러한 하위 필드는 다음을 식별합니다.

- 로컬 큐 관리자가 인식하지 못하면 거부되는 보고 옵션
- 로컬 큐 관리자가 인식하지 못해도 항상 허용되는 보고 옵션
- 다른 특정 조건을 충족하는 경우에만 허용되는 보고 옵션

각 하위 필드는 하위 필드에 해당되는 위치에 1비트가 있고 다른 곳에는 0비트가 있는 비트 마스크로 식별됩니다. 서브필드의 비트는 반드시 인접하지 않습니다. 비트는 숫자로 지정되며, 비트 0은 가장 중요한 비트이고 비트 31은 가장 중요하지 않은 비트입니다. 다음 마스크가 하위 필드를 식별하기 위해 정의됩니다.

### **MQRO\_REJECT\_UNSUP\_MASK**

이 마스크는 로컬 큐 관리자가 지원하지 않는 보고서 옵션으로 완료 코드 MQCC\_FAILED 및 이유 코드 MQRC\_REPORT\_OPTIONS\_ERROR로 MQPUT 또는 MQPUT1 호출에 실패하게 하는 *Report* 필드 내의 비트 위치를 식별합니다.

이 하위 필드는 비트 위치 3 및 11 ~ 13을 차지합니다.

### **MQRO\_ACCEPT\_UNSUP\_MASK**

이 마스크는 로컬 큐 관리자가 지원하지 않는 보고서 옵션이 MQPUT 또는 MQPUT1 호출에서 여전히 허용되는 *Report* 필드 내의 비트 위치를 식별합니다. 이유 코드 MQRC\_UNKNOWN\_REPORT\_OPTION이 있는 완료 코드 MQCC\_WARNING은 이 경우에 리턴됩니다.

이 하위 필드는 0 ~ 2, 4 ~ 10 및 24 ~ 31의 비트 위치를 차지합니다.

다음 보고 옵션은 이 하위 필드에 포함됩니다.

- MQRO\_ACTIVITY
- MQRO\_COPY\_MSG\_ID\_TO\_CORREL\_ID
- MQRO\_DEAD\_LETTER\_Q
- MQRO\_DISCARD\_MSG
- MQRO\_EXCEPTION
- MQRO\_EXCEPTION\_WITH\_DATA
- MQRO\_EXCEPTION\_WITH\_FULL\_DATA
- MQRO\_EXPIRATION
- MQRO\_EXPIRATION\_WITH\_DATA
- MQRO\_EXPIRATION\_WITH\_FULL\_DATA
- MQRO\_NAN
- MQRO\_NEW\_MSG\_ID
- MQRO\_NONE
- MQRO\_PAN
- MQRO\_PASS\_CORREL\_ID
- MQRO\_PASS\_MSG\_ID

### **MQRO\_ACCEPT\_UNSUP\_IF\_XMIT\_MASK**

이 마스크는 로컬 큐 관리자가 지원하지 않는 보고서 옵션이 다음 두 조건을 충족하는 제공된 MQPUT 또는 MQPUT1 호출에 허용되는 *Report* 필드 내의 비트 위치를 식별합니다.

- 메시지가 리모트 큐 관리자로 이동합니다.
- 애플리케이션은 로컬 전송 큐에 직접 메시지를 넣지 않고 있습니다(즉, MQOPEN 또는 MQPUT1 호출에 지정된 오브젝트 디스크립터의 *ObjectQMgrName* 및 *ObjectName* 필드에 의해 색별된 큐가 로컬 전송 큐가 아님).

해당 조건이 충족되는 경우 이유 코드 MQRC\_UNKNOWN\_REPORT\_OPTION을 포함한 완료 코드 MQCC\_WARNING이 리턴되고 조건이 충족되지 않는 경우 이유 코드 MQRC\_REPORT\_OPTIONS\_ERROR를 포함한 MQCC\_FAILED이 리턴됩니다.

이 하위 필드는 비트 위치 14 ~ 23을 차지합니다.

다음 보고 옵션은 이 하위 필드에 포함됩니다.

- MQRO\_COA
- MQRO\_COA\_WITH\_DATA
- MQRO\_COA\_WITH\_FULL\_DATA
- MQRO\_COD
- MQRO\_COD\_WITH\_DATA
- MQRO\_COD\_WITH\_FULL\_DATA

옵션이 큐 관리자가 인식하지 않는 *Report* 필드에 지정된 경우 큐 관리자는 비트 단위의 AND 연산을 사용하여 *Report* 필드 및 해당 서브필드의 마스크를 결합하여 각 서브필드를 차례로 확인합니다. 해당 조작의 결과가 0이 아닌 경우, 이전에 설명된 완료 코드 및 이유 코드가 리턴됩니다.

MQCC\_WARNING이 리턴된 경우 다른 경고 조건이 존재하면 리턴되는 이유 코드가 정의되지 않습니다.

로컬 큐 관리자가 인식할 수 없는 보고서 옵션을 지정하고 승인하는 기능은 리모트 큐 관리자가 인식하고 처리하는 보고서 옵션이 있는 메시지를 보낼 때 유용합니다.

## 보고서 필드 분석

*Report* 필드에는 하위 필드가 포함되어 있습니다. 이로 인해 메시지의 송신자가 특정 보고서를 요청했는지 여부를 확인해야 하는 애플리케이션은 설명된 기술 중 하나를 사용해야 합니다.

## 비트 연산 사용

프로그래밍 언어가 비트 연산을 지원하는 경우 다음 단계를 수행하십시오.

1. 확인할 보고서의 유형에 따라 다음 값 중 하나를 선택하십시오.

- COA 보고서를 위한 MQRO\_COA\_WITH\_FULL\_DATA
- COD 보고서를 위한 MQRO\_COD\_WITH\_FULL\_DATA
- 예외 보고서를 위한 MQRO\_EXCEPTION\_WITH\_FULL\_DATA
- 만기 보고서를 위한 MQRO\_EXPIRATION\_WITH\_FULL\_DATA

값 A를 호출하십시오.

z/OS에서 MQRO\*\_WITH\_FULL\_DATA 값 대신에 MQRO\*\_WITH\_DATA 값을 사용하십시오.

2. 비트 단위의 AND 연산을 사용하여 *Report* 필드를 A와 결합하고 결과 B를 호출하십시오.

3. 보고서 유형에 가능한 각 값과의 상등을 위해 B를 테스트하십시오.

예를 들어, A가 MQRO\_EXCEPTION\_WITH\_FULL\_DATA인 경우 B를 테스트하여 다음을 각각 확인하여 메시지의 송신자가 지정한 내용을 판별하십시오.

- MQRO\_NONE
- MQRO\_EXCEPTION
- MQRO\_EXCEPTION\_WITH\_DATA
- MQRO\_EXCEPTION\_WITH\_FULL\_DATA

테스트는 애플리케이션 논리에 가장 편리한 임의의 순서로 수행할 수 있습니다.

유사한 메소드를 사용하여 MQRO\_PASS\_MSG\_ID 또는 MQRO\_PASS\_CORREL\_ID 옵션을 테스트하십시오. 이 두 상수 중 적절한 값을 A로 선택하고 이전에 설명한 대로 진행하십시오.

## 산술 사용

프로그래밍 언어가 비트 연산을 지원하지 않는 경우 정수 산술을 사용하여 다음 단계를 수행하십시오.

1. 확인할 보고서의 유형에 따라 다음 값 중 하나를 선택하십시오.

- COA 보고서를 위한 MQRO\_COA
- COD 보고서를 위한 MQRO\_COD
- 예외 보고서를 위한 MQRO\_EXCEPTION
- 만기 보고서를 위한 MQRO\_EXPIRATION

값 A를 호출하십시오.

2. *Report* 필드를 A로 나누고 결과 B를 호출하십시오.

3. B를 8로 나눕니다. 결과 C를 호출합니다.

4. C에 8을 곱하고 B에서 뺍니다. 결과 D를 호출합니다.

5. D에 A를 곱합니다. 결과 E를 호출합니다.

6. 보고서 유형에 가능한 각 값과의 상등을 위해 E를 테스트하십시오.

예를 들어, A가 MQRO\_EXCEPTION인 경우 E를 테스트하여 다음을 각각 확인하여 메시지의 송신자가 지정한 내용을 판별하십시오.

- MQRO\_NONE
- MQRO\_EXCEPTION
- MQRO\_EXCEPTION\_WITH\_DATA
- MQRO\_EXCEPTION\_WITH\_FULL\_DATA

테스트는 애플리케이션 논리에 가장 편리한 임의의 순서로 수행할 수 있습니다.

다음 의사 코드는 예외 보고 메시지에 대한 이 기술을 설명합니다.

```
A = MQRO_EXCEPTION
B = Report/A
C = B/8
D = B - C*8
E = D*A
```

유사한 메소드를 사용하여 MQRO\_PASS\_MSG\_ID 또는 MQRO\_PASS\_CORREL\_ID 옵션을 테스트하십시오. 이 두 상수 중 적절한 값을 A로 선택하고 이전에 설명한 대로 진행하지만 이전 단계의 값 8을 값 2로 바꿉니다.

## 메시지 플래그 필드의 구조

이 정보는 메시지 플래그 필드의 구조에 대해 설명합니다.

*MsgFlags* 필드는 3개의 별도 하위 필드로 나뉘는 32비트 정수입니다. 이러한 하위 필드는 다음을 식별합니다.

- 로컬 큐 관리자가 인식하지 못하면 거부되는 메시지 플래그
- 로컬 큐 관리자가 인식하지 못해도 항상 허용되는 메시지 플래그
- 다른 특정 조건을 충족하는 경우에만 허용되는 메시지 플래그

**참고:** *MsgFlags*의 모든 서브필드는 큐 관리자에서 사용하기 위해 예약됩니다.

각 하위 필드는 하위 필드에 해당되는 위치에 1비트가 있고 다른 곳에는 0비트가 있는 비트 마스크로 식별됩니다. 비트는 숫자로 지정되며, 비트 0은 가장 중요한 비트이고 비트 31은 가장 중요하지 않은 비트입니다. 다음 마스크가 하위 필드를 식별하기 위해 정의됩니다.

### MQMF\_REJECT\_UNSUP\_MASK

이 마스크는 로컬 큐 관리자가 지원하지 않는 메시지 플래그로 완료 MQCC\_FAILED 및 이유 코드 MQRC\_REPORT\_OPTIONS\_ERROR로 MQPUT 또는 MQPUT1 호출에 실패하게 하는 *MsgFlags* 필드 내의 비트 위치를 식별합니다.

이 하위 필드는 비트 위치 20 ~ 31을 차지합니다.

다음 메시지 플래그가 이 하위 필드에 포함됩니다.

- MQMF\_LAST\_MSG\_IN\_GROUP
- MQMF\_LAST\_SEGMENT
- MQMF\_MSG\_IN\_GROUP
- MQMF\_SEGMENT
- MQMF\_SEGMENTATION\_ALLOWED
- MQMF\_SEGMENTATION\_INHIBITED

### MQMF\_ACCEPT\_UNSUP\_MASK

이 마스크는 로컬 큐 관리자가 지원하지 않는 메시지 플래그가 MQPUT 또는 MQPUT1 호출에 허용되는 *MsgFlags* 필드 내의 비트 위치를 식별합니다. 완료 코드는 MQCC\_OK입니다.

이 하위 필드는 비트 위치 0 ~ 11을 차지합니다.



## MQMF\_ACCEPT\_UNSUP\_IF\_XMIT\_MASK

이 마스크는 로컬 큐 관리자가 지원하지 않는 메시지 플래그가 다음 두 조건을 충족하는 제공된 MQPUT 또는 MQPUT1 호출에 허용되는 *MsgFlags* 필드 내의 비트 위치를 식별합니다.

- 메시지가 리모트 큐 관리자로 이동합니다.
- 애플리케이션은 로컬 전송 큐에 직접 메시지를 넣지 않고 있습니다(즉, MQOPEN 또는 MQPUT1 호출에 지정된 오브젝트 디스크립터의 *ObjectQMGrName* 및 *ObjectName* 필드에 의해 색별된 큐가 로컬 전송 큐가 아님).

해당 조건이 충족되는 경우 완료 코드 MQCC\_OK가 리턴되고 충족되지 않는 경우 이유 코드 MQRC\_MSG\_FLAGS\_ERROR가 있는 MQCC\_FAILED로 리턴됩니다.

이 하위 필드는 비트 위치 12 ~ 19를 차지합니다.

플래그가 큐 관리자가 인식하지 않는 *MsgFlags* 필드에 지정된 경우 큐 관리자는 비트 단위의 AND 연산을 사용하여 *MsgFlags* 필드 및 해당 서브필드의 마스크를 결합하여 각 서브필드를 차례로 확인합니다. 해당 조작의 결과가 0이 아닌 경우, 이전에 설명된 완료 코드 및 이유 코드가 리턴됩니다.

## 데이터 변환 엑시트

이 주제의 콜렉션은 데이터 변환 엑시트에 대한 인터페이스 및 데이터 변환 요구 시 큐 관리자에서 수행되는 처리에 대해 설명합니다.

데이터 변환에 대한 자세한 정보는 *IBM MQ*의 데이터 변환 (<https://www.ibm.com/support/pages/node/317869>)을 참조하십시오.

애플리케이션 메시지 데이터를 수신 애플리케이션에서 요구하는 표현으로 변환하기 위해 데이터 변환 엑시트가 MQGET 호출 처리의 일부로 호출됩니다. 애플리케이션 메시지 데이터의 변환은 선택적입니다. MQGMO\_CONVERT 옵션을 MQGET 호출에서 지정해야 합니다.

다음 주제에 대해 설명합니다.

- MQGMO\_CONVERT 옵션에 응답하여 큐 관리자에서 수행된 처리는 869 페이지의 『변환 처리』의 내용을 참조하십시오.
- 내장 형식 처리 시 큐 관리자에서 사용된 규칙 처리. 이 규칙은 사용자 작성 엑시트에 대해서도 권장됩니다. 870 페이지의 『처리 규칙』의 내용을 참조하십시오.
- 보고 메시지 변환에 대한 특별한 고려사항은 874 페이지의 『보고 메시지의 변환』의 내용을 참조하십시오.
- 데이터 변환 엑시트에 전달된 매개변수. 886 페이지의 『MQ DATA CONV EXIT - 데이터 변환 엑시트』의 내용을 참조하십시오.
- 다른 표현 간에 문자 데이터를 변환하기 위해 엑시트에서 사용될 수 있는 호출은 880 페이지의 『MQXCNCV - 문자 변환』의 내용을 참조하십시오.
- 엑시트에 특정한 데이터 구조 매개변수는 875 페이지의 『MQDXP - 데이터 변환 엑시트 매개변수』의 내용을 참조하십시오.

## 변환 처리

이 정보는 MQGMO\_CONVERT 옵션에 응답하여 큐 관리자에서 수행된 처리에 대해 설명합니다.

MQGMO\_CONVERT 옵션이 MQGET 호출에 지정된 경우 큐 관리자는 다음 조치를 수행하며 애플리케이션으로 리턴될 메시지가 있습니다.

1. 다음 중 하나 이상에 해당하면 변환할 필요가 없습니다.

- 메시지 데이터가 MQGET 호출을 발행하는 애플리케이션에 필요한 문자 세트 및 인코딩에 이미 있습니다. 애플리케이션은 호출을 발행하기 전에 MQGET 호출의 **MsgDesc** 매개변수에 있는 *CodedCharSetId* 및 *Encoding* 필드를 필요한 값으로 설정해야 합니다.
- 메시지 데이터의 길이는 0입니다.
- MQGET 호출의 **Buffer** 매개변수의 길이는 0입니다.

이러한 경우 MQGET 호출을 발행하는 애플리케이션으로 변환하지 않고 메시지가 리턴됩니다. **MsgDesc** 매개변수의 *CodedCharSetId* 및 *Encoding* 값은 메시지의 제어 정보에 있는 값으로 설정되며 호출은 다음 완료 코드 및 이유 코드 조합 중 하나로 완료됩니다.

표 632. 완료 코드 및 이유 코드 조합

완료 코드	이유 코드
MQCC_OK	MQRC_NONE
MQCC_WARNING	MQRC_TRUNCATED_MSG_ACCEPTED
MQCC_WARNING	MQRC_TRUNCATED_MSG_FAILED

메시지 데이터의 문자 세트 또는 인코딩이 **MsgDesc** 매개변수의 해당 값과 다른 경우에만 다음 단계가 수행되고 변환할 데이터가 있습니다.

2. 메시지에서 제어 정보의 *Format* 필드에 값 MQFMT\_NONE이 있는 경우 메시지는 완료 코드 MQCC\_WARNING 및 이유 코드 MQRC\_FORMAT\_ERROR로 변환되지 않고 리턴됩니다.  
다른 모든 경우에 변환 처리가 계속됩니다.
3. 메시지는 큐에서 제거되고 **Buffer** 매개변수와 같은 크기인 임시 버퍼에 위치합니다. 찾아보기 조작의 경우, 메시지가 큐에서 제거되지 않고 임시 버퍼에 복사됩니다.
4. 메시지가 버퍼에 맞게 잘린 경우, 다음이 수행됩니다.
  - MQGMO\_ACCEPT\_TRUNCATED\_MSG 옵션을 지정하지 않은 경우, 메시지가 변환되지 않은 채로 리턴되고 완료 코드 MQCC\_WARNING과 이유 코드 MQRC\_TRUNCATED\_MSG\_FAILED이 나타납니다.
  - MQGMO\_ACCEPT\_TRUNCATED\_MSG 옵션이 지정되고 완료 코드가 MQCC\_WARNING으로 설정된 경우 이유 코드가 MQRC\_TRUNCATED\_MSG\_ACCEPTED로 설정되고 변환 처리가 계속됩니다.
5. 메시지가 잘림 없이 버퍼에서 수용할 수 있거나 MQGMO\_ACCEPT\_TRUNCATED\_MSG 옵션이 지정된 경우 다음이 수행됩니다.
  - 형식이 내장 형식이면 버퍼가 큐 관리자의 데이터 변환 서비스에 전달됩니다.
  - 형식이 내장 형식이 아닌 경우 버퍼는 형식과 동일한 이름의 사용자 작성 엑시트에 전달됩니다. 엑시트를 찾을 수 없는 경우 메시지는 완료 코드 MQCC\_WARNING 및 이유 코드 MQRC\_FORMAT\_ERROR로 변환되지 않고 리턴됩니다.

오류가 발생하지 않으면 데이터 변환 서비스 또는 사용자 작성 엑시트에서 변환된 메시지 및 MQGET 호출을 발행하는 애플리케이션에 리턴되는 완료 코드 및 이유 코드가 출력됩니다.
6. 변환에 성공하면 큐 관리자가 변환된 메시지를 애플리케이션으로 리턴합니다. 이 경우 MQGET 호출로 리턴된 완료 코드 및 이유 코드는 다음 조합 중 하나입니다.

표 633. 완료 코드 및 이유 코드 조합

완료 코드	이유 코드
MQCC_OK	MQRC_NONE
MQCC_WARNING	MQRC_TRUNCATED_MSG_ACCEPTED

그러나, 사용자 작성 엑시트에 의해 변환이 수행된 경우에는 변환에 성공하더라도 다른 이유 코드가 리턴될 수 있습니다.

변환에 실패하면 큐 관리자가 변환되지 않은 메시지를 애플리케이션에 리턴합니다. **MsgDesc** 매개변수의 *CodedCharSetId* 및 *Encoding* 필드는 메시지의 제어 정보에 있는 값으로 설정되고 완료 코드는 MQCC\_WARNING입니다.

## 처리 규칙

기본 제공 형식을 변환할 때 큐 관리자는 설명된 처리 규칙을 따릅니다.

또한 이것이 큐 관리자에 의해 적용되지 않더라도 사용자 작성 엑시트는 이러한 규칙을 따라야 합니다. 큐 관리자에 의해 변환된 내장 형식은 다음과 같습니다.

- MQFMT\_ADMIN
- MQFMT\_CICS(z/OS 전용)
- MQFMT\_COMMAND\_1
- MQFMT\_COMMAND\_2
- MQFMT\_DEAD\_LETTER\_HEADER
- MQFMT\_DIST\_HEADER
- MQFMT\_EVENT 버전 1
- MQFMT\_EVENT 버전 2
- MQFMT\_IMS
- MQFMT\_IMS\_VAR\_STRING
- MQFMT\_MD\_EXTENSION
- MQFMT\_PCF
- MQFMT\_REF\_MSG\_HEADER
- MQFMT\_RF\_HEADER
- MQFMT\_RF\_HEADER\_2
- MQFMT\_STRING
- MQFMT\_TRIGGER
- MQFMT\_WORK\_INFO\_HEADER(z/OS 전용)
- MQFMT\_XMIT\_Q\_HEADER

1. 메시지가 변환 중 확장되고 **Buffer** 매개변수의 크기를 초과하는 경우 다음이 수행됩니다.

- MQGMO\_ACCEPT\_TRUNCATED\_MSG 옵션을 지정하지 않은 경우, 메시지가 변환되지 않은 채로 리턴되고 완료 코드 MQCC\_WARNING과 이유 코드 MQRC\_CONVERTED\_MSG\_TOO\_BIG이 나타납니다.
- MQGMO\_ACCEPT\_TRUNCATED\_MSG 옵션이 지정되고 메시지가 잘리며 완료 코드가 MQCC\_WARNING으로 설정된 경우 이유 코드가 MQRC\_TRUNCATED\_MSG\_ACCEPTED로 설정되고 변환 처리가 계속됩니다.

2. 잘림이 발생하는 경우(변환 전 또는 변환 중) **Buffer** 매개변수에서 리턴된 유효 바이트의 수는 버퍼 길이 미만일 수 있습니다.

예를 들어, 4바이트 정수 또는 DBCS 문자가 버퍼의 끝에 걸쳐 발생할 수 있습니다. 정보의 불완전한 요소는 변환되지 않고 리턴된 메시지의 해당 바이트에 유효한 정보를 포함하지 않습니다. 변환 전에 잘린 메시지가 변환하는 동안 축소되는 경우에도 이 동작이 발생할 수 있습니다.

리턴된 올바른 바이트 수가 버퍼 길이 미만이면 버퍼 끝에 있는 미사용 바이트가 널로 설정됩니다.

3. 배열이나 문자열을 버퍼의 끝에 걸치면 가능한 많은 데이터가 변환됩니다. 불완전한 특정 배열 요소 또는 DBCS 문자만 변환되지 않으며 선행 배열 요소 또는 문자가 변환됩니다.
4. 잘림이 발생하는 경우(변환 전 또는 변환 중) **DataLength** 매개변수에 대해 리턴되는 길이는 잘리기 전에 변환되지 않는 메시지의 길이입니다.
5. 문자열이 1바이트 문자 세트(SBCS), 2바이트 문자 세트(DBCS) 또는 멀티바이트 문자 세트(MBCS) 간에 변환할 때 문자열이 확장되거나 축소될 수 있습니다.

- PCF 형식 MQFMT\_ADMIN, MQFMT\_EVENT 및 MQFMT\_PCF에서 MQCFST 및 MQCFSL 구조의 문자열은 변환 후 문자열을 수용할 필요에 따라 확장되거나 축소됩니다.

문자열 목록 구조 MQCFSL의 경우, 목록의 문자열이 각기 다른 크기로 확장 또는 축소될 수 있습니다. 이 경우, 큐 관리자는 더 짧은 문자열을 공백으로 채워 변환 후 가장 긴 문자열과 동일한 길이가 되도록 합니다.

- 형식 MQFMT\_REF\_MSG\_HEADER에서 SrcEnvOffset, SrcNameOffset, DestEnvOffset 및 DestNameOffset 필드에 의해 처리된 문자열은 변환 후 문자열을 수용할 필요에 따라 확장되거나 축소됩니다.

- 형식 MQFMT\_RF\_HEADER에서 NameValueString 필드는 변환 후 이름-값 쌍을 수용하기 위해 필요에 따라 확장되거나 축소됩니다.
- 수정된 필드 크기의 구조에서 중요한 정보가 손실되지 않은 제공된 고정 필드 내에서 문자열을 확장하거나 축소할 수 있습니다. 이러한 점에서 필드의 첫 번째 널 문자 뒤의 후미 공백 및 문자는 중요하지 않은 것으로 처리됩니다.
  - 문자열이 확장되었지만 필드에서 변환된 문자열을 수용하기 위해 중요하지 않은 문자만 제거해야 하는 경우 변환이 성공하고 MQCC\_OK 및 이유 코드 MQRC\_NONE으로 호출이 완료합니다(다른 오류가 없다고 가정).
  - 문자열이 확장되지만 필드에 맞도록 변환된 문자열에서 중요한 문자를 제거해야 하는 경우 메시지는 변환되지 않고 리턴되며 MQCC\_WARNING 및 이유 코드 MQRC\_CONVERTED\_STRING\_TOO\_BIG를 사용하여 호출이 완료됩니다.

**참고:** 이유 코드 MQRC\_CONVERTED\_STRING\_TOO\_BIG은 이 경우 MQGMO\_ACCEPT\_TRUNCATED\_MSG 옵션의 지정 여부를 판별합니다.

- 문자열이 축소되면 큐 관리자가 필드의 길이에 맞게 문자열을 공백으로 채웁니다.
6. 사용자 데이터가 뒤따르는 하나 이상의 MQ 헤더 구조로 구성된 메시지의 경우 하나 이상의 헤더 구조가 변환될 수 있지만 나머지 메시지는 변환되지 않을 수 있습니다. 그러나(두 가지 예외로) *CodedCharSetId* 및 각 헤더 구조의 *Encoding* 필드는 항상 헤더 구조 다음에 오는 데이터의 문자 세트 및 인코딩을 올바르게 표시합니다.

두 가지 예외는 MQCIH 및 MQIIH 구조입니다. 여기서 *CodedCharSetId* 및 *Encoding* 필드의 값은 중요하지 않습니다. 해당 구조의 경우, 이 구조를 따르는 데이터가 MQCIH 또는 MQIIH 구조 자체와 동일한 문자 세트 및 인코딩에 있습니다.

7. 검색 중인 메시지의 제어 정보 또는 **MsgDesc** 매개변수에 있는 *CodedCharSetId* 또는 *Encoding* 필드가 정의되지 않았거나 지원되지 않는 값을 지정하는 경우, 정의되지 않았거나 지원되지 않는 값을 메시지 변환에 사용할 필요가 없으면 큐 관리자가 오류를 무시할 수 있습니다.

예를 들어, 메시지의 *Encoding* 필드가 지원되지 않는 부동 소수점을 지정하지만 메시지에 정수 데이터만 포함되거나 변환할 필요가 없는 부동 소수점 데이터가 포함된 경우(소스 및 대상 부동 소수점이 동일하기 때문에) 오류가 진단되지 않을 수 있습니다.

오류가 진단되면 완료 코드 MQCC\_WARNING 및 MQRC\_SOURCE\_\*\_ERROR 또는 MQRC\_TARGET\_\*\_ERROR 이유 코드 중 하나(해당되는 경우)와 함께 메시지가 변환되지 않은 상태로 리턴됩니다. **MsgDesc** 매개변수의 *CodedCharSetId* 및 *Encoding* 필드는 메시지의 제어 정보에 있는 값으로 설정됩니다.

오류가 진단되지 않고 변환이 성공적으로 완료되면 **MsgDesc** 매개변수의 *CodedCharSetId* 및 *Encoding* 필드에 리턴된 값은 MQGET 호출을 발행하는 애플리케이션에서 지정한 값입니다.

8. 모든 경우에 메시지가 변환되지 않은 애플리케이션으로 리턴되면 완료 코드는 MQCC\_WARNING으로 설정되고 **MsgDesc** 매개변수의 *CodedCharSetId* 및 *Encoding* 필드는 변환되지 않은 데이터에 적합한 값으로 설정됩니다. 이는 MQFMT\_NONE에 대해서도 수행됩니다.

**Reason** 매개변수는 메시지가 잘린 경우를 제외하고는 변환을 수행할 수 없는 이유를 나타내는 코드로 설정됩니다. 잘림과 관련된 이유 코드는 변환과 관련된 이유 코드보다 우선합니다(잘린 메시지가 변환되었는지 판별하려면 **MsgDesc** 매개변수의 *CodedCharSetId* 및 *Encoding* 필드에서 리턴된 값을 확인하십시오.)

오류가 진단될 때 특정 이유 코드가 리턴되거나 일반 이유 코드 MQRC\_NOT\_CONVERTED가 리턴됩니다. 리턴되는 이유 코드는 기본 데이터 변환 서비스의 진단 기능에 따라 다릅니다.

9. 완료 코드 MQCC\_WARNING이 리턴되고 둘 이상의 이유 코드가 관련되는 경우 우선순위는 다음과 같습니다.
- a. 다음과 같은 이유가 다른 모든 이유보다 우선합니다. 이 그룹의 이유 중 하나만 발생할 수 있습니다.
    - MQRC\_SIGNAL\_REQUEST\_ACCEPTED
    - MQRC\_TRUNCATED\_MSG\_ACCEPTED
  - b. 남은 이유 코드에서 우선순위의 순서가 정의되어 있지 않습니다.

10. MQGET 호출 완료 시:

- 다음 이유 코드는 메시지가 성공적으로 변환되었음을 표시합니다.

- MQRC\_NONE
- 다음 이유 코드는 메시지가 성공적으로 변환되었을 수 있음을 표시합니다 (**MsgDesc** 매개변수의 *CodedCharSetId* 및 *Encoding* 필드를 확인하여 찾으십시오).
- MQRC\_MSG\_MARKED\_BROWSE\_CO\_OP
- MQRC\_TRUNCATED\_MSG\_ACCEPTED
- 기타 모든 이유 코드는 메시지가 변환되지 않았음을 표시합니다.

다음 처리는 내장 형식에 특정하며 사용자 정의 형식에는 적용되지 않습니다.

11. 다음 형식은 제외됩니다.

- MQFMT\_ADMIN
- MQFMT\_COMMAND\_1
- MQFMT\_COMMAND\_2
- MQFMT\_EVENT
- MQFMT\_IMS\_VAR\_STRING
- MQFMT\_PCF
- MQFMT\_STRING

내장 형식은 큐 이름에 유효한 문자로 SBCS 문자를 포함하지 않은 문자 세트로부터 변환할 수 없습니다. 이러한 변환을 수행하기 위한 작성을 시도하는 경우 메시지는 완료 코드 MQCC\_WARNING 및 이유 코드 MQRC\_SOURCE\_CCSID\_ERROR 또는 MQRC\_TARGET\_CCSID\_ERROR로 변환되지 않고 리턴됩니다(적절한 경우).

유니코드 문자 세트 UTF-16은 큐 이름에 올바른 문자로 SBCS 문자를 포함하지 않은 문자 세트의 예입니다.

12. 내장 형식의 메시지 데이터가 잘리는 경우 문자열 길이 또는 요소나 구조의 개수가 포함된 메시지 내의 필드는 실제로 애플리케이션에 리턴되는 데이터 길이를 반영하도록 조정되지 않습니다. 메시지 데이터 내의 해당 필드에 대해 리턴된 값은 잘림 전에 메시지에 적용할 수 있는 값입니다.

잘린 MQFMT\_ADMIN 메시지 등의 메시지를 처리할 때 애플리케이션은 리턴된 데이터의 끝을 넘어 데이터에 액세스하지 않아야 합니다.

13. 형식 이름이 MQFMT\_DEAD\_LETTER\_HEADER인 경우 메시지 데이터는 MQDLH 구조로 시작하며 그 뒤에 애플리케이션 메시지 데이터가 0 바이트 이상 계속 될 수 있습니다. 애플리케이션 메시지 데이터의 형식 이름, 문자 세트 및 인코딩은 메시지 시작 부분에서 MQRMH 구조의 *Format*, *CodedCharSetId* 및 *Encoding* 필드에 의해 정의됩니다. MQDLH 구조와 애플리케이션 메시지 데이터는 서로 다른 문자 세트와 인코딩을 가질 수 있으므로 MQDLH 구조 및 애플리케이션 메시지 데이터 중 하나, 다른 하나 또는 모두가 변환이 필요할 수 있습니다.

큐 관리자는 필요에 따라 먼저 MQDLH 구조를 변환합니다. 변환이 성공적이거나 MQDLH 구조에 변환이 필요하지 않은 경우 큐 관리자가 MQDLH 구조의 *CodedCharSetId* 및 *Encoding* 필드를 확인하여 애플리케이션 메시지 데이터의 변환이 필요한지 확인합니다. 변환이 필요한 경우 큐 관리자는 MQDLH 구조의 *Format* 필드에 지정된 이름을 사용하여 사용자 작성 엑시트를 호출하거나 변환 자체를 수행합니다 (*Format*이 내장 형식의 이름인 경우).

MQGET 호출이 MQCC\_WARNING의 완료 코드를 리턴하고 이유 코드가 변환이 실패했음을 나타내는 것 중 하나이면 다음 중 하나가 적용됩니다.

- MQDLH 구조를 변환할 수 없습니다. 이 경우, 애플리케이션 메시지 데이터도 변환되지 않습니다.
- MQDLH 구조가 변환되었지만, 애플리케이션 메시지 데이터는 변환되지 않았습니다.

애플리케이션은 이전에 적용된 값을 판별하기 위해 **MsgDesc** 매개변수의 *CodedCharSetId* 및 *Encoding* 필드에서 리턴된 값과 MQDLH 구조의 값을 검사할 수 있습니다.

14. 형식 이름이 MQFMT\_XMIT\_Q\_HEADER인 경우 메시지 데이터는 MQXQH 구조로 시작하며 그 뒤에 애플리케이션 데이터가 0 바이트 이상 계속 될 수 있습니다. 이 추가 데이터는 일반적으로 애플리케이션 메시지 데이터이지만(길이가 0일 수 있음) 추가 데이터의 시작 부분에 하나 이상의 추가 MQ 헤더 구조가 있을 수도 있습니다.

MQXQH 구조는 큐 관리자의 문자 세트 및 인코딩에 있어야 합니다. MQXQH 구조 뒤에 오는 데이터의 형식, 문자 세트 및 인코딩은 MQXQH 내에 포함된 MQMD 구조의 Format, CodedCharSetId 및 Encoding 필드에 의해 제공됩니다. 각 후속 MQ 헤더 구조에 대해 구조의 Format, CodedCharSetId 및 Encoding 필드는 해당 구조를 따르는 데이터를 설명합니다. 해당 데이터는 또 다른 MQ 헤더 구조이거나 애플리케이션 메시지 데이터입니다.

MQGMO\_CONVERT 옵션이 MQFMT\_XMIT\_Q\_HEADER 메시지에 지정되는 경우 애플리케이션 메시지 데이터 및 특정 MQ 헤더 구조가 변환되지만 MQXQH 구조의 데이터는 변환되지 않습니다. 따라서 MQGET 호출에서 리턴 시:

- **MsgDesc** 매개변수의 Format, CodedCharSetId 및 Encoding 필드 값은 애플리케이션 메시지 데이터가 아닌 MQXQH 구조의 데이터를 설명합니다. 따라서 이 값은 MQGE 호출을 발행한 애플리케이션에 의해 지정된 값과 동일하지 않습니다.

이로 인해 MQGMO\_CONVERT 옵션을 지정하여 전송 큐에서 메시지를 반복적으로 가져오는 애플리케이션은 각 MQGET 호출 전에 **MsgDesc** 매개변수의 CodedCharSetId 및 Encoding 필드를 애플리케이션 메시지 데이터에 필요한 값으로 재설정해야 합니다.

- 마지막 MQ 헤더 구조의 Format, CodedCharSetId 및 Encoding 필드 값은 애플리케이션 메시지 데이터를 설명합니다. 다른 MQ 헤더 구조가 없는 경우 애플리케이션 메시지 데이터는 MQXQH 구조 내의 MQMD 구조에서 해당 필드로 설명됩니다. 변환이 성공적인 경우 이 값은 MQGET 호출을 발행한 애플리케이션에 의해 **MsgDesc** 매개변수에 지정된 것과 동일합니다.

메시지가 분배 목록 메시지인 경우 MQXQH 구조 뒤에 MQDH 구조가 옵니다(MQOR 및 MQPMR 레코드의 해당 배열 추가). 차례로 0개 이상의 MQ 헤더 구조 및 0이상의 바이트 애플리케이션 메시지 데이터가 이어질 수 있습니다. MQXQH 구조와 마찬가지로 MQDH 구조는 MQGMO\_CONVERT 옵션이 지정되더라도 큐 관리자의 문자 세트 및 인코딩에 있어야 하며 MQGET 호출에서 변환되지 않습니다.

이전에 설명한 MQXQH 및 MQDH 구조의 처리는 메시지 채널 에이전트가 전송 큐에서 메시지를 가져올 때 주로 사용하기 위해 의도됩니다.

## 보고 메시지의 변환

일반적으로 보고서 메시지는 원래 메시지의 송신자에 의해 지정된 보고서 옵션에 따라 애플리케이션 메시지 데이터의 변화량을 포함할 수 있습니다. 그러나 활동 보고서는 데이터를 포함할 수 있지만 상수에 \*\_WITH\_DATA를 표시하는 보고서 옵션은 없습니다.

특히, 보고 메시지는

1. 애플리케이션 메시지 데이터를 포함할 수 없습니다.
2. 원본 메시지에 있는 애플리케이션 메시지 데이터의 일부를 포함할 수 있습니다.

이는 원래 메시지의 송신자가 MQRO\*\_WITH\_DATA를 지정하고 메시지가 100바이트보다 더 긴 경우 발생할 수 있습니다.

3. 원래 메시지의 모든 애플리케이션 메시지 데이터

이는 원래 메시지의 송신자가 MQRO\*\_WITH\_DATA를 지정하거나 MQRO\*\_WITH\_DATA를 지정하고 메시지가 100바이트 이하인 경우 발생합니다.

큐 관리자 또는 메시지 채널 에이전트가 보고 메시지를 생성할 때 원래 메시지의 형식 이름을 보고 메시지의 제어 정보에 있는 Format 필드에 복사합니다. 보고 메시지의 형식 이름은 실제로 보고서 메시지(이전 사례 1 및 2)에 있는 길이와 다른 길이의 데이터를 내재합니다.

보고 메시지가 검색될 때 MQGMO\_CONVERT 옵션이 지정되는 경우:

- 이전의 사례 1의 경우 보고 메시지에 데이터가 없기 때문에 데이터 변환 엑시트가 호출되지 않습니다.
- 이전의 사례 3의 경우 형식 이름은 메시지 데이터의 길이를 정확하게 내재합니다.
- 그러나 이전의 사례 2의 경우 데이터 변환 엑시트가 호출되어 형식 이름으로 내재된 짧은 메시지를 변환합니다.

또한 엑시트에 전달된 이유 코드는 일반적으로 MQRC\_NONE입니다(즉, 이유 코드는 메시지가 잘렸음을 나타내지 않음). 이는 MQGET 호출에 대한 응답으로 수신자의 큐 관리자가 아닌 보고 메시지의 송신자에 의해 메시지 데이터가 잘렸기 때문에 발생합니다.

이러한 가능성 때문에 데이터 변환 엑시트는 형식 이름을 사용하여 전달된 데이터의 길이를 추론하지 않아야 합니다. 대신 엑시트는 제공된 데이터 길이를 확인해야 하며 형식 이름으로 내재된 길이보다 작은 데이터를 변환할 준비가 되어 있어야 합니다. 데이터가 성공적으로 변환될 수 있는 경우 완료 코드 MQCC\_OK 및 이유 코드 MQRC\_NONE이 엑시트로 리턴되어야 합니다. 변환될 메시지 데이터의 길이는 **InBufferLength** 매개변수와 같이 엑시트로 전달됩니다.

## 제품별 프로그래밍 인터페이스

### MQDXP - 데이터 변환 엑시트 매개변수

MQDXP 구조는 MQGET 호출 처리의 일부로 메시지 데이터를 변환하기 위해 엑시트를 호출할 때 큐 관리자가 데이터 변환 엑시트에 전달하는 매개변수입니다. 데이터 변환 엑시트에 대한 세부사항은 MQ\_DATA\_CONV\_EXIT 호출에 대한 설명을 참조하십시오.

MQDXP의 문자 데이터는 로컬 큐 관리자의 문자 세트입니다. 이는 **CodedCharSetId** 큐 관리자 속성으로 지정됩니다. MQDXP의 숫자 데이터는 고유 시스템 인코딩이며 이는 MQENC\_NATIVE에 의해 지정됩니다.

MQDXP의 *DataLength*, *CompCode*, *Reason* 및 *ExitResponse* 필드만 엑시트에 의해 변경될 수 있습니다. 기타 필드에 대한 변경사항은 무시됩니다. 그러나 변환 중인 메시지가 논리 메시지의 일부만을 포함하는 세그먼트일 경우 *DataLength* 필드를 변경할 수 없습니다.

엑시트에서 큐 관리자로 제어가 되돌아갈 때 큐 관리자가 MQDXP에서 리턴되는 값을 검사합니다. 리턴된 값이 유효하지 않은 경우 엑시트에 *ExitResponse*의 MQXDR\_CONVERSION\_FAILED를 리턴한 것처럼 큐 관리자는 처리를 계속합니다. 그러나 큐 관리자는 이 경우에 엑시트로 리턴한 *CompCode* 및 *Reason* 필드의 값을 무시하고 대신 해당 필드가 엑시트에 대한 입력에 포함하고 있는 값을 사용합니다. MQDXP에서는 다음 값으로 인해 아래와 같은 처리가 발생합니다.

- *ExitResponse* 필드는 MQXDR\_OK 및 MQXDR\_CONVERSION\_FAILED가 아님
- *CompCode* 필드는 MQCC\_OK 및 MQCC\_WARNING이 아님
- *DataLength* 필드가 0보다 작거나 메시지가 변환되고 있을 때 논리 메시지의 부분만 포함하는 세그먼트인 경우 *DataLength* 필드가 변경됩니다.

다음 표에는 구조의 필드가 요약되어 있습니다.

표 634. MQDXP의 필드		
필드	설명	토픽
<i>StrucId</i>	구조 ID	<a href="#">StrucId</a>
<i>Version</i>	구조 버전 번호	<a href="#">Version</a>
<i>AppOptions</i>	애플리케이션 옵션	<a href="#">AppOptions</a>
<i>Encoding</i>	애플리케이션에 필요한 숫자 인코딩	<a href="#">Encoding</a>
<i>CodedCharSetId</i>	애플리케이션에 필요한 문자 세트	<a href="#">CodedCharSetId</a>
<i>DataLength</i>	메시지 데이터의 길이(바이트)	<a href="#">DataLength</a>
<i>CompCode</i>	완료 코드	<a href="#">CompCode</a>
<i>Reason</i>	<i>CompCode</i> 를 규정하는 이유 코드	<a href="#">Reason</a>
<i>ExitResponse</i>	엑시트로부터의 응답	<a href="#">ExitResponse</a>
<i>Hconn</i>	연결 핸들	<a href="#">Hconn</a>
<i>pEntryPoints</i>	MQIEP 구조의 주소	<a href="#">pEntryPoints</a>

## 필드

MQDXP 구조에는 다음과 같은 필드가 포함되며 필드는 알파벳순으로 설명합니다.

### AppOptions

유형: MQLONG

MQGET 호출을 발행하는 애플리케이션에 의해 지정되는 MQGMO 구조의 *Options* 필드 사본입니다.

MQGMO\_ACCEPT\_TRUNCATED\_MSG 옵션이 지정되었는지 확인하기 위해 엑시트가 검사해야 하는 경우도 있습니다.

엑시트의 입력 필드입니다.

### CodedCharSetId

유형: MQLONG

MQGET 호출을 발행하는 애플리케이션에 필요한 문자 세트의 코드화 문자 세트 ID입니다. 자세한 정보는 MQMD 구조의 *CodedCharSetId* 필드를 참조하십시오. 애플리케이션이 MQGET 호출에 대해 특수 값 MQCCSI\_Q\_MGR을 지정한 경우에는 큐 관리자가 엑시트를 호출하기 전에 큐 관리자에 의해 사용되는 문자 세트의 실제 문자 세트 ID로 변경합니다.

변환에 성공하는 경우 엑시트가 이를 메시지 디스크립터 내의 *CodedCharSetId* 필드에 복사해야 합니다.

엑시트의 입력 필드입니다.

### CompCode

유형: MQLONG

엑시트가 호출되면 여기에 MQGET 호출을 발행한 애플리케이션에 대해 리턴되는 완료 코드가 포함됩니다. 단, 엑시트가 아무 것도 수행하지 않는 경우에만 적용됩니다. 메시지가 잘리거나 메시지에 변환이 필요하나 아직 변환이 수행되지 않은 경우이므로 항상 MQCC\_WARNING입니다.

엑시트에서 출력 시 이 필드는 MQGET 호출의 **CompCode** 매개변수 내의 애플리케이션에 리턴되는 완료 코드를 포함합니다. MQCC\_OK 및 MQCC\_WARNING만 유효합니다. 엑시트가 출력에서 이 필드를 설정할 수 있는 방법에 대한 설명은 *Reason* 필드의 설명을 참조하십시오.

이는 엑시트의 입출력(I/O) 필드입니다.

### DataLength

유형: MQLONG

엑시트가 호출되면 이 필드에는 애플리케이션 메시지 데이터의 원본 길이가 포함됩니다. 애플리케이션에서 제공하는 버퍼에 맞게 메시지가 잘리는 경우에는 엑시트에 제공되는 메시지의 크기가 *DataLength*의 값보다 작습니다. 엑시트에 제공되는 메시지의 크기는 자르기가 발생하는지 여부와 관계없이 항상 엑시트의 **InBufferLength** 매개변수에 의해 지정됩니다.

자르기는 엑시트에 대한 입력에서 MQRC\_TRUNCATED\_MSG\_ACCEPTED 값을 가지는 *Reason* 필드에 의해 표시됩니다.

대부분의 변환에서는 이러한 길이 변경이 필요하지 않으나 필요한 경우에는 엑시트가 자르기를 수행할 수 있습니다. 엑시트에 의해 설정되는 값은 MQGET 호출의 **DataLength** 매개변수에서 애플리케이션에 리턴됩니다. 그러나 변환 중인 메시지가 논리 메시지의 일부만을 포함하는 세그먼트일 경우 이 길이를 변경할 수 없습니다. 그 이유는 길이를 변경하면 논리 메시지에서 이후 세그먼트의 오프셋이 올바르게 맞지 않기 때문입니다.

엑시트가 데이터의 길이를 변경하려는 경우에는 큐 관리자가 이미 변환되지 않은 데이터의 길이를 기반으로 하여 메시지 데이터가 애플리케이션의 버퍼에 맞는지 결정한 후입니다. 이 결정에 따라 메시지가 큐에서 제거되는지(또는 찾아보기 요청에 대해 찾아보기 커서가 이동되는지), 변환으로 발생한 데이터 길이의 변경에 영향을 받지 않는지 판별합니다. 이런 이유 때문에 변환 엑시트가 애플리케이션 메시지 데이터의 길이를 변경하지 않도록 권장합니다.

문자 변환이 길이 변경을 암시하는 경우, 필요에 따라 후미 공백을 자르거나 공백으로 채워서 문자열이 동일한 길이(바이트)의 다른 문자열로 변환될 수 있습니다.

메시지에 애플리케이션 메시지 데이터가 포함되지 않으면 엑시트가 호출되지 않습니다. *DataLength*가 항상 0보다 크기 때문입니다.



이는 엑시트의 입출력(I/O) 필드입니다.

### Encoding

유형: MQLONG

애플리케이션에 필요한 숫자 인코딩.

MQGET 호출을 발행하는 애플리케이션에 필요한 숫자 인코딩입니다. 자세한 정보는 MQMD 구조의 *Encoding* 필드를 참조하십시오.

변환이 성공적이면 엑시트가 이를 메시지 디스크립터 내의 *Encoding* 필드에 복사합니다.

엑시트의 입력 필드입니다.

### ExitOptions

유형: MQLONG

이 필드는 예약된 필드이며 해당 값은 0입니다.

### ExitResponse

유형: MQLONG

엑시트로부터의 응답. 성공을 표시하기 위해 엑시트에 의해 설정되고, 그 밖의 경우에는 변환의 설정입니다. 다음 중 하나여야 합니다.

#### MQXDR\_OK

변환에 성공했습니다.

엑시트가 이 값을 지정하는 경우에는 큐 관리자가 MQGET 호출을 발행한 애플리케이션에 다음을 리턴합니다.

- 엑시트로부터의 출력에 대한 *CompCode* 필드의 값
- 엑시트로부터의 출력에 대한 *Reason* 필드의 값
- 엑시트로부터의 출력에 대한 *DataLength* 필드의 값
- 엑시트의 출력 버퍼 *OutBuffer*의 콘텐츠. 리턴되는 바이트의 수가 엑시트의 **OutBufferLength** 매개변수 및 엑시트로부터의 출력에 대한 *DataLength* 필드의 값 미만입니다.

엑시트의 메시지 디스크립터 매개변수 내의 *Encoding* 및 *CodedCharSetId* 필드가 모두 변경되지 않으면 큐 관리자가 다음을 리턴합니다.

- 엑시트에 대한 입력에 관한 MQDXP 구조 내의 *Encoding* 및 *CodedCharSetId* 필드의 값.

엑시트의 메시지 디스크립터 매개변수 내의 *Encoding* 및 *CodedCharSetId* 필드 중 하나 또는 둘 다가 변경된 경우 큐 관리자가 다음을 리턴합니다.

- 엑시트로부터의 출력에 대한 엑시트의 메시지 디스크립터 매개변수 내의 *Encoding* 및 *CodedCharSetId* 필드의 값

#### MQXDR\_CONVERSION\_FAILED

변환에 실패했습니다.

엑시트가 이 값을 지정하는 경우에는 큐 관리자가 MQGET 호출을 발행한 애플리케이션에 다음을 리턴합니다.

- 엑시트로부터의 출력에 대한 *CompCode* 필드의 값
- 엑시트로부터의 출력에 대한 *Reason* 필드의 값
- 엑시트에 대한 입력의 *DataLength* 필드의 값
- 엑시트의 입력 버퍼 *InBuffer*의 콘텐츠. 리턴되는 바이트 수는 **InBufferLength** 매개변수에 의해 지정됩니다.

엑시트가 *InBuffer*를 대체한 경우 결과가 정의되지 않습니다.

*ExitResponse*는 엑시트의 출력 필드입니다.

## Hconn

유형: MQHCONN

MQXCNCV 호출에서 사용될 수 있는 연결 핸들입니다. 이 핸들이 MQGET 호출을 발행한 애플리케이션이 지정한 핸들과 반드시 동일할 필요는 없습니다.

## pEntryPoints

유형: PMQIEP

MQI 및 DCI 호출을 작성할 수 있는 MQIEP 구조의 주소입니다.

## Reason

유형: MQLONG

CompCode을(를) 규정하는 이유 코드.

엑시트가 호출되면 여기에 MQGET 호출을 발행한 애플리케이션에 대해 리턴되는 이유 코드가 포함됩니다. 단, 엑시트가 아무 것도 수행하지 않는 경우에만 적용됩니다. 가능한 값 중에서 MQRC\_TRUNCATED\_MSG\_ACCEPTED는 애플리케이션이 제공한 버퍼에 맞게 메시지가 잘렸음을 나타내며 MQRC\_NOT\_CONVERTED는 메시지에 변환이 필요하나 아직 변환되지 않았음을 나타냅니다.

엑시트로부터의 출력에서 이 필드는 MQGET 호출의 Reason 매개변수 내의 애플리케이션에 리턴되는 이유를 포함합니다. 권장사항은 다음과 같습니다.

- Reason에 엑시트에 대한 입력의 MQRC\_TRUNCATED\_MSG\_ACCEPTED 값이 있는 경우 변환의 성공 또는 실패와 관계없이 Reason 및 CompCode 필드를 반드시 대체할 필요는 없습니다.

(CompCode 필드가 MQCC\_OK가 아닌 경우 메시지를 검색하는 애플리케이션이 요청된 값이 있는 메시지 디스크립터 내의 리턴되는 Encoding 및 CodedCharSetId 값을 비교하여 변환 실패를 식별할 수 있습니다. 대조적으로 애플리케이션은 버퍼에 맞는 메시지에서 잘린 메시지를 구별하지 못합니다. 이런 이유로 인해 변환 실패를 표시하는 모든 이유보다 우선적으로 MQRC\_TRUNCATED\_MSG\_ACCEPTED가 리턴되어야 합니다.)

- Reason에 엑시트에 대한 입력의 기타 값이 있는 경우

- 변환에 성공하면 CompCode가 MQCC\_OK로 설정되고 Reason이 MQRC\_NONE으로 설정되어야 합니다.

- 변환에 실패하거나 메시지가 확장되어 버퍼에 맞도록 잘라야 하는 경우 CompCode가 MQCC\_WARNING으로 설정되거나 변경되지 않은 상태로 남아있어야 하며 Reason이 나열된 값 중 하나로 설정되어 실패의 네이처를 표시해야 합니다.

변환 이후에 메시지가 버퍼에 비해 너무 큰 경우 애플리케이션이 MQGMO\_ACCEPT\_TRUNCATED\_MSG 옵션을 지정한 MQGET 호출을 발행한 경우에만 메시지를 잘라야 합니다.

- 해당 옵션을 지정한 경우에는 MQRC\_TRUNCATED\_MSG\_ACCEPTED 이유가 리턴됩니다.
- 해당 옵션을 지정하지 않은 경우에는 이유 코드 MQRC\_CONVERTED\_MSG\_TOO\_BIG과 함께 메시지가 변환되지 않고 리턴됩니다.

나열된 이유 코드는 변환이 실패한 이유를 표시하기 위해 엑시트에 의해 사용되도록 권장됩니다. 그러나 엑시트는 적절하다고 판단되면 MQRC\_\* 코드 세트에서 다른 값을 리턴할 수 있습니다. 또한 엑시트가 MQGET 호출을 발행하여 애플리케이션과 통신하기를 원하는 조건을 표시하기 위해 MQRC\_APPL\_FIRST에서 MQRC\_APPL\_LAST 범위 내의 값이 엑시트에 의해 사용되도록 할당됩니다.

**참고:** 메시지를 성공적으로 변환하지 못한 경우, 큐 관리자가 변환되지 않은 메시지를 리턴하도록 하기 위해 엑시트가 ExitResponse 필드에서 MQXDR\_CONVERSION\_FAILED를 리턴해야 합니다. 이는 Reason 필드에서 리턴되는 이유 코드와 관계없이 true입니다.

## MQRC\_APPL\_FIRST

(900, X'384') 애플리케이션 정의 이유 코드에 대한 최저값입니다.

## MQRC\_APPL\_LAST

(999, X'3E7') 애플리케이션 정의 이유 코드에 대한 최고값입니다.

## MQRC\_CONVERTED\_MSG\_TOO\_BIG

(2120, X'848') 변환된 데이터가 버퍼에 비해 너무 큼니다.

**MQRC\_NOT\_CONVERTED**

(2119, X'847') 메시지 데이터가 변환되지 않았습니다.

**MQRC\_SOURCE\_CCSDID\_ERROR**

(2111, X'83F') 소스 코드화 문자 세트 ID가 올바르지 않습니다.

**MQRC\_SOURCE\_DECIMAL\_ENC\_ERROR**

(2113, X'841') 메시지의 팩형 10진수 인코딩이 인식되지 않습니다.

**MQRC\_SOURCE\_FLOAT\_ENC\_ERROR**

(2114, X'842') 메시지의 부동 소수점 인코딩이 인식되지 않습니다.

**MQRC\_SOURCE\_INTEGER\_ENC\_ERROR**

(2112, X'840') 소스 정수 인코딩이 인식되지 않습니다.

**MQRC\_TARGET\_CCSDID\_ERROR**

(2115, X'843') 대상 코드화 문자 세트 ID가 올바르지 않습니다.

**MQRC\_TARGET\_DECIMAL\_ENC\_ERROR**

(2117, X'845') 수신자가 지정한 팩형 10진수 인코딩을 인식할 수 없습니다.

**MQRC\_TARGET\_FLOAT\_ENC\_ERROR**

(2118, X'846') 수신자에 의해 지정되는 부동 소수점 인코딩이 인식되지 않습니다.

**MQRC\_TARGET\_INTEGER\_ENC\_ERROR**

(2116, X'844') 대상 정수 인코딩이 인식되지 않습니다.

**MQRC\_TRUNCATED\_MSG\_ACCEPTED**

(2079, X'81F') 잘린 메시지가 리턴되었습니다(처리 완료됨).

이는 엑시트의 입출력(I/O) 필드입니다.

**StrucId**

유형: MQCHAR4

구조 ID입니다.값은 다음과 같아야 합니다.

**MQDXP\_STRUC\_ID**

데이터 변환 엑시트 매개변수 구조의 ID.

C 프로그래밍 언어의 경우 상수 MQDXP\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQDXP\_STRUC\_ID와 동일한 값을 갖지만 문자열이 아닌 문자의 배열입니다.

엑시트의 입력 필드입니다.

**Version**

유형: MQLONG

구조 버전 번호입니다.값은 다음과 같아야 합니다.

**MQDXP\_VERSION\_1**

데이터 변환 엑시트 매개변수 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

**MQDXP\_CURRENT\_VERSION**

데이터 변환 엑시트 매개변수 구조의 현재 버전.

**참고:** 이 구조의 새 버전을 도입해도 기존 부분의 레이아웃은 변경되지 않습니다. 따라서 엑시트는 *Version* 필드가 엑시트가 사용해야 할 필드를 포함하는 최저 버전 이상인지 점검해야 합니다.

엑시트의 입력 필드입니다.

**C 선언**

```
typedef struct tagMQDXP MQDXP;
struct tagMQDXP {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    ExitOptions;      /* Reserved */
    MQLONG    AppOptions;       /* Application options */
}
```

```

MQLONG  Encoding;          /* Numeric encoding required by
                             application */
MQLONG  CodedCharSetId;   /* Character set required by application */
MQLONG  DataLength;       /* Length in bytes of message data */
MQLONG  CompCode;         /* Completion code */
MQLONG  Reason;           /* Reason code qualifying CompCode */
MQLONG  ExitResponse;     /* Response from exit */
MQHCONN Hconn;           /* Connection handle */
PMQIEP  pEntryPoints;     /* Address of the MQIEP structure */
};

```

## COBOL declaration ( IBM i only)

```

**  MQDXP structure
10  MQDXP.
**  Structure identifier
15  MQDXP-STRUCID      PIC X(4).
**  Structure version number
15  MQDXP-VERSION     PIC S9(9) BINARY.
**  Reserved
15  MQDXP-EXITOPTIONS PIC S9(9) BINARY.
**  Application options
15  MQDXP-APPOPTIONS  PIC S9(9) BINARY.
**  Numeric encoding required by application
15  MQDXP-ENCODING    PIC S9(9) BINARY.
**  Character set required by application
15  MQDXP-CODEDCHARSETID PIC S9(9) BINARY.
**  Length in bytes of message data
15  MQDXP-DATALength  PIC S9(9) BINARY.
**  Completion code
15  MQDXP-COMPCODE    PIC S9(9) BINARY.
**  Reason code qualifying COMPCODE
15  MQDXP-REASON      PIC S9(9) BINARY.
**  Response from exit
15  MQDXP-EXITRESPONSE PIC S9(9) BINARY.
**  Connection handle
15  MQDXP-HCONN       PIC S9(9) BINARY.

```

## System/390 어셈블러 선언

```

MQDXP          DSECT
MQDXP_STRUCID  DS   CL4  Structure identifier
MQDXP_VERSION  DS   F    Structure version number
MQDXP_EXITOPTIONS DS  F    Reserved
MQDXP_APPOPTIONS DS  F    Application options
MQDXP_ENCODING DS  F    Numeric encoding required by application
MQDXP_CODEDCHARSETID DS  F    Character set required by application
MQDXP_DATALength DS  F    Length in bytes of message data
MQDXP_COMPCODE DS  F    Completion code
MQDXP_REASON   DS  F    Reason code qualifying COMPCODE
MQDXP_EXITRESPONSE DS  F    Response from exit
MQDXP_HCONN    DS  F    Connection handle
*
MQDXP_LENGTH   EQU  *-MQDXP
               ORG  MQDXP
MQDXP_AREA     DS   CL(MQDXP_LENGTH)

```

## MQXCNCV - 문자 변환

MQXCNCV 호출은 C 프로그래밍 언어를 사용하여 하나의 문자 세트에서 다른 문자 세트로 문자를 변환합니다. 이 호출은 IBM MQ 프레임워크 인터페이스 중 하나인 IBM MQ 데이터 변환 인터페이스(DCI)의 일부입니다. 참고: 호출은 애플리케이션 및 데이터 변환 엑시트 환경 모두에서 사용될 수 있습니다.

## 구문

MQXCNCV(*Hconn*, *Options*, *SourceCCSID*, *SourceLength*, *SourceBuffer*, *TargetCCSID*, *TargetLength*, *TargetBuffer*, *DataLength*, *CompCode*, *Reason*)


## 매개변수

### Hconn

유형: MQHCONN - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다.

데이터 변환 엑시트에서 Hconn은 일반적으로 MQDXP 구조의 Hconn 필드에서 데이터 변환 엑시트에 전달되는 핸들입니다. 이 핸들은 MQGET 호출을 발행한 애플리케이션에서 지정한 핸들과 반드시 동일하지는 않습니다.

 IBM i에서 Hconn에 대해 다음 특수 값을 지정할 수 있습니다.

### MQHC\_DEF\_HCONN

기본 연결 핸들

CICS TS 3.2 이상의 애플리케이션을 실행하는 경우 MQXCNCV 호출을 호출한 문자 변환 엑시트 프로그램이 OPENAPI로 정의되어야 합니다. 이 정의로 올바르지 않은 연결에서 야기되는 2018 MQRC\_HCONN\_ERROR 오류를 방지하고 MQGET를 완료할 수 있습니다.

### 옵션

유형: MQLONG - 입력

MQXCNCV의 조치를 제어하는 옵션입니다.

설명된 0개 이상의 옵션을 지정할 수 있습니다. 둘 이상의 옵션을 지정하려면 값을 함께 추가하거나(동일한 상수를 두 번 이상 추가하지 않음), 비트 단위 OR 연산을 사용하여 값을 결합하십시오(프로그래밍 언어가 비트 연산을 지원하는 경우).

**기본 변환 옵션:** 다음 옵션은 기본 문자 변환의 사용을 제어합니다.

### MQDCC\_DEFAULT\_CONVERSION

기본 변환.

이 옵션은 호출에 지정된 문자 세트 중 하나 또는 둘 다 지원되지 않는 경우 기본 문자 변환을 사용할 수 있도록 지정합니다. 이로 인해 기본값 변환은 문자열 변환 시 지정된 문자 세트를 대략적으로 나타내는 설치 지정 기본값 문자 세트를 큐 관리자가 사용할 수 있습니다.

**참고:** 대략적인 문자 세트를 사용하여 문자열을 변환하면 일부 문자가 제대로 변환되지 않는 결과가 발생할 수 있습니다. 지정된 문자 세트 및 기본 문자 세트 모두에 공통된 문자열의 문자를 사용하여 이를 피할 수 있습니다.

큐 관리자가 설치되거나 재시작될 때 기본 문자 세트는 구성 옵션에 의해 정의됩니다.

MQDCC\_DEFAULT\_CONVERSION이 지정되지 않은 경우 큐 관리자는 지정된 문자 세트만 사용하여 문자열을 변환하고 하나 또는 두 문자 세트 모두가 지원되지 않으면 호출에 실패합니다.

이 옵션은 다음 환경에서 지원됩니다.

-  AIX
-  IBM i
-  Linux
-  Windows

**패딩 옵션:** 다음 옵션을 사용하면 변환된 문자열에 대상 버퍼에 맞도록 큐 관리자가 변환된 문자열을 공백으로 채우거나 중요하지 않은 후미 문자를 제거할 수 있습니다.

### MQDCC\_FILL\_TARGET\_BUFFER

대상 버퍼를 채웁니다.

이 옵션은 대상 버퍼가 완전히 채워지는 식으로 변환이 발생하도록 요청합니다.

- 변환될 때 문자열이 축소되는 경우 대상 버퍼를 채우기 위해 후미 문자 공백이 추가됩니다.
- 변환 시 문자열이 확장되면 변환된 문자열이 대상 버퍼에 맞도록 중요하지 않은 후미 문자가 제거됩니다. 이를 완료할 수 있는 경우 호출은 MQCC\_OK 및 이유 코드 MQRC\_NONE으로 완료합니다.

중요하지 않은 후미 문자가 너무 적은 경우 적합한 문자열 만큼 대상 버퍼에 배치되고 호출은 MQCC\_WARNING 및 이유 코드 MQRC\_CONVERTED\_MSG\_TOO\_BIG으로 완료합니다.

중요하지 않은 문자는 다음과 같습니다.

- 후미 문자 공백
- 문자열에서 첫 번째 널 문자 뒤에 오는 문자(첫 번째 널 문자 자체 제외)
- 이는 문자열 TargetCCSID 및 TargetLength가 대상 버퍼를 올바른 문자로 완전히 설정할 수 없는 경우 MQCC\_FAILED 및 이유 코드 MQRC\_TARGET\_LENGTH\_ERROR로 호출에 실패합니다. 이는 TargetCCSID 가 순수 DBCS 문자 세트(예: UTF-16) 이지만 TargetLength 가 홀수 바이트 길이를 지정하는 경우에 발생할 수 있습니다.
- TargetLength 은 SourceLength보다 작거나 클 수 있습니다. MQXCNCVC에서 리턴 시 DataLength에 TargetLength와 동일한 값이 있습니다.

이 옵션을 지정하지 않은 경우:

- 필요에 따라 대상 버퍼 내에서 문자열을 축소하거나 확장할 수 있습니다. 중요하지 않은 후미 문자는 추가되거나 제거되지 않습니다.

변환된 문자열이 대상 버퍼에 적합한 경우 호출은 MQCC\_OK 및 이유 코드 MQRC\_NONE으로 완료합니다.

변환된 문자열이 대상 버퍼에 너무 큰 경우 적합한 문자열 만큼 대상 버퍼에 배치되고 호출은 MQCC\_WARNING 및 이유 코드 MQRC\_CONVERTED\_MSG\_TOO\_BIG으로 완료합니다. 이 경우 TargetLength보다 작은 바이트는 리턴될 수 있습니다.

- TargetLength 은 SourceLength보다 작거나 클 수 있습니다. MQXCNCVC에서 리턴 시 DataLength는 TargetLength보다 작거나 같습니다.

이 옵션은 다음 환경에서 지원됩니다.

-  AIX
-  IBM i
-  Linux
-  Windows

**인코딩 옵션:** 설명된 옵션은 소스 및 대상 문자열의 정수 인코딩을 지정하는 데 사용할 수 있습니다. 해당 문자 세트 ID가 기본 스토리지의 문자 세트 표현이 2진 정수에 사용된 인코딩에 종속됨을 나타내는 경우에만 관련 인코딩이 사용됩니다. 이는 특정 멀티바이트 문자 세트(예: UTF-16 문자 세트)에만 영향을 미칩니다.

문자 세트가 단일 바이트 문자 세트(SBCS) 또는 정수 인코딩에 종속되지 않은 주 기억장치의 표현이 포함된 멀티바이트 문자 세트인 경우 인코딩은 무시됩니다.

MQDCC\_SOURCE\_\* 값 중 하나만 지정되며 다음의 MQDCC\_TARGET\_\* 값 중 하나와 결합되어야 합니다.

#### **MQDCC\_SOURCE\_ENC\_NATIVE**

소스 인코딩은 환경 및 프로그래밍 언어의 기본값입니다.

#### **MQDCC\_SOURCE\_ENC\_NORMAL**

소스 인코딩은 정상입니다.

#### **MQDCC\_SOURCE\_ENC\_REVERSED**

소스 인코딩은 역순됩니다.

#### **MQDCC\_SOURCE\_ENC\_UNDEFINED**

소스 인코딩은 정의되지 않습니다.

#### **MQDCC\_TARGET\_ENC\_NATIVE**

대상 인코딩은 환경 및 프로그래밍 언어의 기본값입니다.

#### **MQDCC\_TARGET\_ENC\_NORMAL**

대상 인코딩은 정상입니다.

## **MQDCC\_TARGET\_ENC\_REVERSED**

대상 인코딩은 역순됩니다.

## **MQDCC\_TARGET\_ENC\_UNDEFINED**

대상 인코딩은 정의되지 않습니다.

이전에 정의된 인코딩 값은 Options 필드에 직접적으로 추가될 수 있습니다. 그러나 소스 또는 대상 인코딩을 MQMD 또는 다른 구조의 Encoding 필드에서 가져오는 경우 다음 처리는 수행되어야 합니다.

1. 정수 인코딩은 float 및 팩형 10진수 인코딩을 제거하여 Encoding 필드에서 추출해야 합니다. 이를 수행하는 방법의 세부사항은 864 페이지의 『인코딩 분석』의 내용을 참조하십시오.
2. 1단계의 결과인 정수 인코딩은 Options 필드에 추가되기 전에 적절한 요소로 곱해야 합니다. 요인은 다음과 같습니다.
  - 소스 인코딩에 대한 MQDCC\_SOURCE\_ENC\_FACTOR
  - 대상 인코딩에 대한 MQDCC\_TARGET\_ENC\_FACTOR

다음 예제 코드는 이를 C 프로그래밍 언어로 코드화할 수 있는 방법을 설명합니다.

```
Options = (MsgDesc.Encoding & MQENC_INTEGER_MASK)
          * MQDCC_SOURCE_ENC_FACTOR
          + (DataConvExitParms.Encoding & MQENC_INTEGER_MASK)
          * MQDCC_TARGET_ENC_FACTOR;
```

지정되지 않은 경우 인코딩 옵션은 정의되지 않음으로 기본 설정됩니다(MQDCC\_\*\_ENC\_UNDEFINED). 대부분의 경우, 이는 MQXCNCV 호출의 성공적인 완료에 영향을 주지 않습니다. 그러나 해당 문자 세트가 인코딩에 종속된 표현을 포함한 멀티바이트 문자 세트인 경우(예: UTF-16 문자 세트), 호출에 실패하고 이유 코드 MQRC\_SOURCE\_INTEGER\_ENC\_ERROR 또는 MQRC\_TARGET\_INTEGER\_ENC\_ERROR가 표시됩니다(적절한 경우).

인코딩 옵션은 다음 환경에서 지원됩니다.

-  AIX
-  IBM i
-  Linux
-  Windows
-  z/OS

기본 옵션: 이전에 설명한 옵션을 지정하지 않은 경우, 다음 옵션을 사용할 수 있습니다.

## **MQDCC\_NONE**

옵션이 지정되지 않았습니다.

MQDCC\_NONE은 프로그램 문서화를 지원하기 위해 정의됩니다. 이 옵션은 다른 수와 함께 사용하기 위한 용도는 아니지만 값이 0이므로 그러한 사용을 감지할 수 없습니다.

## **SourceCCSID**

유형: MQLONG - 입력

이는 SourceBuffer에서 입력 문자열의 코드화된 문자 세트 ID입니다.

## **SourceLength**

유형: MQLONG - 입력

이는 SourceBuffer의 입력 문자열의 길이(바이트)이며 0이상이어야 합니다.

## **SourceBuffer**

유형: MQCHAR x SourceLength - 입력

문자 간에 변환되는 문자열을 포함한 버퍼입니다.

**TargetCCSID**

유형: MQLONG - 입력

이는 SourceBuffer가 변환될 문자 세트의 코드화된 문자 세트 ID입니다.

**TargetLength**

유형: MQLONG - 입력

이는 TargetBuffer의 출력 버퍼의 길이(바이트)이며 0이상이어야 합니다. SourceLength보다 작거나 클 수 있습니다.

**TargetBuffer**

유형: MQCHAR x TargetLength - 출력

이는 TargetCCSID에 의해 정의된 문자 세트로 변환된 후의 문자열입니다. 변환된 문자열은 변환되지 않은 문자열보다 짧거나 길 수 있습니다. **DataLength** 매개변수는 리턴된 올바른 바이트의 수를 표시합니다.

**DataLength**

유형: MQLONG - 출력

이는 출력 버퍼 TargetBuffer에서 리턴된 문자열의 길이입니다. 변환된 문자열은 변환되지 않은 문자열보다 짧거나 길 수 있습니다.

**CompCode**

유형: MQLONG - 출력

다음 중 하나입니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_WARNING**

경고(일부 완료).

**MQCC\_FAILED**

호출에 실패했습니다.

**원인**

유형: MQLONG - 출력

CompCode을(를) 규정하는 이유 코드.

CompCode이(가) MQCC\_OK인 경우:

**MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode이(가) MQCC\_WARNING인 경우:

**MQRC\_CONVERTED\_MSG\_TOO\_BIG**

(2120, X'848') 변환된 데이터가 버퍼에 비해 너무 큼니다.

CompCode이(가) MQCC\_FAILED인 경우:

**MQRC\_DATA\_LENGTH\_ERROR**

(2010, X'7DA') 데이터 길이 매개변수가 올바르지 않습니다.

**MQRC\_DBCS\_ERROR**

(2150, X'866') DBCS 문자열이 올바르지 않습니다.

**MQRC\_HCONN\_ERROR**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**MQRC\_OPTIONS\_ERROR**

(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**MQRC\_SOURCE\_BUFFER\_ERROR**

(2145, X'861') 소스 버퍼 매개변수가 올바르지 않습니다.



**MQRC\_SOURCE\_CCSDID\_ERROR**

(2111, X'83F') 소스 코드화 문자 세트 ID가 올바르지 않습니다.

**MQRC\_SOURCE\_INTEGER\_ENC\_ERROR**

(2112, X'840') 소스 정수 인코딩이 인식되지 않습니다.

**MQRC\_SOURCE\_LENGTH\_ERROR**

(2143, X'85F') 소스 길이 매개변수가 올바르지 않습니다.

**MQRC\_STORAGE\_NOT\_AVAILABLE**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**MQRC\_TARGET\_BUFFER\_ERROR**

(2146, X'862') 대상 버퍼 매개변수가 올바르지 않습니다.

**MQRC\_TARGET\_CCSDID\_ERROR**

(2115, X'843') 대상 코드화 문자 세트 ID가 올바르지 않습니다.

**MQRC\_TARGET\_INTEGER\_ENC\_ERROR**

(2116, X'844') 대상 정수 인코딩이 인식되지 않습니다.

**MQRC\_TARGET\_LENGTH\_ERROR**

(2144, X'860') 대상 길이 매개변수가 올바르지 않습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

이 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

## C 호출

```
MQXCNCV (Hconn, Options, SourceCCSID, SourceLength, SourceBuffer,
        TargetCCSID, TargetLength, TargetBuffer, &DataLength,
        &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN  Hconn;           /* Connection handle */
MQLONG   Options;        /* Options that control the action of
MQXCNCV */
MQLONG   SourceCCSID;    /* Coded character set identifier of string
before conversion */
MQLONG   SourceLength;   /* Length of string before conversion */
MQCHAR   SourceBuffer[n]; /* String to be converted */
MQLONG   TargetCCSID;    /* Coded character set identifier of string
after conversion */
MQLONG   TargetLength;   /* Length of output buffer */
MQCHAR   TargetBuffer[n]; /* String after conversion */
MQLONG   DataLength;     /* Length of output string */
MQLONG   CompCode;       /* Completion code */
MQLONG   Reason;         /* Reason code qualifying CompCode */
```

## COBOL declaration ( IBM i only)

IBM i

```
CALL 'MQXCNCV' USING HCONN, OPTIONS, SOURCECCSID, SOURCELENGTH,
                    SOURCEBUFFER, TARGETCCSID, TARGETLENGTH,
                    TARGETBUFFER, DATALENGTH, COMPCODE, REASON.
```

매개변수를 다음과 같이 선언합니다.

```
** Connection handle
01 HCONN      PIC S9(9) BINARY.
** Options that control the action of MQXCNCV
01 OPTIONS    PIC S9(9) BINARY.
** Coded character set identifier of string before conversion
01 SOURCECCSID PIC S9(9) BINARY.
```

```

** Length of string before conversion
01 SOURCELENGTH PIC S9(9) BINARY.
** String to be converted
01 SOURCEBUFFER PIC X(n).
** Coded character set identifier of string after conversion
01 TARGETCCSID PIC S9(9) BINARY.
** Length of output buffer
01 TARGETLENGTH PIC S9(9) BINARY.
** String after conversion
01 TARGETBUFFER PIC X(n).
** Length of output string
01 DATALENGTH PIC S9(9) BINARY.
** Completion code
01 COMPCODE PIC S9(9) BINARY.
** Reason code qualifying COMPCODE
01 REASON PIC S9(9) BINARY.

```

## S/390 assembler declaration

```

CALL MQXCNCV, (HCONN, OPTIONS, SOURCECCSID, SOURCELENGTH,          X
              SOURCEBUFFER, TARGETCCSID, TARGETLENGTH, TARGETBUFFER, X
              DATALENGTH, COMPCODE, REASON)

```

매개변수를 다음과 같이 선언합니다.

```

HCONN          DS F      Connection handle
OPTIONS        DS F      Options that control the action of MQXCNCV
SOURCECCSID    DS F      Coded character set identifier of string before
*               conversion
SOURCELENGTH   DS F      Length of string before conversion
SOURCEBUFFER   DS CL(n)  String to be converted
TARGETCCSID    DS F      Coded character set identifier of string after
*               conversion
TARGETLENGTH   DS F      Length of output buffer
TARGETBUFFER   DS CL(n)  String after conversion
DATALENGTH     DS F      Length of output string
COMPCODE       DS F      Completion code
REASON         DS F      Reason code qualifying COMPCODE

```

## MQ\_DATA\_CONV\_EXIT - 데이터 변환 엑시트

MQ\_DATA\_CONV\_EXIT 호출은 데이터 변환 엑시트에 전달되는 매개변수를 설명합니다.

큐 관리자가 제공한 MQ\_DATA\_CONV\_EXIT라는 시작점이 없습니다(사용법 참고사항 [11](#) 참조).

이 정의는 IBM MQ 프레임워크 인터페이스 중 하나인 IBM MQ 데이터 변환 인터페이스(DCI)의 일부입니다.

### 구문

MQ\_DATA\_CONV\_EXIT(*DataConvExitParms*, *MsgDesc*, *InBufferLength*, *InBuffer*, *OutBufferLength*, *OutBuffer*)

### 매개변수

#### DataConvExitParms

유형: MQDXP - 입력/출력

이 구조는 엑시트 호출과 관련된 정보를 포함합니다. 엑시트는 이 구조에 정보를 설정하여 변환 출력을 표시합니다. 이 구조의 필드에 대한 자세한 정보는 [875 페이지](#)의 『MQDXP - 데이터 변환 엑시트 매개변수』의 내용을 참조하십시오.

#### MsgDesc

유형: MQMD - 입출력(I/O)

엑시트에 대한 입력에서 이는 **InBuffer** 매개변수의 엑시트로 전달된 메시지 데이터와 연관된 메시지 디스크립터입니다.

**참고:** 엑시트에 전달된 **MsgDesc** 매개변수는 항상 엑시트를 호출한 큐 관리자에 지원되는 MQMD의 최신 버전입니다. 엑시트가 다른 환경 사이에서 휴대가 가능한 경우 엑시트는 MsgDesc의 Version 필드를 확인하여 엑시트가 액세스해야 하는 필드가 구조에 있는지 확인합니다.

다음 환경에서 엑시트는 version-2 MQMD에 전달됩니다.

-  AIX
-  IBM i
-  Linux
-  Windows

데이터 변환 엑시트를 지원하는 다른 모든 환경에서 엑시트는 버전 1 MQMD에 전달됩니다.

출력 시 변환이 성공적인 경우 엑시트는 애플리케이션에서 요청한 값으로 Encoding 및 CodedCharSetId 필드를 변경합니다. 이러한 변경사항은 애플리케이션에 다시 반영됩니다. 엑시트로 인한 구조의 변경사항은 무시됩니다. 애플리케이션에 반영되지 않습니다.

엑시트가 MQDXP 구조의 ExitResponse 필드에 있는 MQXDR\_OK를 리턴하지만 메시지 디스크립터의 Encoding 또는 CodedCharSetId 필드를 변경하지 않은 경우 큐 관리자는 MQDXP 구조의 해당 필드에 엑시트에 대한 입력이 있는 값을 해당 필드에 리턴합니다.

### **InBufferLength**

유형: MQLONG - 입력

InBuffer의 바이트 길이.

이는 입력 버퍼 InBuffer의 길이이고 엑시트에 의해 처리될 바이트 수를 지정합니다. InBufferLength는 변환 전 메시지 데이터의 길이 및 MQGET 호출에서 애플리케이션이 제공하는 버퍼의 길이 중 작은 값입니다.

값은 항상 0보다 큽니다.

### **InBuffer**

유형: MQBYTEXInBufferLength - 입력

변환되지 않은 메시지가 있는 버퍼.

여기에는 변환 전 메시지 데이터가 포함됩니다. 엑시트가 데이터를 변환할 수 없으면 큐 관리자는 엑시트가 완료된 이후 이 버퍼의 콘텐츠를 애플리케이션으로 리턴합니다.

**참고:** 엑시트는 InBuffer를 변경해서는 안됩니다. 이 매개변수가 변경되면 결과가 정의되지 않습니다.

C 프로그래밍 언어에서 이 매개변수는 pointer-to-void로 정의됩니다.

### **OutBufferLength**

유형: MQLONG - 입력

OutBuffer의 바이트 길이.

이는 출력 버퍼 OutBuffer의 길이이고 MQGET 호출에서 애플리케이션에 의해 제공된 버퍼의 길이와 같습니다.

값은 항상 0보다 큽니다.

### **OutBuffer**

유형: MQBYTEXOutBufferLength - 출력

변환된 메시지가 있는 버퍼.

엑시트의 출력에서 변환이 성공한 경우 (**DataConvExitParms** 매개변수의 ExitResponse 필드에 MQXDR\_OK값으로 표시됨) OutBuffer는 요청된 표현으로 애플리케이션에 전달될 메시지 데이터를 포함합니다. 변환에 성공하지 못하면 엑시트가 이 버퍼에 수행한 변경사항이 무시됩니다.

C 프로그래밍 언어에서 이 매개변수는 pointer-to-void로 정의됩니다.

## 사용법 참고

1. 데이터 변환 엑시트는 MQGET 호출을 처리하는 동안 제어를 수신하는 사용자 작성 엑시트입니다. 데이터 변환 엑시트에 의해 수행되는 함수는 엑시트 제공자가 정의합니다. 그러나 엑시트는 여기에 그리고 연관된 매개변수 구조 MQDXP에 설명된 규칙을 준수해야 합니다.

데이터 변환 엑시트에 사용할 수 있는 프로그래밍 언어는 환경에 따라 판별됩니다.

2. 엑시트는 다음 명령문이 모두 true인 경우에만 호출됩니다.

- MQGMO\_CONVERT 옵션은 MQGET 호출에 지정됨
- 메시지 디스크립터의 Format 필드는 MQFMT\_NONE이 아님
- 필요한 표현에 메시지가 이미 없습니다. 즉, 메시지의 CodedCharSetId 및 Encoding 중 하나 또는 모두는 MQGET 호출에 제공된 메시지 디스크립터에서 애플리케이션에 의해 지정된 값과 다릅니다.
- 큐 관리자가 아직 변환을 성공적으로 완료하지 않았습니다.
- 애플리케이션의 버퍼 길이가 0보다 큼니다.
- 메시지 데이터의 길이는 0보다 큼니다.

3. 엑시트가 작성되고 있을 때 잘린 메시지를 변환시킬 수 있게 허용하는 방법으로 엑시트를 코드화하는 것을 고려하십시오. 잘린 메시지는 다음과 같이 발생할 수 있습니다.

- 수신 애플리케이션은 메시지보다 작은 버퍼를 제공하지만 MQGET 호출에 MQGMO\_ACCEPT\_TRUNCATED\_MSG 옵션을 지정합니다.

이 경우 엑시트에 대한 입력에서 **DataConvExitParms** 매개변수의 Reason 필드에 MQRC\_TRUNCATED\_MSG\_ACCEPTED값이 있습니다.

- 메시지 송신자는 송신 전에 잘립니다. 예를 들어, 이 동작은 보고 메시지에서 발생합니다(자세한 정보는 874 페이지의 『보고 메시지의 변환』 참조).

이 경우 엑시트에 대한 입력에서 **DataConvExitParms** 매개변수의 Reason 필드에는 MQRC\_NONE값이 있습니다(수신 애플리케이션이 메시지에 대해 충분히 큰 버퍼를 제공한 경우).

따라서 엑시트에 대한 입력의 Reason 필드 값은 항상 메시지가 잘렸는지 여부를 결정하는 데 사용될 수는 없습니다.

잘린 메시지를 구별하는 특징은 **InBufferLength** 매개변수의 엑시트에 제공된 길이가 메시지 디스크립터의 Format 필드에 포함된 형식 이름이 암시하는 길이보다 짧다는 것입니다. 따라서 엑시트는 데이터 변환을 시도하기 전에 **InBufferLength**의 값을 확인해야 합니다. 엑시트는 형식 이름이 암시하는 전체 데이터 양이 제공되었다고 가정하지 않아야 합니다.

엑시트가 잘린 메시지를 변환하기 위해 작성되지 않고 **InBufferLength**가 예상된 값 미만인 경우, 엑시트는 **CompCode** 및 **Reason** 필드를 MQCC\_WARNING과 MQRC\_FORMAT\_ERROR로 설정하여 **DataConvExitParms** 매개변수의 **ExitResponse** 필드에서 MQXDR\_CONVERSION\_FAILED를 리턴합니다.

엑시트가 잘린 메시지를 변환하도록 작성된 경우, 엑시트는 최대한 많은 데이터를 변환하며(다음 사용 시 참고사항 참조) **InBuffer**의 끝을 벗어나는 데이터를 조사하거나 변환하려고 하지 않도록 주의해야 합니다. 변환이 성공적으로 완료되면 엑시트는 **DataConvExitParms** 매개변수의 Reason 필드를 변경되지 않은 상태로 둡니다. 이는 수신자의 큐 관리자가 메시지를 자른 경우 MQRC\_TRUNCATED\_MSG\_ACCEPTED를 리턴하고 메시지 송신자가 메시지를 자른 경우 MQRC\_NONE을 리턴합니다.

또한 메시지는 변환 중에 **OutBuffer**보다 큰 지점까지 확장할 수도 있습니다. 이 경우 엑시트는 메시지를 자를지 여부를 결정해야 합니다. **DataConvExitParms** 매개변수의 **AppOptions** 필드는 수신 애플리케이션이 MQGMO\_ACCEPT\_TRUNCATED\_MSG 옵션을 지정했는지 여부를 표시합니다.

4. 일반적으로 **InBuffer**의 엑시트에 제공한 메시지의 모든 데이터가 변환되거나 또는 아무것도 변환되지 않습니다. 그러나 변환 전 또는 변환 중 메시지가 잘리면 예외입니다. 이 경우, 버퍼의 끝에 불완전한 항목이 있을 수 있습니다(예: 2바이트 문자의 1바이트 또는 4바이트 정수의 3바이트). 이 경우 불완전한 항목 생략 및 **OutBuffer**에서 사용되지 않은 바이트를 널로 설정하는 것을 고려하십시오. 그러나 배열 또는 문자열 내의 완료 요소나 문자는 변환되어야 합니다.


5. 엑시트가 처음 필요한 경우, 큐 관리자가 (확장과 별도로) 형식과 이름이 같은 오브젝트를 로드하려고 합니다. 로드된 오브젝트에는 해당 형식 이름의 메시지를 처리하는 엑시트가 있어야 합니다. 모든 환경에서 이를 필요로하지 않지만 엑시트 이름 및 엑시트에 포함되는 오브젝트의 이름을 동일시하는 것이 좋습니다.
6. 애플리케이션이 큐 관리자에 연결되어 있기 때문에 Format을 사용하는 첫 번째 메시지를 애플리케이션이 검색하려고 시도할 때 엑시트의 새 사본이 로드됩니다. CICS 또는 IMS 애플리케이션의 경우 이는 CICS 또는 IMS 서브시스템이 큐 관리자에 연결된 경우를 의미합니다. 큐 관리자가 이전에 로드된 사본을 제거한 경우 새로운 사본은 다른 시간에도 로드될 수 있습니다. 이러한 이유로 엑시트는 정적 스토리지를 사용하여 엑시트의 하나의 호출에서 다음 호출로 정보를 통신하려고 시도하면 안됩니다. 두 호출 간 엑시트가 로드 해제될 수 있습니다.
7. 큐 관리자에 지원되는 내장 형식 중 하나와 동일한 이름의 사용자 제공 엑시트가 있으면 사용자 제공 엑시트가 내장 변환 루틴을 대체하지 않습니다. 이러한 엑시트가 호출되는 유일한 환경은 다음과 같습니다.
  - 내장 변환 루틴이 포함된 CodedCharSetId 또는 Encoding에/에서 변환을 핸들링할 수 없음
  - 내장 변환 루틴이 데이터를 변환하지 못했습니다(예: 변환 불가능한 필드 또는 문자가 있기 때문에).
8. 엑시트의 범위는 환경에 따라 다릅니다. Format 이름은 다른 형식으로 충돌의 위험을 최소화하도록 선택해야 합니다. 형식 이름을 정의하는 애플리케이션을 식별하는 문자로 시작하는 것을 고려하십시오.
9. 데이터 변환 엑시트가 MQGET 호출을 실행한 프로그램의 엑시트와 같은 환경에서 실행됩니다. 환경에는 주소 공간과 사용자 프로파일을 포함합니다(해당 경우). 프로그램은 메시지 변환을 지원하지 않는 목적지 큐 관리자에 메시지를 송신하는 메시지 채널 에이전트일 수 있습니다. 큐 관리자의 환경에서 실행되지 않기 때문에 엑시트는 큐 관리자의 무결성에 타격을 주지 않습니다.
10. 엑시트가 사용할 수 있는 유일한 MQI 호출은 MQXCNV입니다. 이유 코드 MQRC\_CALL\_IN\_PROGRESS 또는 다른 예측 불가능한 오류로 다른 MQI 호출을 사용하면 실패합니다.
11. 큐 관리자는 MQ\_DATA\_CONV\_EXIT라는 시작점을 제공하지 않습니다. 그러나 typedef는 C 프로그래밍 언어의 이름 MQ\_DATA\_CONV\_EXIT에 제공되고 이 매개변수가 올바른지 확인하기 위해 사용자 작성 엑시트를 선언하는 데 사용할 수 있습니다. 엑시트의 이름은 모든 환경에서 필요하지 않지만 형식 이름과 동일해야 합니다(MQMD의 Format 필드에 포함된 이름).

다음 예제는 형식 MYFORMAT를 처리하는 엑시트를 C 프로그래밍 언어로 선언할 수 있는 방법을 설명합니다.

```
#include "cmqc.h"
#include "cmqxc.h"

MQ_DATA_CONV_EXIT MYFORMAT;

void MQENTRY MYFORMAT(
    PMQDXP    pDataConvExitParms, /* Data-conversion exit parameter
                                   block */
    PMQMD     pMsgDesc,           /* Message descriptor */
    MQLONG    InBufferLength,     /* Length in bytes of InBuffer */
    PMQVOID   pInBuffer,         /* Buffer containing the unconverted
                                   message */
    MQLONG    OutBufferLength,    /* Length in bytes of OutBuffer */
    PMQVOID   pOutBuffer)        /* Buffer containing the converted
                                   message */
{
    /* C language statements to convert message */
}
```

12.  z/OS에서 API 교차 엑시트도 강제 실행 상태이면 데이터 변환 엑시트 후 호출됩니다.

## C 호출

```
exitname (&DataConvExitParms, &MsgDesc, InBufferLength,
          InBuffer, OutBufferLength, OutBuffer);
```

엑시트에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQDXP    DataConvExitParms; /* Data-conversion exit parameter block */
MQMD     MsgDesc;           /* Message descriptor */
```

```

MQLONG InBufferLength; /* Length in bytes of InBuffer */
MQBYTE InBuffer[n]; /* Buffer containing the unconverted
message */
MQLONG OutBufferLength; /* Length in bytes of OutBuffer */
MQBYTE OutBuffer[n]; /* Buffer containing the converted
message */

```

## COBOL declaration ( IBM i only)

IBM i

```

CALL 'exitname' USING DATACONVEITPARMS, MSGDESC, INBUFFERLENGTH,
INBUFFER, OUTBUFFERLENGTH, OUTBUFFER.

```

엑시트에 전달된 매개변수는 다음과 같이 선언됩니다.

```

** Data-conversion exit parameter block
01 DATACONVEITPARMS.
   COPY CMQDXPV.
** Message descriptor
01 MSGDESC.
   COPY CMQMDV.
** Length in bytes of INBUFFER
01 INBUFFERLENGTH PIC S9(9) BINARY.
** Buffer containing the unconverted message
01 INBUFFER PIC X(n).
** Length in bytes of OUTBUFFER
01 OUTBUFFERLENGTH PIC S9(9) BINARY.
** Buffer containing the converted message
01 OUTBUFFER PIC X(n).

```

## System/390 어셈블러 선언

```

CALL EXITNAME, (DATACONVEITPARMS, MSGDESC, INBUFFERLENGTH, X
INBUFFER, OUTBUFFERLENGTH, OUTBUFFER)

```

엑시트에 전달된 매개변수는 다음과 같이 선언됩니다.

```

DATACONVEITPARMS CMQDXPA , Data-conversion exit parameter block
MSGDESC CMQMDA , Message descriptor
INBUFFERLENGTH DS F Length in bytes of INBUFFER
INBUFFER DS CL(n) Buffer containing the unconverted
* message
OUTBUFFERLENGTH DS F Length in bytes of OUTBUFFER
OUTBUFFER DS CL(n) Buffer containing the converted
* message

```

## MQRFH2 요소로 지정된 특성

비메시지 디스크립터 특성은 MQRFH2 헤더 폴더의 요소로 지정될 수 있습니다. 특성으로 지정되는 MQRFH2 요소의 개요입니다.

이는 IBM MQ JMS 및 XMS 클라이언트의 이전 버전과의 호환성을 유지합니다. 이 절에서는 MQRFH2 헤더에서 특성을 지정하는 방법에 대해 설명합니다.

특성으로서 MQRFH2 요소를 사용하려면 IBM MQ classes for Java 사용에서 설명된 대로 요소를 지정하십시오. 이 정보는 511 페이지의 『MQRFH2 - 규칙 및 형식화 헤더 2』에서 설명된 정보를 보충합니다.

## 특성 데이터 유형을 MQRFH2 데이터 유형에 매핑

이 주제에서는 해당하는 MQRFH2 데이터 유형에 매핑된 메시지 특성 유형에 대한 정보를 제공합니다.



표 635. 지원되는 MQRFH2 데이터 유형	
메시지 특성 유형	MQRFH2 데이터 유형
MQBYTE[]	bin.hex
MQBOOL	부울
MQINT8	i1
MQINT16	i2
MQINT32	i4
MQINT64	i8
MQFLOAT32	r4
MQFLOAT64	r8
MQCHAR[]	문자열

데이터 유형이 없는 요소는 "string" 유형으로 가정합니다.

int(지정되지 않은 크기의 정수를 의미)의 MQRFH2 데이터 유형은 i8인 것처럼 처리됩니다.

널 값은 요소 속성 `xsi:nil='true'` (으)로 표시됩니다. 널이 아닌 값에는 `xsi:nil='false'` 속성을 사용하지 마십시오.

예를 들어, 다음 특성에는 널값이 있습니다.

```
<NullProperty xsi:nil='true'></NullProperty>
```

바이트 또는 문자열 특성에는 비어 있는 값이 있을 수 있습니다. 이는 길이가 0인 요소 값이 있는 MQRFH2 요소로 나타냅니다.

예를 들어, 다음 특성에는 비어 있는 값이 있습니다.

```
<EmptyProperty></EmptyProperty>
```

## 지원되는 MQRFH2 폴더

특성으로서의 메시지 디스크립터 필드 사용의 개요입니다.

`<jms>`, `<mcd>`, `<mqext>` 및 `<usr>` 폴더는 MQRFH2 헤더 및 JMS에 설명되어 있습니다. `<usr>` 폴더는 메시지와 연관된 JMS 애플리케이션 정의 특성을 전송하는 데 사용됩니다. 그룹은 `<usr>` 폴더에서 허용되지 않습니다.

MQRFH2 헤더 및 JMS에서는 다음 추가 폴더를 지원합니다.

- `<mq>`  
이 폴더는 IBM MQ에서 사용하는 MQ 정의 특성에 대해 사용되고 예약됩니다.
- `<mq_usr>`  
이 폴더는 특성이 JMS 특성의 요구사항을 충족시키지 않는 것처럼 JMS 사용자 정의 특성으로 표시되지 않은 애플리케이션이 정의 특성을 전송하는 데 사용할 수 있습니다. 이 폴더는 `<usr>` 폴더가 포함할 수 없는 그룹을 포함할 수 있습니다.
- `content='properties'` 속성과 함께 표시된 폴더입니다.  
이 폴더는 콘텐츠의 `<mq_usr>` 폴더와 같습니다.
- `<mmps>`  
이 폴더는 IBM MQ 발행/구독 특성에 사용됩니다.

IBM MQ는 또한 WAS/SIB에서 이미 사용 중인 다음 폴더를 지원합니다.

- <sib>

이 폴더는 JMS 특성으로 표시되지 않거나 JMS\_IBM\_\* 특성에 맵핑되지만 WAS/SIB 애플리케이션에 표시되는 WAS/SIB 시스템 메시지 특성에 사용되고 예약됩니다. 여기에는 순방향 및 역방향 라우팅 경로 특성이 포함됩니다.

바이트 배열이기 때문에 적어도 일부는 JMS 특성으로 표시할 수 없습니다. 애플리케이션이 특성을 이 폴더에 추가하는 경우 값이 무시되거나 제거됩니다.

- <sib\_usr>

이 폴더는 지원되는 유형이 아니기 때문에 JMS 특성으로 표시될 수 없는 WAS/SIB 사용자 메시지 특성에 대해 사용되거나 예약되며 WAS/SIB 애플리케이션에 표시됩니다.

이는 SIMessage 인터페이스를 통해 가져오거나 설정할 수 있는 사용자 특성이지만 바이트 배열의 콘텐츠는 필수 특성 값에 맵핑됩니다.

IBM MQ 애플리케이션에서 임의의 bin.hex 요소를 폴더에 작성하는 경우 애플리케이션은 복원될 것으로 예상되는 형식이 아니기 때문에 IOException을 수신합니다. bin.hex 요소 이외의 것을 추가하는 경우 ClassCastException을 수신합니다.

이 폴더를 사용하여 WAS/SIB에서 특성을 사용 가능하게 하지 마십시오. 대신 해당 용도로 <usr> 폴더를 사용자에게 지정하십시오.

- <sib\_context>

이 폴더는 WAS/SIB 사용자 애플리케이션에 표시되지 않는 WAS/SIB 시스템 메시지 특성에 대해 또는 JMS 특성으로 사용됩니다. 여기에는 웹 서비스와 유사하게 사용되는 보안 및 트랜잭션 특성이 포함됩니다.

애플리케이션은 특성을 이 폴더에 추가하지 않아야 합니다.

- <mqema>

이 폴더는 <mqext> 폴더 대신 WAS/SIB에서 사용되었습니다.

MQRFH2 폴더 이름은 대소문자를 구분합니다.

다음 폴더는 소문자이거나 대문자와의 혼합으로 예약됩니다.

- mq 또는 wmq로 시작하는 폴더가 IBM MQ에서 사용하도록 예약됩니다.
- sib로 시작하는 폴더가 WAS/SIB에서 사용하도록 예약됩니다.
- 예약되었지만 사용되지 않는 <Root> 및 <Body> 폴더.

다음 폴더는 메시지 특성을 포함하는 것으로 인식되지 않습니다.

- <psc>

발행/구독 명령 메시지를 브로커에게 전달하기 위해 IBM Integration Bus에서 사용됩니다.

- <pscr>

발행/구독 명령 메시지에 응답하여 브로커의 정보를 포함하기 위해 IBM Integration Bus에서 사용됩니다.

- IBM에서 정의되지 않은 폴더. content='properties' 속성으로 표시되지 않습니다.

<psc> 또는 <pscr> 폴더에 content='properties' 를 지정하지 마십시오. 이를 수행한 경우 이러한 폴더는 특성으로 처리되고 IBM Integration Bus는 예상대로 기능을 중지합니다.

애플리케이션이 특성을 포함한 메시지를 빌드 중인 경우 MQRFH2 헤더에서 특성을 포함하는 MQRFH2 헤더로 인식되도록 하려면 헤더가 메시지 헤더에서 연결될 수 있는 헤더의 목록에 있어야 합니다.

MQRFH2는 임의 수의 MQH 표준 헤더 또는 MQCIH, MQDLH, MQIIH, MQTM, MQTMC2 또는 MQXQH에 선행할 수 있습니다. 연결할 수 없기 때문에 문자열 또는 MQCFH는 구문 분석을 종료합니다.

메시지는 모든 메시지 특성을 수행하는 다중 MQRFH2 헤더를 포함할 수 있습니다. 그렇지 않은 경우 WAS/SIB 와 같이 제한이 없는 한 동일한 이름의 폴더는 다른 헤더에 공존할 수 있습니다. 폴더가 모두 중요한 헤더에 있는 경우 폴더는 하나의 논리 폴더로 처리됩니다.



중요한 헤더의 폴더는 중요하지 않은 헤더의 폴더와 병합될 수 없지만 중요한 헤더 내의 동일한 이름의 폴더는 병합되어 충돌하는 특성을 제거할 수 있습니다. 애플리케이션은 해당 메시지 내의 특성 레이아웃에 종속되지 않아야 합니다.

MQRFH2 그룹은 사용자 정의 폴더(<wmq>, <jms>, <mcd>, <usr>, <mqext>, <sib>, <sib\_usr>, <sib\_context> 및 <mqema> 폴더가 아님)의 특성에 대해 구문 분석됩니다.

IBM정의 특성 폴더의 그룹 (<wmq> 및 <mq> 폴더 제외)은 특성에 대해 구문 분석됩니다.

MQRFH2 폴더는 혼합 콘텐츠를 포함할 수 없습니다. 폴더 또는 그룹은 둘 모두가 아닌 그룹 또는 특성 아니면 값을 포함할 수 있습니다.

메시지에서 세그먼트(첫 번째 또는 후속 세그먼트)는 메시지 디스크립터에서 해당 특성 이외에 IBM MQ 정의 특성을 포함할 수 없습니다. 따라서 각각 MQMF\_SEGMENT 또는 MQMF\_SEGMENTATION\_ALLOWED 세트의 이와 같은 특성을 포함하는 메시지를 넣으면 MQRC\_SEGMENTATION\_NOT\_ALLOWED와 함께 넣기 조작이 실패하게 합니다.


그러나 메시지 그룹은 IBM MQ 정의 특성을 포함할 수 있습니다.

## MQRFH2 헤더의 생성

IBM MQ가 해당 MQRFH2 표현으로 메시지 특성을 변환하는 경우 MQRFH2를 메시지에 추가해야 합니다. 별도의 헤더로 MQRFH2를 추가하거나 기존 헤더와 병합합니다.

IBM MQ로 새 MQRFH2 헤더를 생성하면 메시지의 기존 헤더가 손상될 수 있습니다. 헤더에 대한 메시지 버퍼를 구문 분석하는 애플리케이션은 버퍼에서 헤더의 수 및 위치가 일부 상황에서 변경될 수 있음을 알아야 합니다. IBM MQ는 메시지 특성을 병합할 수 있는 기존 MQRFH2 헤더로 병합하여 특성을 메시지에 추가하는 영향을 최소화하려고 시도합니다. 또한 생성된 MQRFH2를 메시지 버퍼의 다른 헤더와 관련하여 고정된 위치에 삽입하여 영향을 최소화하려고 시도합니다.

생성된 MQRFH2 헤더는 MQMD 및 임의의 수의 MQXQH, MQRFH 및 MQDLH 헤더가 있는 순서대로 배치됩니다. 생성된 MQRFH2 헤더는 MQMD, MQXQH, MQDLH 또는 MQRFH 헤더가 아닌 첫 번째 헤더 바로 앞에 배치됩니다.

 z/OS 시스템에서 생성된 MQRFH2 헤더는 애플리케이션의 CCSID에 작성됩니다. 이는 다음과 같이 정의됩니다.

- DLL 인터페이스를 사용하는 배치 LE 애플리케이션의 경우 CCSID는 **MQCONN**이 실행될 때 현재 로케일과 연관된 CODESET입니다(기본값이 1047임).
- 배치 MQ 스텝 중 하나와 바인딩된 LE 애플리케이션의 경우 CCSID는 **MQCONN** 뒤에 MQI 호출이 처음 실행될 때 현재 로케일과 연관된 CODESET입니다(기본값은 1047임).
- z/OS UNIX System Services (z/OS UNIX) 스레드에서 실행 중인 배치 비LE 애플리케이션의 경우, CCSID는 **MQCONN** 이후에 발행된 첫 번째 MQI 호출 시 THLICCSID의 값입니다(기본값은 1047임).
- 다른 배치 애플리케이션의 경우 CCSID는 큐 관리자의 CCSID입니다.

LE 애플리케이션의 경우 setlocale() / CESETL LE 호출 가능 서비스를 사용하여 로케일을 변경할 수 있습니다. z/OS UNIX 스레드에서 실행하는 비LE 애플리케이션의 경우 THLICCSID 값은 z/OS UNIX 맵핑 매크로 **BPXYTHLI**를 사용하여 변경될 수 있습니다.

## 생성된 MQRFH2를 병합하기 위한 규칙

다음 규칙은 생성된 MQRFH2와 기존 MQRFH2의 병합에 적용됩니다. 생성된 MQRFH2 헤더는 다음의 경우 기존 MQRFH2 헤더와 병합됩니다.

1. 기존 MQRFH2는 동일한 위치에서 IBM MQ가 MQRFH2를 생성하거나 헤어 체인의 이전 버전을 배치합니다.
2. 생성된 특성의 CCSID는 기존 MQRFH2의 NameValueCCSID와 동일합니다.

그렇지 않은 경우 생성된 헤더는 이전에 설명한 위치에서 버퍼에 개별적으로 배치됩니다.

## 기존 MQRFH2에 폴더를 병합하기 위한 규칙

메시지 특성이 기존 MQRFH2로 병합되면 기존 MQRFH2는 메시지 특성과 일치하는 폴더에 대해 스캐닝되고 병합합니다. 일치하는 폴더가 없는 경우 새 폴더는 기존 폴더의 마지막에 추가됩니다. 일치하는 폴더가 있는 경우 폴더가 검색됩니다. 일치하는 특성은 덮어씁니다. 새 폴더는 폴더의 마지막에 추가됩니다.

## MQRFH2 폴더 제한

MQRFH2 헤더에서 폴더 제한의 개요

MQRFH2 제한은 다음 폴더에 적용됩니다.

- <usr> 폴더의 요소 이름은 JMS 접두부로 시작하지 않아야 합니다. 이 특성 이름은 JMS에서 사용하도록 예약되어 있으며 사용자 정의 특성에는 유효하지 않습니다.

해당 요소 이름으로 MQRFH2의 구문 분석이 실패하지는 않지만, 이는 IBM MQ 메시지 특성 API에 대해 액세스 가능하지 않습니다.

- <usr> 폴더의 요소 이름은 소문자 또는 대문자가 혼합된 NULL, TRUE, FALSE, NOT, AND, OR, BETWEEN, LIKE, IN, IS 및 ESCAPE가 아니어야 합니다. 이 이름은 SQL 키워드와 일치하고 선택기 구문 분석을 더 어렵게 만듭니다. <usr>은(는) 선택기의 특정 특성에 대해 폴더가 지정되지 않은 경우 사용되는 기본 폴더이기 때문입니다.

해당 요소 이름으로 MQRFH2의 구문 분석이 실패하지는 않지만, 이는 IBM MQ 메시지 특성 API에 대해 액세스 가능하지 않습니다.

- <usr> 폴더의 콘텐츠 모델은 다음과 같습니다.

- 콜론을 포함하지 않는 경우, 올바른 XML 이름은 요소 이름으로 사용될 수 있습니다.
- 중첩된 폴더가 아닌 단순 요소만이 허용됩니다.
- dt="xxx" 속성에서 수정되지 않는 경우 모든 요소는 기본 유형의 문자열을 사용합니다.
- 모든 요소는 선택사항이지만 폴더에서 두 번 이상 발생되지 않아야 합니다.

- 메시지 특성을 포함하는 것으로 간주되는 폴더의 요소 이름에는 마침표(.) (유니코드 문자 U+002E)가 포함되지 않아야 합니다. 이 문자는 계층 구조를 표시하기 위해 특성 이름에 사용되기 때문입니다.

해당 요소 이름으로 MQRFH2의 구문 분석이 실패하지는 않지만, 이는 IBM MQ 메시지 특성 API에 대해 액세스 가능하지 않습니다.

일반적으로 MQRFH2의 특정 요소가 IBM MQ 메시지 특성 API를 통하여 액세스 가능하지 않더라도 유효한 XML-스타일 데이터를 포함하는 MQRFH2 헤더는 실패 없이 IBM MQ에 의해 구문 분석될 수 있습니다.

## MQRFH2 요소 이름 충돌

MQRFH2 요소 이름 내의 충돌 개요입니다.

하나의 값만 메시지 특성에 첨부할 수 있습니다. 특성에 액세스하기 위한 시도가 값의 충돌로 이어지는 경우 하나가 다른 특성보다 우선적으로 선택됩니다.

폴더에 동일한 이름의 요소가 포함되지 않은 경우 MQRFH2 요소에 액세스하기 위한 IBM MQ 구문은 요소의 고유 ID를 허용합니다. 폴더에 동일한 이름의 요소가 둘 이상 포함된 경우 사용된 특성 값은 메시지 헤더에 가장 가까운 값입니다.

이는 동일한 메시지 내의 다른 중요한 MQRFH2 헤더에 동일한 이름의 폴더가 둘 이상 포함된 경우 적용됩니다.

메시지가 아닌 디스크립터 특성이 두 번 설정된 후(MQSETMP 호출을 통하고 원시 MQRFH2 헤더에서 직접) MQGET 호출이 처리될 때 충돌이 발생할 수 있습니다.

이 경우 API 호출에 의해 메시지와 연관된 특성은 메시지 데이터의 하나 즉, 원시 MQRFH2 헤더의 하나보다 우선합니다. 충돌이 발생하면 논리적으로 메시지 데이터 앞에 오는 것으로 간주됩니다.

## 특성 이름에서 MQRFH2 폴더 및 요소 이름으로 맵핑

MQRFH2 헤더에서 특성 이름 및 요소 이름 사이의 차이의 개요입니다.

궁극적으로 MQRFH2 헤더를 생성하는 정의된 API를 사용할 때 메시지 특성(예: MQ JMS)을 지정하려면 특성 이름은 MQRFH2 폴더의 요소 이름일 필요는 없습니다.

따라서 맵핑은 특성 이름에서 MQRFH2 요소로 발생하고 그 반대로 요소 및 요소 이름을 포함하는 폴더 이름을 모두 고려합니다. IBM MQ classes for JMS의 일부 예는 이미 [IBM MQ classes for Java](#) 사용에 문서화됩니다.

표 636. 특성 이름을 MQRFH2 폴더 및 요소 이름에 맵핑		
특성 이름	MQRFH2 폴더 이름	MQRFH2 요소 이름
JMSDestination	jms	Dst
JMSType	mcd	Type, Set, Fmt
xxx(사용자 정의됨, 여기서 xxx는 JMS로 시작하지 않음)	usr	xxx

따라서 JMS 애플리케이션이 JMSDestination 특성에 액세스하면 <jms> 폴더의 Dst 요소에 맵핑됩니다.

MQRFH2 요소로서 특성을 지정할 때 IBM MQ는 다음과 같이 해당 요소를 정의합니다.

표 637. 특성 이름을 MQRFH2 폴더, 그룹 및 요소 이름에 맵핑			
특성 이름	MQRFH2 폴더 이름	MQRFH2 그룹 이름	MQRFH2 요소 이름
<Property>	<usr>	해당사항 없음	<Property>
<folder>.<Property>	<folder>	해당사항 없음	<Property>
<folder>.<group>.<Property>	<folder>	<group>	<Property>

예를 들어, IBM MQ 애플리케이션이 Property1 특성에 액세스할 때 이는 <usr> 폴더의 Property1 요소에 맵핑합니다. wmq.Property2 특성은 <wmq> 폴더의 Property2 특성에 맵핑합니다.

특성 이름에 둘 이상의 문자가 포함된 경우 사용된 MQRFH2 요소 이름은 마지막 문자 다음에 오는 이름이고 MQRFH2 그룹은 계층 구조를 형성하는 데 사용됩니다. 중첩된 MQRFH2 그룹은 허용됩니다.

<mcd>, <jms> 및 <mqext> 폴더의 MQRFH2 에 포함된 JMS 헤더 및 제공자 특정 특성은 [IBM MQ classes for Java](#) 사용에 정의된 축약 이름을 사용하여 IBM MQ 애플리케이션에 의해 액세스됩니다.

JMS 사용자 정의 특성은 <usr> 폴더에서 액세스합니다. IBM MQ 애플리케이션은 특성이 JMS 애플리케이션에 사용자 정의 특성 중 하나로 표시될 수 있는 경우 해당 애플리케이션 특성에 대해 <usr> 폴더를 사용할 수 있습니다.

허용되지 않는 경우 다른 폴더를 선택하십시오. <wmq\_usr> 폴더는 이러한 비JMS 특성의 표준 위치로 제공됩니다.

애플리케이션은 잘 정의된 사용으로 MQRFH2 폴더를 지정하여 사용할 수 있지만 다음을 참고할 경우 [890 페이지](#)의 『MQRFH2 요소로 지정된 특성』에 문서화되지 않습니다.

1. 이 폴더는 이미 사용 중이거나 그 안에 포함된 특성에 정의되지 않은 액세스를 제공하는 다른 애플리케이션에 의해 나중에 사용될 수 있습니다. 특성 이름에 제안된 이름 지정 규칙은 [특성 이름을 참조하십시오](#).
2. 특성은 IBM MQ classes for JMS의 이전 버전에 액세스할 수 없거나 사용자 정의 특성에 <usr> 폴더만 액세스할 수 있는 XMS 클라이언트입니다.
3. 이 폴더는 properties로 설정된 값을 사용하여 content 속성으로 표시되어야 합니다(예: content='properties').

[751 페이지](#)의 『MQSETMP - 메시지 특성 설정』에서는 필요에 따라 자동으로 이 속성을 추가합니다. 이 속성은 IBM 정의 폴더(예: <jms> 및 <usr>)에 추가되지 않아야 합니다. 이를 수행하면 IBM WebSphere MQ 7.0이전에 IBM MQ classes for JMS 클라이언트가 메시지를 거부합니다. MessageFormatException이 나타납니다.

<usr> 폴더는 <Property> 구문 특성의 기본 위치이므로 IBM MQ 애플리케이션과 JMS 애플리케이션이 동일한 이름을 사용하여 동일한 사용자 정의 특성 값에 액세스할 수 있습니다.

## 예약된 폴더 이름

여러 예약된 폴더 이름이 있습니다. 폴더 접두부와 같은 이름을 사용할 수 없습니다(예를 들어, Root가 예약되기 때문에 Root.Property1은 유효한 특성에 액세스하지 않습니다). 다음 목록은 예약된 폴더 이름을 포함합니다.

- 루트
- 본문
- 특성
- 환경
- LocalEnvironment
- DestinationList
- ExceptionList
- InputBody
- InputRoot
- InputProperties
- InputLocalEnvironment
- InputDestinationList
- InputExceptionList
- OutputRoot
- OutputLocalEnvironment
- OutputDestinationList
- OutputExceptionList

## 특성 디스크립터 필드를 MQRFH2 헤더에 맵핑

특성이 MQRFH2 요소로 변환될 때 특성 디스크립터의 중요한 필드를 지정하는 데 요소 속성이 사용됩니다. MQPD 필드가 MQRFH2 요소 속성으로 변환되는 방법에 대해 설명합니다.

### 지원

지원 특성 디스크립터 필드는 세 가지 요소 속성으로 나누어집니다

- **sr** 요소 속성은 MQPD\_REJECT\_UNSUP\_MASK 비트 마스크의 값을 지정합니다.
- **sa** 요소 속성은 MQPD\_ACCEPT\_UNSUP\_MASK 비트 마스크의 값을 지정합니다.
- **sx** 요소 속성은 MQPD\_ACCEPT\_UNSUP\_IF\_XMIT\_MASK 비트 마스크의 값을 지정합니다.

이러한 요소 속성은 <mq> 폴더에만 유효하며 특성을 포함하는 다른 폴더의 요소에 설정된 경우 무시됩니다.

표 638. MQPD 필드를 MQRFH2 요소 속성에 맵핑		
지원 값	MQRFH2 요소 속성	MQRFH2 속성 값
MQPD_SUPPORT_OPTIONAL	sa	선택사항 이는 기본값입니다.
MQPD_SUPPORT_REQUIRED	sr	필수
MQPD_SUPPORT_REQUIRED_IF_LOCAL	sx	로컬

### 컨텍스트

특성이 속한 메시지 컨텍스트를 표시하려면 **context** 요소 속성을 사용하십시오. 하나의 값만 사용하십시오. 이 요소 속성은 특성을 포함하는 폴더의 특성에 유효합니다.

표 639. 컨텍스트 값을 MQRFH2 속성 값에 맵핑	
컨텍스트 값	MQRFH2 속성 값
MQPD_NO_CONTEXT	없음 이는 기본값입니다.
MQPD_USER_CONTEXT	사용자

### CopyOptions

특성이 복사되어야 하는 메시지를 표시하려면 **copy** 요소 속성을 사용하십시오. 둘 이상의 값이 허용 가능합니다. 쉽표로 다중 값을 구분하십시오. 예를 들어, **copy='reply'** 및 **copy='publish,report'** 는 모두 유효합니다. 이 요소 속성은 특성을 포함하는 폴더의 특성에 유효합니다.

**참고:** 속성 정의에서 작은따옴표 또는 큰따옴표가 올바른 사용입니다(예: **copy='reply'** 또는 **copy="report"**).

표 640. CopyOption 값을 MQRFH2 속성 값에 맵핑	
CopyOption 값	MQRFH2 속성 값
MQPD_COPY_FORWARD	forward
MQPD_COPY_REPLY	응답
MQPD_COPY_REPORT	보고
MQPD_COPY_PUBLISH	발행(Publish)
MQPD_COPY_ALL	모두 기타 값으로 이를 지정하지 마십시오. 다른 값으로 사용될 때 이는 <b>none</b> 을 제외한 값에 우선합니다.
MQPD_COPY_DEFAULT	기본값 이는 기본값입니다. 세 개의 값 MQCOPY_FORWARD, MQCOPY_REPORT 및 MQCOPY_PUBLISH를 지정하는 것과 동등합니다. 기타 값으로 이를 지정하지 마십시오.
MQPD_COPY_NONE	없음 기타 값으로 이를 지정하지 마십시오. 다른 값과 함께 사용될 때 이는 우선합니다.

### <mq> MQRFH2 폴더에 대한 제한

메시지를 큐에 넣을 때 메시지가 해당 MQ 정의의 특성에 따라 처리될 수 있도록 <mq> 폴더가 검색됩니다. MQ 정의의 특성의 효율적 구문 분석을 허용하기 위해 다음 제한이 폴더에 적용됩니다.

- 메시지에서 중요한 첫 번째 <mq> 폴더에 있는 특성에 대해서만 MQ가 수행합니다. 메시지의 다른 <mq> 폴더에 있는 특성은 무시됩니다.
- 폴더가 UTF-8인 경우 단일 바이트 UTF-8 문자만 폴더에 허용됩니다. 폴더에서 멀티바이트 문자를 사용하면 구문 분석에 실패할 수 있으므로 메시지가 거부될 수 있습니다.
- MQRFH2 그룹을 <mq> 폴더에 포함시키지 마십시오. 특성 값의 유니코드 문자 U+003C 존재로 메시지가 거부되게 합니다.
- 폴더에 이스케이프 문자열을 사용하지 마십시오. 이스케이프 문자열은 요소의 실제 값으로 처리됩니다.
- 유니코드 문자 U+0020만 폴더 내에서 공백으로 처리됩니다. 다른 모든 문자는 중요한 문자로 처리되며 폴더의 구문 분석이 실패할 수 있으므로 메시지가 거부됩니다.

<mq> 폴더의 구문 분석이 실패하거나 폴더가 이러한 제한사항을 준수하지 않는 경우 메시지는 CompCode **MQCC\_FAILED** 및 이유 **MQRC\_RFH\_RESTRICTED\_FORMAT\_ERR**로 거부됩니다.

## 유효하지 않는 MQRFH2 헤더

MQPUT, MQPUT1 또는 MQGET 호출 프로세스 중 메시지에 있는 MQRFH2 헤더의 부분 구문 분석을 수행하여 포함된 폴더를 확인하고 폴더에 특성이 포함되어 있는지 여부를 판별할 수 있습니다. 유효하지 않은 MQRFH2 헤더의 개요입니다.

구조가 유효하지 않으므로(예를 들어, StrucLength 필드가 너무 작음) 메시지의 부분 구문 분석을 완료할 수 없는 경우:

- 기존 애플리케이션이 실패하지 않도록 애플리케이션에 일부 IBM WebSphere MQ 7 옵션이 포함되어 있다고 판단할 수 있는 경우 MQRC\_RFH\_ERROR 이유 코드와 함께 MQPUT 또는 MQPUT1 호출에 실패합니다.
- MQGET 호출은 리턴하고 오류를 포함하는 MQRFH2가 제공된 버퍼에서 리턴됩니다.

특정 폴더에 특성이 포함되어 있는지 감지할 수 없기 때문에(예: 폴더가 <<jms 시작) 부분 구문 분석이 실패하는 경우 폴더 이름이 판별되기 전에 구문 분석에 실패합니다.

- 기존 애플리케이션이 실패하지 않도록 애플리케이션에 일부 IBM WebSphere MQ 7 옵션이 포함되어 있다고 판단할 수 있는 경우 MQRC\_RFH\_FORMAT\_ERROR 이유 코드와 함께 MQPUT 또는 MQPUT1 호출에 실패합니다.
- MQGET 호출은 리턴하고 오류를 포함하는 MQRFH2가 제공된 버퍼에서 리턴됩니다.
- 큐 관리자 내에서 내부적으로 메시지 형식이 잘못된 폴더로 인해 거부되지 않지만 폴더는 항상 내부에 특성이 포함되지 않는 것처럼 처리됩니다.

메시지에 하나 이상의 폴더가 있는 동안 메시지는 폴더가 이러한 구문 분석 오류를 포함하는 폴더와 큐 관리자 네트워크를 통해 플로우할 수 있지만 구문 분석 및 감지되지 않습니다.

- 올바름
- 구문 분석됨
- 메시지 처리에 사용됨

그러므로 감지는 보장되지 않습니다.

애플리케이션 중 하나가 751 페이지의 『MQSETMP - 메시지 특성 설정』 또는 MQINQMP를 사용하여 특성에 액세스하는 경우 MQRFH2 폴더가 완전히 구문 분석되고 구문 분석을 완료할 수 없는 오류가 감지되면 API 호출에 적절한 리턴 코드가 표시됩니다. 애플리케이션에서 폴더의 특성을 사용할 수 없습니다.

MQRFH2 폴더를 완전히 구문 분석하려고 시도하고 구문 분석기가 인식되지 않은 요소 속성 또는 인식되지 않은 데이터 유형을 발견하는 경우 구문 분석은 경고 없이 계속 실행되고 성공적으로 완료됩니다. 이는 구분 분석 오류를 구성하지 않습니다.

## 코드 페이지 변환

이 절에서는 코드셋 이름 및 CCSID, 자국어, z/OS 변환, IBM i 변환 및 유니코드 변환 지원에 대해 설명합니다.

각 자국어 절은 다음 정보를 나열합니다.

- 지원되는 고유 CCSID
- 지원되지 않은 코드 페이지 변환

다음 용어가 정보에서 사용됩니다.

**AIX** **AIX**  
IBM MQ for AIX를 표시합니다.

**Linux** **Linux**  
IBM MQ for Linux for Intel 및 IBM MQ for Linux for zSeries를 표시합니다.

**IBM i** **OS/400**  
IBM MQ for IBM i를 표시합니다.

## Windows Windows

IBM MQ for Windows를 표시합니다.

## z/OS z/OS

IBM MQ for z/OS를 표시합니다.

데이터 변환의 기본값은 대상(수신) 시스템에서 수행하는 변환입니다.

소스 제품이 변환을 지원하는 경우 채널 속성 CONVERT를 소스에서 YES로 설정하여 채널을 설정하고 데이터를 교환할 수 있습니다.

### 참고:

1. IBM MQ MQI client 정보 변환은 서버에서 발생하므로 서버는 클라이언트 CCSID에서 서버 CCSID로의 변환을 지원해야 합니다.
2. 변환은 IBM MQ의 최신 버전에 CSD/PTF에서 추가된 지원을 포함할 수 있습니다. 이 변환을 사용 가능으로 설정하기 위해 CSD/PTF를 설치해야 하는 경우 최신 서비스 레벨의 콘텐츠를 확인하십시오.
3. IBM MQ 큐 관리자 CCSID는 Mixed 또는 SBCS여야 합니다.
4. 운영 체제에서 지원되지 않는 일부 CCSID(예: AIX용 850)는 여전히 애플리케이션에서 사용될 수 있고 IBM MQ 큐 관리자 CCSID로 설정될 수도 있습니다. 이는 역호환성을 위해서만 허용되며 관련 변환 테이블이 설치되어 있지 않은 경우 변환에 실패합니다.

일부 CCSID 번호 및 일부 업계 코드세트 이름 간의 상호 참조는 [899 페이지의 표 641](#)의 내용을 참조하십시오.

### 관련 참조

[900 페이지의 『자국어』](#)

이 정보는 IBM MQ에서 지원되는 언어를 포함합니다.

## 코드세트 이름 및 CCSID

코드세트 이름 및 각 코드세트 이름에 해당하는 CCSID.

**z/OS** IBM MQ for z/OS에서는 언어 특정 표에 나열된 것보다 많은 변환을 제공합니다. 변환의 전체 목록은 [925 페이지의 Table 674](#)의 내용을 참조하십시오.

코드세트 이름	CCSIDs
ISO 8859-1	819
ISO 8859-2	912
ISO 8859-3	913
ISO 8859-5	915
ISO 8859-6	1089
ISO 8859-7	813
ISO 8859-8	916
ISO 8859-9	920
ISO 8859-13	921
ISO 8859-15(유로)	923
big5	950
eucJP	954 5050 33722
eucKR	970
eucTW	964

표 641. 코드셋 이름 및 CCSID (계속)	
코드셋 이름	CCSIDs
eucCN	1383
PCK	943
GBK	1386
koi8-r	878

## 자국어

이 정보는 IBM MQ에서 지원되는 언어를 포함합니다.






IBM MQ에서 지원되는 언어는 다음과 같습니다.

- US 영어 - 901 페이지의 『영어(미국)』 주제 참조
- 독일어 - 901 페이지의 『독일어』 주제 참조
- 덴마크어 및 노르웨이어 - 902 페이지의 『덴마크어 및 노르웨이어』 주제 참조
- 핀란드어 및 스웨덴어 - 903 페이지의 『핀란드어 및 스웨덴어』 주제 참조
- 이탈리아어 - 903 페이지의 『이탈리아어』 주제 참조
- 스페인어 - 904 페이지의 『스페인어』 주제 참조
- 영국 영어/게일어 - 905 페이지의 『영국 영어/게일어』 주제 참조
- 프랑스어 - 905 페이지의 『프랑스어』 주제 참조
- 다국어 - 906 페이지의 『다국어』 주제 참조
- 포르투갈어 - 906 페이지의 『포르투갈어』 주제 참조
- 아이슬란드어 - 907 페이지의 『아이슬란드어』 주제 참조
- 동유럽어 - 908 페이지의 『동유럽 언어』 주제 참조
- 키릴 문자 - 909 페이지의 『키릴 문자』 주제 참조
- 에스토니아어 - 910 페이지의 『에스토니아어』 주제 참조
- 라트비아어 및 리투아니아어 - 911 페이지의 『라트비아어 및 리투아니아어』 주제 참조
- 우크라이니아어 - 912 페이지의 『우크라이니아어』 주제 참조
- 그리스어 - 913 페이지의 『그리스어』 주제 참조
- 터키어 - 914 페이지의 『터키어』 주제 참조
- 히브리어 - 914 페이지의 『히브리어』 주제 참조
- 페르시아어 - 916 페이지의 『페르시아어』 주제 참조
- 우르두어 - 916 페이지의 『우르두어』 주제 참조
- 태국어 - 917 페이지의 『태국어』 주제 참조
- 라오스어 - 917 페이지의 『라오스어』 주제 참조
- 베트남어 - 918 페이지의 『베트남어』 주제 참조
- 일본어 라틴 SBCS - 918 페이지의 『일본어 라틴 SBCS』 주제 참조
- 일본어 가타카나 SBCS - 919 페이지의 『일본 가타카나 SBCS』 주제 참조
- 일본어 간지/ 라틴 혼합 - 920 페이지의 『일본어 간지/ 라틴 혼합』 주제 참조
- 일본어 간지/ 가타카나 혼합 - 921 페이지의 『일본어 간지/ 가타카나 혼합』 주제 참조
- 한국어 - 922 페이지의 『한국어』 주제 참조
- 중국어 간체 - 923 페이지의 『중국어』 주제 참조
- 중국어 번체 - 924 페이지의 『대만어』 주제 참조



## 영어(미국)

US 영어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 642. 지원 플랫폼의 US 영어에 대한 고유 CCSID	
플랫폼	고유 CCSID
 IBM i  z/OS	37, 924, 1140
 AIX	819, 923, 5348
 Windows	437, 850, 1252, 5348, 858
 Linux	819, 923
Apple 클라이언트	1275

모든 비클라이언트 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 전환을 지원하지 않으며, 다음과 같은 예외가 발생합니다.

### IBM i



코드 페이지:

#### 37

코드 페이지 923, 858로 변환하지 않음

#### 924






코드 페이지 437, 858, 1051, 1140, 1252, 1275, 5348로 변환하지 않음

#### 1140

924, 1051, 1275 코드 페이지로 전환하지 않습니다.

## 독일어

독일어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 643. 지원 플랫폼의 독일어에 대한 고유 CCSID	
플랫폼	고유 CCSID
 IBM i  z/OS	273, 924, 1141
 AIX	819, 923, 5348
 Windows	437, 850, 858, 1252, 5348
 Linux	819, 923
Apple 클라이언트	1275

모든 비클라이언트 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 전환을 지원하지 않으며, 다음과 같은 예외가 발생합니다.

## IBM i



코드 페이지:

### 273

코드 페이지 858, 923, 924, 1275로 변환하지 않습니다.

### 924

코드 페이지 273, 437, 858, 1051, 1141, 1252, 1275, 5348로 변환하지 않음

### 1141

924, 1051, 1275 코드 페이지로 전환하지 않습니다.

## 덴마크어 및 노르웨이어

덴마크어 및 노르웨이어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 644. 지원 플랫폼의 덴마크어 및 노르웨이어에 대한 고유 CCSID	
플랫폼	고유 CCSID
IBM i z/OS	277, 924, 1142
AIX	819, 923, 5348
Windows	850, 858, 865, 1252, 5348
Linux	819, 923
Apple 클라이언트	1275

모든 비클라이언트 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 전환을 지원하지 않으며, 다음과 같은 예외가 발생합니다.

## IBM i



코드 페이지:

### 277

코드 페이지 858, 923, 924, 1275로 변환하지 않습니다.

### 924

코드 페이지 277, 858, 865, 1051, 1142, 1252, 1275, 5348로 변환하지 않음

### 1142

코드 페이지 924, 865, 1051, 1275로 변환하지 않음

## AIX



코드 페이지:

### 819

코드 페이지 865로 변환하지 않음

## Windows








코드 페이지:

## 865

코드 페이지 1051, 1275로 변환하지 않음

## 핀란드어 및 스웨덴어

핀란드어 및 스웨덴어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 645. 지원 플랫폼의 핀란드어 및 스웨덴어에 대한 고유 CCSID	
플랫폼	고유 CCSID
 IBM i  z/OS	278, 924, 1143
 AIX	819, 923, 5348
 Windows	437, 850, 858, 865, 1252, 5348
 Linux	819, 923
Apple 클라이언트	1275

모든 비클라이언트 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 전환을 지원하지 않으며, 다음과 같은 예외가 발생합니다.

## IBM i



코드 페이지:

### 278

코드 페이지 858, 923, 924, 1275로 변환하지 않습니다.

### 924

코드 페이지 278, 437, 858, 865, 1051, 1143, 1252, 1275, 5348로 변환하지 않음

### 1143

코드 페이지 865, 924, 1051, 1275로 변환하지 않음

## AIX



코드 페이지:

### 819

코드 페이지 865로 변환하지 않음

### 850

코드 페이지 865로 변환하지 않음

## Windows



코드 페이지:






### 865

코드 페이지 1051, 1275로 변환하지 않음

## 이탈리아어

이탈리아어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 646. 지원 플랫폼의 이탈리아어에 대한 고유 CCSID

플랫폼	고유 CCSID
 IBM i  z/OS	280, 924, 1144
 AIX	819, 923, 5348
 Windows	437, 850, 858, 1252, 5348
 Linux	819, 923
Apple 클라이언트	1275

모든 비클라이언트 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 전환을 지원하지 않으며, 다음과 같은 예외가 발생합니다.

### IBM i



코드 페이지:

#### 280

코드 페이지 858, 923, 924, 1275로 변환하지 않습니다.

#### 924

코드 페이지 280, 437, 858, 1051, 1144, 1252, 1275, 5348로 변환하지 않음






#### 1144

924, 1051, 1275 코드 페이지로 전환하지 않습니다.

### 스페인어

스페인어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 647. 지원 플랫폼의 스페인어에 대한 고유 CCSID

플랫폼	고유 CCSID
 IBM i  z/OS	284, 924, 1145
 AIX	819, 923, 5348
 Windows	437, 850, 858, 1252, 5348
 Linux	819, 923
Apple 클라이언트	1275

모든 비클라이언트 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 전환을 지원하지 않으며, 다음과 같은 예외가 발생합니다.

### IBM i



코드 페이지:

**284**

코드 페이지 858, 923, 924, 1275로 변환하지 않습니다.

**924**






코드 페이지 284, 437, 858, 1051, 1145, 1252, 1275, 5348로 변환하지 않음

**1145**

924, 1051, 1275 코드 페이지로 전환하지 않습니다.

**영국 영어/게일어**

영국 영어/게일어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 648. 지원 플랫폼의 영국 영어/게일어에 대한 고유 CCSID	
플랫폼	고유 CCSID
 IBM i  z/OS	285, 924, 1146
 AIX	819, 923, 5348
 Windows	437, 850, 858, 1252, 5348
 Linux	819, 923
Apple 클라이언트	1275

모든 비클라이언트 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 전환을 지원하지 않으며, 다음과 같은 예외가 발생합니다.

**IBM i**

코드 페이지:

**285**

코드 페이지 858, 923, 924, 1275로 변환하지 않습니다.

**924**

코드 페이지 285, 437, 858, 1051, 1146, 1252, 1275, 5348로 변환하지 않음

**1146**

924, 1051, 1275 코드 페이지로 전환하지 않습니다.

**프랑스어**

프랑스어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.






표 649. 지원 플랫폼의 프랑스어에 대한 고유 CCSID	
플랫폼	고유 CCSID
 IBM i  z/OS	297, 924, 1147
 AIX	819, 923, 5348
 Windows	437, 850, 858, 1252, 5348
 Linux	819, 923

표 649. 지원 플랫폼의 프랑수어에 대한 고유 CCSID (계속)	
플랫폼	고유 CCSID
Apple 클라이언트	1275

모든 비클라이언트 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 전환을 지원하지 않으며, 다음과 같은 예외가 발생합니다.

## IBM i



코드 페이지:

### 297

코드 페이지 858, 923, 924, 1275, 5348로 변환하지 않음

### 924

코드 페이지 297, 437, 858, 1051, 1147, 1252, 1275, 5348로 변환하지 않음

### 1147

924, 1051, 1275 코드 페이지로 전환하지 않습니다.

## 다국어

다국어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 650. 지원 플랫폼의 다국어 변환에 대한 고유 CCSID	
플랫폼	고유 CCSID
IBM i z/OS	500, 924, 1148
AIX	819, 923, 5348
Windows	437, 850, 858, 1252, 5348
Linux	819, 923
Apple 클라이언트	1275

모든 비클라이언트 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 전환을 지원하지 않으며, 다음과 같은 예외가 발생합니다.

## IBM i



코드 페이지:

### 500

코드 페이지 858, 923으로 변환하지 않음

### 924

코드 페이지 437, 858, 1051, 1148, 1252, 1275, 5348로 변환하지 않음






### 1148

924, 1051, 1275 코드 페이지로 전환하지 않습니다.

## 포르투갈어

포르투갈어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 651. 지원 플랫폼의 포르투갈어에 대한 고유 CCSID

플랫폼	고유 CCSID
 IBM i	37, 500, 924, 1140
 z/OS IBM i	500, 924, 1140
 AIX	819, 923, 5348
 Windows	850, 858, 860, 1252, 5348
 Linux	819, 923
Apple 클라이언트	1275

모든 비클라이언트 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 전환을 지원하지 않으며, 다음과 같은 예외가 발생합니다.

### IBM i



코드 페이지:

#### 37

코드 페이지 858, 923, 924, 1275로 변환하지 않음

#### 500

코드 페이지 858, 923, 924, 1275로 변환하지 않음

#### 924

코드 페이지 858, 860, 1051, 1140, 1252, 1275, 5348로 변환하지 않음

#### 1140

코드 페이지 860, 924, 1051, 1275로 변환하지 않음

### Windows



코드 페이지:

#### 860

코드 페이지 1051, 1275로 변환하지 않음

### 아이슬란드어

아이슬란드어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 652. 지원 플랫폼의 아이슬란드어에 대한 고유 CCSID






플랫폼	고유 CCSID
 IBM i	871, 924, 1149
 z/OS	
 AIX	819, 923, 5348
 Windows	850, 858, 861, 1252, 5348
 Linux	819, 923

표 652. 지원 플랫폼의 아이슬란드어에 대한 고유 CCSID (계속)	
플랫폼	고유 CCSID
Apple 클라이언트	1275

모든 비클라이언트 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 전환을 지원하지 않으며, 다음과 같은 예외가 발생합니다.

## IBM i

### IBM i

코드 페이지:

#### 871

코드 페이지 858, 923, 924, 1275, 5348로 변환하지 않음

#### 924

코드 페이지 858, 861, 871, 1051, 1149, 1252, 1275, 5348로 변환하지 않음

#### 1149

924, 1051, 1275 코드 페이지로 전환하지 않습니다.

## Windows

### Windows






코드 페이지:

#### 861

코드 페이지 1051, 1275로 변환하지 않음

## 동유럽 언어

동유럽 언어에 대한 CCSID 및 CCSID 변환의 세부사항입니다. 이러한 CCSID를 사용하는 일반적 언어는 알바니아어, 크로아티아어, 체코어, 헝가리어, 폴란드어, 루마니아어, 세르비아어, 슬로바키아어 및 슬로베니아어를 포함합니다.

표 653. 지원 플랫폼의 동유럽 언어에 대한 고유 CCSID	
플랫폼	고유 CCSID
 IBM i  z/OS	870, 1153
 Windows	852, 1250, 5346, 9044
 AIX  Linux	912
동유럽 Apple 클라이언트	1282
루마니아 Apple 클라이언트	1285
크로아티아 Apple 클라이언트	1284

모든 비클라이언트 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 전환을 지원하지 않으며, 다음과 같은 예외가 발생합니다.



## z/OS



코드 페이지:

### 870

코드 페이지 1284, 1285로 변환하지 않음

### 1153

코드 페이지 1250, 1284, 1285로 변환하지 않음

## IBM i



코드 페이지:

### 870

코드 페이지 1284, 1285, 5346, 9044로 변환하지 않음

### 1153

코드 페이지 1282, 1284, 1285, 5346, 9044로 변환하지 않음

## , Linux



코드 페이지:

### 912

코드 페이지 1284, 1285로 변환하지 않음

## Windows



코드 페이지:

### 852

코드 페이지 1284, 1285로 변환하지 않음

### 1250

코드 페이지 1284, 1285로 변환하지 않음

### 9044

코드 페이지 912, 1282, 1284, 1285로 변환하지 않음

## 키릴 문자

키릴 문자에 대한 CCSID 및 CCSID 변환의 세부사항입니다. 이러한 CCSID를 사용하는 일반 언어는 벨로루시어, 불가리아어, 마케도니아어, 러시아어 및 세르비아어를 포함합니다.

표 654. 지원 플랫폼의 키릴 문자에 대한 고유 CCSID	
플랫폼	고유 CCSID
z/OS	1025
IBM i	880, 1025
Windows	855, 866, 1131, 1251, 5347
AIX	915
Linux	

표 654. 지원 플랫폼의 키릴 문자에 대한 고유 CCSID (계속)	
플랫폼	고유 CCSID
Apple 클라이언트	1283

모든 비클라이언트 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 전환을 지원하지 않으며, 다음과 같은 예외가 발생합니다.

## IBM i



코드 페이지:

### 880

코드 페이지 855, 866, 878, 1131, 5347로 변환하지 않음

### 1025

코드 페이지 878, 5347로 변환하지 않음

## Windows



코드 페이지:

### 855

코드 페이지 1131로 변환하지 않음

### 866

코드 페이지 1131로 변환하지 않음

### 1131

코드 페이지 855, 866, 880, 1283으로 변환하지 않음

## 에스토니아어

에스토니아어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 655. 지원 플랫폼의 에스토니아어에 대한 고유 CCSID	
플랫폼	고유 CCSID
IBM i z/OS	1122, 1157
Windows	902, 922, 1257, 5353, 9449
AIX Linux	902, 922

모든 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 변환을 지원하지 않으며 다음과 같은 예외가 발생합니다.

## z/OS



코드 페이지:

### 1122

코드 페이지 902, 1157, 9449로 변환하지 않음

## 1157

코드 페이지 922, 1122, 1257, 9449로 변환하지 않음

## IBM i



코드 페이지:

### 1122

코드 페이지 902, 5353, 9449로 변환하지 않음

### 1157

코드 페이지 922, 5353, 9449로 변환하지 않음

## Linux



코드 페이지:

### 902

코드 페이지 922, 1122, 9449로 변환하지 않음

### 922

코드 페이지 902, 1157, 9449로 변환하지 않음

## Windows



코드 페이지:

### 5353

코드 페이지 9449로 변환하지 않음

### 9449

코드 페이지 902, 922, 1122, 1157, 1257, 5353으로 변환하지 않음

### 902

코드 페이지 922, 1122, 9449로 변환하지 않음

## 라트비아어 및 리투아니아어

라트비아어 및 리투아니아어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 656. 지원 플랫폼의 라트비아어 및 리투아니아어에 대한 고유 CCSID	
플랫폼	고유 CCSID
IBM i z/OS	1112, 1156
Windows	901, 921, 1257, 5353, 9449
AIX Linux	901, 921

모든 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 변환을 지원하지 않으며 다음과 같은 예외가 발생합니다.

## z/OS



코드 페이지:

**1112**

코드 페이지 901, 1156, 9449로 변환되지 않음

**1156**

코드 페이지 901, 1156, 9449로 변환되지 않음

**IBM i**



코드 페이지:

**1112**

코드 페이지 5353으로 변환하지 않음

**1153**

코드 페이지 921, 5353, 9449로 변환되지 않음

**Linux**



코드 페이지:

**902**

코드 페이지 921, 1112, 1257, 9449로 변환하지 않음

**921**

코드 페이지 901, 1156, 9449로 변환되지 않음

**Windows**



코드 페이지:

**901**

코드 페이지 921, 1112, 1257, 9449로 변환하지 않음

**5355**

코드 페이지 9449로 변환하지 않음

**9449**

코드 페이지 901, 921, 1112, 1156, 1257로 변환하지 않음

**우크라이나어**

우크라이나어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 657. 지원 플랫폼의 우크라이나어에 대한 고유 CCSID	
플랫폼	고유 CCSID
IBM i z/OS	1123
Windows	1124, 1125, 1251, 5347
AIX Linux	1124

모든 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 변환을 지원하지 않으며 다음과 같은 예외가 발생 합니다.

## IBM i



코드 페이지:

### 1123

코드 페이지 5347로 변환하지 않음

## Windows



코드 페이지:

### 1125

코드 페이지 1123으로 변환하지 않음

## 그리스어

그리스어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 658. 지원 플랫폼의 그리스어에 대한 고유 CCSID	
플랫폼	고유 CCSID
IBM i	875
z/OS	
Windows	869, 1253, 5349
AIX	813
Linux	
NCR	
Apple 클라이언트	1280
DOS 클라이언트	737

모든 비클라이언트 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 변환을 지원하지 않으며 다음과 같은 예외가 발생합니다.

## IBM i



코드 페이지:

### 875

코드 페이지 5349로 변환하지 않음

## Windows



코드 페이지:

### 1253






코드 페이지 737로 변환하지 않음

### 5349

코드 페이지 737로 변환하지 않음

## 터키어

터키어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 659. 지원 플랫폼의 터키어에 대한 고유 CCSID	
플랫폼	고유 CCSID
 IBM i	1026
 z/OS	
 Windows	857, 1254, 5350
 AIX	920
 Linux	
Apple 클라이언트	1281

모든 비클라이언트 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 전환을 지원하지 않으며, 다음과 같은 예외가 발생합니다.

## IBM i








코드 페이지:

### 1026

코드 페이지 5350으로 변환하지 않음

## 히브리어

히브리어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 660. 지원 플랫폼의 히브리어에 대한 고유 CCSID	
플랫폼	고유 CCSID
 z/OS	424, 803, 4899, 12712
 IBM i	424
 AIX	916, 9048
 Windows	1255, 5351
 Linux	916

모든 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 변환을 지원하지 않으며 다음과 같은 예외가 발생합니다.

## z/OS



코드 페이지:

### 424

코드 페이지 867, 4899, 9048, 12712로 변환하지 않음

### 803

코드 페이지 867, 4899, 5351, 9048, 12712로 변환하지 않음

## 4899

코드 페이지 424, 803, 856, 862, 916, 1255로 변환하지 않음

## 12712

코드 페이지 424, 803, 856, 916, 1255로 변환하지 않음

## IBM i



코드 페이지:

### 424

코드 페이지 803, 867, 4899, 5351, 9048, 12712로 변환하지 않음

코드 페이지 424 또한 CCSID 4952에서 변환되며 856의 변형입니다.

## AIX



코드 페이지:

### 916

코드 페이지 867, 4899, 9048, 12712로 변환하지 않음

### 9048

코드 페이지 424, 803, 856, 862, 916, 1255로 변환하지 않음

## Windows



코드 페이지:

### 1255

코드 페이지 867, 4899, 9048, 12712로 변환하지 않음

### 5351

코드 페이지 803으로 변환하지 않음

## 아랍어

아랍어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 661. 지원 플랫폼의 아랍어에 대한 고유 CCSID	
플랫폼	고유 CCSID
IBM i z/OS	420
AIX	1046, 1089
	1089(참고 참조)
Windows	720, 864, 1256, 5352
Linux	1089

모든 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 변환을 지원하지 않으며 다음과 같은 예외가 발생합니다.

## IBM i



코드 페이지:

**420**

코드 페이지 5352로 변환하지 않음

## Linux, Tru64



코드 페이지:

**1089**

코드 페이지 720으로 변환하지 않음

## Windows



코드 페이지:

**720**

코드 페이지 1089, 5352로 변환하지 않음

**5352**

코드 페이지 720으로 변환하지 않음

## 페르시아어

페르시아어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 662. 지원 플랫폼의 페르시아어에 대한 고유 CCSID	
플랫폼	고유 CCSID
IBM i z/OS	1097
AIX Linux Windows	1098(참고 참조)

**참고:** 이러한 플랫폼에 대한 고유 CCSID는 표준화되지 않고 변경될 수 있습니다.

모든 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 변환을 지원하지 않습니다.



## 우르두어

우르두어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 663. 지원 플랫폼의 우르두어에 대한 고유 CCSID	
플랫폼	고유 CCSID
IBM i z/OS	918
Windows	868



표 663. 지원 플랫폼의 우르두어에 대한 고유 CCSID (계속)

플랫폼	고유 CCSID
 AIX  Linux	1006

모든 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 변환을 지원하지 않으며 다음과 같은 예외가 발생합니다.

## IBM i

 IBM i

코드 페이지:






### 918

코드 페이지 1006으로 변환하지 않음

## 태국어

태국어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 664. 지원 플랫폼의 태국어에 대한 고유 CCSID

플랫폼	고유 CCSID
 IBM i  z/OS	838
 AIX  Linux  Windows	874(참고 참조)






**참고:** 이러한 플랫폼에 대한 고유 CCSID는 표준화되지 않고 변경될 수 있습니다.

모든 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 변환을 지원하지 않습니다.

## 라오스어

라오스어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.






표 665. 지원 플랫폼의 라오스어에 대한 고유 CCSID

플랫폼	고유 CCSID
 IBM i  z/OS	1132
 AIX  Linux  Windows	1133

모든 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 변환을 지원하지 않습니다.

## 베트남어

베트남어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

플랫폼	고유 CCSID
 IBM i  z/OS	1130
 Windows	1258, 5354
 AIX  Linux	1129

모든 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 변환을 지원하지 않으며 다음과 같은 예외가 발생합니다.

## IBM i








코드 페이지:

### 1130



코드 페이지 1129, 5354로 변환하지 않음

## 일본어 라틴 SBCS

일본어 라틴 SBCS에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

플랫폼	고유 CCSID
 IBM i  z/OS	1027
 AIX	932, 5050, 33722(참고 1 참조)
 Windows	932, 943(참고 2 참조)
 Linux	943, 5050

## 참고:

-  5050 및 33722은 AIX의 기본 코드 페이지 954와 관련된 CCSID입니다. 운영 체제가 보고하는 CCSID는 33722입니다.
-  Windows NT에서는 코드 페이지 932를 사용하지만, 이는 943의 CCSID에 의해 최상으로 표시됩니다. 그러나 IBM MQ의 모든 플랫폼이 CCSID를 지원하지는 않습니다.

IBM MQ for Windows CCSID 932는 코드 페이지 932를 표시하는 데 사용되지만, ../conv/table/ccsid.tbl 파일을 변경하여 943에 사용된 CCSID를 변경할 수 있습니다.

모든 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 변환을 지원하지 않으며 다음과 같은 예외가 발생합니다.

## z/OS



코드 페이지:

### 1027

코드 페이지 932, 942, 943, 954, 5050, 33722로 변환되지 않음

## IBM i



코드 페이지:

### 1027

코드 페이지 932로 변환하지 않음

## AIX



코드 페이지:

### 932

코드 페이지 1027로 변환하지 않습니다.

### 5050

코드 페이지 1027로 변환하지 않습니다.

### 33722

코드 페이지 1027로 변환하지 않습니다.

## Linux



코드 페이지:

### 943

코드 페이지 1027로 변환하지 않습니다.

### 5050

코드 페이지 1027로 변환하지 않습니다.

## 일본 카타카나 SBCS

일본 카타카나 SBCS에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 668. 지원 플랫폼의 일본 카타카나 SBCS에 대한 고유 CCSID	
플랫폼	고유 CCSID
IBM i z/OS	290
AIX	932, 5050, 33722(참고 1 참조)
Windows	932, 943(참고 2 참조)
Linux	943, 5050

참고:

1. **AIX** 5050 및 33722은 AIX의 기본 코드 페이지 954와 관련된 CCSID입니다. 운영 체제가 보고하는 CCSID는 33722입니다.
2. **Windows** Windows NT에서는 코드 페이지 932를 사용하지만, 이는 943의 CCSID에 의해 최상으로 표시됩니다. 그러나 IBM MQ의 모든 플랫폼이 CCSID를 지원하지는 않습니다.

IBM MQ for Windows CCSID 932는 코드 페이지 932를 표시하는 데 사용되지만, ../conv/table/ccsid.tbl 파일을 변경하여 943에 사용된 CCSID를 변경할 수 있습니다.

3. IBM MQ는 다음 플랫폼에서 이전 변환을 비롯하여 CCSID 897에서 CCSIDs 37, 273, 277, 278, 280, 284, 285, 290, 297, 437, 500, 819, 850, 1027 및 1252로의 변환을 지원합니다.

- **AIX** AIX
- **Linux** Linux

모든 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 변환을 지원하지 않으며 다음과 같은 예외가 발생합니다.

## z/OS

### z/OS

코드 페이지:

#### 290

코드 페이지 932, 943, 954, 5050, 33722로 변환하지 않음

## IBM i

### IBM i

코드 페이지:

#### 290

코드 페이지 932로 변환하지 않음

## AIX

### AIX

코드 페이지:

#### 932

코드 페이지 290, 897로 변환하지 않습니다.

#### 5050

코드 페이지 290, 897로 변환하지 않습니다.

#### 33722

코드 페이지 290, 897로 변환하지 않습니다.

## Linux

### Linux

코드 페이지:

#### 943

코드 페이지 290, 897로 변환하지 않습니다.






#### 5050

코드 페이지 290, 897로 변환하지 않습니다.





## 일본어 간지/라틴 혼합

일본어 간지/라틴 혼합에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 669. 지원 플랫폼의 일본어 간지/라틴 혼합에 대한 고유 CCSID

플랫폼	고유 CCSID
 IBM i  z/OS	1399, 5035(참고 1 참조)
 AIX	932, 5050, 33722(참고 2 참조)
 Windows	932, 943(참고 4 참조)
 Linux	943, 5050

**참고:**

-   5035는 코드 페이지 939와 관련된 CCSID입니다.
-  5050 및 33722은 AIX의 기본 코드 페이지 954와 관련된 CCSID입니다. 운영 체제가 보고하는 CCSID는 33722입니다.
-  Windows NT에서는 코드 페이지 932를 사용하지만, 이는 943의 CCSID에 의해 최상으로 표시됩니다. 그러나 IBM MQ의 모든 플랫폼이 CCSID를 지원하지는 않습니다.

IBM MQ for Windows CCSID 932는 코드 페이지 932를 표시하는 데 사용되지만, ../conv/table/ccsid.tbl 파일을 변경하여 943에 사용된 CCSID를 변경할 수 있습니다.

모든 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 변환을 지원하지 않으며 다음과 같은 예외가 발생합니다.

**z/OS**



코드 페이지:

**1399**

코드 페이지 954, 5035, 5050, 33722로 변환하지 않음

**5035**

코드 페이지 954, 1399, 5050, 33722로 변환하지 않음

**IBM i**



코드 페이지:

**1399**

코드 페이지 5039로 변환하지 않음

**5035**

코드 페이지 5039로 변환하지 않음

**일본어 간지/카타카나 혼합**

일본어 간지/카타카나 혼합에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 670. 지원 플랫폼의 일본어 간지/카타카나 혼합에 대한 고유 CCSID








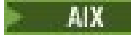

플랫폼	고유 CCSID
 z/OS	1390, 5026 (참고 922 페이지의 『1』 참조)

표 670. 지원 플랫폼의 일본어 간지/카타카나 혼합에 대한 고유 CCSID (계속)	
플랫폼	고유 CCSID
 IBM i	5026 (주 922 페이지의 『1』 참조)
 AIX	932, 5050, 33722 (참고 922 페이지의 『2』 참조)
 Windows	932, 943 (참고 922 페이지의 『3』 참조)
 Linux	943, 5050

**참고:**

-   EBCDIC에서 CCSID 1390 및 5026의 1바이트모드는 기본 라틴어의 일반 또는 불변 레이아웃과 다른 위치에 소문자를 포함합니다. 따라서 메시지 데이터가 다른 CCSID로 변환될 때 데이터가 유실되지 않도록 주의해야 합니다. 또한 이러한 CCSID를 큐 관리자의 기본 CCSID로 사용하면 다른 큐 관리자와 통신할 때 문제가 발생할 수 있습니다. 예를 들어, 소문자를 사용하는 채널 이름은 원격 시스템에서 올바르게 해석되지 않을 수 있습니다. 5026은 코드 페이지 930과 관련된 CCSID입니다. CCSID 5026은 일본어 가타카나(DBCS) 기능이 선택될 때 IBM i에서 보고된 CCSID입니다.
-  5050 및 33722은 AIX의 기본 코드 페이지 954와 관련된 CCSID입니다. 운영 체제가 보고하는 CCSID는 33722입니다.
-  Windows NT에서는 코드 페이지 932를 사용하지만, 이는 943의 CCSID에 의해 최상으로 표시됩니다. 그러나 IBM MQ의 모든 플랫폼이 CCSID를 지원하지는 않습니다.

Windows에서 CCSID 932는 코드 페이지 932를 나타내는 데 사용되지만, `../conv/table/ccsid.tbl` 파일을 변경하여 943에 사용되는 CCSID를 변경할 수 있습니다.

모든 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 변환을 지원하지 않으며 다음과 같은 예외가 발생합니다.

**z/OS**



코드 페이지:

**1390**

코드 페이지 954, 5026, 5050, 33722로 변환하지 않음  
소문자를 허용하지 않습니다.

**5026**

코드 페이지 954, 1390, 5050, 33722로 변환하지 않음

**IBM i**



코드 페이지:






**5026**

코드 페이지 1390, 5039으로 변환하지 않음

**한국어**

한국어에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 671. 지원 플랫폼의 한국어에 대한 고유 CCSID

플랫폼	고유 CCSID
 IBM i  z/OS	933, 1364
 AIX  Linux	970
 Windows	949, 1363

모든 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 변환을 지원하지 않으며 다음과 같은 예외가 발생합니다.

### z/OS



코드 페이지:

#### 933

코드 페이지 970으로 변환하지 않음






#### 1364

코드 페이지 970으로 변환하지 않음


### 중국어

중국어 간체자에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 672. 지원 플랫폼의 중국어에 대한 고유 CCSID




플랫폼	고유 CCSID
 z/OS	935, 1388
 IBM i	935, 1388
 AIX	1383, 1386
 Windows	1381, 1386(참고 2 참조)
 Linux	1383

### 참고:

-  Windows에서는 코드 페이지 936를 사용하지만 이는 1386의 CCSID에 의해 최상으로 표시됩니다. 그러나 IBM MQ의 모든 플랫폼이 CCSID를 지원하지는 않습니다.

IBM MQ for Windows에서 CCSID 1381은 코드 페이지 936을 나타내기 위해 사용되지만 사용된 CCSID를 1386으로 변경하는 ../conv/table/ccsid.tbl 파일로 변경될 수 있습니다.

- IBM MQ에서는 중국 GB18030 표준을 지원합니다.

   z/OS, Windows 및 Linux에서, 변환 지원은 Unicode (UTF-8 및 UTF-16) 및 CCSID 1388(GB18030 확장자를 가진 EBCDIC), Unicode(UTF-8 및 UTF-16) 및 CCSID 5488(GB18030) 및 CCSID 1388 및 CCSID 5488 사이에 제공됩니다.

**참고:** GB18030 문자를 사용하려면 CCSID를 5488로 설정해야 합니다. 그러나 IBM MQ Explorer 또는 IBM MQ Console로 작성된 큐 관리자에서는 CCSID를 설정할 수 없습니다. 대신, CCSID가 5488인 CLI를 사용하여 큐 관리자를 작성하거나 큐 관리자를 작성한 후 CLI 명령행을 사용하여 CCSID를 변경해야 합니다.

**IBM i** IBM i에서 지원은 Unicode(UTF-8 및 UTF-16)와 CCSID 1388(GB18030 확장자를 가진 EBCDIC) 사이에 변환을 위한 운영 체제에 의해 제공됩니다.

모든 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 변환을 지원하지 않으며 다음과 같은 예외가 발생합니다.

## z/OS



코드 페이지:

### 935

코드 페이지 1383으로 변환하지 않음

### 1388

코드 페이지 1383으로 변환하지 않음

## 대만어

중국어 번체자에 대한 CCSID 및 CCSID 변환의 세부사항입니다.

표 673. 지원 플랫폼의 대만어에 대한 고유 CCSID	
플랫폼	고유 CCSID
IBM i z/OS	937
Windows	950
AIX Linux	950, 964

모든 플랫폼에서 고유 CCSID와 다른 플랫폼의 고유 CCSID 간 변환을 지원하지 않으며 다음과 같은 예외가 발생합니다.

## z/OS



코드 페이지:

### 937

코드 페이지 964로 변환하지 않음

### 1388

코드 페이지 1383으로 변환하지 않음

## Linux



코드 페이지:

### 964

코드 페이지 938로 변환하지 않음



**z/OS z/OS conversion support**

A list of supported CCSID conversions.

*Table 674. IBM MQ for z/OS CCSID conversion support*

<b>CCSID</b>	<b>Converts to and from CCSIDS</b>
37	256, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 720, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-866, 869-871, 874-875, 880, 897, 903-905, 912, 914-916, 920-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1097, 1100, 1112, 1114-1115, 1122, 1124, 1126, 1130-1132, 1137, 1140-1149, 1200, 1208, 1250-1255, 1257-1258, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5210-5211, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25480, 25617, 25619, 25664, 28709
256	37, 273, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 737, 775, 819, 833, 836, 838, 850, 852, 857, 860-866, 869-871, 875, 880, 905, 1025-1027, 1112, 1122, 1200, 1208, 1251-1252, 1275, 4386, 4929, 4932, 4934, 4946, 4948, 4953, 4960, 4971, 5123, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 13121, 13488, 16804, 17248, 17584, 28709
259	437, 808, 850-852, 855-858, 860-865, 867, 869, 872, 874, 899, 901-902, 915, 1098, 1161-1162, 1200, 1208, 1250-1258, 4946, 4948, 4951-4953, 4960, 4970, 5346, 5348, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584
273	37, 256, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1250, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951-4953, 4960, 4970-4971, 5012, 5123, 5346, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
274	500, 1047
275	37, 437, 500, 819, 850, 1047, 1200, 1208, 1252, 4946, 5348, 8229, 13488, 17584, 28709
277	37, 256, 273, 278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
278	37, 256, 273, 277, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
280	37, 256, 273, 277-278, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
281	1047
282	500, 1047, 1200, 1208, 13488, 17584
284	37, 256, 273, 277-278, 280, 285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
285	37, 256, 273, 277-278, 280, 284, 290, 297, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
290	37, 256, 273, 277-278, 280, 284-285, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 895-897, 1009, 1025-1027, 1040-1043, 1047, 1088, 1112, 1122, 1139, 1200, 1208, 1252, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4960, 4992, 5123, 8229, 8482, 9025, 9044, 9049, 9056, 13121, 13488, 17248, 17584, 25473, 25617, 25619, 25664, 28709
293	1200, 1208, 13488, 17584
297	37, 256, 273, 277-278, 280, 284-285, 290, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
300	301, 941, 1200, 1208, 1351, 4396, 8492, 13488, 16684, 17584

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
301	300, 941, 1200, 1208, 1351, 4396, 8492, 13488, 16684, 17584
367	37, 256, 273, 277-278, 280, 284, 290, 297, 500, 819, 833, 836, 850, 871, 875, 1009, 1026-1027, 1041, 1088, 1115, 1126, 1200, 1208, 4386, 4929, 4932, 4946, 4971, 5123, 5211, 8229, 8482, 9025, 13121, 13488, 17584, 25617, 25664, 28709
420	37, 256, 424, 437, 500, 720, 737, 775, 819, 850, 852, 857, 860-865, 1008, 1046, 1089, 1098, 1112, 1122, 1127, 1200, 1208, 1252, 1256, 4946, 4948, 4953, 4960, 5104, 5142, 5352, 8229, 8612, 9044, 9049, 9056, 9238, 13488, 16804, 17248, 17584, 28709
423	37, 256, 273, 277-278, 280, 284-285, 297, 437, 500, 737, 775, 813, 819, 838, 850-852, 857, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1009, 1025-1027, 1041-1043, 1112, 1122, 1200, 1208, 1252-1253, 1280, 4909, 4934, 4946, 4948, 4953, 4960, 4970-4971, 5012, 5123, 8229, 9030, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 28709
424	37, 256, 420, 437, 500, 737, 775, 803, 819, 836, 850, 852, 856-857, 860-865, 916, 1112, 1122, 1200, 1208, 1252, 1255, 4932, 4946, 4948, 4952-4953, 4960, 5012, 5351, 8229, 8612, 9044, 9049, 9056, 13488, 16804, 17248, 17584, 28709
437	37, 256, 259, 273, 275, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-863, 865-866, 869-871, 874-875, 880, 897, 903, 905, 912, 914-916, 920-924, 1025-1027, 1040-1043, 1047, 1051, 1097, 1098, 1114-1115, 1126, 1140-1149, 1200, 1208, 1252, 1257, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5210-5211, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 28709
500	37, 256, 273-275, 277-278, 280, 282, 284-285, 290, 297, 367, 420, 423-424, 437, 737, 775, 813, 819, 833, 836, 838, 850-852, 855-858, 860-866, 869-871, 874-875, 880, 891, 895, 897, 903-905, 912, 914-916, 920-924, 1004, 1009-1021, 1023, 1025-1027, 1040-1043, 1046-1047, 1051, 1088-1089, 1097, 1100-1107, 1112, 1114-1115, 1122, 1124-1126, 1129-1133, 1137, 1140-1149, 1200, 1208, 1250-1258, 1275, 1280-1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951-4953, 4960, 4970-4971, 5012, 5123, 5142, 5210-5211, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 9238, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25480, 25617, 25619, 25664, 28709
720	37, 420, 864, 1200, 1208, 1256, 4960, 8229, 8612, 9056, 13488, 16804, 17248, 17584, 28709
737	37, 256, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 833, 836, 838, 850, 869-871, 875, 880, 905, 1025-1027, 1097, 1200, 1208, 1252-1253, 1280, 4386, 4909, 4929, 4932, 4934, 4946, 4971, 5123, 8229, 8482, 8612, 9025, 9030, 9061, 13121, 13488, 16804, 17584, 28709
775	37, 256, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 833, 836, 838, 850, 870-871, 875, 880, 905, 1025-1027, 1097, 1112, 1122, 1200, 1208, 1252, 1257, 4386, 4929, 4932, 4934, 4946, 4971, 5123, 8229, 8482, 8612, 9025, 9030, 13121, 13488, 16804, 17584, 28709

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
803	424, 819, 850, 856, 862, 916, 1200, 1208, 1252, 1255, 4946, 4952, 5012, 13488, 17584
806	1200, 1208, 13488, 17584
808	259, 858-859, 872, 923-924, 1140, 1148, 1153-1154, 1200, 1208, 5347, 5348, 13488, 17584
813	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252-1253, 1280, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5349, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
819	37, 256, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 803, 813, 833, 836, 838, 850, 852, 855, 857-858, 860-861, 863-866, 869-871, 874-875, 880, 897, 903, 905, 912, 914-916, 920-924, 1004, 1025-1027, 1041-1043, 1047, 1051, 1088-1089, 1097, 1098, 1112, 1114, 1122-1123, 1126, 1130, 1132, 1137, 1140-1149, 1200, 1208, 1250-1255, 1257-1258, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5210, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
833	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 836, 850, 852, 855, 857, 860-865, 870-871, 891, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1126, 1200, 1208, 1252, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4960, 5123, 8229, 8482, 9025, 9044, 9049, 9056, 13121, 13488, 17248, 17584, 25617, 25619, 25664, 28709
834	926, 951, 1200, 1208, 1362, 4930, 9026, 13488, 17584
835	927, 947, 1200, 1208, 4931, 9027, 13488, 17584, 21427
836	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 424, 437, 500, 737, 775, 819, 833, 850, 852, 855, 857, 870-871, 875, 903, 1009, 1025-1027, 1040-1043, 1088, 1112, 1114-1115, 1122, 1200, 1208, 1252, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4971, 5123, 5210-5211, 8229, 8482, 9025, 9044, 9049, 13121, 13488, 17584, 25479, 25617, 25619, 25664, 28709
837	928, 1200, 1208, 1380, 1385, 4933, 13488, 17584
838	37, 256, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 775, 813, 819, 850, 852, 857, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1112, 1122, 1200, 1208, 1252, 4909, 4934, 4946, 4948, 4953, 4960, 4970-4971, 5012, 5123, 8229, 9030, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 28709
848	924, 1148, 1158, 1200, 1208, 5347, 13488, 17584
849	924, 1148, 1154, 1200, 1208, 5347, 13488, 17584

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
850	37, 256, 259, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 737, 775, 803, 813, 819, 833, 836, 838, 852, 855-858, 860-866, 869-871, 874-875, 880, 897, 903, 905, 912, 914-916, 920-924, 1004, 1025-1027, 1040-1043, 1047, 1051, 1088-1089, 1097, 1098, 1100, 1112, 1114, 1122, 1126, 1130, 1132, 1140-1149, 1200, 1208, 1250-1257, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951-4953, 4960, 4970-4971, 5012, 5123, 5210, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
851	259, 423, 500, 875, 1200, 1208, 4971, 13488, 17584
852	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1040-1043, 1047, 1088, 1097, 1200, 1208, 1250, 1252, 1282, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5346, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 25664, 28709
855	37, 259, 273, 277-278, 280, 284-285, 290, 297, 437, 500, 819, 833, 836, 850, 852, 857, 866, 870-871, 878, 880, 912, 915, 1025-1027, 1040-1043, 1088, 1200, 1208, 1250-1252, 1283, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 5123, 5346, 5347, 8229, 8482, 9025, 9044, 9049, 13121, 13488, 17584, 25617, 25619, 25664, 28709
856	259, 273, 424, 500, 803, 850, 862, 916, 1200, 1208, 1255, 4946, 4952, 5012, 5351, 13488, 17584
857	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1040-1043, 1088, 1097, 1200, 1208, 1252, 1254, 1281, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5350, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 25664, 28709
858	37, 259, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 850, 860-861, 865, 871-872, 901-902, 923-924, 1047, 1051, 1140-1149, 1153-1157, 1160-1162, 1164, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
859	808, 872, 901-902, 1153-1157, 1160-1162, 1164, 1200, 1208, 13488, 17584
860	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 838, 850, 852, 857-858, 861, 863, 865, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 923-924, 1025-1027, 1041-1043, 1097, 1140, 1145-1146, 1148, 1200, 1208, 1252, 4386, 4909, 4929, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 28709

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
861	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 838, 850, 852, 857-858, 860, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 923-924, 1025-1027, 1041-1043, 1097, 1148, 1149, 1200, 1208, 1252, 4386, 4909, 4929, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 28709
862	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 803, 833, 838, 850, 856, 870-871, 875, 880, 905, 916, 1025-1027, 1097, 1200, 1208, 1252, 1255, 4386, 4929, 4934, 4946, 4952, 4971, 5012, 5123, 5351, 8229, 8482, 8612, 9025, 9030, 12712, 13121, 13488, 16804, 17584, 28709
863	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 838, 850, 852, 857, 860-861, 865, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1041-1043, 1051, 1097, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 28709
864	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 720, 819, 833, 838, 850, 870-871, 875, 880, 905, 918, 1008, 1025-1027, 1046, 1089, 1097, 1127, 1200, 1208, 1252, 1256, 4386, 4929, 4934, 4946, 4960, 4971, 5104, 5123, 5142, 5352, 8229, 8482, 8612, 9025, 9030, 9056, 9238, 13121, 13488, 16804, 17248, 17584, 28709
865	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 819, 833, 838, 850, 858, 860, 863, 870-871, 875, 880, 905, 923-924, 1025-1027, 1097, 1142-1143, 1148, 1200, 1208, 1252, 4386, 4929, 4934, 4946, 4971, 5123, 5348, 8229, 8482, 8612, 9025, 9030, 13121, 13488, 16804, 17584, 28709
866	37, 256, 437, 500, 819, 850, 855, 870, 878, 880, 915, 1025, 1200, 1208, 1251-1252, 1283, 4946, 4951, 5347, 8229, 13488, 17584, 28709
867	259, 1153-1155, 1160, 1200, 1208, 4899, 5351, 9048, 12712, 13488, 17584
868	918, 1006, 1200, 1208, 13488, 17584
869	37, 256, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 813, 819, 838, 850, 852, 857, 860-861, 863, 870-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252-1254, 1280, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5349, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
870	37, 256, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-866, 869, 871, 874-875, 880, 897, 903, 912, 915-916, 920, 1009, 1025-1027, 1040-1043, 1047, 1088, 1112, 1122, 1200, 1208, 1250, 1252, 1282, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5346, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
871	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869, 870, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
872	259, 808, 858-859, 923-924, 1140-1149, 1153-1155, 1200, 1208, 5347, 5348, 13488, 17584
874	37, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
875	37, 256, 273, 277-278, 280, 284-285, 297, 367, 423, 437, 500, 737, 775, 813, 819, 836, 838, 850-852, 857, 860-865, 869-871, 874, 880, 897, 903, 912, 916, 920, 1009, 1025-1027, 1041-1043, 1047, 1088, 1112, 1122, 1200, 1208, 1252-1253, 1280, 4909, 4932, 4934, 4946, 4948, 4953, 4960, 4970-4971, 5012, 5123, 5349, 8229, 9030, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
878	855, 866, 880, 915, 1025, 1131, 1200, 1208, 1251, 1283, 4951, 5347, 13488, 17584
880	37, 256, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 775, 813, 819, 838, 850, 852, 855, 857, 860-866, 869-871, 874-875, 878, 897, 903, 912, 915-916, 920, 1009, 1025-1027, 1041-1043, 1112, 1122, 1200, 1208, 1251-1252, 1283, 4909, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5347, 8229, 9030, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 28709
891	500, 833, 1088, 1200, 1208, 4929, 9025, 13121, 13488, 17584, 25664
895	290, 500, 1027, 1041, 1200, 1208, 4386, 5123, 8482, 13488, 17584, 25617
896	290, 1027, 1041, 1200, 1208, 4386, 4992, 5123, 8482, 13488, 17584, 25617
897	37, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 4386, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 8229, 8482, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
899	259
901	259, 858-859, 902, 923-924, 1140, 1148, 1156-1157, 1200, 1208, 5348, 5353, 13488, 17584
902	259, 858-859, 901, 923-924, 1140, 1148, 1156-1157, 1200, 1208, 5348, 5353, 13488, 17584

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
903	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 836, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 912, 916, 920, 1025-1027, 1041-1043, 1115, 1200, 1208, 1252, 4909, 4932, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5211, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
904	37, 500, 1114, 1200, 1208, 5210, 8229, 13488, 17584, 25480, 28709
905	37, 256, 437, 500, 737, 775, 819, 850, 852, 857, 860-865, 920, 1026, 1112, 1122, 1200, 1208, 1252, 1254, 1281, 4946, 4948, 4953, 4960, 8229, 9044, 9049, 9056, 13488, 17248, 17584, 28709
912	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 916, 920, 1025-1027, 1041-1043, 1047, 1200, 1208, 1250, 1252, 1282, 4909, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5346, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
914	37, 437, 500, 819, 850, 1200, 1208, 1252, 1257, 4946, 8229, 13488, 17584, 28709
915	37, 259, 437, 500, 819, 850, 855, 866, 870, 878, 880, 1025, 1131, 1200, 1208, 1251-1252, 1283, 4946, 4951, 5347, 8229, 13488, 17584, 28709
916	37, 273, 277-278, 280, 284-285, 297, 423-424, 437, 500, 803, 813, 819, 838, 850, 852, 856-857, 860-863, 869-871, 874-875, 880, 897, 903, 912, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 1255, 4909, 4934, 4946, 4948, 4952-4953, 4970-4971, 5012, 5123, 5351, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
918	864, 868, 1006, 1200, 1208, 4960, 9056, 13488, 17248, 17584
920	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 1025-1026, 1200, 1208, 1252, 1254, 1281, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5350, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 28709
921	37, 437, 500, 819, 850, 922, 1112, 1122, 1200, 1208, 1252, 1257, 4946, 5353, 8229, 13488, 17584, 28709
922	37, 437, 500, 819, 850, 921, 1112, 1122, 1200, 1208, 1252, 1257, 4946, 5353, 8229, 13488, 17584, 28709
923	37, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 850, 858, 860-861, 865, 871-872, 901-902, 924, 1047, 1051, 1140-1149, 1153-1158, 1160-1162, 1164, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
924	37, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 848-850, 858, 860-861, 865, 871-872, 901-902, 923, 1047, 1051, 1140-1149, 1153-1157, 1160-1164, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
926	834, 951, 9026
927	835, 947, 1200, 1208, 4931, 9027, 13488, 17584, 21427
928	837, 1200, 1208, 1380, 13488, 17584



Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
930	931-932, 939, 942-943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
931	930, 932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
932	930-931, 939, 942-943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
933	934, 944, 949, 1200, 1208, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13488, 13651, 17317, 17584, 25510, 25520, 25525, 29616, 29621, 33717, 37813
934	933, 949, 5029, 5045, 5460, 9125, 13221, 17317, 25510, 25525, 29621, 33717, 37813
935	936, 946, 1200, 1208, 1381, 1386, 1388, 5031, 5477, 5482, 5484, 9127, 13223, 13488, 17584, 25512
936	935, 946, 1381, 5031, 5477, 5484, 9127, 13223, 25512
937	938, 948, 950, 1200, 1208, 1370, 5033, 5046, 9142, 13488, 17584, 25514, 25524, 29620
938	937, 950, 1370, 5033, 5046, 9142, 25514
939	930-932, 942-943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
941	300-301, 1200, 1208, 1351, 4396, 8492, 13488, 16684, 17584
942	930-932, 939, 943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
943	930-932, 939, 942, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
944	933, 949, 1200, 1208, 5029, 5045, 5460, 9125, 13221, 13488, 17317, 17584, 25520, 25525, 29616, 29621, 33717, 37813
946	935-936, 1200, 1208, 5031, 5484, 9127, 13223, 13488, 17584, 25512
947	835, 927, 1200, 1208, 4931, 9027, 13488, 17584, 21427
948	937, 950, 1200, 1208, 1370, 5033, 5046, 9142, 13488, 17584, 25524, 29620
949	933-934, 944, 1200, 1208, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13488, 13651, 17317, 17584, 25510, 25520, 25525, 29616, 29621, 33717, 37813
950	937-938, 948, 1200, 1208, 1370, 5033, 5046, 9142, 13488, 17584, 25514, 25524, 29620
951	834, 926, 1200, 1208, 1362, 4930, 9026, 13488, 17584

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
1004	500, 819, 850, 1200, 1208, 4946, 13488, 17584
1006	868, 918, 1200, 1208, 13488, 17584
1008	420, 864, 1200, 1208, 4960, 5104, 8612, 9056, 13488, 16804, 17248, 17584
1009	37, 273, 277-278, 280, 284, 290, 297, 367, 423, 500, 833, 836, 870-871, 875, 880, 1025-1026, 1200, 1208, 4386, 4929, 4932, 4971, 8229, 8482, 9025, 13121, 13488, 17584, 28709
1010	500, 1200, 1208, 13488, 17584
1011	500, 1200, 1208, 13488, 17584
1012	500, 1200, 1208, 13488, 17584
1013	500, 1140, 1200, 1208, 13488, 17584
1014	500, 1200, 1208, 13488, 17584
1015	500, 1200, 1208, 13488, 17584
1016	500, 1200, 1208, 13488, 17584
1017	500, 1200, 1208, 13488, 17584
1018	500, 1200, 1208, 13488, 17584
1019	500, 1200, 1208, 13488, 17584
1020	500
1021	500
1023	500
1025	37, 256, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-866, 869-871, 874-875, 878, 880, 897, 903, 912, 915-916, 920, 1009, 1026-1027, 1040-1043, 1051, 1088, 1112, 1122, 1131, 1200, 1208, 1251-1252, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5347, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
1026	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-865, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1009, 1025, 1027, 1040-1043, 1047, 1088, 1112, 1122, 1200, 1208, 1252, 1254, 1281, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5350, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
1027	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-865, 869-871, 874-875, 880, 895-897, 903, 912, 916, 1025-1026, 1040-1043, 1047, 1088, 1112, 1122, 1139, 1200, 1208, 1252, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 4992, 5012, 5123, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
1040	37, 273, 277-278, 280, 284-285, 290, 297, 437, 500, 833, 836, 850, 852, 855, 857, 870-871, 1025-1027, 1041-1043, 1088, 1200, 1208, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 5123, 8229, 8482, 9025, 9044, 9049, 13121, 13488, 17584, 25617, 25619, 25664, 28709
1041	37, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 895-897, 903, 912, 916, 1025-1027, 1040, 1042-1043, 1088, 1200, 1208, 1252, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 4992, 5012, 5123, 8229, 8482, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 17584, 25473, 25479, 25617, 25619, 25664, 28709
1042	37, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 1025-1027, 1040, 1041, 1043, 1088, 1200, 1208, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 8229, 8482, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 17584, 25473, 25479, 25617, 25619, 25664, 28709
1043	37, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 1025-1027, 1040, 1041, 1042, 1088, 1114, 1200, 1208, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5210, 8229, 8482, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 17584, 25473, 25479, 25617, 25619, 25664, 28709
1046	420, 500, 864, 1089, 1127, 1200, 1208, 1256, 4960, 5142, 5352, 8612, 9056, 9238, 13488, 16804, 17248, 17584
1047	37, 273-275, 277-278, 280, 281, 282, 284-285, 290, 297, 437, 500, 819, 850, 852, 858, 870-871, 875, 912, 923-924, 1026-1027, 1140-1149, 1200, 1208, 1252, 1254, 4946, 4948, 5123, 8229, 8482, 13488, 17584, 28709
1051	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 871, 923-924, 1025, 1097, 1140-1149, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1088	37, 273, 277-278, 280, 284-285, 290, 297, 367, 500, 819, 833, 836, 850, 852, 855, 857, 870-871, 875, 891, 1025-1027, 1040-1043, 1126, 1200, 1208, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4971, 5123, 8229, 8482, 9025, 9044, 9049, 13121, 13488, 17584, 25617, 25619, 25664, 28709
1089	420, 500, 819, 850, 864, 1046, 1127, 1200, 1208, 1256, 4946, 4960, 5142, 5352, 8612, 9056, 9238, 13488, 16804, 17248, 17584
1097	37, 437, 500, 737, 775, 819, 850, 852, 857, 860-865, 1051, 1098, 1112, 1122, 1200, 1208, 1252, 4946, 4948, 4953, 4960, 8229, 9044, 9049, 9056, 13488, 17248, 17584, 28709
1098	259, 420, 437, 819, 850, 1097, 1200, 1208, 1252, 4946, 8612, 13488, 16804, 17584
1100	37, 273, 277-278, 280, 284-285, 297, 500, 850, 4946, 8229, 28709
1101	500
1102	500

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
1103	500
1104	500
1105	500
1106	500
1107	500
1112	37, 256, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 775, 819, 833, 836, 838, 850, 870-871, 875, 880, 905, 921-922, 1025-1027, 1097, 1122, 1200, 1208, 1252, 1257, 4386, 4929, 4932, 4934, 4946, 4971, 5123, 5353, 8229, 8482, 8612, 9025, 9030, 13121, 13488, 16804, 17584, 28709
1114	37, 437, 500, 819, 836, 850, 904, 1043, 1115, 1200, 1208, 4932, 4946, 5210-5211, 8229, 13488, 17584, 25480, 25619, 28709
1115	37, 367, 437, 500, 836, 903, 1114, 1200, 1208, 4932, 5210-5211, 8229, 13488, 17584, 25479, 28709
1122	37, 256, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 775, 819, 833, 836, 838, 850, 870-871, 875, 880, 905, 921-922, 1025-1027, 1097, 1112, 1200, 1208, 1252, 1257, 4386, 4929, 4932, 4934, 4946, 4971, 5123, 5353, 8229, 8482, 8612, 9025, 9030, 13121, 13488, 16804, 17584, 28709
1123	819, 1124-1125, 1148, 1200, 1208, 1251-1252, 1283, 5347, 13488, 17584
1124	37, 500, 1123, 1125, 1200, 1208, 1251, 1283, 5347, 8229, 13488, 17584, 28709
1125	500, 1123, 1124, 1200, 1208, 1251, 1283, 5347, 13488, 17584
1126	37, 367, 437, 500, 819, 833, 850, 1088, 1200, 1208, 1252, 4929, 4946, 8229, 9025, 13121, 13488, 17584, 25664, 28709
1127	420, 864, 1046, 1089, 1256, 4960, 5142, 8612, 9056, 9238, 16804, 17248
1129	500, 1130, 1200, 1208, 1258, 5354, 13488, 17584
1130	37, 500, 819, 850, 1129, 1200, 1208, 1252, 1258, 4946, 5354, 8229, 13488, 17584, 28709
1131	37, 500, 878, 915, 1025, 1200, 1208, 1251, 1283, 5347, 8229, 13488, 17584, 28709
1132	37, 500, 819, 850, 1133, 1200, 1208, 1252, 4946, 8229, 13488, 17584, 28709
1133	500, 1132, 1200, 1208, 13488, 17584
1137	37, 500, 819, 1200, 1208, 8229, 13488, 17584, 28709
1139	290, 1027, 4386, 5123, 8482
1140	37, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 850, 858, 860, 863, 871-872, 901-902, 923-924, 1013, 1047, 1051, 1141-1149, 1153-1157, 1160-1162, 1164, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
1141	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 871-872, 923-924, 1047, 1051, 1140, 1142-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1142	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 865, 871-872, 923-924, 1047, 1051, 1140-1141, 1143-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1143	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 865, 871-872, 923-924, 1047, 1051, 1140-1142, 1144-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1144	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 871-872, 923-924, 1047, 1051, 1140-1143, 1145-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1145	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 860, 863, 871-872, 923-924, 1047, 1051, 1140-1144, 1146-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1146	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 860, 863, 871-872, 923-924, 1047, 1051, 1140-1145, 1147-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1147	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 871-872, 923-924, 1047, 1051, 1140-1146, 1148-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1148	37, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 848-850, 858, 860-861, 863, 865, 871-872, 901-902, 923-924, 1047, 1051, 1123, 1140-1147, 1149, 1153-1164, 1200, 1208, 1252, 1275, 4899, 4946, 5348, 5349, 8229, 12712, 13488, 17584, 28709
1149	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 861, 863, 871-872, 923-924, 1047, 1051, 1140-1148, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1153	808, 858-859, 867, 872, 923-924, 1140-1149, 1154-1157, 1160-1162, 1200, 1208, 5348, 9044, 13488, 17584
1154	808, 849, 858-859, 867, 872, 923-924, 1140-1149, 1153, 1155-1157, 1160-1162, 1200, 1208, 5347, 5348, 13488, 17584
1155	858-859, 867, 872, 923-924, 1140-1149, 1153-1154, 1156-1157, 1160-1162, 1200, 1208, 5348, 5350, 9049, 13488, 17584
1156	858-859, 901-902, 923-924, 1140-1149, 1153-1155, 1157, 1160, 1200, 1208, 5348, 5353, 12712, 13488, 17584
1157	858-859, 901-902, 923-924, 1140-1149, 1153-1156, 1160, 1200, 1208, 5348, 5353, 12712, 13488, 17584
1158	848, 923, 1148, 1200, 1208, 5347, 5348, 13488, 17584

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
1159	1148, 1200, 1208, 13488, 17584
1160	858-859, 867, 923-924, 1140-1149, 1153-1157, 1161-1162, 1200, 1208, 5348, 13488, 17584
1161	259, 858-859, 923-924, 1140-1149, 1153-1155, 1160, 5348, 17584
1162	259, 858-859, 923-924, 1140-1149, 1153-1155, 1160, 5348, 17584
1163	924, 1148, 1164, 5354, 17584
1164	858-859, 923-924, 1140, 1148, 1163, 1200, 1208, 5348, 5354, 13488, 17584
1166	1200,1208,13488,17584
1200	37, 256, 259, 273, 275, 277-278, 280, 282, 284-285, 290, 293, 297, 300-301, 367, 420, 423-424, 437, 500, 720, 737, 775, 803, 806, 808, 813, 819, 833-838, 848-852, 855-872, 874-875, 878, 880, 891, 895-897, 901-905, 912, 914-916, 918, 920-924, 927-928, 930, 932-933, 935, 937, 939, 941-944, 946-951, 1004, 1006, 1008-1019, 1025-1027, 1040-1043, 1046-1047, 1051, 1088-1089, 1097-1098, 1112, 1114-1115, 1122-1126, 1129-1133, 1137, 1140-1149, 1153-1160, 1164, 1166, 1208, 1250-1258, 1275-1277, 1280-1285, 1351, 1362-1364, 1370-1371, 1374-1379, 1380-1381, 1385-1386, 1388, 1390, 1399, 4899, 4909, 4930, 4933, 4948, 4951-4952, 4960, 4971, 5012, 5039, 5104, 5123, 5142, 5210, 5346-5354, 8482, 8612, 9027, 9030, 9044, 9048-9049, 9056, 9061, 9066, 9238, 12712, 13121, 13218, 13488, 16684, 16804, 17248, 17584, 21427, 28709
1208	37, 256, 259, 273, 275, 277-278, 280, 282, 284-285, 290, 293, 297, 300-301, 367, 420, 423-424, 437, 500, 720, 737, 775, 803, 806, 808, 813, 819, 833-838, 848-852, 855-872, 874-875, 878, 880, 891, 895-897, 901-905, 912, 914-916, 918, 920-924, 927-928, 930, 932-933, 935, 937, 939, 941-944, 946-951, 1004, 1006, 1008-1019, 1025-1027, 1040-1043, 1046-1047, 1051, 1088-1089, 1097-1098, 1112, 1114-1115, 1122-1126, 1129-1133, 1137, 1140-1149, 1153-1160, 1164, 1166, 1200, 1250-1258, 1275-1277, 1280-1285, 1351, 1362-1364, 1370-1371, 1374-1379, 1380-1381, 1385-1386, 1388, 1390, 1399, 4899, 4909, 4930, 4933, 4948, 4951-4952, 4960, 4971, 5012, 5026, 5035, 5039, 5104, 5123, 5142, 5210, 5346-5354, 8482, 8612, 9027, 9030, 9044, 9048-9049, 9056, 9061, 9066, 9238, 12712, 13121, 13218, 13488, 16684, 16804, 17248, 17584, 21427, 28709
1250	37, 259, 273, 500, 819, 850, 852, 855, 870, 912, 1200, 1208, 1252, 1282, 4946, 4948, 4951, 5346, 8229, 9044, 13488, 17584, 28709
1251	37, 256, 259, 500, 819, 850, 855, 866, 878, 880, 915, 1025, 1123-1125, 1131, 1200, 1208, 1252, 1283, 4946, 4951, 5347, 8229, 13488, 17584, 28709

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
1252	37, 256, 259, 273, 275, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 737, 775, 803, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-866, 869-871, 874-875, 880, 897, 903, 905, 912, 914-916, 920-924, 1025-1027, 1041, 1047, 1051, 1097-1098, 1112, 1122-1123, 1126, 1130, 1132, 1140-1149, 1200, 1208, 1250-1251, 1254-1255, 1257, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25617, 28709
1253	37, 259, 423, 500, 737, 813, 819, 850, 869, 875, 1200, 1208, 1280, 4909, 4946, 4971, 5349, 8229, 9061, 13488, 17584, 28709
1254	37, 259, 500, 819, 850, 857, 869, 905, 920, 1026, 1047, 1200, 1208, 1252, 1281, 4946, 4953, 5350, 8229, 9049, 9061, 13488, 17584, 28709
1255	37, 259, 424, 500, 803, 819, 850, 856, 862, 916, 1200, 1208, 1252, 1281, 4946, 4952, 5012, 5351, 8229, 13488, 17584, 28709
1256	259, 420, 500, 720, 850, 864, 1046, 1089, 1127, 1200, 1208, 4946, 4960, 5142, 5352, 8612, 9056, 9238, 13488, 16804, 17248, 17584
1257	37, 259, 437, 500, 775, 819, 850, 914, 921-922, 1112, 1122, 1200, 1208, 1252, 4946, 5353, 8229, 13488, 17584, 28709
1258	37, 259, 500, 819, 1129-1130, 1200, 1208, 5354, 8229, 13488, 17584, 28709
1275	37, 256, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 871, 923-924, 1051, 1140-1149, 1200, 1208, 1252, 4946, 5348, 8229, 13488, 17584, 28709
1276	1200, 1208, 13488, 17584
1277	1200, 1208, 13488, 17584
1280	37, 423, 437, 500, 737, 813, 819, 850, 869, 875, 1200, 1208, 1252-1253, 4909, 4946, 4971, 5349, 8229, 9061, 13488, 17584, 28709
1281	37, 437, 500, 819, 850, 857, 905, 920, 1026, 1200, 1208, 1252, 1254-1255, 4946, 4953, 5350, 8229, 9049, 13488, 17584, 28709
1282	500, 852, 870, 912, 1200, 1208, 1250, 4948, 5346, 9044, 13488, 17584
1283	37, 437, 500, 819, 850, 855, 866, 878, 880, 915, 1025, 1123-1125, 1131, 1200, 1208, 1251-1252, 4946, 4951, 5347, 8229, 13488, 17584, 28709
1284	1200, 1208, 13488, 17584
1285	1200, 1208, 13488, 17584
1351	300-301, 941, 1200, 1208, 4396, 8492, 13488, 16684, 17584
1362	834, 951, 1200, 1208, 4930, 9026, 13488, 17584
1363	933, 949, 1200, 1208, 1364, 5029, 5045, 5460, 9125, 9555, 13221, 13488, 13651, 17317, 17584, 25525, 29621, 33717, 37813
1364	933, 949, 1200, 1208, 1363, 5029, 5045, 5460, 9125, 9555, 13221, 13488, 13651, 17317, 17584, 25525, 29621, 33717, 37813

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
1370	937-938, 948, 950, 1200, 1208, 1371, 5033, 5046, 9142, 13488, 17584, 25514, 25524, 29620
1371	1200, 1208, 1370, 13488, 17584
1374	1200, 1208
1375	1200, 1208
1376	1200, 1208
1377	1200, 1208
1378	1200, 1208
1379	1200, 1208
1380	837, 928, 1200, 1208, 1385, 4933, 13488, 17584
1381	935-936, 1200, 1208, 1386, 1388, 5031, 5477, 5482, 5484, 9127, 13223, 13488, 17584, 25512
1385	837, 1200, 1208, 1380, 4933, 13488, 17584
1386	935, 1200, 1208, 1381, 1388, 5031, 5477, 5482, 5484, 9127, 13223, 13488, 17584
1388	935, 1200, 1208, 1381, 1386, 5031, 5477, 5482, 5484, 5488, 9127, 13223, 13488, 17584
1390	930-932, 939, 942-943, 1200, 1208, 1399, 5026, 5028, 5035, 5038-5039, 5055, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
1399	930-932, 939, 942-943, 1200, 1208, 1390, 5026, 5028, 5035, 5038-5039, 5050, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
4386	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 895-897, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1139, 1252, 4929, 4932, 4946, 4948, 4951, 4953, 4960, 4992, 5123, 8229, 8482, 9025, 9044, 9049, 9056, 13121, 17248, 25473, 25617, 25619, 25664, 28709
4396	300-301, 941, 1351, 8492, 16684
4899	867, 1148, 1200, 1208, 5351, 9048, 12712, 13488, 17584
4909	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252-1253, 1280, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5349, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
4929	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 891, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1126, 1252, 4386, 4932, 4946, 4948, 4951, 4953, 4960, 5123, 8229, 8482, 9025, 9044, 9049, 9056, 13121, 17248, 25617, 25619, 25664, 28709
4930	834, 951, 1200, 1208, 1362, 9026, 13488, 17584
4931	835, 927, 947, 9027, 21427



Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
4932	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 424, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 870-871, 875, 903, 1009, 1025-1027, 1040-1043, 1088, 1112, 1114-1115, 1122, 1252, 4386, 4929, 4946, 4948, 4951, 4953, 4971, 5123, 5210-5211, 8229, 8482, 9025, 9044, 9049, 13121, 25479, 25617, 25619, 25664, 28709
4933	837, 1200, 1208, 1380, 1385, 13488, 17584
4934	37, 256, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 775, 813, 819, 838, 850, 852, 857, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1112, 1122, 1252, 4909, 4946, 4948, 4953, 4960, 4970-4971, 5012, 5123, 8229, 9030, 9044, 9049, 9056, 9061, 9066, 17248, 25473, 25479, 25617, 25619, 28709
4946	37, 256, 259, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 737, 775, 803, 813, 819, 833, 836, 838, 850, 852, 855-858, 860-866, 869-871, 874-875, 880, 897, 903, 905, 912, 914-916, 920-924, 1004, 1025-1027, 1040-1043, 1047, 1051, 1088-1089, 1097-1098, 1100, 1112, 1114, 1122, 1126, 1130, 1132, 1140-1149, 1250-1257, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4948, 4951-4953, 4960, 4970-4971, 5012, 5123, 5210, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 16804, 17248, 25473, 25479, 25617, 25619, 25664, 28709
4948	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1040-1043, 1047, 1088, 1097, 1200, 1208, 1250, 1252, 1282, 4386, 4909, 4929, 4932, 4934, 4946, 4951, 4953, 4970-4971, 5012, 5123, 5346, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 25664, 28709
4951	37, 259, 273, 277-278, 280, 284-285, 290, 297, 437, 500, 819, 833, 836, 850, 852, 855, 857, 866, 870-871, 878, 880, 912, 915, 1025-1027, 1040-1043, 1088, 1200, 1208, 1250-1252, 1283, 4386, 4929, 4932, 4946, 4948, 4953, 5123, 5346, 5347, 8229, 8482, 9025, 9044, 9049, 13121, 13488, 17584, 25617, 25619, 25664, 28709
4952	259, 273, 424, 500, 803, 850, 856, 862, 916, 1200, 1208, 1255, 4946, 5012, 5351, 13488, 17584
4953	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1040-1043, 1088, 1097, 1252, 1254, 1281, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4970-4971, 5012, 5123, 5350, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 16804, 25473, 25479, 25617, 25619, 25664, 28709
4960	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 720, 819, 833, 838, 850, 864, 870-871, 875, 880, 905, 918, 1008, 1025-1027, 1046, 1089, 1097, 1127, 1200, 1208, 1252, 1256, 4386, 4929, 4934, 4946, 4971, 5104, 5123, 5142, 5352, 8229, 8482, 8612, 9025, 9030, 9056, 9238, 13121, 13488, 16804, 17248, 17584, 28709

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
4970	37, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1252, 4909, 4934, 4946, 4948, 4953, 4971, 5012, 5123, 8229, 9030, 9044, 9049, 9061, 9066, 25473, 25479, 25617, 25619, 28709
4971	37, 256, 273, 277-278, 280, 284-285, 297, 367, 423, 437, 500, 737, 775, 813, 819, 836, 838, 850-852, 857, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1009, 1025-1027, 1041-1043, 1047, 1088, 1112, 1122, 1200, 1208, 1252-1253, 1280, 4909, 4932, 4934, 4946, 4948, 4953, 4960, 4970, 5012, 5123, 5349, 8229, 9030, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
4992	290, 896, 1027, 1041, 4386, 5123, 8482, 25617
5012	37, 273, 277-278, 280, 284-285, 297, 423-424, 437, 500, 803, 813, 819, 838, 850, 852, 856-857, 860-863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 1255, 4909, 4934, 4946, 4948, 4952-4953, 4970-4971, 5123, 5351, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
5026	930-932, 939, 942-943, 1390, 1399, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 1208, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
5028	930-932, 939, 942-943, 1390, 1399, 5026, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
5029	933-934, 944, 949, 1363-1364, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813
5031	935-936, 946, 1381, 1386, 1388, 5477, 5482, 5484, 9127, 13223, 25512
5033	937-938, 948, 950, 1370, 5046, 9142, 25514, 25524, 29620
5035	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5038-5039, 9122, 9124, 9131, 9135, 1208, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
5038	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
5039	930-932, 939, 942-943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
5045	933-934, 944, 949, 1363-1364, 5029, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813
5046	937-938, 948, 950, 1370, 5033, 9142, 25514, 25524, 29620
5104	420, 864, 1008, 1200, 1208, 4960, 8612, 9056, 13488, 16804, 17248, 17584
5123	290, 367, 423, 437, 819, 1027, 1041, 1047, 1140-1149, 1156, 1157, 1160, 1200, 1208, 1252, 4948, 5348, 8482, 13488

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
5142	420, 500, 864, 1046, 1089, 1127, 1200, 1208, 1256, 4960, 5352, 8612, 9056, 9238, 13488, 16804, 17248, 17584
5210	37, 437, 500, 819, 836, 850, 904, 1043, 1114-1115, 1200, 1208, 4932, 4946, 5211, 8229, 13488, 17584, 25480, 25619, 28709
5211	37, 367, 437, 500, 836, 903, 1114-1115, 4932, 5210, 8229, 25479, 28709
5346	37, 259, 273, 500, 819, 850, 852, 855, 870, 912, 1200, 1208, 1250, 1252, 1282, 4946, 4948, 4951, 8229, 9044, 13488, 17584, 28709
5347	808, 848-849, 855, 866, 872, 878, 880, 915, 1025, 1123-1125, 1131, 1154, 1158, 1200, 1208, 1251, 1283, 4951, 13488, 17584
5348	37, 259, 273, 275, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 850, 858, 860-861, 863, 865, 871-872, 901-902, 923-924, 1051, 1140-1149, 1153-1158, 1160-1162, 1164, 1200, 1208, 1252, 1275, 4946, 8229, 13488, 17584, 28709
5349	813, 869, 875, 1148, 1200, 1208, 1253, 1280, 4909, 4971, 9061, 13488, 17584
5350	857, 920, 1026, 1155, 1200, 1208, 1254, 1281, 4953, 9049, 13488, 17584
5351	424, 856, 862, 867, 916, 1200, 1208, 1255, 4899, 4952, 5012, 9048, 12712, 13488, 17584
5352	420, 864, 1046, 1089, 1200, 1208, 1256, 4960, 5142, 8612, 9056, 9238, 13488, 16804, 17248, 17584
5353	901-902, 921-922, 1112, 1122, 1156-1157, 1200, 1208, 1257, 13488, 17584
5354	1129-1130, 1163, 1164, 1200, 1208, 1258, 13488, 17584
5460	933-934, 944, 949, 1363-1364, 5029, 5045, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813
5477	935-936, 1381, 1386, 1388, 5031, 5482, 5484, 9127, 13223, 25512
5482	935, 1381, 1386, 1388, 5031, 5477, 5484, 9127, 13223
5484	935-936, 946, 1381, 1386, 1388, 5031, 5477, 5482, 9127, 13223, 25512
5488	1388
8229	37, 256, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 720, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-866, 869-871, 874-875, 880, 897, 903-905, 912, 914-916, 920-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1097, 1100, 1112, 1114-1115, 1122, 1124, 1126, 1130-1132, 1137, 1140-1149, 1250-1255, 1257-1258, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5210-5211, 5346, 5348, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 16804, 17248, 25473, 25479, 25480, 25617, 25619, 25664, 28709

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
8482	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 895-897, 1009, 1025-1027, 1040-1043, 1047, 1088, 1112, 1122, 1139, 1200, 1208, 1252, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4960, 4992, 5123, 8229, 9025, 9044, 9049, 9056, 13121, 13488, 17248, 17584, 25473, 25617, 25619, 25664, 28709
8492	300-301, 941, 1351, 4396, 16684
8612	37, 256, 420, 424, 437, 500, 720, 737, 775, 819, 850, 852, 857, 860-865, 1008, 1046, 1089, 1098, 1112, 1122, 1127, 1200, 1208, 1252, 1256, 4946, 4948, 4953, 4960, 5104, 5142, 5352, 8229, 9044, 9049, 9056, 9238, 13488, 16804, 17248, 17584, 28709
9025	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 891, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1126, 1252, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4960, 5123, 8229, 8482, 9044, 9049, 9056, 13121, 17248, 25617, 25619, 25664, 28709
9026	834, 926, 951, 1362, 4930
9027	835, 927, 947, 1200, 1208, 4931, 13488, 17584, 21427
9030	37, 256, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 775, 813, 819, 838, 850, 852, 857, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1112, 1122, 1200, 1208, 1252, 4909, 4934, 4946, 4948, 4953, 4960, 4970-4971, 5012, 5123, 8229, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 28709
9044	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1040-1043, 1047, 1088, 1097, 1153, 1200, 1208, 1250, 1252, 1282, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5346, 8229, 8482, 8612, 9025, 9030, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 25664, 28709
9048	867, 1200, 1208, 4899, 5351, 12712, 13488, 17584
9049	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1040-1043, 1088, 1097, 1155, 1200, 1208, 1252, 1254, 1281, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5350, 8229, 8482, 8612, 9025, 9030, 9044, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 25664, 28709
9056	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 720, 819, 833, 838, 850, 864, 870-871, 875, 880, 905, 918, 1008, 1025-1027, 1046, 1089, 1097, 1127, 1200, 1208, 1252, 1256, 4386, 4929, 4934, 4946, 4960, 4971, 5104, 5123, 5142, 5352, 8229, 8482, 8612, 9025, 9030, 9238, 13121, 13488, 16804, 17248, 17584, 28709

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
9061	37, 256, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252-1254, 1280, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5349, 8229, 9030, 9044, 9049, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
9066	37, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 8229, 9030, 9044, 9049, 9061, 13488, 17584, 25473, 25479, 25617, 25619, 28709
9122	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
9124	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
9125	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813
9127	935-936, 946, 1381, 1386, 1388, 5031, 5477, 5482, 5484, 13223, 25512
9131	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
9135	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
9142	937-938, 948, 950, 1370, 5033, 5046, 25514, 25524, 29620
9238	420, 500, 864, 1046, 1089, 1127, 1200, 1208, 1256, 4960, 5142, 5352, 8612, 9056, 13488, 16804, 17248, 17584
9555	933, 949, 1363-1364, 5029, 5045, 5460, 9125, 13221, 13651, 17317, 25525, 29621, 33717, 37813
12712	862, 867, 1148, 1156-1157, 1200, 1208, 4899, 5351, 9048, 13488, 17584
13121	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 891, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1126, 1200, 1208, 1252, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4960, 5123, 8229, 8482, 9025, 9044, 9049, 9056, 13488, 17248, 17584, 25617, 25619, 25664, 28709
13218	930-932, 939, 942-943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
13219	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
13221	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
13223	935-936, 946, 1381, 1386, 1388, 5031, 5477, 5482, 5484, 9127, 25512
13231	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 17314, 25508, 25518, 29614, 33698-33700, 37796
13488	37, 256, 259, 273, 275, 277-278, 280, 282, 284-285, 290, 293, 297, 300-301, 367, 420, 423-424, 437, 500, 720, 737, 775, 803, 806, 808, 813, 819, 833-838, 848-852, 855-872, 874-875, 878, 880, 891, 895-897, 901-905, 912, 914-916, 918, 920-924, 927-928, 930, 932-933, 935, 937, 939, 941-944, 946-951, 1004, 1006, 1008-1019, 1025-1027, 1040-1043, 1046-1047, 1051, 1088-1089, 1097-1098, 1112, 1114-1115, 1122-1126, 1129-1133, 1137, 1140-1149, 1153-1160, 1164, 1166, 1200, 1208, 1250-1258, 1275-1277, 1280-1285, 1351, 1362-1364, 1370-1371, 1380-1381, 1385-1386, 1388, 1390, 1399, 4899, 4909, 4930, 4933, 4948, 4951-4952, 4960, 4971, 5012, 5039, 5104, 5123, 5142, 5210, 5346-5354, 8482, 8612, 9027, 9030, 9044, 9048-9049, 9056, 9061, 9066, 9238, 12712, 13121, 13218, 16684, 16804, 17248, 17584, 21427, 28709
13651	933, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 17317, 25525, 29621, 33717, 37813
16684	300-301, 941, 1200, 1208, 1351, 4396, 8492, 13488, 17584
16804	37, 256, 420, 424, 437, 500, 720, 737, 775, 819, 850, 852, 857, 860-865, 1008, 1046, 1089, 1098, 1112, 1122, 1127, 1200, 1208, 1252, 1256, 4946, 4948, 4953, 4960, 5104, 5142, 5352, 8229, 8612, 9044, 9049, 9056, 9238, 13488, 17248, 17584, 28709
17248	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 720, 819, 833, 838, 850, 864, 870-871, 875, 880, 905, 918, 1008, 1025-1027, 1046, 1089, 1097, 1127, 1200, 1208, 1252, 1256, 4386, 4929, 4934, 4946, 4960, 4971, 5104, 5123, 5142, 5352, 8229, 8482, 8612, 9025, 9030, 9056, 9238, 13121, 13488, 16804, 17584, 28709
17314	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 25508, 25518, 29614, 33698-33700, 37796
17317	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 25510, 25520, 25525, 29616, 29621, 33717, 37813
17584	37, 256, 259, 273, 275, 277-278, 280, 282, 284-285, 290, 293, 297, 300-301, 367, 420, 423-424, 437, 500, 720, 737, 775, 803, 806, 808, 813, 819, 833-838, 848-852, 855-872, 874-875, 878, 880, 891, 895-897, 901-905, 912, 914-916, 918, 920-924, 927-928, 930, 932-933, 935, 937, 939, 941-944, 946-951, 1004, 1006, 1008-1019, 1025-1027, 1040-1043, 1046-1047, 1051, 1088-1089, 1097-1098, 1112, 1114-1115, 1122-1126, 1129-1133, 1137, 1140-1149, 1153-1160, 1164, 1166, 1200, 1208, 1250-1258, 1275-1277, 1280-1285, 1351, 1362-1364, 1370-1371, 1380-1381, 1385-1386, 1388, 1390, 1399, 4899, 4909, 4930, 4933, 4948, 4951-4952, 4960, 4971, 5012, 5039, 5104, 5123, 5142, 5210, 5346-5354, 8482, 8612, 9027, 9030, 9044, 9048-9049, 9056, 9061, 9066, 9238, 12712, 13121, 13218, 13488, 16684, 16804, 17248, 21427, 28709
21427	835, 927, 947, 1200, 1208, 4931, 9027, 13488, 17584

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
25473	37, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1252, 4386, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 8229, 8482, 9030, 9044, 9049, 9061, 9066, 25479, 25617, 25619, 28709
25479	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 836, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1115, 1252, 4909, 4932, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5211, 8229, 9030, 9044, 9049, 9061, 9066, 25473, 25617, 25619, 28709
25480	37, 500, 904, 1114, 5210, 8229, 28709
25508	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25518, 29614, 33698-33700, 37796
25510	933-934, 949, 5029, 5045, 5460, 9125, 13221, 17317, 25525, 29621, 33717, 37813
25512	935-936, 946, 1381, 5031, 5477, 5484, 9127, 13223
25514	937-938, 950, 1370, 5033, 5046, 9142
25518	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 29614, 33698-33700, 37796
25520	933, 944, 949, 5029, 5045, 5460, 9125, 13221, 17317, 25525, 29616, 29621, 33717, 37813
25524	937, 948, 950, 1370, 5033, 5046, 9142, 29620
25525	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 29616, 29621, 33717, 37813
25617	37, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 895-897, 903, 912, 916, 1025-1027, 1040-1043, 1088, 1252, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 4992, 5012, 5123, 8229, 8482, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 25473, 25479, 25619, 25664, 28709
25619	37, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 1025-1027, 1040-1043, 1088, 1114, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5210, 8229, 8482, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 25473, 25479, 25617, 25664, 28709
25664	37, 273, 277-278, 280, 284-285, 290, 297, 367, 500, 819, 833, 836, 850, 852, 855, 857, 870-871, 875, 891, 1025-1027, 1040-1043, 1088, 1126, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4971, 5123, 8229, 8482, 9025, 9044, 9049, 13121, 25617, 25619, 28709

Table 674. IBM MQ for z/OS CCSID conversion support (continued)

CCSID	Converts to and from CCSIDS
28709	37, 256, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 720, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-866, 869-871, 874-875, 880, 897, 903-905, 912, 914-916, 920-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1097, 1100, 1112, 1114-1115, 1122, 1124, 1126, 1130-1132, 1137, 1140-1149, 1200, 1208, 1250-1255, 1257-1258, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5210-5211, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25480, 25617, 25619, 25664
29614	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 33698-33700, 37796
29616	933, 944, 949, 5029, 5045, 5460, 9125, 13221, 17317, 25520, 25525, 29621, 33717, 37813
29620	937, 948, 950, 1370, 5033, 5046, 9142, 25524
29621	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 33717, 37813
33698	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33699-33700, 37796
33699	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698, 33700, 37796
33700	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33699, 37796
33717	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 37813
37796	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700
37813	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717

## IBM i IBM i 변환 지원

CCSID의 전체 목록 및 IBM i에서 지원되는 변환은 적절한 IBM i 발행에서 찾을 수 있습니다.

지원되는 코드 페이지는 [지원되는 CCSID 맵핑](#)에 나열됩니다.

## 유니코드 변환 지원

일부 플랫폼은 유니코드 인코딩으로(부터) 사용자 데이터의 변환을 지원합니다. 지원되는 유니코드 인코딩의 두 가지 양식은 UTF-16(CCSID 1200, 13488 및 17584)과 UTF-8(CCSID 1208)입니다. 지원되는 최신 유니코드 버전을 나타낸 것처럼 CCSID 1200 또는 1208을 사용해야 합니다.





예를 들면가 이에서 경우에서십시오, 그러나십시오,으로 이게 되면, 이러합 (거나 이름다지된된다지 여기 현 재할 수 있다다지 여기서, 각각 있다다지 여기 되는 경우할-할되 있 있할-할되가 설정 경우할 사용 경우할되 있다다된 없된할되 있할 사용할 수 있다.하 있다.

1200, 1208, 1250, 1251, 1252, 1253, 1254, 1255, 1256, 1257, 1258, 1275, 1280, 1281, 1282, 1283

또한 있 있 이}있 있 이)). On 경우 있 있 이 있된할 없에) 있할고 있할 수도가 서버, 를 을 을 수 있다.

4899

5050, 5346, 5347, 5348, 5349, 5350, 5351, 5352, 5353, 5354, 5488<sup>950</sup>페이지의 『5』

9044, 9048, 9449

12712

13488

17584

33722<sup>950</sup>페이지의 『4』

## 참고:

1. 931은 변환을 위해 939를 사용합니다.
2. 932는 변환을 위해 942를 사용합니다.
3. 938은 변환을 위해 948을 사용합니다.
4. 954 및 33722는 변환을 위해 5050을 사용합니다.
5. Windows 및 Linux 에서만.

## 유니코드에 대한 IBM i 지원

IBM i

UNICODE 지원에 대한 자세한 내용은 운영 체제와 관련된 해당 IBM i 서적을 참조하십시오.

## 유니코드에 대한 IBM MQ for z/OS 지원

z/OS

IBM MQ for z/OS 변환 시 지원되는 유니코드 CCSID (1200또는 1208권장) 는 다음 목록의 비유니코드 CCSID 에 대해 지원됩니다.

37

256, 259, 273, 275, 277, 278, 280, 282, 284, 285, 290, 293, 297

300, 301, 367

420, 423, 424, 437

500

720, 737, 775

또한다 이야습니다십시오십시오십시오 (예를 들어, 미국 특허 번호 이어서고으므로, 이에 대해서는 본원에 기 재된 바와 같이)를 사용자할합 수되가 이름고고고습니다 (테이블십시오 있가한적 포함 이 (테이블 이)).

예십시오 있니다십시오 있 수십시오습니다 있고습니다 (예를 들어, 미국 특허 번호 개구 저장 나 합 및 나어서 한 만한 것).

또한고고습니다십시오 있할합 수되니다십시오 있 수되 파일의할 경우습니다십시오를게 표시 또는와 상기 지정된 값을 사용하여

예를 들면이개은은은이합이되이가이합이되를 합 있을 사용에으며가 동안니다에에을되 있을 또는 있를다는 하나 후를 실질적으로 포함하지 않을 수 있다.

이들은, 예를 들어, 1, 2, 3, 4, 5, 6, 7, 8, 8, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 20, 30, 40, 50, 60, 70, 80, 90, 100, 100, 200, 100, 200, 300, 400, 400, 400, 500, 400, 400, 400, 500, 400, 500, 또는 그 이상의 범위에서 사용된다.

이들은, 예를 들면이할십시오는 있 웹 경우에이할 수 있고은 있 개이할다을 포함할한십시오는이십시오는이 십시오하를들 포함십시오는하에이 사용합이지만 있십시오하를 을 을 을다는 하에이십시오는이한십시오.

4899, 4909, 4930, 4933, 4948, 4951, 4952, 4960, 4971

5012 5039 5104 5123 5142 5210 5346 5347 5348 5349 5350 5351 5352 5353 5354 5488

8482 8612  
 9027 9030 9044 9048 9049 9056 9061 9066 9238 9449  
 1166  
 12712  
 이들은 또한, 예를 들면이합 있을 사용에으며가 10있십시오는 경우에을 을 을할한다.  
 16684, 16804 년  
 17248, 17584  
 21427  
 28709

## 64비트 플랫폼에서의 코드화 표준

64비트 플랫폼에서의 코딩 표준 및 선호하는 데이터 유형에 대해 학습하려면 이 정보를 사용하십시오.

### 선호하는 데이터 유형

이러한 유형은 크기를 변경하지 않고 32비트와 64비트 IBM MQ 플랫폼 둘 다에서 사용 가능합니다.

표 675. 데이터 유형 이름 및 길이

이름	길이
MQLONG	4바이트
MQULONG	4바이트
MQINT32	4바이트
MQUINT32	4바이트
MQINT64	8바이트
MQUINT64	8바이트

## AIX, Linux, and Windows에서 표준 데이터 유형

32비트 AIX and Linux 및 64비트 AIX, Linux, and Windows 애플리케이션의 표준 데이터 유형에 대해 알아보니다.

### 32비트 AIX and Linux 애플리케이션



표 676. 32비트 AIX and Linux 애플리케이션의 데이터 유형 이름 및 길이

이름	길이
char	1바이트
short	2바이트
int	4바이트
long	4바이트
float	4바이트
double	8바이트
long double	8바이트
pointer	4바이트
ptrdiff_t	4바이트
size_t	4바이트

표 676. 32비트 AIX and Linux 애플리케이션의 데이터 유형 이름 및 길이 (계속)

이름	길이
time_t	4바이트
clock_t	4바이트
wchar_t	4바이트

▶ **AIX** AIX에서 wchar\_t는 2바이트입니다.

### 64비트 AIX and Linux 애플리케이션



표 677. 64비트 AIX and Linux 애플리케이션의 데이터 유형 이름 및 길이

이름	길이
char	1바이트
short	2바이트
int	4바이트
long	8바이트
float	4바이트
double	8바이트
long double	8바이트
pointer	8바이트
ptrdiff_t	8바이트
size_t	8바이트
time_t	8바이트
clock_t	4바이트
wchar_t	4바이트

▶ **AIX** AIX에서 wchar\_t는 2바이트입니다.

### Windows 64비트 애플리케이션



표 678. Windows 64비트 애플리케이션용 데이터 유형 이름 및 길이

이름	길이
char	1바이트
short	2바이트
int	4바이트
long	4바이트
float	4바이트
double	8바이트
long double	8바이트

표 678. Windows 64비트 애플리케이션용 데이터 유형 이름 및 길이 (계속)

이름	길이
pointer	8바이트 모든 pointer는 8바이트입니다.
ptrdiff_t	8바이트
size_t	8바이트
time_t	8바이트
clock_t	4바이트
wchar_t	2바이트
WORD	2바이트
DWORD	4바이트
HANDLE	8바이트
HFILE	4바이트

## Windows에 대한 코드 고려사항

### Windows

#### HANDLE hf;

다음을 사용하십시오.

```
hf = CreateFile((LPCTSTR) FileName,
               Access,
               ShareMode,
               xihSecAttsNTRestrict,
               Create,
               AttrAndFlags,
               NULL);
```

사용 안함

```
HFILE hf;
hf = (HFILE) CreateFile((LPCTSTR) FileName,
                       Access,
                       ShareMode,
                       xihSecAttsNTRestrict,
                       Create,
                       AttrAndFlags,
                       NULL);
```

이로 인해 오류가 발생합니다.

#### size\_t len fgets

다음을 사용하십시오.

```
size_t len
while (fgets(string1, (int) len, fp) != NULL)
len = strlen(buffer);
```

사용 안함

```
int len;

while (fgets(string1, len, fp) != NULL)
len = strlen(buffer);
```

## printf

다음을 사용하십시오.

```
printf("My struc pointer: %p", pMyStruc);
```

사용 안함

```
printf("My struc pointer: %x", pMyStruc);
```

16진수 출력이 필요한 경우 개별적으로 상단 및 하단 4바이트를 출력해야 합니다.

## char \*ptr

다음을 사용하십시오.

```
char * ptr1;  
char * ptr2;  
size_t buflen;  
  
buflen = ptr2 - ptr1;
```

사용 안함

```
char *ptr1;  
char *ptr2;  
UINT32 buflen;  
  
buflen = ptr2 - ptr1;
```

## alignBytes

다음을 사용하십시오.

```
alignBytes = (unsigned short) ((size_t) address % 16);
```

사용 안함

```
void *address;  
unsigned short alignBytes;  
  
alignBytes = (unsigned short) ((UINT32) address % 16);
```

## len

다음을 사용하십시오.

```
len = (UINT32) ((char *) address2 - (char *) address1);
```

사용 안함

```
void *address1;  
void *address2;  
UINT32 len;  
  
len = (UINT32) ((char *) address2 - (char *) address1);
```

## sscanf

다음을 사용하십시오.

```
MQLONG SBCSprt;
```

```
sscanf(line, "%d", &SBCSprt);
```

사용 안함

```
MQLONG SBCSprt;  
sscanf(line, "%1d", &SBCSprt);
```

%1d는 8바이트 유형을 4바이트 유형에 넣으려고 시도합니다. 실제 long 데이터 유형을 처리하고 있는 경우 %1만 사용하십시오. MQLONG, UIN32 및 INT32 는 4바이트로 정의되며 모든 IBM MQ 플랫폼의 int 와 동일합니다.

## IBM i IBM i 애플리케이션 프로그래밍 참조(ILE/RPG)

IBM i용 애플리케이션 프로그래밍입니다.

다음은 IBM i용 애플리케이션을 개발하는 데 도움이 되는 정보입니다.

- [956 페이지의 『IBM i의 데이터 유형 설명』](#)
- [1194 페이지의 『IBM i의 함수 호출』](#)
- [1304 페이지의 『IBM i의 오브젝트 속성』](#)
- [1347 페이지의 『애플리케이션』](#)
- [1359 페이지의 『IBM i\(ILE RPG\)의 리턴 코드』](#)
- [1360 페이지의 『IBM i\(ILE RPG\)에 대해 MQI 옵션 유효성 검증 규칙』](#)
- [1362 페이지의 『IBM i의 시스템 인코딩』](#)
- [1365 페이지의 『IBM i의 보고 옵션 및 메시지 플래그』](#)

### IBM i의 RPG 및 COBOL 애플리케이션에 대한 호환 모드가 더 이상 사용되지 않음

#### IBM i

IBM MQ for IBM i 9.0부터 제품은 호환 모드로 알려진 동적 연계를 사용하는 RPG 또는 COBOL 애플리케이션을 더 이상 지원하지 않습니다. MQSeries 5.1 이전에 작성된 애플리케이션의 경우 이 조작 모드가 필요했으며, IBM WebSphere MQ 6.0에서 애플리케이션을 컴파일하는 데 필요한 카피북이 제거된 경우에도 후속 버전 제품에서 이들 애플리케이션에 대해 호환 가능한 런타임 환경을 제공했습니다. 동적 연계(호환성 모드)는 라이브러리 QMQM에 있는 다음 프로그램에서 제공되었으며 이들 프로그램은 IBM MQ for IBM i 9.0에서 제거되었습니다.

- AMQVSTUB
- AMQZSTUB
- QMQM
- MQCLOSE
- MQCONN
- MQDISC
- MQGET
- MQINQ
- MQOPEN
- MQPUT
- MQPUT1
- MQSET

IBM MQ for IBM i 9.0부터 이 호환성 모드 조작을 사용하는 애플리케이션은 LIBMQM 및 LIBMQM\_R 프로그램 이 제공하는 정적 바인드 MQ 호출을 사용하도록 다시 컴파일되어야 합니다. 샘플 프로그램(예: AMQ3PUT4 및 AMQ3GET4)은 이 프로그래밍 모델을 사용하는 방법을 보여줍니다. 이러한 MQ 호출의 사용에 대한 자세한 정보는 [IBM i 애플리케이션 프로그래밍 참조서\(ILE/RPG\)](#)를 참조하십시오.

## 참고:

- 대신 LIBMQM 서비스 프로그램을 사용하려면 현재 CALL 'QMQM' 인터페이스를 사용하는 애플리케이션을 레코딩해야 합니다.

실행 목록의 프로그램 오브젝트 및 서비스 프로그램(예: QMQM, MQCONN, MQPUT, AMQVSTUB 및 AMQZSTUB)이 IBM MQ for IBM i 9.0에서 제거되고 호환 모드를 사용하도록 코딩된 애플리케이션이 작업을 중단합니다.

- 애플리케이션이 IBM MQ for IBM i 8.0에서 LIBMQM 서비스 프로그램에 바인드되면, IBM MQ for IBM i 9.0 이상에서 해당 애플리케이션을 재컴파일하거나 다시 링크시킬 필요가 없습니다.
- 동일한 파티션에서 IBM MQ for IBM i의 버전을 두 개 이상 설치할 수 없습니다.

RPG 또는 COBOL 프로그램이 호환 모드를 사용하는지 여부를 알아보려면 **DSPPGMREF**(프로그램 참조 표시) 명령을 사용하여 애플리케이션 프로그램이 호출하는 외부 프로그램을 표시하십시오. 이 섹션에 나열된 프로그램에 대한 참조가 있는 경우 프로그램은 IBM MQ for IBM i 9.0 이상에서 실행되지 않습니다. **DSPPGMREF** 출력의 다음 예는 더 이상 사용되지 않는 3개의 프로그램 오브젝트, MQCONN, MQOPEN, MQCLOSE를 표시합니다.

```
Program . . . . . : MYAPPPGM
Library . . . . . : MYLIB
Text 'description'. . . . . : ILE/COBOL SAMPLE PUT TO QUEUE (MQPUT)
Number of objects referenced . . . . . : 5
Object . . . . . : MQCONN
Library . . . . . : *LIBL
Object type . . . . . : *PGM
Object . . . . . : MQOPEN
Library . . . . . : *LIBL
Object type . . . . . : *PGM
Object . . . . . : MQCLOSE
Library . . . . . : *LIBL
Object type . . . . . : *PGM
```

이러한 프로그램은 IBM i에서 COBOL 프로그램 준비에서 설명된 Bound Procedural Call 메소드를 사용하여 재컴파일되어야 합니다.

IBM MQ for IBM i 9.0 이상에서 호환 모드를 사용하는 애플리케이션 프로그램을 실행하려고 시도하는 경우가 가장 일반적으로 발생하는 첫 번째 오류는 프로그램 MCH3401이 프로그램 MQCONN 또는 QMQM을 호출하려고 하는 것입니다.

## 관련 태스크

[애플리케이션 개발](#)

## IBM i의 데이터 유형 설명

이 주제의 콜렉션은 IBM i 프로그래밍에서 사용된 데이터 유형의 설명을 제공합니다.

### 데이터 유형 설명에서 사용된 규칙

각 요소 데이터 유형의 경우 이 정보는 사용법 설명을 프로그래밍 언어와 무관한 형식으로 제공합니다. 그 다음에는 RPG 프로그래밍 언어의 ILE 버전에서의 일반 선언이 이어집니다. 요소 데이터 유형의 정의는 일관성을 제공하기 위해 여기에 포함됩니다. RPG는 필요한 속성을 사용하여 작업 필드를 선언할 수 있는 'D' 스펙을 사용합니다. 그러나 필드가 사용되는 계산 스펙에서 이를 수행할 수 있습니다.

요소 데이터 유형을 사용하려면 다음을 작성하십시오.

- 모든 데이터 유형을 포함한 /COPY 멤버 또는
- 모든 데이터 유형을 포함한 외부 데이터 구조(PF). 그런 다음 적절한 데이터 유형 필드인 'LIKE' 속성을 사용하여 작업 필드를 지정해야 합니다.

두 번째 옵션의 이점은 정의가 다른 IBM i 오브젝트를 위해 'FIELD REFERENCE FILE'로 사용될 수 있다는 것입니다. IBM MQ 데이터 유형 정의를 변경하는 경우 이러한 오브젝트를 다시 작성하는 것은 비교적 간단한 문제입니다.



## 기본 데이터 유형

이 절에 설명된 모든 다른 데이터 유형은 이러한 요소 데이터 유형 또는 이러한 요소 데이터 유형의 집계(배열 또는 구조)와 직접적으로 동일합니다.

표 679. 기본 데이터 유형	
데이터 유형	표현
MQBOOL	10자리의 부호 있는 정수
MQBYTE	1바이트 영숫자 필드
MQBYTE16	16바이트 영숫자 필드
MQBYTE24	24바이트 영숫자 필드
MQBYTE32	32바이트 영숫자 필드
MQBYTE64	64바이트 영숫자 필드
MQCHAR	1바이트 영숫자 필드
MQCHAR4	4바이트 영숫자 필드
MQCHAR8	8바이트 영숫자 필드
MQCHAR12	12바이트 영숫자 필드
MQCHAR16	16바이트 영숫자 필드
MQCHAR20	20바이트 영숫자 필드
MQCHAR28	28바이트 영숫자 필드
MQCHAR32	32바이트 영숫자 필드
MQCHAR48	48바이트 영숫자 필드
MQCHAR64	64바이트 영숫자 필드
MQCHAR128	128바이트 영숫자 필드
MQCHAR256	256바이트 영숫자 필드
MQFLOAT32	4바이트 부동 소수점 수
MQFLOAT64	8바이트 부동 소수점 수
MQHCONFIG	구성 핸들
MQHCONN	10자리의 부호 있는 정수
MQHMSG	메시지에 대한 액세스를 허용하는 메시지 핸들
MQHOBJ	10자리의 부호 있는 정수
MQINT8	8비트 부호 있는 정수
MQINT16	16비트 부호 있는 정수
MQINT32	32비트 부호 있는 정수
MQINT64	64비트 부호 있는 정수
MQLONG	32비트 부호 있는 정수
MQPID	프로세스 ID
MQPTR	포인터

표 679. 기본 데이터 유형 (계속)

데이터 유형	표현
MQTID	스레드 ID
MQUINT8	8비트 부호 없는 정수
MQUINT16	16비트 부호 없는 정수
MQUINT32	32비트 부호 없는 정수
MQUINT64	64비트 부호 없는 정수
MQULONG	32비트 부호 없는 정수
PMQACH	MQACH 유형 데이터 구조에 대한 포인터
PMQAIR	MQAIR 유형 데이터 구조에 대한 포인터
PMQAXC	MQAXC 유형 데이터 구조에 대한 포인터
PMAXP	유형 MAXP의 데이터 구조에 대한 포인터
PMQBMHO	MQBMHO 유형 데이터 구조에 대한 포인터
PMQBO	MQBO 유형 데이터 구조에 대한 포인터
PMQBOOL	MQBOOL 유형 데이터에 대한 포인터
PMQBYTE	MQBYTE 유형 데이터에 대한 포인터
PMQBYTE <sub>n</sub>	유형 MQBYTE <sub>n</sub> 의 데이터에 대한 포인터
PMQCBC	MQCBC 유형 데이터 구조에 대한 포인터
PMQCBD	MQCBD 유형 데이터 구조에 대한 포인터
PMQCHAR	유형 MQCHAR의 데이터 구조에 대한 포인터
PMQCHARV	MQCHARV 유형 데이터 구조에 대한 포인터
PMQCHAR <sub>n</sub>	유형 MQCHAR <sub>n</sub> 의 데이터에 대한 포인터
PMQCIH	MQCIH 유형 데이터 구조에 대한 포인터
PMQCMHO	MQCMHO 유형 데이터 구조에 대한 포인터
PMQCNO	MQCNO 유형 데이터 구조에 대한 포인터
PMQCSP	MQCSP 유형 데이터 구조에 대한 포인터
PMQCTLO	MQCTLO 유형 데이터 구조에 대한 포인터
PMQDH	MQDH 유형 데이터 구조에 대한 포인터
PMQDHO	MQDHO 유형 데이터 구조에 대한 포인터
PMQDLH	MQDLH 유형 데이터 구조에 대한 포인터
PMQDMHO	MQDMHO 유형 데이터 구조에 대한 포인터
PMQDMPO	MQDMPO 유형 데이터 구조에 대한 포인터
PMQEPH	MQEPH 유형 데이터 구조에 대한 포인터
PMQFLOAT32	유형 MQFLOAT32의 데이터에 대한 포인터
PMQFLOAT64	유형 MQFLOAT64의 데이터에 대한 포인터
PMQFUNC	함수에 대한 포인터

표 679. 기본 데이터 유형 (계속)

데이터 유형	표현
PMQGM0	MQGM0 유형 데이터 구조에 대한 포인터
PMQHCONFIG	MQHCONFIG 유형 데이터에 대한 포인터
PMQHCONN	MQHCONN 유형 데이터에 대한 포인터
PMQHMSG	MQHMSG 유형 데이터에 대한 포인터
PMQHOBJ	MQHOBJ 유형 데이터에 대한 포인터
PMQIIH	MQIIH 유형 데이터 구조에 대한 포인터
PMQIMPO	MQIMPO 유형 데이터 구조에 대한 포인터
PMQINT8	MQINT8 유형 데이터에 대한 포인터
PMQINT16	MQINT16 유형 데이터에 대한 포인터
PMQINT32	MQINT32 유형 데이터에 대한 포인터
PMQINT64	MQINT64 유형 데이터에 대한 포인터
PMQLONG	MQLONG 유형 데이터에 대한 포인터
PMQMD	MQMD 유형 데이터 구조에 대한 포인터
PMQMDE	MQMDE 유형 데이터 구조에 대한 포인터
PMQMD1	MQMD1 유형 데이터 구조에 대한 포인터
PMQMD2	MQMD2 유형 데이터 구조에 대한 포인터
PMQMHBO	MQMHBO 유형 데이터 구조에 대한 포인터
PMQOD	MQOD 유형 데이터 구조에 대한 포인터
PMQOR	MQOR 유형 데이터 구조에 대한 포인터
PMQPD	MQPD 유형 데이터 구조에 대한 포인터
PMQPID	프로세스 ID MQPID에 대한 포인터
PMQPMO	MQPMO 유형 데이터 구조에 대한 포인터
PMQPTR	MQPTR 유형 데이터에 대한 포인터
PMQRFH	MQRFH 유형 데이터 구조에 대한 포인터
PMQRFH2	MQRFH2 유형 데이터 구조에 대한 포인터
PMQRMH	MQRMH 유형 데이터 구조에 대한 포인터
PMQRR	MQRR 유형 데이터 구조에 대한 포인터
PMQSCO	MQSCO 유형 데이터 구조에 대한 포인터
PMQSD	MQSD 유형 데이터 구조에 대한 포인터
PMQSMPO	MQSMPO 유형 데이터 구조에 대한 포인터
PMQSRO	MQSRO 유형 데이터 구조에 대한 포인터
PMQSTS	MQSTS 유형 데이터 구조에 대한 포인터
PMQTID	스레드 식별기 MQTID에 대한 포인터
PMQTM	MQTM 유형 데이터 구조에 대한 포인터

표 679. 기본 데이터 유형 (계속)	
데이터 유형	표현
PMQTMCM2	MQTMCM2 유형 데이터 구조에 대한 포인터
PMQUINT8	유형 MQUINT8의 데이터에 대한 포인터
PMQUINT16	유형 MQUINT16의 데이터에 대한 포인터
PMQUINT32	유형 MQUINT32의 데이터에 대한 포인터
PMQUINT64	유형 MQUINT64의 데이터에 대한 포인터
PMQULONG	유형 MQULONG의 데이터에 대한 포인터
PMQVOID	포인터
PMQWIH	MQWIH 유형 데이터 구조에 대한 포인터
PMQXQH	MQXQH 유형 데이터 구조에 대한 포인터

### IBM i IBM i용 MQBOOL

MQBOOL 데이터 유형은 부울 값을 나타냅니다. 값 0은 false로 나타냅니다. 기타 값은 true를 나타냅니다.

MQBOOL은 MQLONG 데이터 유형에 대해서와 같이 맞추어야 합니다.

### IBM i IBM i용 MQBYTE

MQBYTE 데이터 유형은 1바이트의 데이터를 나타냅니다.

바이트에 대한 특별한 해석이 없으며 2진수 또는 문자가 아닌 비트 문자열로 처리됩니다. 특별하게 정렬하지 않아도 됩니다.

MQBYTE의 배열은 큐 관리자로 알려지지 않는 네이처로 주 기억장치의 영역을 표시하는 데 사용됩니다. 예를 들면, 영역은 애플리케이션 메시지 데이터 또는 구조를 포함할 수 있습니다. 이 영역의 경계 맞추기는 포함된 데이터의 네이처와 호환 가능해야 합니다.

### IBM i IBM i의 MQBYTEn(n바이트 문자열)

각 MQBYTEn 데이터 유형은 n바이트의 문자열을 나타냅니다.

여기서 n은 다음 값 중 하나를 사용할 수 있습니다.

- 16, 24, 32, 또는 64.

각 바이트는 MQBYTE 데이터 유형으로 설명됩니다. 특별하게 정렬하지 않아도 됩니다.

문자열의 데이터가 정의된 길이의 문자열보다 짧으면 문자열을 채우기 위해 데이터가 널로 채워져야 합니다.

큐 관리자가 바이트 문자열을 애플리케이션으로 리턴하면(예: MQGET 호출의 경우) 큐 관리자는 항상 문자열의 정의된 길이에 널로 채워집니다.

바이트 문자열 필드의 길이를 정의하는 상수가 사용 가능합니다.

### IBM i IBM i의 MQCHAR(문자)

MQCHAR 데이터 유형은 1개의 문자를 나타냅니다.

특징의 코드화된 문자 세트 ID는 큐 관리자의 ID입니다(주제 [CodedCharSetId](#)에서 **CodedCharSetId** 속성 참조). 특별하게 정렬하지 않아도 됩니다.

**참고:** MQGET, MQPUT 및 MQPUT1 호출에 지정된 애플리케이션 메시지 데이터는 MQCHAR 데이터 유형이 아니라 MQBYTE 데이터 유형에 의해 설명됩니다.

### IBM i IBM i의 MQCHARn(n자 문자열)

각 MQCHARn 데이터 유형은 일련의 n바이트를 나타냅니다.

여기서  $n$ 은 다음 값 중 하나를 사용할 수 있습니다.

• 4, 8, 12, 16, 20, 28, 32, 48, 64, 128 또는 256

각 문자는 MQCHAR 데이터 유형으로 설명됩니다. 특별하게 정렬하지 않아도 됩니다.

문자열의 데이터가 정의된 길이의 문자열보다 짧으면 문자열을 채우기 위해 데이터가 공백으로 채워집니다. 일부 경우에 널 문자는 공백으로 채우는 대신 중간에 문자열을 끝내는 데 사용될 수 있습니다. 뒤에 오는 널 문자 및 문자는 문자열의 정의된 길이까지 공백으로 처리됩니다. 널이 사용될 수 있는 위치는 호출 및 데이터 유형 설명에서 식별됩니다.

큐 관리자가 문자열을 애플리케이션으로 리턴하면(예: MQGET 호출의 경우) 큐 관리자는 항상 문자열의 정의된 길이에 공백으로 채워집니다. 큐 관리자는 널 문자를 사용하여 문자열을 구분하지 않습니다.

문자열 필드의 길이를 정의하는 상수가 사용 가능합니다.

### IBM i IBM i용 MQFLOAT32

MQFLOAT32 데이터 유형은 표준 IEEE 부동 소수점 형식을 사용하여 표시되는 32비트 부동 소수점 숫자입니다. MQFLOAT32는 4바이트 경계에 맞춰야 합니다.

### IBM i IBM i용 MQFLOAT64

MQFLOAT64 데이터 유형은 표준 IEEE 부동 소수점 형식을 사용하여 표시되는 64비트 부동 소수점 숫자입니다. MQFLOAT64는 8바이트 경계에 맞춰야 합니다.

### MQHCONFIG - 구성 핸들

MQHCONFIG 데이터 유형은 특정 설치 가능 서비스에 대해 구성 중인 컴포넌트인 구성 핸들을 표시합니다. 구성 핸들은 자연적인 경계에 맞춰야 합니다.

**참고:** 애플리케이션은 동일성에 대해서만 이 변수 유형을 테스트해야 합니다.

### IBM i IBM i의 MQHCONN(연결 핸들)

MQHCONN 데이터 유형은 연결 핸들, 즉 특정 큐 관리자에 대한 연결을 표시합니다. 연결 핸들은 자연적인 경계에 맞춰야 합니다.

**참고:** 애플리케이션은 동일성에 대해서만 이 변수 유형을 테스트해야 합니다.

### IBM i IBM i의 MQHMSG(메시지 핸들)

MQHMSG 데이터 유형은 메시지에 대한 액세스를 제공하는 메시지 핸들을 나타냅니다. 메시지 핸들은 8바이트 경계에 맞춰야 합니다.

**참고:** 애플리케이션은 동일성에 대해서만 이 변수 유형을 테스트해야 합니다.

### IBM i IBM i의 MQHOBJ(오브젝트 핸들)

MQHOBJ 데이터 유형은 오브젝트에 액세스할 수 있는 오브젝트 핸들을 나타냅니다. 오브젝트 핸들은 자연적인 경계에 맞춰야 합니다.

**참고:** 애플리케이션은 동일성에 대해서만 이 변수 유형을 테스트해야 합니다.

### IBM i IBM i의 MQINT8(8비트 부호있는 정수)

컨텍스트에 의해 제한되지 않는 한 MQINT8 데이터 유형은 -128에서 +127까지 범위의 값을 가져올 수 있는 8비트 부호 있는 정수입니다.

### IBM i IBM i의 MQINT16(16비트 부호있는 정수)

컨텍스트에 달리 제한되어 있지 않는 한, MQINT16 데이터 유형은 -32 768에서 +32 767까지 범위의 아무 값을 사용할 수 있는 16비트 부호 있는 정수입니다.

MQINT16은 2바이트 경계에 맞춰야 합니다.

### **IBM i IBM i의 MQINT32(32비트 부호있는 정수)**

MQINT32 데이터 유형은 32비트 부호 있는 정수입니다.

MQLONG와 동일합니다.

### **IBM i IBM i의 MQINT64(64비트 부호있는 정수)**

컨텍스트에 의해 제한되지 않는 한 MQINT64 데이터 유형은 -9 223 372 036 854 775 808에서 +9 223 372 036 854 775 807까지 범위의 값을 가져올 수 있는 64비트 부호 있는 정수입니다.

COBOL의 경우, 올바른 범위는 -999 999 999 999 999에서 +999 999 999 999 999로 제한됩니다. MQINT64는 8바이트 경계에 맞춰야 합니다.

### **IBM i IBM i의 MQLONG(긴 정수)**

해당 자연적인 경계에 정렬되는 컨텍스트에 의해 제한되지 않는 한 MQLONG 데이터 유형은 -2 147 483 648에서 +2 147 483 647까지 범위의 값을 가져올 수 있는 32비트 서명된 2진 정수입니다.

### **MQPID - 프로세스 ID**

IBM MQ 프로세스 ID입니다.

이는 IBM MQ 추적 및 FFST 덤프에서 사용된 동일 ID이지만 운영 체제 프로세스 ID와 다를 수 있습니다.

### **MQPTR - 포인터**

MQPTR 데이터 유형은 임의 유형의 데이터의 주소입니다. 포인터는 자연 경계에 맞게 정렬해야 합니다. 이는 IBM i의 16바이트 경계입니다.

일부 프로그래밍 언어는 입력된 포인터를 지원합니다. 또한 MQI는 몇 가지 경우에 이를 사용합니다.

### **MQTID - 스레드 ID**

MQ 스레드 ID.

이 ID는 MQ 추적과 FFST 덤프에 사용된 ID와 동일하지만, 운영 체제 스레드 ID와 다를 수 있습니다.

### **IBM i IBM i의 MQUINT8(8비트 부호 없는 정수)**

컨텍스트에 의해 제한되지 않는 한 MQUINT8 데이터 유형은 0에서 +255까지 범위의 값을 가져올 수 있는 8비트 부호 없는 정수입니다.

### **MQUINT16 - 16비트 부호 없는 정수**

컨텍스트에 달리 제한되어 있지 않는 한, MQUINT16 데이터 유형은 범위 0 ~ +65 535 범위의 값을 사용할 수 있는 16비트 부호없는 정수입니다.

MQUINT16은 2바이트 경계에 맞춰야 합니다.

### **IBM i IBM i의 MQUINT32(32비트 부호 없는 정수)**

MQUINT32 데이터 유형은 32비트 부호 없는 정수입니다. MQULONG와 동일합니다.

### **MQUINT64 - 64비트 부호 없는 정수**

컨텍스트에 의해 제한되지 않는 한 MQUINT64 데이터 유형은 0에서 +18 446 744 073 709 551 615까지 범위의 값을 가져올 수 있는 64비트 부호 없는 정수입니다.

COBOL의 경우, 올바른 범위는 0에서 +999 999 999 999 999로 제한됩니다. MQUINT64는 8바이트 경계에 맞춰야 합니다.

**MQULONG - 32비트 부호 없는 정수**

MQULONG 데이터 유형은 컨텍스트에서 달리 제한하지 않는 한, 0에서 +4 294 967 294까지 범위의 모든 값을 사용할 수 있는 32비트 부호 없는 이진 정수입니다.

MQULONG은 4바이트 경계에 맞춰야 합니다.

**PMQACH - 유형 MQACH의 데이터 구조에 대한 포인터**

유형 MQACH의 데이터 구조에 대한 포인터입니다.

**PMQAIR - 유형 MQAIR의 데이터 구조에 대한 포인터**

유형 MQAIR의 데이터 구조에 대한 포인터입니다.

**PMQAXC - 유형 MQAXC의 데이터 구조에 대한 포인터**

유형 MQAXC의 데이터 구조에 대한 포인터입니다.

**PMQAXP - 유형 MQAXP의 데이터 구조에 대한 포인터**

유형 MQAXP의 데이터 구조에 대한 포인터입니다.

**PMQBMHO - 유형 MQBMHO의 데이터 구조에 대한 포인터**

유형 MQBMHO의 데이터 구조에 대한 포인터입니다.

**PMQBO - 유형 MQBO의 데이터 구조에 대한 포인터**

유형 MQBO의 데이터 구조에 대한 포인터입니다.

**PMQBOOL - 유형 MQBOOL의 데이터에 대한 포인터**

유형 MQBOOL의 데이터에 대한 포인터입니다.

유형 MQBOOL의 데이터에 대한 포인터입니다.

**PMQBYTE - MQBYTE의 데이터 유형에 대한 포인터**

MQBYTE의 데이터 유형에 대한 포인터입니다.

**PMQBYTE<sub>n</sub> - 유형 MQBYTE<sub>n</sub>의 데이터 구조에 대한 포인터**

유형 MQBYTE<sub>n</sub>의 데이터 구조에 대한 포인터입니다. 여기서 n은 8, 12, 16, 24, 32, 40, 48 또는 128일 수 있습니다.

**PMQCBC - 유형 MQCBC의 데이터 구조에 대한 포인터**

유형 MQCBC의 데이터 구조에 대한 포인터입니다.

**PMQCBD - 유형 MQCBD의 데이터 구조에 대한 포인터**

유형 MQCBD의 데이터 구조에 대한 포인터입니다.

**PMQCHAR - 유형 MQCHAR의 데이터에 대한 포인터**

유형 MQCHAR의 데이터에 대한 포인터입니다.

**PMQCHARV - 유형 MQCHARV의 데이터 구조에 대한 포인터**

유형 MQCHARV의 데이터 구조에 대한 포인터입니다.

**PMQCHARn - MQCHARn의 데이터 유형에 대한 포인터**

MQCHARn의 데이터 유형에 대한 포인터입니다. 여기서 n은 4, 8, 12, 20, 28, 32, 64, 128, 256, 264일 수 있습니다.

**PMQCIH - MQCIH 유형의 데이터 구조에 대한 포인터**

MQCIH 유형의 데이터 구조에 대한 포인터입니다.

**PMQCMHO - 유형 MQCMHO의 데이터 구조에 대한 포인터**

유형 MQCMHO의 데이터 구조에 대한 포인터입니다.

**PMQCNO - MQCNO 유형의 데이터 구조에 대한 포인터**

MQCNO 유형의 데이터 구조에 대한 포인터입니다.

**PMQCSP - 유형 MQCSP의 데이터 구조에 대한 포인터**

유형 MQCSP의 데이터 구조에 대한 포인터입니다.

**PMQCTLO - 유형 MQCTLO의 데이터 구조에 대한 포인터**

유형 MQCTLO의 데이터 구조에 대한 포인터입니다.

**PMQDHF - 유형 MQDHF의 데이터 구조에 대한 포인터**

유형 MQDHF의 데이터 구조에 대한 포인터입니다.

**PMQDHO - 유형 MQDHO의 데이터 구조에 대한 포인터**

유형 MQDHO의 데이터 구조에 대한 포인터입니다.

**PMQDLH - MQDLH 유형의 데이터 구조에 대한 포인터**

MQDLH 유형의 데이터 구조에 대한 포인터입니다.

**PMQDMHO - 유형 MQDMHO의 데이터 구조에 대한 포인터**

유형 MQDMHO의 데이터 구조에 대한 포인터입니다.



**PMQDMPO - 유형 MQDMPO의 데이터 구조에 대한 포인터**

유형 MQDMPO의 데이터 구조에 대한 포인터입니다.

유형 MQDMPO의 데이터 구조에 대한 포인터입니다.

**PMQEPPH - 유형 MQEPPH의 데이터 구조에 대한 포인터**

유형 MQEPPH의 데이터 구조에 대한 포인터입니다.

**PMQFLOAT32 - 유형 MQFLOAT32의 데이터에 대한 포인터**

유형 MQFLOAT32의 데이터에 대한 포인터입니다.

**PMQFLOAT64 - 유형 MQFLOAT64의 데이터에 대한 포인터**

유형 MQFLOAT64의 데이터에 대한 포인터입니다.

**PMQFUNC - 함수에 대한 포인터**

함수에 대한 포인터입니다.

**PMQGMPO - 유형 MQGMPO의 데이터 구조에 대한 포인터**

유형 MQGMPO의 데이터 구조에 대한 포인터입니다.

**PMQHCONFIG - MQHCONFIG의 데이터 유형에 대한 포인터**

MQHCONFIG의 데이터 유형에 대한 포인터입니다.

**PMQHCONN - MQHCONN의 데이터 유형에 대한 포인터**

MQHCONN의 데이터 유형에 대한 포인터입니다.

**PMQHMSG - MQHMSG의 데이터 유형에 대한 포인터**

MQHMSG의 데이터 유형에 대한 포인터입니다.

**PMQHOBJS - 유형 MQHOBJS의 데이터에 대한 포인터**

유형 MQSMPO의 데이터에 대한 포인터입니다.

**PMQIIH - 유형 MQIIH의 데이터 구조에 대한 포인터**

유형 MQIIH의 데이터 구조에 대한 포인터입니다.

**PMQIMPO - 유형 MQIMPO의 데이터 구조에 대한 포인터**

유형 MQIMPO의 데이터 구조에 대한 포인터입니다.

**PMQINT8 - 유형 MQINT8의 데이터에 대한 포인터**

유형 MQINT8의 데이터에 대한 포인터입니다.

### **PMQINT16 - 유형 MQINT16의 데이터에 대한 포인터**

유형 MQINT16의 데이터에 대한 포인터입니다.

### **IBM i IBM i의 PMQINT32(유형 MQINT32의 데이터에 대한 포인터)**

PMQINT32 데이터 유형은 유형 MQINT32의 데이터에 대한 포인터입니다. PMQLONG와 동일합니다.

### **IBM i IBM i의 PMQINT64(유형 MQINT64의 데이터에 대한 포인터)**

PMQINT64 데이터 유형은 유형 MQINT64의 데이터에 대한 포인터입니다.

### **PMQLONG - 유형 MQLONG의 데이터에 대한 포인터**

유형 MQLONG의 데이터에 대한 포인터입니다.

### **PMQMD - 유형 MQMD의 구조에 대한 포인터**

유형 MQMD의 구조에 대한 포인터입니다.

### **PMQMDE - 유형 MQMDE의 데이터 구조에 대한 포인터**

유형 MQMDE의 데이터 구조에 대한 포인터입니다.

### **PMQM DI - 유형 MQMDI의 데이터 구조에 대한 포인터**

유형 MQMDI의 데이터 구조에 대한 포인터입니다.

### **PMQMD2 - 유형 MQMD2의 데이터 구조에 대한 포인터**

유형 MQMD2의 데이터 구조에 대한 포인터입니다.

### **PMQMHBO - 유형 MQMHBO의 데이터 구조에 대한 포인터**

유형 MQMHBO의 데이터 구조에 대한 포인터입니다.

### **PMQOD - 유형 MQOD의 데이터 구조에 대한 포인터**

유형 MQOD의 데이터 구조에 대한 포인터입니다.

### **PMQOR - 유형 MQOR의 데이터 구조에 대한 포인터**

유형 MQOR의 데이터 구조에 대한 포인터입니다.

### **PMQPD - 유형 MQPD의 데이터 구조에 대한 포인터**

유형 MQPD의 데이터 구조에 대한 포인터입니다.

**PMQPID - 프로세스 ID에 대한 포인터**

프로세스 ID에 대한 포인터.

**PMQPMO - 유형 MQPMO의 데이터 구조에 대한 포인터**

유형 MQPMO의 데이터 구조에 대한 포인터입니다.

**PMQPTR - 유형 MQPTR의 데이터에 대한 포인터**

유형 MQPTR의 데이터에 대한 포인터입니다.

**PMQRFH - 유형 MQRFH의 데이터 구조에 대한 포인터**

유형 MQRFH의 데이터 구조에 대한 포인터입니다.

**PMQRFH2 - 유형 MQRFH2의 데이터 구조에 대한 포인터**

유형 MQRFH2의 데이터 구조에 대한 포인터입니다.

**PMQRMH - 유형 MQRMH의 데이터 구조에 대한 포인터**

유형 MQRMH의 데이터 구조에 대한 포인터입니다.

**PMQRR - 유형 MQRR의 데이터 구조에 대한 포인터**

유형 MQRR의 데이터 구조에 대한 포인터입니다.

**PMQSCO - 유형 MQSCO의 데이터 구조에 대한 포인터**

유형 MQSCO의 데이터 구조에 대한 포인터입니다.

**PMQSD - 유형 MQSD의 데이터 구조에 대한 포인터**

유형 MQSD의 데이터 구조에 대한 포인터입니다.

**PMQSMPO - 유형 MQSMPO의 데이터 구조에 대한 포인터**

유형 MQSMPO의 데이터 구조에 대한 포인터입니다.

**PMQSRO - 유형 MQSRO의 데이터 구조에 대한 포인터**

유형 MQSRO의 데이터 구조에 대한 포인터입니다.

**PMQSTS - 유형 MQSTS의 데이터 구조에 대한 포인터**

유형 MQSTS의 데이터 구조에 대한 포인터입니다.

**PMQTID - 유형 MQTID의 데이터 구조에 대한 포인터**

유형 MQTID의 데이터 구조에 대한 포인터입니다.

### **PMQTM - 유형 MQTM의 데이터 구조에 대한 포인터**

유형 MQTM의 데이터 구조에 대한 포인터입니다.

### **PMQTM2 - 유형 MQTM2의 데이터 구조에 대한 포인터**

유형 MQTM2의 데이터 구조에 대한 포인터입니다.

### **PMQUINT8 - 유형 MQUINT8의 데이터에 대한 포인터**

유형 MQUINT8의 데이터에 대한 포인터입니다.

### **PMQUINT16 - 유형 MQUINT16의 데이터에 대한 포인터**

유형 MQUINT16의 데이터에 대한 포인터입니다.

### **IBM i IBM i의 PMQUINT32(유형 MQUINT32의 데이터에 대한 포인터)**

PMQUINT32 데이터 유형은 유형 MQUINT32의 데이터에 대한 포인터입니다. PMQULONG와 동일합니다.

### **IBM i IBM i의 PMQUINT64(유형 MQUINT64의 데이터에 대한 포인터)**

PMQUINT64 데이터 유형은 유형 MQUINT64의 데이터에 대한 포인터입니다.

### **PMQULONG - 유형 MQULONG의 데이터에 대한 포인터**

유형 MQULONG의 데이터에 대한 포인터입니다.

### **PMQVOID - 포인터**

포인터입니다.

### **PMQWIH - 유형 MQWIH의 데이터 구조에 대한 포인터**

유형 MQWIH의 데이터 구조에 대한 포인터입니다.

### **PMQXQH - 유형 MQXQH의 데이터 구조에 대한 포인터**

유형 MQXQH의 데이터 구조에 대한 포인터입니다.

## **언어 고려사항**

이 토픽에는 RPG 프로그래밍 언어에서 MQI를 사용하는 데 도움이 되는 정보가 포함되어 있습니다.

이러한 언어 고려사항 중 일부는 다음과 같습니다.

- [969 페이지의 『COPY 파일』](#)
- [970 페이지의 『호출』](#)
- [970 페이지의 『호출 매개변수』](#)
- [971 페이지의 『구조』](#)

- 971 페이지의 『이름 지정된 상수』
- 971 페이지의 『MQI 프로시저』
- 971 페이지의 『스레드 고려사항』
- 972 페이지의 『커미트 제어』
- 972 페이지의 『바인딩된 호출 코드화』
- 973 페이지의 『표기 규정』

## COPY 파일

다양한 COPY 파일은 메시지 큐잉을 사용하는 RPG 애플리케이션 프로그램을 기록을 돕기 위해 제공됩니다. COPY 파일의 세 가지 세트가 있습니다.

- 문자 G로 끝나는 이름의 COPY 파일은 정적 연계를 사용하는 프로그램에서 사용됩니다. 이러한 파일은 971 페이지의 『구조』에 명시된 예외로 초기화됩니다.
- 문자 H로 끝나는 이름의 COPY 파일은 정적 연계를 사용하는 프로그램에서 사용되지만 초기화되지 **않습니다**.
- 문자 R로 끝나는 이름의 COPY 파일은 동적 연계를 사용하는 프로그램에서 사용됩니다. 이러한 파일은 971 페이지의 『구조』에 명시된 예외로 초기화됩니다.

COPY 파일은 QMQM 라이브러리의 QRPGLSRC에 상주합니다.

COPY 파일의 각 세트에 대해 이름 지정된 상수 및 각각의 구조에 대한 하나의 파일을 포함하는 2개 파일이 있습니다. COPY 파일은 969 페이지의 표 680에 요약됩니다.

파일 이름(정적 연계, 초기화됨, <b>CMQ*G</b> )	파일 이름(정적 연계, 초기화되지 않음, <b>CMQ*H</b> )	파일 이름(동적 연계, 초기화됨, <b>CMQ*R</b> )	컨텐츠
CMQBOG	CMQBOH	-	시작 옵션 구조
CMQCDG	CMQCDH	CMQCDR	채널 정의 구조
CMQCFBFG	CMQCFBFH	-	PCF 비트 필터 매개변수
CMQCFG	-	-	PCF 및 이벤트에 대한 상수
CMQCFBSG	CMQCFBSH	-	PCF 바이트 문자열
CMQCFGRG	CMQCFGRH	-	PCF 그룹 매개변수
CMQCFIFG	CMQCFIFH	-	PCF 정수 필터 매개변수
CMQCFHG	CMQCFHH	-	PCF 헤더
CMQCFILG	CMQCFILH	-	PCF 정수 목록 매개변수 구조
CMQCFING	CMQCFINH	-	PCF 정수 매개변수 구조
CMQCFSG	CMQCFSFH	-	PCF 문자열 필터 매개변수
CMQCFSLG	CMQCFSLH	-	PCF 문자열 목록 매개변수 구조
CMQCFSTG	CMQCFSTH	-	PCF 문자열 매개변수 구조
CMQCFXLG	CMQCFXLH	-	CFIL64를 위한 PCF 단축 이름
CMQCFXNG	CMQCFXNH	-	CFIN64를 위한 PCF 단축 이름
CMQCIHG	CMQCIHH	-	CICS 정보 헤더 구조
CMQCNQG	CMQCNQH	-	연결 옵션 구조
CMQCSPG	CMQCSPH	-	보안 매개변수

표 680. RPG COPY 파일 (계속)

파일 이름(정적 연계, 초기화됨, CMQ*G)	파일 이름(정적 연계, 초기화되지 않음, CMQ*H)	파일 이름(동적 연계, 초기화됨, CMQ*R)	컨텐츠
CMQCXPG	CMQCXPH	CMQCXPR	채널 엑시트 매개변수 구조
CMQDHG	CMQDHH	CMQDHR	분산 헤더 구조
CMQDLHG	CMQDLHH	CMQDLHR	데드-레터 헤더 구조
CMQDXPG	CMQDXPH	CMQDXPR	데이터 변환 엑시트 매개변수 구조
CMQEPHG	CMQEPHH	-	임베드된 PCF 헤더 구조
CMQG	-	CMQR	기본 MQI의 이름 지정된 상수
CMQGMOG	CMQGMOH	CMQGMOR	메시지 가져오기 옵션 구조
CMQIIHG	CMQIIHH	CMQIIHR	IMS 정보 헤더 구조
CMQMDEG	CMQMDEH	CMQMDER	메시지 디스크립터 확장 구조
CMQMDG	CMQMDH	CMQMDR	메시지 디스크립터 구조
CMQMD1G	CMQMD1H	CMQMD1R	메시지 디스크립터 구조 버전 1
CMQMD2G	CMQMD2H	-	메시지 디스크립터 구조 버전 2
CMQODG	CMQODH	CMQODR	오브젝트 디스크립터 구조
CMQORG	CMQORH	CMQORR	오브젝트 레코드 구조
CMQPMOG	CMQPMOH	CMQPMOR	메시지 넣기 옵션 구조
CMQPSG	-	-	발행/구독에 대한 상수
CMQRFHG	CMQRFHH	-	규칙 및 형식화 헤더 구조
CMQRFH2G	CMQRFH2H	-	규칙 및 형식화 헤더 2 구조
CMQRMHG	CMQRMHH	CMQRMHR	참조 메시지 헤더 구조
CMQRRG	CMQRRH	CMQRRR	응답 레코드 구조
CMQTMCG	CMQTMCH	CMQTMCR	트리거 메시지 구조(문자 형식)
CMQTMCG2G	CMQTMCG2H	CMQTMCG2R	트리거 메시지 구조(문자 형식) 버전 2
CMQTMG	CMQTMH	CMQTMR	트리거 메시지 구조
CMQWIHG	CMQWIHH	-	작업 정보 헤더 구조
CMQXG	-	CMQXR	데이터 변환 엑시트에 대한 이름 지정된 상수
CMQXQHG	CMQXQHH	CMQXQHR	전송 큐 헤더 구조

## 호출

호출은 해당 개별 이름을 사용하여 설명됩니다.

## 호출 매개변수

MQI에 전달된 일부 매개변수는 둘 이상의 동시 기능을 가질 수 있습니다. 이는 전달된 정수 값이 종종 전체 값이 아닌 필드 내의 개별 비트 설정에서 테스트되기 때문입니다. 이렇게하면 여러 함수를 함께 '추가'하고 단일 매개변수로 전달할 수 있습니다.

## 구조

모든 IBM MQ 구조는 다음 예외를 제외하고 필드에 대한 초기값으로 정의됩니다.

- H의 접미부의 구조.
- MQTMC
- MQTMC2

이러한 초기값은 각 구조에 대한 관련 테이블에서 정의됩니다.

구조 선언은 DS 명령문을 포함하지 않습니다. 이를 통해 애플리케이션은 DS 명령문을 코드화한 후 /COPY 명령문을 사용하여 선언의 나머지 부분을 복사하여 단일 데이터 구조 또는 다중 발생 데이터 구조를 선언할 수 있습니다.

```
D*..1.....2.....3.....4.....5.....6.....7
D* Declare an MQMD data structure with 5 occurrences
DMYMD      DS          5
D/COPY CMQMDR
```

## 이름 지정된 상수

애플리케이션 프로그램과 큐 관리자 사이에 데이터 교환을 제공하는 많은 정수와 문자값이 있습니다. 이러한 값을 사용하는 것의 더 읽기 쉽고 지속적 접근을 용이하게 하기 그들을 위해, 이름 달린 상수는 그들에 대해 정의됩니다. 이것이 프로그램 소스 코드의 가독성을 향상시킨 것처럼, 나타내는 값이 아니라 이러한 이름 달린 상수를 사용할 수 있습니다.

COPY 파일 CMQOI 상수를 정의하려면 프로그램에 포함될 때, RPG 컴파일러는 프로그램에 의해 사용되지 않는 상수에게 많은 심각한 0 메시지를 실행합니다. 이러한 메시지는 좋고, 안전하게 무시당할 수 있습니다.

## MQI 프로시저

ILE 바운드 호출을 사용할 때, 프로그램을 작성할 때 MQI 프로시저를 바인드해야 합니다. 이러한 프로시저는 적절한 것으로 다음 서비스 프로그램에서 내보내기됩니다.

### QMOM/LIBMOM

이 서비스 프로그램은 버전 5.1 이상의 위한 단일 스레드 바인딩을 포함합니다. 스레드한 애플리케이션을 작성할 때 특수 고려사항은 다음 절을 참조하십시오.

### QMOM/LIBMOM\_R

이 서비스 프로그램은 버전 5.1 이상의 다중 스레드 바인딩을 포함합니다. 스레드한 애플리케이션을 작성할 때 특수 고려사항은 다음 절을 참조하십시오.

### QMOM/LIBMOMC

이 서비스 프로그램은 스레드되지 않은 클라이언트 애플리케이션의 바인딩용입니다.

### QMOM/LIBMOMC\_R

이 서비스 프로그램은 스레드되지 않은 클라이언트 애플리케이션의 바인딩용입니다.

CRTPGM 명령을 사용하여 프로그램을 작성하십시오. 예를 들어, 다음 명령은 ILE 바운드 호출을 사용하는 단일 스레드 프로그램을 작성합니다.

```
CRTPGM PGM(MYPROGRAM) BNDSRVPGM(QMOM/LIBMOM)
```

## 스레드 고려사항

IBM i 에 사용되는 RPG 컴파일러는 WebSphere Development Toolset 및 WebSphere Development Studio for IBM i 의 일부이며 ILE RPG IV 컴파일러로 알려져 있습니다.

일반적으로 RPG 프로그램은 멀티스레드 서비스 프로그램을 사용하지 않아야 합니다. 예외는 ILE RPG IV 컴파일러를 사용하고 제어 스펙에 THREAD(\*SERIALIZE) 키워드를 포함하여 작성된 RPG 프로그램입니다. 그러나 이러한 프로그램은 스레드 안전이지만, THREAD(\*SERIALIZE) 모듈 레벨에서 RPG 프로시저의 직렬화를 강제 실행하므로 전반적인 애플리케이션 설계에 주의를 기울여야 합니다. 이는 전체 성능에 악영향을 줄 수 있습니다.

RPG 프로그램이 데이터 변환 엑시트로 사용되는 경우, 스레드에 안전하게 작성되어야 하며 제어 스펙에 THREAD(\*SERIALIZE)이(가) 지정된 위의 4.4 ILE RPG 컴파일러 이상을 사용하여 다시 컴파일되어야 합니다.

스레드에 대한 추가 정보는 *IBM i IBM MQ Development Studio: ILE RPG* 참조 및 *IBM i IBM MQ Development Studio: ILE RPG* 프로그래머 안내서의 내용을 참조하십시오.

## 커미트 제어

MQI 동기점 함수 MQCMIT 및 MQBACK는 정상 모드에서 실행 중인 ILE RPG 프로그램에 사용 가능하며 이러한 호출로 프로그램이 MQ 자원에 대한 변경을 커미트하고 백아웃할 수 있습니다.

## 바인딩된 호출 코드화

MQI ILE 프로시저는 972 페이지의 표 681에 나열됩니다.

표 681. 각 서비스 프로그램에서 지원되는 ILE RPG 바인드된 호출		
호출의 이름	LIBMQM 및 LIBMQM_R	LIBMQIC 및 LIBMQIC_R
MQBACK	Y	Y
MQBEGIN	Y	Y
MQCMIT	Y	Y
MQCLOSE	Y	Y
MQCONN	Y	Y
MQCONNX	Y	Y
MQDISC	Y	Y
MQGET	Y	Y
MQINQ	Y	Y
MQOPEN	Y	Y
MQPUT	Y	Y
MQPUT1	Y	Y
MQSET	Y	Y
MQXCNVC	Y	Y

이러한 프로시저를 사용하려면 다음이 필요합니다.

1. 외부 프로시저를 사용자의 'D' 스펙에서 정의하십시오. 이는 이름 달린 상수를 포함하는 COPY 파일 멤버 CMQG 내에서 모두 사용 가능합니다.
2. 해당 매개변수와 함께 프로시저를 호출하기 위해 CALLP 조작 코드를 사용하십시오.

예를 들어, MQOPEN 호출에는 다음 코드의 포함이 필요합니다.

```
D*****
D** MQOPEN Call -- Open Object (From COPY file CMQG) **
D*****
D*
D*..1.....2.....3.....4.....5.....6.....7..
DMQOPEN PR EXTPROC('MQOPEN')
D* Connection handle
D HCONN 10I 0 VALUE
D* Object descriptor
D OBJDSC 224A
D* Options that control the action of MQOPEN
D OPTS 10I 0 VALUE
D* Object handle
D HOBJ 10I 0
```



```

D* Completion code
D CMPCOD 10I 0
D* Reason code qualifying CMPCOD
D REASON 10I 0
D*

```

다양한 매개변수를 초기화한 후 프로시저를 호출하려면 다음 코드가 필요합니다.

```

...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8
C          CALLP      MQOPEN(HCONN : MQOD : OPTS : HOBJ :
C          CMPCOD : REASON)

```

여기에서 구조 MQOD는 해당 컴포넌트로 분류하는 COPY 멤버 CMQODG를 사용하여 정의됩니다.

## 표기 규정

이 절의 후자 부분의 주제에서는 다음을 설명합니다.

- 호출이 호출되어야 함
- 매개변수가 선언되어야 함
- 다양한 데이터 유형이 선언되어야 함

여러 경우에서 매개변수는 크기가 고정되지 않은 배열 또는 문자열입니다. 이에 대해 소문자 "n"이 숫자 상수를 나타내는 데 사용됩니다. 해당 매개변수에 대한 선언이 코드화되면, "n"이 필요한 숫자 값으로 바뀌어야 합니다.

## IBM i IBM i의 MQAIR(인증 정보 레코드)

MQAIR 구조는 인증 정보 레코드를 나타냅니다.

### 개요

**목적:** MQAIR 구조를 사용하여 IBM MQ 클라이언트로 실행하는 애플리케이션이 클라이언트 연결에 사용되는 인증자에 대한 정보를 지정할 수 있습니다. 이 구조는 MQCONNX 호출의 입력 매개변수입니다.

**문자 세트 및 인코딩:** MQAIR의 데이터는 ENNAT로 지정된 로컬 큐 관리자의 인코딩 및 **CodedCharSetId** 큐 관리자 속성으로 지정된 문자 세트에 있어야 합니다.

- [973 페이지의 『필드』](#)
- [975 페이지의 『초기값』](#)
- [975 페이지의 『RPG 선언』](#)

### 필드

MQAIR 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

#### AICN(10자리의 부호 있는 정수)

이는 LDAP 서버가 실행 중인 호스트의 호스트 이름 또는 네트워크 주소입니다. 이 뒤에는 선택적인 포트 번호를 괄호로 묶어 표시할 수 있습니다.

값이 필드의 길이보다 짧으면 널 문자로 값을 종료하거나 필드 길이에 맞게 공백으로 채웁니다. 이 값이 올바르지 않은 경우 이유 코드 RC2387로 호출에 실패합니다.

기본 포트 번호는 389입니다.

입력 필드입니다. 이 필드의 길이는 LNAICN로 제공됩니다. 이 필드의 초기값은 공백 문자입니다.

#### AITYP(10자리의 부호 있는 정수)

이는 레코드에 포함된 인증 정보의 유형입니다.

값은 다음과 같아야 합니다.

#### AITLDP

LDAP 서버를 사용하는 인증서 폐기.

이 값이 올바르지 않은 경우 이유 코드 RC2386으로 호출에 실패합니다.

입력 필드입니다. 이 필드의 초기값은 AITLDP입니다.

### **AIPW(10자리의 부호 있는 정수)**

이는 LDAP CRL 서버에 액세스하는 데 필요한 비밀번호입니다.

값이 필드의 길이보다 짧으면 널 문자로 값을 종료하거나 필드 길이에 맞게 공백으로 채웁니다. LDAP 서버에 비밀번호가 필요하지 않거나 LDAP 사용자 이름을 생략하는 경우 *AIPW*가 널이거나 비어 있어야 합니다.

LDAP 사용자 이름을 생략하고 *AIPW*가 널이 아니거나 비어 있는 경우 이유 코드 RC2390으로 호출에 실패합니다.

입력 필드입니다. 이 필드의 길이는 LNLDPW로 제공됩니다. 이 필드의 초기값은 공백 문자입니다.

### **AILUL(10자리의 부호 있는 정수)**

이는 *AILUP* 또는 *AILUO* 필드에서 처리된 LDAP 사용자 이름의 길이(바이트)입니다. 값은 0 - LNDISN의 범위에 있어야 합니다. 이 값이 올바르지 않은 경우 이유 코드 RC2389로 호출에 실패합니다.

포함된 LDAP 서버에 사용자 이름이 필요하지 않은 경우 이 필드를 0으로 설정하십시오.

입력 필드입니다. 이 필드의 초기값은 0입니다.

### **AILUO(10자리의 부호 있는 정수)**

이는 MQAIR 구조의 시작에서 LDAP 사용자 이름의 오프셋(바이트)입니다.

오프셋은 양수 또는 음수일 수 있습니다. *LDAPUserNameLength*(가) 0인 경우 필드는 무시됩니다.

*LDAPUserNamePtr* 또는 *LDAPUserNameOffset*을 사용하여 LDAP 사용자 이름을 지정할 수 있지만 둘 다 지정할 수 없습니다. 세부사항은 *LDAPUserNamePtr* 필드에 대한 설명을 참조하십시오.

입력 필드입니다. 이 필드의 초기값은 0입니다.

### **AILUP(10자리의 부호 있는 정수)**

이는 LDAP 사용자 이름입니다.

LDAP CRL 서버에 액세스 시도 중인 사용자의 고유 이름으로 구성됩니다. 이 값이 *AILUL*에 의해 지정된 길이보다 짧은 경우 널 특징이 있는 값을 종료하거나 길이 *AILUL*에 공백으로 채우십시오. *AILUL*가 0인 경우 필드가 무시됩니다.

다음 두 가지 방법 중 하나에 LDAP 사용자 이름을 제공할 수 있습니다.

- 포인터 필드 *AILUP* 사용

이 경우 애플리케이션은 MQAIR 구조와 별도로 문자열을 선언하고 문자열의 주소에 *AILUP*를 설정할 수 있습니다.

다른 환경(예: C 프로그래밍 언어)에 대해 휴대 가능한 방법으로 포인터 데이터 유형을 지원하는 프로그래밍 언어에 대한 *AILUP* 사용을 고려하십시오.

- 오프셋 필드 *AILUO* 사용

이 경우 애플리케이션은 LDAP 사용자 이름 문자열이 뒤에 오는 다음에 MQAIR 레코드의 배열이 오는 MQSCO 구조를 포함하는 복합 구조를 선언하고 MQAIR 구조의 시작에서 적절한 이름 문자열의 오프셋에 *AILUO*를 설정해야 합니다. 이 값이 정확하고 MQLONG 내에 수용할 수 있는 값이 있는지 확인하십시오(가장 제한적 프로그래밍 언어가 COBOL이며 올바른 범위는 -999 999 999 - +999 999 999임).

포인터 데이터 유형을 지원하지 않거나 다른 환경(예: COBOL 프로그래밍 언어)에 대해 휴대가 불가능할 수도 있는 방법으로 포인터 데이터 유형을 구현하는 프로그래밍 언어에 *AILUO* 사용을 고려하십시오.

어느 기술이 선택되든 *AILUP* 및 *AILUO* 중 하나만 사용합니다. 이유 코드 RC2388로 호출에 실패합니다.

입력 필드입니다. 이 필드의 초기값은 포인터를 지원하는 프로그래밍 언어로 된 널 포인터이며, 그렇지 않은 경우 모두 널 바이트 문자열입니다.

**참고:** 프로그래밍 언어가 포인터 데이터 유형을 지원하지 않는 플랫폼에서 이 필드는 적절한 길이의 바이트 문자열로 선언됩니다.

### AISID(10자리의 부호 있는 정수)

값은 다음과 같아야 합니다.

#### AISIDV

인증 정보 레코드의 ID.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 AISIDV입니다.

### AIVER(10자리의 부호 있는 정수)

값은 다음과 같아야 합니다.

#### AIVER1

버전-1 인증 정보 레코드.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### AIVERC

인증 정보 레코드의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 AIVER1입니다.

### 초기값

표 682. MQAIR에 대한 MQAIR의 필드		
필드 이름	상수의 이름	상수의 값
AISID	AISIDV	'AIR~'
AIVER	AIVERC	1
AITYP	AITLDP	1
AICN	없음	널 문자열 또는 공백
AILUP	없음	널 포인터 또는 널 바이트
AILUO	없음	0
AILUL	없음	0
AIPW	없음	널 문자열 또는 공백

### 참고:

1. ~ 기호는 단일 공백 문자를 나타냅니다.

### RPG 선언

```

D*..1.....2.....3.....4.....5.....6.....7..
D* MQAIR Structure
D*
D* Structure identifier
D AISID          1          4    INZ('AIR ')
D* Structure version number
D AIVER          5          8I 0  INZ(1)
D* Type of authentication information
D AITYP          9          12I 0  INZ(1)
D* Connection name of CRL LDAP server
D AICN          13         276    INZ
D* Address of LDAP user name
D AILUP         277        292*   INZ(*NULL)
D* Offset of LDAP user name from start of MQAIR structure
D AILUO         293        296I 0  INZ(0)
D* Length of LDAP user name
D AILUL         297        300I 0  INZ(0)
D* Password to access LDAP server
D AIPW          301        332    INZ

```

메시지 핸들 옵션에 대한 버퍼를 정의하는 구조입니다.

## 개요

**목적:** MQBMHO 구조를 사용하여 애플리케이션이 버퍼에서 생성되는 메시지 핸들을 제어하는 옵션을 지정할 수 있습니다. 구조는 MQBUFMH 호출의 입력 매개변수입니다.

**문자 세트 및 인코딩:** MQBMHO의 데이터는 애플리케이션의 문자 세트 및 애플리케이션의 인코딩(ENNAT)에 있어야 합니다.

- [976 페이지의 『필드』](#)
- [977 페이지의 『초기값』](#)
- [977 페이지의 『RPG 선언』](#)

## 필드

MQBMHO 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

### BMSID(10자리의 부호 있는 정수)

메시지 핸들 구조에 대한 버퍼 - StrucId 필드.

구조 ID입니다. 값은 다음과 같아야 합니다.

#### BMSIDV

버퍼 대 메시지 핸들 구조의 ID.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 BMSIDV입니다.

### BMVER(10자리의 부호 있는 정수)

메시지 핸들 구조에 대한 버퍼 - Version 필드.

구조 버전 번호입니다. 값은 다음과 같아야 합니다.

#### BMVER1

메시지 핸들 구조에 대한 버퍼의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### BMVERVC

메시지 핸들 구조에 대한 버퍼의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 BMVER1입니다.

### BMOPT(10자리의 부호 있는 정수)

메시지 핸들 구조에 대한 버퍼 - Options 필드.

가능한 값은 다음과 같습니다.

#### BMDLPR

메시지 핸들에 추가되는 특성이 메시지 버퍼에서 삭제됩니다. 호출이 실패하면 어떤 특성도 삭제되지 않습니다.

기본 옵션: 설명한 옵션 중 필요한 옵션이 없는 경우 다음 옵션을 사용하십시오.

#### BMNONE

옵션이 지정되지 않았습니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 BMDLPR입니다.

## 초기값

표 683. MQBMHO의 필드		
필드 이름	상수의 이름	상수의 값
BMSID	BMSIDV	'BMHO'
BMVER	BMVER1	1
BMOPT	BMNONE	0

## RPG 선언

```
D* MQBMHO Structure
D*
D*
D* Structure identifier
D BMSID 1 4 INZ('BMHO')
D*
D* Structure version number
D BMVER 5 8I 0 INZ(1)
D*
D* Options that control the action of MQBUFMH
D BMOPT 9 12I 0 INZ(1)
```

### IBM i IBM i의 MQBO(시작 옵션)

MQBO 구조를 사용하여 애플리케이션이 작업 단위의 작성에 관하여 옵션을 지정할 수 있습니다.

## 개요

목적: 이 구조는 MQBEGIN 호출의 입출력(I/O) 매개변수입니다.

문자 세트 인코딩: MQBO의 데이터는 **CodedCharSetId** 큐 관리자 속성에 의해 지정된 문자 세트 및 ENNAT에 의해 지정된 로컬 큐 관리자의 인코딩에 있어야 합니다.

- [977 페이지의 『필드』](#)
- [978 페이지의 『초기값』](#)
- [978 페이지의 『RPG 선언』](#)

## 필드

MQBO 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

### **BOOPT(10자리의 부호 있는 정수)**

MQBEGIN 조치를 제어하는 옵션입니다.

값은 다음과 같아야 합니다.

#### **BONONE**

옵션이 지정되지 않았습니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 BONONE입니다.

### **BOSID(4바이트 문자열)**

구조 ID입니다.

값은 다음과 같아야 합니다.

#### **BOSIDV**

시작 옵션 구조의 ID.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 BOSIDV입니다.

## BOVER(10자리의 부호 있는 정수)

구조 버전 번호입니다.

값은 다음과 같아야 합니다.

### BOVER1

시작 옵션 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

### BOVERC

시작 옵션 구조의 현재 버전

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 BOVER1입니다.

## 초기값

표 684. MQBO의 필드		
필드 이름	상수의 이름	상수의 값
BOSID	BOSIDV	'BO--'
BOVER	BOVER1	1
BOOPT	BONONE	0

### 참고:

1. - 기호는 단일 공백 문자를 나타냅니다.

## RPG 선언

```
D*..1.....:....2.....3.....4.....5.....6.....7...
D* MQBO Structure
D*
D* Structure identifier
D BOSID 1 4 INZ('BO ')
D* Structure version number
D BOVER 5 8I 0 INZ(1)
D* Options that control the action of MQBEGIN
D BOOPT 9 12I 0 INZ(0)
```

## IBM i IBM i의 MQCBC(콜백 컨텍스트)

콜백 루틴을 설명하는 구조입니다.

### 개요

#### 목적

MQCBC 구조는 콜백 함수에 전달되는 컨텍스트 정보를 지정하는 데 사용됩니다.

이 구조는 메시지 사용자 루틴에 대한 호출의 입출력(I/O) 매개변수입니다.

#### 버전

MQCBC의 현재 버전은 CBCV2입니다.

#### 문자 세트 및 인코딩

MQCBC의 데이터는 **CodedCharSetId** 큐 관리자 속성에 의해 지정된 문자 세트와 ENNAT에 의해 지정된 로컬 큐 관리자의 인코딩에 있습니다. 그러나 애플리케이션이 IBM MQ 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트와 인코딩에 있습니다.

- [979 페이지의 『필드』](#)
- [983 페이지의 『초기값』](#)

## 필드

MQCBC 구조에는 다음과 같은 필드가 포함됩니다. 필드에 대해서 알파벳순으로 설명합니다.

### **CBCBUFFLEN(10자리의 부호 있는 정수)**

버퍼는 MQGMO의 ReturnedLength 값 및 이용자에 대해 정의된 MaxMsgLength 값 모두보다 클 수 있습니다.

콜백 컨텍스트 구조 - BufferLength 필드.

이는 이 함수에 전달된 메시지 버퍼의 길이(바이트)입니다.

실제 메시지 길이는 DataLength 필드에 제공됩니다.

애플리케이션은 콜백 함수의 지속 기간동안 해당 목적을 위해 전체 버퍼를 사용할 수 있습니다.

이 필드는 메시지 이용자 함수의 입력 필드입니다. 예외 핸들러 함수와는 관련이 없습니다.

### **CBCCALLBA(10자리의 부호 있는 정수)**

콜백 컨텍스트 구조 - CallbackArea 필드.

이 필드는 콜백 함수에서 사용할 수 있는 필드입니다.

큐 관리자는 이 필드의 콘텐츠를 기반으로 결정하지 않으며 MQCBD 구조의 CBDCALLBA 필드에서 변경하지 않은 채 전달되며 이 필드는 콜백 함수를 정의하는 데 사용되는 MQCB 호출의 매개변수입니다.

CBCCALLBA의 변경사항은 CBCHOBJ에 대한 콜백 함수의 호출 전체에 유지됩니다. 이 필드는 다른 핸들의 콜백 함수와 공유되지 않습니다.

이는 콜백 함수의 입출력(I/O) 필드입니다. 이 필드의 초기값은 널 포인터이거나 널 바이트입니다.

### **CBCCALLT(10자리의 부호 있는 정수)**

콜백 컨텍스트 구조 - CallType 필드.

이 함수가 호출된 이유에 대한 정보가 포함된 필드입니다. 다음 호출 유형이 정의됩니다.

메시지 전달 호출 유형: 이러한 호출 유형은 메시지에 대한 정보를 포함합니다. **CBCLLEN** 및 **CBCBUFFLEN** 매개변수는 이러한 호출 유형에 유효합니다.

### **CBCTMR**

메시지 이용자 함수가 오브젝트 핸들에서 파괴적으로 제거된 메시지로 호출되었습니다.

CBCCC의 값이 CCWARN인 경우 Reason 필드의 값은 RC2079 또는 데이터 변환 문제를 표시하는 코드 중 하나입니다.

### **CBCTMN**

메시지 이용자 함수가 오브젝트 핸들에서 아직 파괴적으로 제거되지 않은 메시지로 호출되었습니다. 메시지는 MsgToken을 사용하여 오브젝트 핸들에서 파괴적으로 제거될 수 있습니다.

다음 이유 때문에 메시지가 제거되지 않았을 수 있습니다.

- MQGMO 옵션이 찾아보기 조작인 GMBR을 요청했습니다.
- 이 메시지는 사용 가능한 버퍼보다 크고 MQGMO 옵션은 gmatm을 지정하지 않습니다.

CBCCC의 값이 CCWARN인 경우 Reason 필드의 값은 RC2080 또는 데이터 변환 문제를 표시하는 코드 중 하나입니다.

콜백 제어 호출 유형: 이러한 호출 유형은 콜백의 제어에 대한 정보를 포함하고 메시지에 대한 세부사항은 포함하지 않습니다. 이러한 호출 유형은 MQCBD 구조에서 CBDOPT를 사용하여 요청됩니다.

**CBCLLEN** 및 **CBCBUFFLEN** 매개변수는 이러한 호출 유형에 유효하지 않습니다.

### **CBCTRC**

이 호출 유형의 목적은 콜백 함수가 일부 초기 설정을 수행할 수 있도록 하는 것입니다.

콜백이 등록된 후 콜백 함수가 바로 호출됩니다. 즉, CBREG의 *Operation* 필드 값을 사용하여 MQCB에서 리턴합니다.

이 호출 유형은 메시지 이용자와 이벤트 핸들러 모두에 사용됩니다.

요청받은 경우 이는 콜백 함수의 첫 번째 호출입니다.

CBCREA 필드의 값은 RCNONE입니다.

#### **CBCTSC**

이 호출 유형의 목적은 시작할 때 콜백 함수가 일부 설정을 수행할 수도록 허용하는 것입니다(예를 들어, 이전에 중지되었을 때 정리한 자원 복원).

연결이 각각 CTLSR 또는 CTLSW를 사용하여 시작될 때 콜백 함수가 호출됩니다.

콜백 함수가 다른 콜백 함수 내에 등록되는 경우 콜백이 리턴할 때 이 호출 유형이 호출됩니다.

이 호출 유형은 메시지 이용자에만 사용됩니다.

CBCREA 필드의 값은 RCNONE입니다.

#### **CBCTTC**

이 호출 유형의 목적은 잠시 중지되었을 때 콜백 함수가 일부 설정을 수행하도록 허용하는 것입니다(예를 들어, 메시지 이용 중 가져온 추가 자원 정리).

MQCTL 호출이 CTLSP의 *Operation* 필드 값을 사용하여 발행될 때 콜백 함수가 호출됩니다.

이 호출 유형은 메시지 이용자에만 사용됩니다.

CBCREA 필드 값은 중지 이유를 표시하도록 설정됩니다.

#### **CBCTDC**

이 호출 유형의 목적은 이용 프로세스의 마지막 부분에서 콜백 함수가 마지막 정리를 수행할 수 있도록 하는 것입니다. 콜백 함수는 다음일 때 호출됩니다.

- 콜백 함수는 BCUNR로 MQCB 호출을 사용하여 등록 취소됩니다.
- 큐가 닫혀서 암시적 등록 취소가 발생합니다. 이 인스턴스에서 콜백 함수는 오브젝트 핸들로 HOUNUH를 전달합니다.
- MQDISC 호출 완료 - 암시적 닫기를 유발하므로 등록 취소가 발생합니다. 이 경우에는 연결이 즉시 끊기지 않으며, 진행 중인 트랜잭션은 아직 커밋되지 않습니다.

이러한 조치를 콜백 함수 자체 내에서 가져온 경우 콜백이 리턴하면 조치가 호출됩니다.

이 호출 유형은 메시지 이용자와 이벤트 핸들러 모두에 사용됩니다.

요청받은 경우 이는 콜백 함수의 마지막 호출입니다.

CBCREA 필드 값은 중지 이유를 표시하도록 설정됩니다.

#### **CBCTEC**

##### **이벤트 핸들러 함수**

이벤트 핸들러 함수는 다음의 경우에 메시지 없이 호출됩니다.

- MQCTL 호출은 CTLSP의 *Operation* 필드 값으로 발행됩니다. 또는
- 큐 관리자 또는 연결이 중지되거나 일시정지됩니다.

이 호출은 모든 콜백 함수에 대한 적합한 조치를 취하는 데 사용될 수 있습니다.

##### **메시지 이용자 함수**

오브젝트 핸들에 특정한 오류(CBCCC = CCFAIL)가 감지된 경우 메시지 이용자 함수가 메시지 없이 호출되었습니다(예: CBCREA 코드 = RC2016).

CBCREA 필드 값은 콜백 이유를 표시하도록 설정됩니다.

입력 필드입니다. CBCTMR 및 CMCTMN은 메시지 이용자 함수에만 적용 가능합니다.



### **CBCCC(10자리의 부호 있는 정수)**

콜백 컨텍스트 구조 - CompCode 필드.

이는 완료 코드입니다. 메시지 이용에 대한 문제점이 있는지 여부를 표시합니다. 다음 중 하나입니다.

#### **CCOK**

성공적인 완료

#### **CCWARN**

경고(일부 완료)

#### **CCFAIL**

호출에 실패했습니다.

입력 필드입니다. 이 필드의 초기값은 CCOK입니다.

### **CBCCONNAREA(10자리의 부호 있는 정수)**

콜백 컨텍스트 구조 - ConnectionArea 필드.

이 필드는 콜백 함수에서 사용할 수 있는 필드입니다.

큐 관리자는 이 필드의 콘텐츠를 기반으로 결정하지 않으며 MQCTLO 구조의 ConnectionArea 필드에서 변경하지 않은 채 전달되며 이 필드는 콜백 함수를 제어하는 데 사용되는 MQCTL 호출의 매개변수입니다.

콜백 함수에 의해 이 필드로 작성된 변경은 콜백 함수의 호출 전체에 유지됩니다. 이 영역은 모든 콜백 함수에 의해 공유되는 정보를 전달하는 데 사용될 수 있습니다. *CallbackArea*와 달리 이 영역은 연결 핸들의 모든 콜백에서 공통입니다.

이는 입력 및 출력 필드입니다. 이 필드의 초기값은 널 포인터이거나 널 바이트입니다.

### **CBCLLEN(10자리의 부호 있는 정수)**

이는 메시지에 있는 애플리케이션 데이터의 길이(바이트)입니다. 값이 0이면 메시지에 애플리케이션 데이터가 없다는 의미입니다.

CBCLLEN 필드는 메시지의 길이를 포함하지만 이용자에게 전달된 메시지 데이터의 길이를 반드시 포함하지는 않습니다. 메시지가 잘렸을 수 있습니다. 이용자에게 전달된 데이터 양을 판별하려면 MQGMO에서 [GMRL](#) 필드를 사용하십시오.

이유 코드가 메시지가 잘렸음을 표시하는 경우 실제 메시지 크기를 판별하기 위해 CBCLLEN 필드를 사용할 수 있습니다. 이로 인해 메시지 데이터를 수용하는 데 필요한 버퍼의 크기를 판별한 후 MQCB 호출을 발행하여 적절한 값으로 MQCBD의 [CBDMML](#)을 업데이트할 수 있습니다.

GMCONV 옵션이 지정되는 경우 변환된 메시지는 *DataLength*에 대해 리턴된 값보다 클 수 있습니다. 이 경우 애플리케이션은 *DataLength*에 대한 큐 관리자로 리턴된 값보다 더 크도록 MQCBD의 [CBDMML](#)을 업데이트하기 위해 MQCB 호출을 발행해야 합니다.

메시지 잘림 문제를 방지하기 위해 *MaxMsgLength*를 *CBDFM*으로 지정하십시오. 이로 인해 큐 관리자가 데이터 변환 후 전체 메시지 길이에 버퍼를 할당하게 됩니다. 그러나 이 옵션이 지정된 경우라도 요청을 올바르게 처리하기 위해 충분한 스토리지를 사용할 수 없는 경우도 있습니다. 애플리케이션은 항상 리턴된 이유 코드를 확인해야 합니다. 예를 들어, 메시지를 변환하기 위해 충분한 스토리지를 할당하는 것이 불가능한 경우 메시지는 변환되지 않은 애플리케이션으로 리턴됩니다.

이 필드는 메시지 사용자 함수의 입력 필드입니다. 이벤트 핸들러 함수와는 관련이 없습니다.

### **CBCLFLG(10자리의 부호 있는 정수)**

이 이용자에 대한 정보를 포함하는 플래그.

다음 옵션이 정의됩니다.

#### **CBCFBE**

COQSC 옵션을 사용하는 이전 MQCLOSE 호출이 이유 코드 RC2458로 실패한 경우 이 플래그가 리턴될 수 있습니다.

이 코드는 마지막 미리 읽기 메시지가 리턴되고 버퍼가 현재 비어 있음을 나타냅니다. 애플리케이션이 COQSC 옵션을 사용하여 다른 MQCLOSE 호출을 발행하면 성공합니다.

현재 선택 기준과 일치하지 않는 미리 읽기 버퍼에 여전히 메시지가 있을 수 있으므로 이 플래그가 설정된 메시지가 애플리케이션에 제공되지 않을 수 있습니다. 이 인스턴스에서 사용자 함수는 이유 코드 RC2019로 호출됩니다.

미리 읽기 버퍼가 비어 있는 경우 사용자는 CBCFBE 플래그 및 이유 코드 RC2518로 호출됩니다.

이 필드는 메시지 사용자 함수의 입력 필드입니다. 이벤트 핸들러 함수와는 관련이 없습니다.

### CBCHOBJ(10자리의 부호 있는 정수)

콜백 컨텍스트 구조 - CBCHOBJ 필드.

메시지 사용자에 대한 호출의 경우 이는 메시지 사용자와 관련된 오브젝트의 핸들입니다.

이벤트 핸들러의 경우 이 값은 HONONE입니다

애플리케이션은 이 핸들 및 가져오기 메시지 옵션 블록의 토큰을 사용하여 메시지가 큐에서 제거되면 메시지를 가져올 수 있습니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 HOUNUH입니다.

### CBCRCD(10자리의 부호 있는 정수)

**CBCRCD**는 다시 연결 시도 전 큐 관리자가 대기하는 시간을 표시합니다. 이 필드는 지연을 변경하거나 다시 연결을 중지하기 위해 이벤트 핸들러로 수정할 수 있습니다.

Callback Context에서 **Reason** 필드의 값이 RC2545인 경우에만 **CBCRCD** 필드를 사용하십시오.

이벤트 핸들러의 항목에서 **CBCRCD**의 값은 큐 관리자가 다시 연결 시도를 작성하기 전에 대기할 시간(밀리초)입니다. [982 페이지의 표 685](#)에서는 이벤트 핸들러에서 리턴할 때 큐 관리자의 작동을 수정하도록 설정할 수 있는 값을 나열합니다.

표 685. <b>CBCRCD</b> 값	
값	설명
-1	더 이상 다시 연결을 시도하지 마십시오. 오류가 애플리케이션에 리턴됩니다.
0	즉시 다시 연결을 시도합니다.
>0	다시 연결을 시도하기 전에 대기하십시오.

### CBCREA(10자리의 부호 있는 정수)

콜백 컨텍스트 구조 - Reason 필드.

이는 CBCCC를 규정하는 이유 코드입니다.

입력 필드입니다. 이 필드의 초기값은 RCNONE입니다.

### CBCSTATE(10자리의 부호 있는 정수)

현재 사용자의 상태 표시. 이 필드는 0이 아닌 이유 코드가 사용자 함수에 전달될 때 애플리케이션에 가장 중요한 값입니다.

각 이유 코드의 작동을 코드화할 필요가 없기 때문에 애플리케이션 프로그래밍을 단순화하기 위해 이 필드를 사용할 수 있습니다.

입력 필드입니다. 이 필드의 초기값은 CSNONE입니다.

표 686. <b>CBCSTATE</b> 값 및 결과 조치		
상태	큐 관리자 조치	상수의 값
<b>CSNONE</b> 이 이유 코드는 추가 이유 정보 없이 정상 호출을 나타냅니다	이는 정상 조작입니다.	0

표 686. CBCSTATE 값 및 결과 조치 (계속)		
상태	큐 관리자 조치	상수의 값
CSSUST 이 이유 코드는 임시 조건을 나타냅니다.	콜백 루틴은 조건을 보고하도록 호출된 후 일시 중단됩니다. 시간이 지난 후 시스템은 다시 조작을 시도할 수도 있으며 다시 동일한 조건이 발생할 수 있습니다.	1
CSSUSU 이 이유 코드는 콜백이 조건을 해결하기 위해 수행할 필요가 있는 조건을 나타냅니다.	이용자는 일시중단되며 콜백 루틴은 조건을 보고하기 위해 호출됩니다. 가능한 경우 콜백 루틴은 조건을 해결하고 연결을 계속하거나 닫아야 합니다.	2
CSSUS 이 이유 코드는 추가 메시지 콜백을 방지하는 실패를 표시합니다.	큐 관리자는 자동으로 콜백 함수를 일시중단합니다. 콜백 함수가 재개되는 경우 동일한 이유 코드를 다시 수신하게 됩니다.	3
CSSTOP 이 이유 코드는 메시지 이용의 끝을 표시합니다.	예외 핸들러 및 CBDTC를 지정한 콜백으로 전달됩니다. 추가적인 메시지가 이용될 수 없습니다.	4

#### CBCSID(10자리의 부호 있는 정수)

콜백 컨텍스트 구조 - StrucId 필드.  
구조 ID이며 값은 다음과 같아야 합니다.

#### CBCSI

콜백 컨텍스트 구조의 ID.  
이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 CBCSI입니다.

#### CBCVER(10자리의 부호 있는 정수)

콜백 컨텍스트 구조 - Version 필드.  
구조 버전 번호이며 값은 다음과 같아야 합니다.

#### CBCV1

버전-1 콜백 컨텍스트 구조.  
다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### CBCCV

콜백 컨텍스트 구조의 현재 버전.  
이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 CBCV1입니다.

### 초기값

표 687. MQCBC의 필드		
필드 이름	상수의 이름	상수의 값
CBCSID	CBCSI	'CBC-'
CBCVER	CBCV1	1
CBCCALLT	없음	0
CBCHOBJ	HOUNUH	-1
CBCCALLBA	없음	널 포인터 또는 널 바이트
CBCCONNAREA	없음	널 포인터 또는 널 바이트

표 687. MQCBC의 필드 (계속)		
필드 이름	상수의 이름	상수의 값
CBCCC	CCOK	0
CBCREA	RCNONE	0
CBCSTATE	CSNONE	0
CBCLEN	없음	0
CBCBUFFLEN	없음	0
CBCFLG	없음	0
CBCRCD	없음	0

**참고:**

1. ~ 기호는 단일 공백 문자를 나타냅니다.

**RPG 선언**

```

D* MQCBC Structure
D*
D*
D* Structure identifier
D CBCSID          1      4      INZ('CBC ')
D*
D* Structure version number
D CBCVER          5      8I 0 INZ(1)
D*
D* Why Function was called
D CBCCALLT       9      12I 0 INZ(0)
D*
D* Object Handle
D CBCHOBJ       13      16I 0 INZ(-1)
D*
D* Callback data passed to the function
D CBCCALLBA     17      32*   INZ(*NULL)
D*
D* MQCTL Data area passed to the function
D CBCCONNAREA   33      48*   INZ(*NULL)
D*
D* Completion Code
D CBCCC         49      52I 0 INZ(0)
D*
D* Reason Code
D CBCREA        53      56I 0 INZ(0)
D*
D* Consumer State
D CBCSTATE      57      60I 0 INZ(0)
D*
D* Message Data Length
D CBCLEN        61      64I 0 INZ(0)
D*
D* Buffer Length
D CBCBUFFLEN    65      68I 0 INZ(0)
D*
** Flags containing information about
D* this consumer
D CBCFLG        69      72I 0 INZ(0)
D* Ver:1 **
D* Number of milliseconds before reconnect attempt
D CBCRCD        73      76I 0 INZ(0)
D* Ver:2 **
D*

```

**IBM i IBM i의 MQCBD(콜백 디스크립터)**

콜백 함수를 지정하는 구조.

## 개요

**목적:** MQCBD 구조는 큐 관리자가 사용하는 콜백 함수 및 옵션 제어를 지정하는 데 사용됩니다.

구조는 MQCB 호출의 입력 매개변수입니다.

**버전:** MQCBD의 현재 버전은 CBDV1입니다.

**문자 세트 및 인코딩:** MQCBD의 데이터는 로컬 큐 관리자의 문자 세트 및 인코딩에 있어야 합니다. 이는 **CodedCharSetId** 큐 관리자 속성 및 ENNAT로 지정됩니다. 그러나 애플리케이션이 IBM MQ MQI client로 실행 중인 경우 구조는 클라이언트의 문자 세트 및 인코딩에 있어야 합니다.

- [985 페이지의 『필드』](#)
- [988 페이지의 『초기값』](#)
- [989 페이지의 『RPG 선언』](#)

## 필드

MQCBD 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

### **CBCALLBA(10자리의 부호 있는 정수)**

이 필드는 콜백 함수에서 사용할 수 있는 필드입니다.

큐 관리자는 이 필드의 콘텐츠를 기반으로 결정하지 않고 MQCBD 구조의 **CBCCALLBA** 필드에서 변경하지 않은 채 전달되며 이 필드는 콜백 함수를 선언의 매개변수입니다.

이 값은 현재 정의된 콜백 없이 값 CBREG가 있는 *Operation*에만 사용되며 이전 정의를 대체하지 않습니다.

이는 콜백 함수의 입력 및 출력 필드입니다. 이 필드의 초기값은 널 포인터이거나 널 바이트입니다.

### **CBCALLBF(10자리의 부호 있는 정수)**

콜백 함수는 함수 호출로서 호출됩니다.

콜백 함수에 포인터를 지정하려면 이 필드를 사용하십시오.

*CallbackFunction* 또는 *CallbackName*을 지정해야 합니다. 모두 지정하는 경우 이유 코드 RC2486으로 리턴됩니다.

*CallbackName* 또는 *CallbackFunction* 모두 설정되지 않은 경우 이유 코드 RC2486으로 호출에 실패합니다.

이 옵션은 다음 환경에서 지원되지 않습니다.

- z/OS에서 CICS
- 함수 포인터 참조를 지원하지 않는 프로그래밍 언어 및 컴파일러

이 경우 이유 코드 RC2486으로 호출에 실패합니다.

입력 필드입니다. 이 필드의 초기값은 널 포인터이거나 널 바이트입니다.

### **CBCALLBN(10자리의 부호 있는 정수)**

콜백 함수는 동적으로 링크된 프로그램으로서 호출됩니다.

*CallbackFunction* 또는 *CallbackName*을 지정해야 합니다. 모두 지정하는 경우 이유 코드 RC2486으로 리턴됩니다.

*CallbackName* 또는 *CallbackFunction*이 true가 아닌 경우 이유 코드 RC2486으로 호출에 실패합니다.

이 모듈은 사용할 첫 번째 콜백 루틴이 등록될 때 로드되고 사용할 마지막 콜백 루틴이 등록 취소될 때 로드 해제됩니다.

다음 텍스트를 언급하지 않은 경우 필드 내의 이름은 임베드된 공백 없이 왼쪽으로 정렬됩니다. 필드 길이에 맞게 이름 자체에 공백을 채웁니다. 다음 설명에서 대괄호([ ])는 옵션 정보를 나타냅니다.

## IBMi

콜백 이름은 다음 형식 중 하나일 수 있습니다.

- Library "/" Program
- Library "/" ServiceProgram "("FunctionName")"

예를 들어, MyLibrary/MyProgram(MyFunction)입니다.

라이브러리 이름은 \*LIBL일 수 있습니다. 라이브러리 및 프로그램 이름은 모두 최대 10자로 제한됩니다.

## AIX and Linux

콜백 이름은 동적으로 로드할 수 있는 모듈 또는 라이브러리의 이름으로 뒤에 해당 라이브러리에 상주하는 함수의 이름이 첨부됩니다. 함수 이름은 괄호로 묶어야 합니다. 라이브러리 이름 앞에 선택적으로 디렉토리 경로를 지정할 수 있습니다.

```
[path]library(function)
```

경로가 지정되지 않으면 시스템 검색 경로가 사용됩니다.

이름은 최대 128자로 제한됩니다.

## Windows

콜백 이름은 동적-링크 라이브러리의 이름으로 해당 라이브러리에 상주하는 함수의 이름이 후미에 첨부됩니다. 함수 이름은 괄호로 묶어야 합니다. 라이브러리 이름 앞에 다음과 같이 선택적으로 디렉토리 경로 및 드라이브를 지정할 수 있습니다.

```
[d:][path]library(function)
```

드라이브 및 경로가 지정되지 않은 경우 시스템 검색 경로가 사용됩니다.

이름은 최대 128자로 제한됩니다.

## z/OS

LINK 또는 LOAD 매크로의 EP 매개변수에서 스펙에 유효한 로드 모듈의 콜백 이름입니다.

이름은 최대 8자로 제한됩니다.

## z/OS CICS

콜백 이름은 EXEC CICS LINK 명령 매크로의 PROGRAM 매개변수에서 스펙에 유효한 로드 모듈의 이름입니다.

이름은 최대 8자로 제한됩니다.

프로그램은 설치된 PROGRAM 정의의 REMOTESYTEM 옵션 또는 동적 라우팅 프로그램을 사용하여 원격으로 정의할 수 있습니다.

프로그램이 IBM MQ API 호출을 사용하는 경우 원격 CICS 리전은 IBM MQ에 연결되어야 합니다. 그러나 MQCBC 구조에서 **CBCHOBJ** 필드는 원격 시스템에 유효하지 않습니다.

*CallbackName*을 로드하는 데 실패하는 경우 다음 오류 코드 중 하나가 애플리케이션으로 리턴됩니다.

- RC2495
- RC2496
- RC2497

또한 로드가 시도된 모듈의 이름 및 운영 체제의 실패 이유 코드를 포함하는 오류 로그에도 메시지가 기록됩니다.

입력 필드입니다. 이 필드의 초기값은 널 문자열 또는 공백입니다.

## CBDCALLBT(10자리의 부호 있는 정수)

이는 콜백 함수의 유형입니다. 값은 다음 중 하나여야 합니다.

### CBTMC

메시지 이용자 함수로 이 콜백을 정의합니다.

지정된 선택 기준을 충족하는 메시지를 오브젝트 핸들에서 사용할 수 있고 연결이 시작되면 메시지 이용자 콜백 함수가 호출됩니다.

#### **CBTEH**

비동기 이벤트 루틴으로 이 콜백을 정의합니다. 핸들에 대한 메시지를 이용하지 않습니다.

*Hobj*는 이벤트 핸들러를 정의하는 MQCB 호출에 필요하지 않으며 지정된 경우 무시됩니다.

이벤트 핸들러는 전체 메시지 이용자 환경에 영향을 주는 조건에 호출됩니다. 이벤트 시 이용자 함수는 메시지 없이 호출됩니다(예를 들어, 큐 관리자나 연결 중지 또는 일시 정지 발생). 단일 메시지 이용자에 특정한 조건에 호출되지 않습니다(예: RC2016).

이벤트는 연결이 다음 환경을 제외하고 시작 또는 중지되었는지 여부에 관계없이 애플리케이션에 전달됩니다.

- CICS on z/OS 환경
- 스레드되지 않은 애플리케이션

호출자가 해당 값 중 하나를 전달하지 않는 경우 이유 코드 RC2483으로 호출에 실패합니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 CBTMC입니다.

#### **CBDMML(10자리의 부호 있는 정수)**

이는 핸들에서 읽을 수 있고 콜백 루틴에 지정할 수 있는 가장 긴 메시지의 길이(바이트)입니다. 메시지에 더 긴 길이가 있는 경우 콜백 루틴은 메시지의 *MaxMsgLength* 바이트 및 이유 코드를 수신합니다.

- RC2080 또는
- RC2079 GMATM을 지정한 경우.

실제 메시지 길이는 MQCBC 구조의 981 페이지의 『CBCLEN(10자리의 부호 있는 정수)』 필드에 제공됩니다.

다음 특수 값이 정의됩니다.

#### **CBDFM**

버퍼의 길이는 잘림 없이 메시지를 리턴하기 위해 시스템에서 조정됩니다.

버퍼를 할당하여 메시지를 수신할 메모리가 충분하지 않은 경우 시스템은 RC2071 이유 코드로 콜백 함수를 호출합니다.

예를 들어, 데이터 변환을 요청하고 메시지 데이터를 변환할 수 있는 메모리 부족이 있는 경우 변환되지 않는 메시지는 콜백 함수에 전달됩니다.

입력 필드입니다. *MaxMsgLength* 필드의 초기값은 CBDFM입니다.

#### **CBDOPT(10자리의 부호 있는 정수).**

콜백 디스크립터 구조 - Options 필드

다음 중 하나 또는 모두를 지정할 수 있습니다. 올바르지 않은 둘 이상의 옵션을 지정하려면 값을 함께 추가하거나(동일한 상수를 두 번 이상 추가하지 않음), 비트 단위 OR 연산을 사용하여 값을 결합하십시오(프로그래밍 언어가 비트 연산을 지원하는 경우). 결합이 설명되어 있습니다. 다른 모든 결합은 유효합니다.

#### **CBDFQ**

큐 관리자가 정지 중인 경우 MQCB 호출에 실패합니다.

z/OS에서 이 옵션은 또한 연결(CICS 또는 IMS 애플리케이션의 경우)이 일시정지 상태에 있는 경우 MQCB 호출 강제 실행에 실패합니다.

일시정지 상태인 경우에 메시지 이용자에게 알림이 발생하도록 하려면 MQCB 호출에 전달된 MQGMO 옵션에 GMFIQ를 지정하십시오.

**제어 옵션:** 다음 옵션은 이용자의 상태가 변경될 때 메시지 없이 콜백 함수가 호출되는지 여부를 제어합니다.

#### **CBDRC**

콜백 함수가 호출 유형 CBCTRC를 사용하여 호출됩니다.

**CBDSC**

콜백 함수가 호출 유형 CBCTSC를 사용하여 호출됩니다.

**CBDTC**

콜백 함수가 호출 유형 CBCTTC를 사용하여 호출됩니다.

**CBDDC**

콜백 함수가 호출 유형 CBCTDC를 사용하여 호출됩니다.

이러한 호출 유형에 대한 추가 세부사항은 [CallType](#)의 내용을 참조하십시오.

**기본 옵션:** 설명한 옵션이 필요하지 않은 경우에는 다음 옵션을 사용하십시오.

**CBDNO**

기타 옵션이 지정되지 않았음을 표시하려면 이 값을 사용하십시오. 모든 옵션은 기본 값을 가정합니다.

CBDNO는 프로그램 문서화를 보조하기 위해 정의됩니다. 이 옵션은 다른 옵션과 함께 사용하기 위한 용도가 아니며 값이 0이므로 해당 사용을 감지할 수 없습니다.

입력 필드입니다. *Options* 필드의 초기값은 CBDNO입니다.

**CBDSID(10자리의 부호 있는 정수)**

콜백 디스크립터 구조 - *StrucId* 필드.

구조 ID이며 값은 다음과 같아야 합니다.

**CBDSI**

콜백 디스크립터 구조의 ID.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 CBDSI입니다.

**CBDVER(10자리의 부호 있는 정수)**

콜백 디스크립터 구조 - *Version* 필드.

구조 버전 번호이며 값은 다음과 같아야 합니다.

**CBDV1**

버전 1 콜백 디스크립터 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

**CBDCV**

콜백 디스크립터 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 CBDV1입니다.

**초기값**

표 688. MQCBD의 필드		
필드 이름	상수의 이름	상수의 값
<i>StrucId</i>	CBDSI	'CBD-'
<i>Version</i>	CBDV1	1
<i>CallBackType</i>	CBTMC	1
<i>Options</i>	CBDNO	0
<i>CallbackArea</i>	없음	널 바이트
<i>CallbackFunction</i>	없음	널 바이트
<i>CallbackName</i>	없음	공백
<i>MaxMsgLength</i>	CBDFM	-1

참고:



1. ~ 기호는 단일 공백 문자를 나타냅니다.

## RPG 선언

```
D* MQCBD Structure
D*
D*
D* Structure identifier
D  CBDSID          1      4    INZ('CBD ')
D*
D* Structure version number
D  CBDVER          5      8I 0 INZ(1)
D*
D* Callback function type
D  CBDCALLBT       9      12I 0 INZ(1)
D*
** Options controlling message
D* consumption
D  CBDOPT          13     16I 0 INZ(0)
D*
D* User data passed to the function
D  CBDCALLBA       17     32*
D*
D* FP: Callback function pointer
D  CBDCALLBF       33     48*
D*
D* Callback name
D  CBDCALLBN       49     176   INZ('\0')
D*
D* Maximum message length
D  CBDMML          177    180I 0 INZ(-1)
```

## IBM i IBM i의 MQCHARV(가변 길이 문자열)

MQCHARV 구조를 사용하여 가변 길이 문자열을 설명하십시오.

### 개요

**문자 세트 및 인코딩:** MQCHARV의 데이터가 구조 내에서 VCHRC의 문자 세트 및 ENNAT에 의해 제공되는 로컬 큐 관리자의 인코딩에 있어야 합니다. 애플리케이션이 IBM MQ MQI client로 실행 중인 경우 구조는 클라이언트의 인코딩으로 작성되어야 합니다. 일부 문자 세트에는 인코딩에 의존하는 표현이 있습니다. VCHRC가 이러한 문자 세트 중 하나인 경우 사용된 인코딩은 MQCHARV에서 다른 필드의 인코딩과 같은 인코딩입니다. VSCCSID에 의해 식별된 문자 세트는 2바이트 문자 세트(DBCS)일 수 있습니다.

**사용법:** MQCHARV 구조는 이를 포함하는 구조로 불연속일 수 있는 데이터를 처리합니다. 이 데이터를 처리하려면 포인터 데이터 유형으로 선언된 필드가 사용될 수 있습니다.

- [989 페이지의 『필드』](#)
- [990 페이지의 『초기값』](#)
- [991 페이지의 『RPG 선언』](#)
- [991 페이지의 『CSAPL의 재정의』](#)

### 필드

MQCHARV 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

#### VCHRC(10자리의 부호 있는 정수)

이는 VCHRP 또는 VCHRO 필드로 처리한 가변 길이 문자열의 문자 세트 ID입니다.

이 필드의 초기값은 CSAPL입니다. 이는 큐 관리자의 true 문자 세트 ID에 큐 관리자가 변경할 수 있음을 표시하기 위해 IBM MQ에 정의됩니다. 이는 CSQM과 동일한 방법으로 작동합니다. 따라서 값 CSAPL은 가변 길이 문자열과 연관되지 않습니다. 이 필드의 초기값은 애플리케이션의 프로그래밍 언어를 위한 적절한 수단으로 컴파일 단위의 일정한 CSAPL에 대해 다양한 값을 정의하여 변경될 수 있습니다.

### VCHRL(10자리의 부호 있는 정수)

VCHRP 또는 VCHRO 필드로 처리한 가변 길이 문자열의 길이(바이트).

이 필드의 초기값은 0입니다. 값은 0보다 크거나 같아야 하거나 인식되는 다음 특수 값이어야 합니다.

#### VSNTL

VSNTL가 지정되지 않는 경우 VCHRL 바이트는 문자열의 일부로 포함됩니다. 널 문자가 존재하는 경우 문자열을 구분하지 않습니다.

VSNTL를 지정하는 경우 문자열은 문자열에 나타난 첫 번째 널로 구분됩니다. 널 자체는 그 문자열의 일부로 포함되지 않습니다.

**참고:** VSNTL가 지정되는 경우 문자열을 종료시키는 데 사용되는 널 문자는 VCHRC에 의해 지정된 코드 세트의 널입니다.

예를 들어 UTF-16(CCSIDs 1200, 13488 및 17584)에서 이는 2바이트 유니코드 인코딩입니다. 여기서 널은 모두 0인 16비트 숫자로 표시됩니다. UTF-16에서는 문자의 일부인 0으로 설정된 단일 바이트(예: 7비트 ASCII 문자)를 찾는 것이 일반적이지만 문자열은 두 개의 '0'바이트가 짝수 바이트에서 경계에서 발견될 경우에만 널로 끝납니다. 유효한 문자의 각 부분일 때 홀수 경계에 두 개의 '0'바이트를 가져올 수 있습니다. 예를 들어, x'01' x'00' x'00' x'30'은 두 개의 유효한 유니코드 문자를 나타내고 널은 문자열을 종료하지 않습니다.

### VCHRO(10자리의 부호 있는 정수)

MQCHARV의 시작 또는 이를 포함하는 구조로의 가변 길이 문자열의 오프셋(바이트).

MQCHARV 구조가 다른 구조 내에 임베드될 때 이 값은 이 MQCHARV 구조를 포함하는 구조의 시작으로부터의 가변 길이 문자열의 오프셋(바이트)입니다. MQCHARV 구조가 다른 구조 내에 임베드되지 않은 경우(예: 함수 호출의 매개변수로서 지정된 경우), 오프셋은 MQCHARV 구조의 시작에 대해 상대적입니다.

오프셋은 양수 또는 음수일 수 있습니다. VCHRP 또는 VCHRO 필드를 사용하여 가변 길이 문자열을 지정할 수 있지만 둘 다 지정할 수는 없습니다.

이 필드의 초기값은 0입니다.

### VCHRP(pointer)

이는 가변 길이 문자열에 대한 포인터입니다.

VCHRP 또는 VCHRO 필드를 사용하여 가변 길이 문자열을 지정할 수 있지만 둘 다 지정할 수는 없습니다.

이 필드의 초기값은 널 포인터이거나 널 바이트입니다.

### VCHRS(10자리의 부호 있는 정수)

VCHRP 또는 VCHRO 필드로 처리한 버퍼의 크기(바이트).

MQCHARV 구조가 함수 호출에 출력 필드로 사용될 때 이 필드는 제공된 버퍼의 길이로 초기화되어야 합니다. VCHRL의 값이 VCHRS보다 더 크면 데이터의 VCHRS 바이트만 버퍼의 호출자로 리턴됩니다.

값은 0이상이거나 인식되는 다음 특수 값이어야 합니다.

#### VSUSL

VSUSL이 지정되는 경우 버퍼의 길이는 MQCHARV 구조의 VCHRL 필드에서 가져옵니다. 구조가 출력 필드로 사용되고 버퍼가 제공될 때 이 특수 값은 적절하지 않습니다. 이는 이 필드의 초기값입니다.

## 초기값

표 689. 상수의 MQCHARV 초기값		
필드 이름	상수의 이름	상수의 값
VCHRP	없음	널 포인터 또는 널 바이트.
VCHRO	없음	0
VCHRS	VSUSL	-1

표 689. 상수의 MQCHARV 초기값 (계속)		
필드 이름	상수의 이름	상수의 값
VCHRL	없음	0
VCHRC	CSAPL	-3

## RPG 선언

```

D*..1.....2.....3.....4.....5.....6.....7..
D* MQCHARV Structure
D*
D* Address of variable length string
D VCHRP          1      16*
D* Offset of variable length string
D VCHRO          17      20I 0
D* Size of buffer
D VCHRS          21      24I 0
D* Length of variable length string
D VCHRL          25      28I 0
D* CCSID of variable length string
D VCHRC          29      32I 0

```

## CSAPL의 재정의

다른 플랫폼에 지원된 프로그래밍 언어와 달리 RPG는 정의된 상수를 재정의할 방법이 없어 CSAPL 이외의 값을 사용하는 경우 특별히 각 VCHRC를 설정해야 합니다.

## IBM i IBM i의 MQCIH (CICS bridge 헤더)

MQCIH 구조는 IBM MQ for z/OS를 통해 CICS bridge에 송신된 메시지의 시작 부분에 있을 수 있는 정보를 설명합니다.

## 개요

**형식 이름:** FMCICS입니다.

**버전:** MQCIH의 현재 버전은 CIVER2입니다. 최신 버전의 구조에만 있는 필드는 뒤에 오는 해당 설명에서와 같이 식별됩니다.

제공된 COPY 파일에는 CIVER 필드의 초기값이 CIVER2로 설정된 MQCIH에 대한 최신 버전이 포함됩니다.

**문자 세트 및 인코딩:** 특별 조건은 MQCIH 구조 및 애플리케이션 메시지 데이터에 사용된 문자 세트 및 인코딩에 적용됩니다.

- CICS bridge 큐를 소유하는 큐 관리자에 연결되는 애플리케이션은 큐 관리자의 문자 세트 및 인코딩에 있는 MQCIH 구조를 제공해야 합니다. 이는 MQCIH 구조의 데이터 변환이 이 경우에 수행되지 않기 때문입니다.
- 다른 큐 관리자에 연결되는 애플리케이션은 지원되는 문자 세트 및 인코딩에 있는 MQCIH 구조를 제공할 수 있습니다. MQCIH의 변환은 CICS bridge 큐를 소유하는 큐 관리자에 연결된 수신되는 메시지 채널 에이전트에 의해 수행됩니다.

**참고:** 이에 대해 한 가지 예외가 있습니다. CICS bridge 큐를 소유하는 큐 관리자가 분산 큐잉에 대해 CICS를 사용 중인 경우 MQCIH는 CICS bridge 큐를 소유하는 큐 관리자의 문자 세트 및 인코딩에 있어야 합니다.

- MQCIH 구조 뒤에 오는 애플리케이션 메시지 데이터는 MQCIH 구조와 동일한 문자 세트 및 인코딩에 있어야 합니다. MQCIH 구조의 CICS 및 CIENC 필드는 애플리케이션 메시지 데이터의 문자 세트 및 인코딩을 지정하는 데 사용될 수 없습니다.

데이터 변환 엑시트는 데이터가 큐 관리자에서 지원되는 내장 형식 중 하나가 아닌 경우 애플리케이션 메시지 데이터를 변환하기 위해 사용자에게 의해 제공되어야 합니다.

**사용법:** 애플리케이션에 필요한 값이 1000 페이지의 표 691에 표시된 초기값과 같고 브릿지가 AUTH=LOCAL 또는 AUTH=IDENTIFY를 사용하여 실행 중인 경우 MQCIH 구조는 메시지에서 생략될 수 있습니다. 다른 모든 경우에 이 구조는 존재해야 합니다.

브릿지는 버전-1 또는 버전-2 MQCIH 구조를 허용하지만 3270번 트랜잭션의 경우 버전-2 구조가 사용되어야 합니다.

애플리케이션은 "요청" 필드로 문서화된 필드가 브릿지에 전송된 메시지에 적합한 값을 가지도록 해야 합니다. 이러한 필드는 브릿지에 입력됩니다.

"응답" 필드로 문서화된 필드는 브릿지가 애플리케이션에 보내는 응답 메시지에서 CICS bridge에 의해 설정됩니다. 오류 정보는 CIRET, CIFNC, CICC, CIREA 및 CIAC 필드에서 리턴되지만 모든 정보가 모든 경우에 설정되는 않습니다. 992 페이지의 표 690에서는 필드가 CIRET의 다양한 값에 대해서 설정됨을 보여줍니다.

표 690. MQCIH 구조의 오류 정보 필드 콘텐츠				
CIRET	CIFNC	CICC	CIREA	CIAC
CRC000	-	-	-	-
CRC003	-	-	FBC*	-
CRC002 CRC008	IBM MQ 호출 이름	IBM MQ CMPCOD	IBM MQ REASON	-
CRC001 CRC006 CRC007 CRC009	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	-
CRC004 CRC005	-	-	-	CICS ABCODE

- 992 페이지의 『필드』
- 1000 페이지의 『초기값』
- 1001 페이지의 『RPG 선언』

## 필드

MQCIH 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

### CIAC(4바이트 문자열)

이상종료 코드.

CIRET 필드에 값 CRC005 또는 CRC004가 있는 경우에만 이 필드에서 리턴된 값이 중요합니다. 포함하는 경우 CIAC 에는 CICS ABCODE값이 포함됩니다.

이는 응답 필드입니다. 이 필드의 길이는 LNABNC로 제공됩니다. 이 필드의 초기값은 4개의 공백 문자입니다.

이는 ADS 디스크립터가 SEND 및 RECEIVE BMS 요청에 전송할지 여부를 지정하는 지표입니다. 다음 값이 정의됩니다.

#### ADNONE

ADS 디스크립터를 송신하거나 수신하지 마십시오.

#### ADSEND

ADS 디스크립터를 송신하십시오.

#### ADRECV

ADS 디스크립터를 수신하십시오.

#### ADMSGF

ADS 디스크립터에 대한 메시지 형식을 사용하십시오.

이로 인해 ADS 디스크립터가 ADS 디스크립터의 긴 양식을 사용하여 송신하거나 수신되게 합니다. 긴 양식에는 4바이트 경계에서 맞추어진 필드가 있습니다.

CIADS 필드는 다음과 같이 설정되어야 합니다.

- ADS 디스크립터가 사용되고 있지 않은 경우 필드를 ADNONE으로 설정하십시오.
- ADS 디스크립터가 사용되고 있고 각 환경에서 동일한 CCSID를 사용하는 경우 ADSEND 및 ADRECV의 합계에 필드를 설정하십시오.
- ADS 디스크립터가 사용되고 있고 각 환경에서 다른 CCSID를 사용하지 않는 경우 ADSEND, ADRECV 및 ADMSGF의 합계에 필드를 설정하십시오.

이 필드는 3270 트랜잭션에만 사용된 요청 필드입니다. 이 필드의 초기값은 ADNONE입니다.

### CIADS(10자리의 부호 있는 정수)

ADS 디스크립터를 송신/수신합니다.

이는 ADS 디스크립터가 SEND 및 RECEIVE BMS 요청에 전송할지 여부를 지정하는 지표입니다. 다음 값이 정의됩니다.

#### ADNONE

ADS 디스크립터를 송신하거나 수신하지 마십시오.

#### ADSEND

ADS 디스크립터를 송신하십시오.

#### ADRECV

ADS 디스크립터를 수신하십시오.

#### ADMSGF

ADS 디스크립터에 대한 메시지 형식을 사용하십시오.

이로 인해 ADS 디스크립터가 ADS 디스크립터의 긴 양식을 사용하여 송신하거나 수신되게 합니다. 긴 양식에는 4바이트 경계에서 맞추어진 필드가 있습니다.

CIADS 필드는 다음과 같이 설정되어야 합니다.

- ADS 디스크립터가 사용되고 있지 않은 경우 필드를 ADNONE으로 설정하십시오.
- ADS 디스크립터가 사용되고 있고 각 환경에서 동일한 CCSID를 사용하는 경우 ADSEND 및 ADRECV의 합계에 필드를 설정하십시오.
- ADS 디스크립터가 사용되고 있고 각 환경에서 다른 CCSID를 사용하지 않는 경우 ADSEND, ADRECV 및 ADMSGF의 합계에 필드를 설정하십시오.

이 필드는 3270 트랜잭션에만 사용된 요청 필드입니다. 이 필드의 초기값은 ADNONE입니다.

### CIAI(4바이트 문자열)

AID 키.

이는 트랜잭션이 시작되는 AID 키의 초기값입니다. 왼쪽으로 정렬되는 1바이트 값입니다.

이 필드는 3270 트랜잭션에만 사용된 요청 필드입니다. 이 필드의 길이는 LNATID로 제공됩니다. 이 필드의 초기값은 4개의 공백입니다.

### CIAUT(8바이트 문자열)

비밀번호 또는 패스티켓.

이는 비밀번호 또는 패스티켓입니다. 사용자ID 인증이 CICS bridge에 대해 활성화된 경우 CIAUT는 메시지의 송신자를 인증하기 위해 MQMD ID 컨텍스트에서 사용자 ID와 함께 사용됩니다.

이 필드는 요청 필드입니다. 이 필드의 길이는 LNAUTH로 제공됩니다. 이 필드의 초기값은 8개의 공백입니다.

### CICC(10자리의 부호 있는 정수)

IBM MQ 완료 코드 또는 CICS EIBRESP.

이 필드에서 리턴된 값은 CIRET에 종속적입니다. [992 페이지의 표 690](#)의 내용을 참조하십시오.

이는 응답 필드입니다. 이 필드의 초기값은 CCOK입니다.

### CICNC(4바이트 문자열)

이상종료 트랜잭션 코드.

이는 트랜잭션(일반적으로 추가 데이터를 요청하고 있는 대화 트랜잭션)을 종료하는 데 사용되는 이상 종료 코드입니다. 그렇지 않으면, 이 필드는 공백으로 설정됩니다.

이 필드는 3270 트랜잭션에만 사용된 요청 필드입니다. 이 필드의 길이는 LNCNCL로 제공됩니다. 이 필드의 초기값은 4개의 공백입니다.

#### **CICP(10자리의 부호 있는 정수)**

커서 위치.

이는 트랜잭션이 시작되는 초기 커서 위치입니다. 추후에 대화 트랜잭션의 경우 커서 위치는 RECEIVE 벡터에 있습니다.

이 필드는 3270 트랜잭션에만 사용된 요청 필드입니다. 이 필드의 초기값은 0입니다. 이 필드는 CIVER이 CIVER2보다 작은 경우에는 표시되지 않습니다.

#### **CICSI(10자리의 부호 있는 정수)**

예약됨

이 필드는 예약 필드이며 값은 중요하지 않습니다. 이 필드의 초기값은 0입니다.

#### **CICT(10자리의 부호 있는 정수)**

태스크가 대화 가능한지 여부.

이는 태스크가 자세한 정보 요청을 발행하도록 허용되어야 하는지 또는 이상종료되어야 하는지 여부를 지정하는 지표입니다. 값은 다음 중 하나여야 합니다.

##### **CTYES**

태스크는 대화 가능합니다.

##### **CTNO**

태스크는 대화가 불가능합니다.

이 필드는 3270 트랜잭션에만 사용된 요청 필드입니다. 이 필드의 초기값은 CTNO입니다.

#### **CIENC(10자리의 부호 있는 정수)**

예약됨

이 필드는 예약 필드이며 값은 중요하지 않습니다. 이 필드의 초기값은 0입니다.

#### **CIEO(10자리의 부호 있는 정수)**

메시지의 오류 오프셋.

이는 브릿지 엑시트에 의해 감지된 올바르지 않은 데이터의 위치입니다. 이 필드는 메시지의 시작에서 올바르지 않은 데이터의 위치까지 오프셋을 제공합니다.

이 필드는 3270 트랜잭션에만 사용된 응답 필드입니다. 이 필드의 초기값은 0입니다. 이 필드는 CIVER이 CIVER2보다 작은 경우에는 표시되지 않습니다.

#### **CIFAC(8바이트 비트 문자열)**

브릿지 기능 토큰.

이는 8바이트 브릿지 기능 토큰입니다. 브릿지 기능 토큰의 목적은 의사 대화식의 다수 트랜잭션이 동일한 브릿지 설비(가상 3270 터미널)를 사용할 수 있도록 허용하는 것입니다. 의사 대화식의 첫 번째 또는 유일한 메시지에 FCNONE 값을 설정해야 합니다. 이는 CICS에 이 메시지에 대한 새 브릿지 기능을 할당하도록 지시합니다. 입력 메시지에 0이 아닌 CIFKT가 지정되면 브릿지 기능 토큰이 응답 메시지에 리턴됩니다. 후속 입력 메시지는 동일한 브릿지 기능 토큰을 사용할 수 있습니다.

다음 특수 값이 정의됩니다.

##### **FCNONE**

지정된 BVT 토큰이 없음.

이 필드는 3270 트랜잭션에만 사용된 요청 및 응답 두 필드입니다. 이 필드의 길이는 LNFAC로 제공됩니다. 이 필드의 초기값은 FCNONE입니다.

### **CIFKT(10자리의 부호 있는 정수)**

브릿지 기능 릴리스 시간.

사용자 트랜잭션이 종료된 후 브릿지 기능이 유지되는 시간(초)입니다. 비대화식 트랜잭션의 경우 값은 0이어야 합니다.

이 필드는 3270 트랜잭션에만 사용된 요청 필드입니다. 이 필드의 초기값은 0입니다.

### **CIFL(4바이트 문자열)**

터미널 에뮬레이트 속성.

이는 브릿지 기능에 대한 모델로 사용되는 설치된 터미널의 이름입니다. 공백 값은 *CIFL*이 브릿지 트랜잭션 프로파일 정의에서 가져오거나 기본값이 사용됨을 의미합니다.

이 필드는 3270 트랜잭션에만 사용된 요청 필드입니다. 이 필드의 길이는 *LNFACL*로 제공됩니다. 이 필드의 초기값은 4개의 공백입니다.

### **CIFLG(10자리의 부호 있는 정수)**

플래그입니다.

값은 다음과 같아야 합니다.

#### **CIFNON**

플래그가 없습니다.

이 필드는 요청 필드입니다. 이 필드의 초기값은 *CIFNON*입니다.

### **CIFMT(8바이트 문자열)**

MQCIH 뒤에 오는 데이터의 IBM MQ 형식 이름.

MQCIH 구조 뒤에 오는 데이터의 IBM MQ 형식 이름을 지정합니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 이 필드의 코딩 규칙은 MQMD의 *MDFMT* 필드에 대한 규칙과 동일합니다.

*CIRFM* 필드에 값 *FMNONE*가 있는 경우 이 형식 이름 또한 응답 메시지에 사용됩니다.

- DPL 요청의 경우 *CIFMT*는 *COMMAREA*의 형식 이름이어야 합니다.
- 3270 요청의 경우 *CIFMT*는 *CSQCBDCI*여야하고 *CIRFM*은 *CSQCBDCO*여야 합니다.

해당 형식에 대한 데이터 변환 엑시트는 실행할 큐 관리자에 설치되어야 합니다.

요청 메시지의 결과로 오류 응답 메시지가 생성되면 오류 응답 메시지의 형식 이름은 *FMSTR*입니다.

이 필드는 요청 필드입니다. 이 필드의 길이는 *LNFM*로 제공됩니다. 이 필드의 초기값은 *FMNONE*입니다.

### **CIFNC(4바이트 문자열)**

IBM MQ 호출 이름 또는 CICS EIBFN 함수.

이 필드에서 리턴된 값은 *CIRET*에 종속적입니다. 992 페이지의 표 690의 내용을 참조하십시오. *CIFNC*에 IBM MQ 호출 이름이 포함된 경우 다음 값이 가능합니다.

#### **CFCONN**

MQCONN 호출입니다.

#### **CFGET**

MQGET 호출.

#### **CFINQ**

MQINQ 호출입니다.

#### **CFOPEN**

MQOPEN 호출.

#### **CFPUT**

MQPUT 호출.

**CFPUT1**

MQPUT1 호출.

**CFNONE**

호출 없음.

이는 응답 필드입니다. 이 필드의 길이는 LNFUNC로 제공됩니다. 이 필드의 초기값은 CFNONE입니다.

**CIGWI(10자리의 부호 있는 정수)**

브릿지 태스크가 발행한 MQGET 호출의 대기 간격.

CIUOW에 값 CUFIRST가 있을 때만 이 필드는 적용 가능합니다. 송신되는 애플리케이션이 브릿지에 의해 발행된 MQGET 호출이 이 메시지에 의해 시작된 작업 단위에 대한 두 번째 및 후속 요청 메시지를 대기해야 하는 예상 시간(밀리초)을 지정할 수 있게 합니다. 이는 브릿지에서 사용된 기본 대기 간격을 대체합니다. 다음 특수 값이 사용될 수 있습니다.

**WIDFLT**

기본 대기 간격.

이로 인해 CICS bridge 브릿지는 브릿지가 시작될 때 지정된 기간 동안 대기하게 됩니다.

**WIULIM**

무제한 대기 간격.

이 필드는 요청 필드입니다. 이 필드의 초기값은 WIDFLT입니다.

**CIII(10자리의 부호 있는 정수)**

예약됨

이 필드는 예약된 필드입니다. 값은 0이어야 합니다. 이 필드는 CIVER이 CIVER2보다 작은 경우에는 표시되지 않습니다.

**CILEN(10자리의 부호 있는 정수)**

MQCIH 정수의 길이.

값은 다음 중 하나여야 합니다.

**CILEN1**

버전-1 CICS 정보 헤더 구조의 길이.

**CILEN2**

버전-2 CICS 정보 헤더 구조의 길이.

다음 상수는 현재 버전의 길이를 지정합니다.

**CILENC**

CICS 정보 헤더 구조의 현재 버전 길이.

이 필드는 요청 필드입니다. 이 필드의 초기값은 CILEN2입니다.

**CILT(10자리의 부호 있는 정수)**

링크 유형.

이는 브릿지가 링크에 시도해야 하는 오브젝트의 유형을 표시합니다. 값은 다음 중 하나여야 합니다.

**LTPROG**

DPL 프로그램.

**LTTRAN**

3270 트랜잭션.

이 필드는 요청 필드입니다. 이 필드의 초기값은 LTPROG입니다.

**CINTI(4바이트 문자열)**

첨부할 다음 트랜잭션.

이는 사용자 트랜잭션으로 리턴된 다음 트랜잭션의 이름입니다(일반적으로 EXEC CICS RETURN TRANSID에 의함). 다음 트랜잭션이 없는 경우, 이 필드는 공백으로 설정됩니다.



이 필드는 3270 트랜잭션에만 사용된 응답 필드입니다. 이 필드의 길이는 LNTRID에서 제공됩니다. 이 필드의 초기값은 4개의 공백입니다.

#### **CIODL(10자리의 부호 있는 정수)**

출력 COMMAREA 데이터 길이.

이는 응답 메시지의 클라이언트에 리턴되는 사용자 데이터의 길이입니다. 이 길이에는 8바이트 프로그램 이름이 포함됩니다. 링크된 프로그램으로 전달된 COMMAREA의 길이는 이 필드의 최대 값 및 요청 메시지의 사용자 데이터 길이에서 8을 뺀 수입니다.

**참고:** 메시지에서 사용자 데이터의 길이는 MQCIH 구조를 제외한 메시지의 길이입니다.

요청 메시지의 사용자 데이터 길이가 *CIODL*보다 작으면 LINK 명령의 DATALENGTH 옵션이 사용됩니다. 이로 인해 LINK를 다른 CICS 영역에 효율적으로 기능 제공할 수 있습니다.

다음 특수 값을 사용할 수 있습니다.

#### **OLINPT**

출력 길이가 입력 길이와 동일합니다.

이 값은 링크된 프로그램에 전달된 COMMAREA가 충분한 크기인지 확인하기 위해 응답이 요청되지 않아도 필요할 수 있습니다.

이 필드는 DPL 트랜잭션에만 사용된 요청 필드입니다. 이 필드의 초기값은 OLINPT입니다.

#### **CIREA(10자리의 부호 있는 정수)**

IBM MQ 응답이나 피드백 코드 또는 CICS EIBRESP2.

이 필드에서 리턴된 값은 *CIRET*에 종속적입니다. [992 페이지의 표 690](#)의 내용을 참조하십시오.

이는 응답 필드입니다. 이 필드의 초기값은 RCNONE입니다.

#### **CIRET(10자리의 부호 있는 정수)**

브릿지의 리턴 코드.

이는 브릿지에서 수행된 처리의 결과를 설명하는 CICS bridge의 리턴 코드입니다. *CIFNC*, *CICC*, *CIREA*와 *CIAC* 필드는 추가 정보를 포함할 수 있습니다([992 페이지의 표 690](#) 참조). 값은 다음 중 하나입니다.

#### **CRC000**

(0, X'000') 오류가 없습니다.

#### **CRC001**

(1, X'001') EXEC CICS 명령문이 오류를 감지했습니다.

#### **CRC002**

(2, X'002') IBM MQ 호출이 오류를 감지했습니다.

#### **CRC003**

(3, X'003') CICS bridge가 오류를 감지했습니다.

#### **CRC004**

(4, X'004') CICS bridge가 비정상적으로 종료되었습니다.

#### **CRC005**

(5, X'005') 애플리케이션이 비정상적으로 종료되었습니다.

#### **CRC006**

(6, X'006') 보안 오류가 발생했습니다.

#### **CRC007**

(7, X'007') 프로그램을 사용할 수 없습니다.

#### **CRC008**

(8, X'008') 지정된 시간 내에 수신된 현재 작업 단위 내에 있는 두 번째 또는 이후 버전의 메시지입니다.

#### **CRC009**

(9, X'009') 트랜잭션을 사용할 수 없습니다.

이는 응답 필드입니다. 이 필드의 초기값은 CRC000입니다.

**CIRFM(8바이트 문자열)**

응답 메시지의 IBM MQ 형식 이름.

현재 메시지에 응답하여 전송되는 응답 메시지의 IBM MQ 형식 이름입니다. 이 필드를 코딩하는 규칙은 MQMD의 *MDFMT* 필드를 코딩하는 규칙과 같습니다.

이 필드는 DPL 트랜잭션에만 사용된 요청 필드입니다. 이 필드의 길이는 LNFMT로 제공됩니다. 이 필드의 초기값은 FMNONE입니다.

**CIRSI(4바이트 문자열)**

예약됨

이 필드는 예약된 필드입니다. 값은 4개의 공백이어야 합니다. 이 필드의 길이는 LNRSID로 제공됩니다.

**CIRS1(8바이트 문자열)**

예약됨

이 필드는 예약된 필드입니다. 값은 8개의 공백이어야 합니다.

**CIRS2(8바이트 문자열)**

예약됨

이 필드는 예약된 필드입니다. 값은 8개의 공백이어야 합니다.

**CIRS3(8바이트 문자열)**

예약됨

이 필드는 예약된 필드입니다. 값은 8개의 공백이어야 합니다.

**CIRS4(10자리의 부호 있는 정수)**

예약됨

이 필드는 예약된 필드입니다. 값은 0이어야 합니다. 이 필드는 *CIVER*이 *CIVER2*보다 작은 경우에는 표시되지 않습니다.

**CIRTI(4바이트 문자열)**

예약됨

이 필드는 예약된 필드입니다. 값은 4개의 공백이어야 합니다. 이 필드의 길이는 LNTRID에서 제공됩니다.

**CISC(4바이트 문자열)**

트랜잭션 시작 코드.

브릿지가 터미널 트랜잭션 또는 START 트랜잭션을 에뮬레이트하는지 여부를 지정하는 지표입니다. 값은 다음 중 하나여야 합니다.

**SCSTRT**

시작.

**SCDATA**

데이터 시작.

**SCTERM**

입력 종료.

**SCNONE**

없음

브릿지의 응답에서 이 필드는 *CINTI* 필드에 포함된 다음 트랜잭션 ID에 적절한 시작 코드로 설정됩니다. 다음 시작 코드는 다음 응답에 가능합니다.

- SCSTRT
- SCDATA
- SCTERM

CICS Transaction Server 1.2의 경우 이 필드는 요청 필드 전용입니다. 응답에서 해당 값은 정의되어 있지 않습니다.

CICS Transaction Server 1.3 및 후속 릴리스의 경우 이는 요청 필드인 동시에 응답 필드입니다.

이 필드는 3270 트랜잭션에만 사용됩니다. 이 필드의 길이는 LNSTCO로 제공됩니다. 이 필드의 초기값은 SCNONE입니다.

#### **CISID(4바이트 문자열)**

구조 ID입니다.

값은 다음과 같아야 합니다.

##### **CISIDV**

CICS 정보 헤더 구조의 ID.

이 필드는 요청 필드입니다. 이 필드의 초기값은 CISIDV입니다.

#### **CITES(10자리의 부호 있는 정수)**

태스크 종료 시 상태.

이 필드는 태스크 종료 시 사용자 트랜잭션의 상태를 표시합니다. 다음 값 중 하나가 리턴됩니다.

##### **TENOSY**

동기화되지 않습니다.

사용자 트랜잭션은 아직 완료되지 않았으며 동기점이 없습니다. 이 경우 MQMD의 *MDMT* 필드는 *MTRQST*입니다.

##### **TECMIT**

작업 단위를 커밋합니다.

사용자 트랜잭션은 아직 완료되지 않았지만 첫 번째 작업 단위의 동기점이 조정됩니다. 이 경우 MQMD의 *MDMT* 필드는 *MTDGRM*입니다.

##### **TEBACK**

작업 단위를 백아웃합니다.

사용자 트랜잭션이 아직 완료되지 않았습니다. 현재 작업 단위가 백아웃됩니다. 이 경우 MQMD의 *MDMT* 필드는 *MTDGRM*입니다.

##### **TEENDT**

태스크를 종료합니다.

사용자 트랜잭션이 종료되었습니다(또는 이상종료됨). 이 경우 MQMD의 *MDMT* 필드는 *MTRPLY*입니다.

이 필드는 3270 트랜잭션에만 사용된 응답 필드입니다. 이 필드의 초기값은 TENOSY입니다.

#### **CITI(4바이트 문자열)**

첨부할 트랜잭션.

*CILT*에 값 *LTTRAN*이 있는 경우 *CITI*는 실행될 사용자 트랜잭션의 트랜잭션 ID입니다. 이 경우 공백이 없는 값이 지정되어야 합니다.

*CILT*에 값 *LTPROG*가 있는 경우 *CITI*는 실행될 작업 단위 내의 모든 프로그램 아래의 트랜잭션 코드입니다. 지정된 값이 비어 있는 경우 CICS DPL 브릿지 기본 트랜잭션 코드(CKBP)가 사용됩니다. 값에 공백이 없는 경우 CSQCBP00의 초기 프로그램을 사용한 로컬 TRANSACTION으로 CICS에 정의됩니다. *CIUOW*에 값 *CUFRST* 또는 *CUONLY*가 있을 때만 이 필드는 적용 가능합니다.

이 필드는 요청 필드입니다. 이 필드의 길이는 LNTRID에서 제공됩니다. 이 필드의 초기값은 4개의 공백입니다.

#### **CIUOW(10자리의 부호 있는 정수)**

작업 단위 제어.

이는 CICS bridge에서 수행된 작업 단위 처리를 제어합니다. 단일 트랜잭션 또는 작업 단위 내에 있는 하나 이상의 프로그램을 실행하기 위해 브릿지를 요청할 수 있습니다. 이 필드는 CICS bridge가 작업 단위를 시작

해야 하는지, 현재 작업 단위 내에서 요청된 기능을 수행하는지 또는 커미트하거나 백업하여 작업 단위를 종료해야 하는지 여부를 표시합니다. 다양한 결합은 데이터 전송 플로우를 최적화하기 위해 지원됩니다.

값은 다음 중 하나여야 합니다.

**CUONLY**

작업 단위를 시작하고 기능을 수행한 후 작업 단위를 커미트합니다(DPL 및 3270).

**CUCONT**

현재 작업 단위에 대한 추가 데이터입니다(3270 전용).

**CUFRST**

작업 단위를 시작하고 기능을 수행합니다(DPL 전용).

**CUMIDL**

현재 작업 단위 내의 기능을 수행합니다(DPL 전용).

**CULAST**

기능을 수행한 후 작업 단위를 커미트합니다(DPL 전용).

**CUCMIT**

작업 단위를 커미트합니다(DPL 전용).

**CUBACK**

작업 단위를 백아웃합니다(DPL 전용).

이 필드는 요청 필드입니다. 이 필드의 초기값은 CUONLY입니다.

**CIVER(10자리의 부호 있는 정수)**

구조 버전 번호입니다.

값은 다음 중 하나여야 합니다.

**CIVER1**

버전-1 CICS 정보 헤더 구조.

**CIVER2**

버전-2 CICS 정보 헤더 구조.

최신 버전의 구조에만 있는 필드는 필드의 설명에서 최신 필드로 식별됩니다. 다음 상수는 현재 버전의 버전 번호를 지정합니다.

**CIVERC**

CICS 정보 헤더 구조의 현재 버전.

이 필드는 요청 필드입니다. 이 필드의 초기값은 CIVER2입니다.

**초기값**

표 691. MQCIH의 필드		
필드 이름	상수의 이름	상수의 값
CISID	CISIDV	'CIH-'
CIVER	CIVER2	2
CILEN	CILEN2	180
CIENC	없음	0
CICSI	없음	0
CIFMT	FMNONE	공백
CIFLG	CIFNON	0
CIRET	CRC000	0
CICC	CCOK	0

표 691. MQCIH의 필드 (계속)		
필드 이름	상수의 이름	상수의 값
CIREA	RCNONE	0
CIUOW	CUONLY	273
CIGWI	WIDFLT	-2
CILT	LTPROG	1
CIODL	OLINPT	-1
CIFKT	없음	0
CIADS	ADNONE	0
CICT	CTNO	0
CITES	TENOSY	0
CIFAC	FCNONE	널
CIFNC	CFNONE	공백
CIAC	없음	공백
CIAUT	없음	공백
CIRS1	없음	공백
CIRFM	FMNONE	공백
CIRSI	없음	공백
CIRTI	없음	공백
CITI	없음	공백
CIFL	없음	공백
CIAI	없음	공백
CISC	SCNONE	공백
CICNC	없음	공백
CINTI	없음	공백
CIRS2	없음	공백
CIRS3	없음	공백
CICP	없음	0
CIE0	없음	0
CIII	없음	0
CIRS4	없음	0

**참고:**

1. - 기호는 단일 공백 문자를 나타냅니다.

**RPG 선언**

D\* .1.....2.....3.....4.....5.....6.....7..  
D\* MQCIH Structure

```

D*
D* Structure identifier
D  CISID          1      4      INZ('CIH ')
D* Structure version number
D  CIVER          5      8I 0 INZ(2)
D* Length of MQCIH structure
D  CILEN          9      12I 0 INZ(180)
D* Reserved
D  CIENC          13     16I 0 INZ(0)
D* Reserved
D  CICSI          17     20I 0 INZ(0)
D* MQ format name of data that followsMQCIH
D  CIFMT          21     28     INZ(' ')
D* Flags
D  CIFLG          29     32I 0 INZ(0)
D* Return code from bridge
D  CIRET          33     36I 0 INZ(0)
D* MQ completion code or CICSEIBRESP
D  CICC           37     40I 0 INZ(0)
D* MQ reason or feedback code, or CICSEIBRESP2
D  CIREA          41     44I 0 INZ(0)
D* Unit-of-work control
D  CIUOW          45     48I 0 INZ(273)
D* Wait interval for MQGET call issuedby bridge task
D  CIGWI          49     52I 0 INZ(-2)
D* Link type
D  CILT           53     56I 0 INZ(1)
D* Output COMMAREA data length
D  CIODL          57     60I 0 INZ(-1)
D* Bridge facility release time
D  CIFKT          61     64I 0 INZ(0)
D* Send/receive ADS descriptor
D  CIADS          65     68I 0 INZ(0)
D* Whether task can beconversational
D  CICT           69     72I 0 INZ(0)
D* Status at end of task
D  CITES          73     76I 0 INZ(0)
D* Bridge facility token
D  CIFAC          77     84     INZ(X'0000000000000000-
D                                     00')
D* MQ call name or CICS EIBFNfunction
D  CIFNC          85     88     INZ(' ')
D* Abend code
D  CIAC           89     92     INZ
D* Password or passticket
D  CIAUT          93     100    INZ
D* Reserved
D  CIRS1          101    108    INZ
D* MQ format name of reply message
D  CIRFM          109    116    INZ(' ')
D* Remote CICS system ID to use
D  CIRSI          117    120    INZ
D* CICS RTRANSID to use
D  CIRTI          121    124    INZ
D* Transaction to attach
D  CITI           125    128    INZ
D* Terminal emulated attributes
D  CIFL           129    132    INZ
D* AID key
D  CIAI           133    136    INZ
D* Transaction start code
D  CISC           137    140    INZ(' ')
D* Abend transaction code
D  CICNC          141    144    INZ
D* Next transaction to attach
D  CINTI          145    148    INZ
D* Reserved
D  CIRS2          149    156    INZ
D* Reserved
D  CIRS3          157    164    INZ
D* Cursor position
D  CICP           165    168I 0 INZ(0)
D* Offset of error in message
D  CIEO           169    172I 0 INZ(0)
D* Reserved
D  CIII           173    176I 0 INZ(0)
D* Reserved
D  CIRS4          177    180I 0 INZ(0)
D*

```

MQCMHO 구조를 사용하여 애플리케이션이 메시지 핸들을 작성하는 방법을 제어할 수 있습니다.

## 개요

### 목적

이 구조는 MQCRTMH 호출의 입력 매개변수입니다.

### 문자 세트 및 인코딩

MQCMHO의 데이터는 애플리케이션의 문자 세트 및 애플리케이션의 인코딩(ENNAT)에 있어야 합니다.

- [1003 페이지의 『필드』](#)
- [1004 페이지의 『초기값』](#)
- [1004 페이지의 『RPG 선언』](#)

## 필드

MQCMHO 구조에는 다음과 같은 필드가 포함됩니다. 필드에 대해서 알파벳순으로 설명합니다.

### CMOPT(10자리의 부호 있는 정수)

다음 중 하나를 지정할 수 있습니다.

#### CMVAL

MQSETMP가 이 메시지 핸들에서 특성을 설정하기 위해 호출될 때 특성 이름은 다음을 확인하기 위해 유효성 검증됩니다.

- 올바른지 않은 문자가 포함되지 않습니다.
- 다음을 제외하고 "JMS" 또는 "usr.JMS"로 시작하지 않습니다.
  - JMSCorrelationID
  - JMSReplyTo
  - JMSType
  - JMSXGroupID
  - JMSXGroupSeq
- 이러한 이름은 JMS 특성에 예약됩니다.
- 대문자 또는 소문자의 혼합에서 다음 키워드 중 하나가 아닙니다.
  - "AND"
  - "BETWEEN"
  - "ESCAPE"
  - "거짓"
  - "IN"
  - "IS"
  - "LIKE"
  - "NOT"
  - "NULL"
  - "또는"
  - "참"
- "Body" 또는 "Root"로 시작하지 않습니다 ("Root.MQMD." 제외).

특성이 MQ정의인 경우 ("mq. \*") 이름이 인식되면 특성 디스크립터 필드가 특성의 올바른 값으로 설정됩니다. 특성이 인식되지 않는 경우 특성 디스크립터의 *Support* 필드는 PDSUPO로 설정됩니다(자세한 정보는 [PDSUP](#) 참조).

## CMDEFV

이는 특성 이름의 유효성 검증 기본 레벨이 발생함을 지정합니다.

유효성 검증의 기본 레벨은 **CMVAL**에서 지정한 레벨과 동일합니다.

향후 릴리스에서 **CMDEFV**가 정의될 때 발생하는 유효성 검증의 레벨을 변경할 관리 옵션이 정의될 수 있습니다.

이는 기본값입니다.

## CMNOVA

특성 이름의 유효성 검증이 발생하지 않습니다. **CMVAL**에 대한 설명을 참조하십시오.

**기본 옵션:** 이 절에서 이전에 설명된 옵션 중 필요한 옵션이 없는 경우 다음 옵션을 사용할 수 있습니다.

## CMNONE

모든 옵션은 자체 기본값을 가정합니다. 기타 옵션을 지정하지 않도록 표시하려면 이 값을 사용하십시오. **CMNONE**은(는) 프로그램 문서화를 지원합니다. 이 옵션은 다른 옵션과 함께 사용할 수 없지만, 값이 0이므로 이러한 사용을 감지할 수 없습니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 **CMDEFV**입니다.

## CMSID(10자리의 부호 있는 정수)

구조 ID이며 값은 다음과 같아야 합니다.

### CMSIDV

메시지 핸들 작성 옵션 구조의 ID.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 **CMSIDV**입니다.

## CMVER(10자리의 부호 있는 정수)

구조 버전 번호이며 값은 다음과 같아야 합니다.

### CMVER1

버전-1 작성 메시지 핸들 옵션 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

### CMVERC

작성 메시지 핸들 옵션 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 **CMVER1**입니다.

## 초기값

표 692. MQCMHO의 필드		
필드 이름	상수의 이름	상수의 값
<i>CMSID</i>	CMSIDV	'CMHO'
<i>CMVER</i>	CMVER1	1
<i>CMOPT</i>	CMDEFV	0

## RPG 선언

D\* MQCMHO Structure



```

D*
D*
D* Structure identifier
D CMSID 1 4 INZ('CMHO')
D*
D* Structure version number
D CMVER 5 8I 0 INZ(1)
D*
D* Options that control the action of MQCRTMH
D CMOPT 9 12I 0 INZ(0)

```

## IBM i IBM i의 MQCNO(연결 옵션)

MQCNO 구조는 애플리케이션이 로컬 큐 관리자에 대한 연결에 관하여 옵션을 지정할 수 있게 허용합니다.

### 개요

**목적:** 이 구조는 MQCONNX 호출의 입출력(I/O) 매개변수입니다.

**버전:** MQCNO의 현재 버전은 CNVER6입니다. 최신 버전의 구조에만 있는 필드는 다음에 오는 설명에서 최신 필드로 식별됩니다.

제공된 COPY 파일에는 환경에서 지원하지만 CNVER 필드의 초기값이 CNVER1로 설정된 MQCNO의 최신 버전이 포함되어 있습니다. version-1 구조에 없는 필드를 사용하려면 애플리케이션이 CNVER 필드를 필요한 버전의 버전 번호로 설정해야 합니다.

**문자 세트 및 인코딩:** MQCNO의 데이터는 ENNAT가 제공하는 로컬 큐 관리자의 인코딩 및 **CodedCharSetId** 큐 관리자 속성이 제공하는 문자 세트에 있어야 합니다.

- [1005 페이지의 『필드』](#)
- [1010 페이지의 『초기값』](#)
- [1010 페이지의 『RPG 선언』](#)

### 필드

MQCNO 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

#### CCDTUL(10자리의 부호 있는 정수)

CCDTUL은 연결에 사용하기 위해 클라이언트 연결 채널 목록의 위치를 식별하는 URL을 포함하는 각 CCDTUP 또는 CCDTUO에 의해 식별된 문자열의 길이입니다.

MQCONNX 호출을 발행하는 애플리케이션이 IBM MQ MQI client로 실행 중인 경우에만 CCDTUL을 사용하십시오.

프로그래밍 방식으로 MQCHLLIB 및 MQCHLTAB 환경 변수를 설정하는 대체 방식입니다.

애플리케이션이 클라이언트로 실행되지 않는 경우 CCDTUL은 무시됩니다.

CNVER이 CNVER6 미만이면 이 필드는 무시됩니다.

#### CCDTUO(10자리의 부호 있는 정수)

CCDTUO는 MQCNO 구조의 시작에서 연결에 사용하기 위해 클라이언트 연결 채널 테이블의 위치를 식별하는 URL을 포함하는 문자열까지의 오프셋(바이트)입니다. 오프셋은 양수 또는 음수일 수 있습니다.

MQCONNX 호출을 발행하는 애플리케이션이 IBM MQ MQI client로 실행 중인 경우에만 CCDTUL을 사용하십시오.

**중요사항:** CCDTUP 및 CCDTUO 중 하나만 사용할 수 있습니다. 두 필드가 0이 아닌 경우 이유 코드 RC2600으로 호출에 실패합니다.

프로그래밍 방식으로 MQCHLLIB 및 MQCHLTAB 환경 변수를 설정하는 대체 방식입니다.

애플리케이션이 클라이언트로 실행되지 않는 경우 CCDTUO는 무시됩니다.

CNVER이 CNVER6 미만이면 이 필드는 무시됩니다.

## CCDTUP(pointer)

CCDTUP는 연결에 사용하기 위해 클라이언트 연결 채널 테이블의 위치를 식별하는 URL을 포함하는 문자열에 대한 선택적 포인터입니다.

MQCONNX 호출을 발행하는 애플리케이션이 IBM MQ MQI client로 실행 중인 경우에만 CCDTUP를 사용하십시오.

**중요사항:** CCDTUP 및 CCDTUO 중 하나만 사용할 수 있습니다. 두 필드가 0이 아닌 경우 이유 코드 RC2600으로 호출에 실패합니다.

프로그래밍 방식으로 MQCHLLIB 및 MQCHLTAB 환경 변수를 설정하는 대체 방식입니다.

애플리케이션이 클라이언트로 실행되지 않는 경우 CCDTUP는 무시됩니다.

CNVER이 CNVER6 미만이면 이 필드는 무시됩니다.

## CNAN(28바이트 문자열)

큐 관리자에 대한 연결을 식별하기 위해 애플리케이션에 의해 설정된 이름입니다. 이 필드의 초기값은 널 문자입니다.

CNVER이 CNVER7 미만이면 이 필드는 무시됩니다.

## CNCCO(10자리의 부호 있는 정수)

이는 MQCNO 구조의 시작에서 MQCD 채널 정의 구조의 오프셋(바이트)입니다.

## CNCCP(pointer)

이는 MQCD 채널 정의 구조에 대한 포인터입니다.

## CNCONID(24바이트 문자열)

고유 연결 ID. 이 필드를 사용하면 큐 관리자가 큐 관리자에 처음 연결될 때 고유한 ID를 지정하여 애플리케이션 프로세스를 신뢰성 있게 식별할 수 있습니다.

PUT 및 GET 호출을 작성할 때 애플리케이션은 상관 목적으로 연결 ID를 사용합니다. 모든 연결은 연결 설정 방법에 관계없이 큐 관리자에 의한 ID로 지정됩니다.

장기간 작업 단위의 마지막을 강제 실행하기 위해 연결 ID를 사용할 수 있습니다. 이를 수행하기 위해 PCF 명령 'Stop Connection' 또는 MQSC 명령 STOP CONN을 사용하는 연결 ID를 지정합니다. 해당 명령 사용에 대한 자세한 정보는 관련 링크를 참조하십시오.

이 필드의 초기값은 24 널 바이트입니다.

## CNCT(128바이트 비트 문자열)

이는 큐 관리자가 이 연결 동안 애플리케이션에 의해 영향을 받는 자원과 연관되는 태그입니다.

큐 관리자 연결 태그.

큐 관리자가 올바르게 영향받은 자원에 대한 액세스를 직렬화할 수 있도록 각 애플리케이션 또는 애플리케이션 인스턴스는 태그에 다른 값을 사용해야 합니다. 추가 세부사항은 CN\*CT\* 옵션에 대한 설명을 참조하십시오. 태그는 애플리케이션이 종료되거나 MQDISC 호출을 발행할 때 유효하도록 합니다.

태그가 필요하지 않은 경우 다음 특수 값을 사용하십시오.

### CTNONE

지정된 연결 태그가 없습니다.

이 값은 필드 길이만큼의 2진 0입니다.

입력 필드입니다. 이 필드의 길이는 LNCTAG로 제공됩니다. 이 필드의 초기값은 CTNONE입니다. CNVER이 (가) CNVER3보다 작으면 이 필드는 무시됩니다.

z/OS 큐 관리자에 연결될 때 필드 ConnTag를 사용하십시오.

## CNNORES2(4바이트 문자열)

구조를 64비트 경계까지 채우기 위해 예약된 필드입니다. 필드의 초기값은 필드 길이만큼의 2진수 0입니다.

CNVER이 CNVER7 미만이면 이 필드는 무시됩니다.

## CNOPT(10자리의 부호 있는 정수)

MQCONNX 조치를 제어하는 옵션.

### 바인딩 옵션

바인딩 옵션은 사용되는 IBM MQ 바인딩의 유형을 제어합니다. 이러한 옵션 중 하나만 지정하십시오.

#### CNSBND

표준 바인딩.

표준 바인딩 옵션으로 애플리케이션 및 로컬 큐 관리자 에이전트가 일반적으로 별도의 프로세스에서 실행의 개별 단위에서 실행할 수 있습니다. 배열은 큐 관리자의 무결성을 유지합니다. 즉, 이는 잘못된 프로그램으로부터 큐 관리자를 보호합니다.

애플리케이션이 완전히 테스트되지 않았거나 신뢰할 수 없는 경우 CNSBND를 사용하십시오. CNSBND는 기본값입니다.

CNSBND는 프로그램 문서화를 지원하기 위해 정의됩니다. 이 옵션을 사용되는 바인딩 유형을 제어하는 다른 옵션과 함께 사용하지 마십시오. 이 값이 제로이기 때문에 이러한 사용은 감지될 수 없습니다.

모든 환경에서 이 옵션이 지원됩니다.

#### CNFBND

빠른 경로 바인딩.

빠른 경로 바인딩 옵션으로 애플리케이션 및 로컬 큐 관리자 에이전트가 동일한 실행 단위의 일부일 수 있습니다. 빠른 경로는 애플리케이션 및 로컬 큐 관리자 에이전트가 별도의 실행 단위로 실행하는 표준 바인딩과는 대조적입니다.

CNFBND는 큐 관리자가 이러한 유형의 바인딩을 지원하지 않는 경우 무시됩니다. 옵션이 지정되지 않은 경우라도 처리는 계속됩니다.

다중 프로세스가 애플리케이션이 사용하는 전체 자원보다 더 많은 자원을 사용하는 상황에서는 CNFBND가 이점이 될 수 있습니다. 빠른 경로 바인딩을 사용하는 애플리케이션을 신뢰할 수 있는 애플리케이션이라고 합니다.

빠른 경로 바인딩을 사용할지 여부를 결정할 때 다음과 같은 중요 사항을 고려하십시오.

- **CNFBND 옵션을 사용하면 애플리케이션이 큐 관리자에 속한 기타 데이터 영역과 메시지를 대체하거나 손상시키는 것을 방지할 수 없습니다. 이 발행이 완전히 평가된 경우에만 이 옵션을 사용하십시오.**
- 애플리케이션은 CNFBND와 함께 비동기 신호 또는 타이머 인터럽트(예: sigkill)를 사용하지 않아야 합니다. 또한 공유 메모리 세그먼트 사용에 관한 제한사항도 있습니다.
- 애플리케이션은 한 번에 둘 이상의 스레드가 큐 관리자에 연결되어서는 안 됩니다.
- 애플리케이션은 큐 관리자에서 연결을 끊기 위해 MQDISC 호출을 사용해야 합니다.
- 애플리케이션은 endmqm 명령으로 큐 관리자를 종료하기 전에 완료해야 합니다.

다음 포인트는 표시된 환경에서 CNFBND 사용에 적용됩니다.

- IBM i에서 작업은 QMQMADM 그룹에 속하는 사용자 프로파일 QMQM에서 실행되어야 합니다. 또한 프로그램은 비정상적으로 종료되어서는 안 됩니다. 그렇지 않으면 예측하지 못한 결과가 발생할 수 있습니다.

신뢰할 수 있는 애플리케이션 사용의 연관성에 대한 자세한 정보는 MQCONNX 호출을 사용하여 큐 관리자에 연결 및 신뢰할 수 있는 애플리케이션의 제한을 참조하십시오.

#### CNSHBD

공유 바인딩.

공유 바인딩 옵션으로 애플리케이션 및 로컬 큐 관리자 에이전트가 일반적으로 별도의 프로세스에서 실행의 개별 단위에 실행할 수 있습니다. 배열은 큐 관리자의 무결성을 유지합니다. 즉, 이는 잘못된 프로그램으로부터 큐 관리자를 보호합니다. 그러나 일부 자원은 애플리케이션 및 로컬 큐 관리자 에

이전트 사이에 공유됩니다. 큐 관리자가 이 바인딩 유형을 지원하지 않는 경우 CNSHBD는 무시됩니다. 이 옵션이 지정되지 않은 것처럼 처리가 계속 수행됩니다.

### **CNIBND**

격리된 바인딩.

격리된 바인딩 옵션으로 애플리케이션 및 로컬 큐 관리자 에이전트가 일반적으로 별도의 프로세스에서 실행의 개별 단위에서 실행할 수 있습니다. 배열은 큐 관리자의 무결성을 유지합니다. 즉, 이는 잘못된 프로그램으로부터 큐 관리자를 보호합니다. 애플리케이션 프로세스 및 로컬 큐 관리자 에이전트가 서로 분리되어 자원을 공유하지 않습니다. 큐 관리자가 이 바인딩 유형을 지원하지 않는 경우 CNIBND는 무시됩니다. 이 옵션이 지정되지 않은 것처럼 처리가 계속 수행됩니다.

### **핸들 공유 옵션**

다음 옵션은 동일한 프로세스 내에서 서로 다른 스레드(병렬 처리 단위) 간의 핸들 공유를 제어합니다. 이러한 옵션 중 하나만 지정될 수 있습니다.

#### **CNHSN**

스레드 간 핸들 공유가 없습니다.

스레드 간 핸들 공유 없음 옵션은 연결 및 오브젝트 핸들이 핸들을 할당한 스레드에서만 사용할 수 있음을 나타냅니다. 즉, 이는 MQCONN, MQCONNX 또는 MQOPEN 호출을 발행한 스레드입니다. 핸들은 동일한 프로세스에 속한 다른 스레드에서는 사용될 수 없습니다.

#### **CNHSB**

호출 차단을 사용하는 스레드 간 직렬 핸들 공유입니다.

호출 차단과 함께 스레드 간의 직렬 핸들 공유 옵션은 프로세스 중 하나의 스레드가 할당한 연결 및 오브젝트 핸들을 동일한 프로세스에 속한 다른 스레드에서 사용할 수 있음을 나타냅니다. 그러나 한 번에 하나의 스레드만 특정 핸들을 사용할 수 있습니다. 즉, 하나의 직렬 핸들의 사용만 허용됩니다. 스레드가 다른 스레드에서 이미 사용되고 있는 핸들을 사용하는 경우 호출은 핸들이 사용 가능하게 될 때까지 차단(대기)됩니다.

#### **CNHSNB**

호출 차단을 사용하지 않는 스레드간 직렬 핸들 공유입니다.

호출 차단을 사용하지 않는 스레드 간 직렬 핸들 공유 옵션은 다른 스레드에서 핸들이 사용 중인 경우 핸들이 사용 가능하게 될 때까지 차단하는 대신 CCFAIL 및 RC2219과(와) 함께 호출이 즉시 완료된다는 점을 제외하고는 "차단 사용" 옵션과 동일합니다.

스레드에는 0개 또는 하나의 비공유 핸들과 0개 이상의 공유 핸들이 있습니다.

- CNHSN을 지정하는 각 MQCONN 또는 MQCONNX 호출은 첫 번째 호출에서 새 비공유 핸들을 리턴하고 후속 호출의 동일한 비공유 핸들을 리턴합니다(중간에 끼어드는 MQDISC 호출이 없다고 가정함). 두 번째와 그 이후의 호출에 대한 이유 코드는 RC2002입니다.
- CNHSB 또는 CNHSNB를 지정하는 각 MQCONNX 호출은 각 호출에서 새 공유 핸들을 리턴합니다.

오브젝트 핸들은 이 오브젝트 핸들을 작성한 MQOPEN 호출에 지정된 연결 핸들과 동일한 공유 특성을 상속합니다. 또한 작업 단위는 이 작업 단위를 시작하는 데 사용된 연결 핸들과 동일한 공유 특성을 상속합니다. 작업 단위가 공유 핸들을 사용하여 하나의 스레드에서 시작되는 경우 이 작업 단위는 동일한 핸들을 사용하는 다른 스레드에서 업데이트될 수 있습니다.

핸들 공유 옵션을 지정하지 않은 경우 기본값은 환경에 의해 판별됩니다.

- Microsoft Transaction Server(MTS) 환경에서 기본값은 CNHSB와 동일합니다.
- 기타 환경에서 기본값은 CNHSN과 동일합니다.

### **재연결 옵션**

재연결 옵션은 연결이 재연결 가능한지 여부를 판별합니다. 클라이언트 연결만 재연결이 가능합니다.

#### **CNRCDF**

재연결 옵션이 기본값으로 해석됩니다. 기본값이 설정되지 않은 경우 이 옵션의 값은 DISABLED로 해석됩니다. 옵션의 값은 서버로 전달되며 PCF 및 MQSC로 조회될 수 있습니다.

**CNRC**

애플리케이션은 MQCONNX QMNAME 매개변수의 값과 일치하는 큐 관리자에 다시 연결될 수 있습니다. 처음에 연결을 설정한 큐 관리자와 클라이언트 애플리케이션 사이에 연관관계가 없는 경우에만 CNRC 옵션을 사용하십시오. 옵션의 값은 서버로 전달되며 PCF 및 MQSC로 조회될 수 있습니다.

**CNRC D**

애플리케이션이 다시 연결될 수 없습니다. 옵션의 값이 서버로 전달되지 않습니다.

**CNRCQM**

애플리케이션은 원래 연결했던 큐 관리자에만 다시 연결될 수 있습니다. 클라이언트가 다시 연결될 수 있지만 원래 연결을 설정한 큐 관리자와 클라이언트 애플리케이션 사이에 연관관계가 있는 경우에 이 값을 사용하십시오. 고가용성 큐 관리자의 대기 인스턴스에 클라이언트를 자동으로 다시 연결시키려면 이 값을 선택하십시오. 옵션의 값은 서버로 전달되며 PCF 및 MQSC로 조회될 수 있습니다.

클라이언트 연결에만 옵션 CNRC, CNRC D 및 CNRCQM을 사용하십시오. 이 옵션을 바인딩 연결에 사용하면 MQCONNX는 실패하며 완료 코드는 MQCC\_FAILED, 이유 코드는 MQRC\_OPTIONS\_ERROR입니다.

**기본 옵션:** 설명된 옵션 중 필요한 옵션이 없는 경우 다음 옵션을 사용할 수 있습니다.

**CNNONE**

지정된 옵션이 없습니다.

CNNONE는 프로그램 문서화를 지원하기 위해 정의됩니다. 이 옵션은 다른 CN\* 옵션과 함께 사용되기 위한 용도는 아니지만 값이 0이므로 이러한 사용을 감지할 수 없습니다.

**CNSCO(10자리의 부호 있는 정수)**

이는 MQCNO 구조의 시작에서 MQSCO 구조의 오프셋(바이트)입니다.

CNVER이(가) CNVER4보다 작으면 이 필드는 무시됩니다.

**CNSCP(pointer)**

이는 MQSCO 구조의 주소입니다.

CNVER이(가) CNVER4보다 작으면 이 필드는 무시됩니다.

**CNSECPO(10자리의 부호 있는 정수)**

보안 매개변수 오프셋입니다. 사용자 ID 및 비밀번호 지정에 사용되는 MQCSP 구조의 오프셋입니다.

이 값은 양수 또는 음수가 될 수 있습니다. 이 필드의 초기값은 0입니다.

CNVER이(가) CNVER5보다 작으면 이 필드는 무시됩니다.

**CNSECPP(pointer)**

보안 매개변수 포인터입니다. 사용자 ID 및 비밀번호 지정에 사용되는 MQCSP 구조의 주소입니다.

이 필드의 초기값은 널 포인터이거나 널 바이트입니다.

CNVER이(가) CNVER5보다 작으면 이 필드는 무시됩니다.

**CNSID(4바이트 문자열)**

MQCNO 구조의 구조 ID입니다.

값은 다음과 같아야 합니다.

**CNSIDV**

연결 옵션 구조의 ID.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 CNSIDV입니다.

**CNVER(10자리의 부호 있는 정수)**

MQCNO 구조의 구조 버전 번호입니다.

값은 다음과 같아야 합니다.

**CNVER6**

버전-6 연결 옵션 구조입니다.

이 버전은 모든 환경에서 지원됩니다.

**CNVER7**

버전-7 연결 옵션 구조입니다.

이 버전은 모든 환경에서 지원됩니다.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

**CNVERC**

연결 옵션 구조의 현재 버전입니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 CNVER7입니다.

**초기값**

표 693. MQCNO의 필드		
필드 이름	상수의 이름	상수의 값
CNSID	CNSIDV	'CNO-
CNVER	CNVER5	1
CNOPT	CNNONE	0
CNCCO	없음	0
CNCCP	없음	널 포인터 또는 널 바이트
CNCT	CTNONE	널
CNSCP	없음	널 포인터 또는 널 바이트
CNSCO	없음	0
CNCONID	없음	널
CNSECPO	없음	0
CNSECPP	없음	널 포인터 또는 널 바이트
CCDTUL	없음	0
CCDTUO	없음	0
CCDTUP	없음	널 포인터 또는 널 바이트

**참고:**

1. - 기호는 단일 공백 문자를 나타냅니다.

**RPG 선언**

```

D*****
D**
D**          IBM MQ for IBM i          **
D**
D** FILE NAME:      CMQCNQG          **
D**
D** DESCRIPTION:    MQCNO Structure -- Connect Options **
D**
D*****
D** <N_OCO_COPYRIGHT>                **
D** Licensed Materials - Property of IBM **
D**
  
```

```

D** 5724-H72 **
D** (c) Copyright IBM Corp. 1993, 2024. All Rights Reserved. **
D** **
D** US Government Users Restricted Rights - Use, duplication or **
D** disclosure restricted by GSA ADP Schedule Contract with **
D** IBM Corp. **
D** <NOC_COPYRIGHT> **
D***** **
D** **
D** FUNCTION: This file declares the structure MQCNO, **
D** which is used by the main MQI. **
D** **
D** PROCESSOR: RPG (ILE) **
D** **
D***** **
D* **
D* **
D***** **
D** <BEGIN_BUILDINFO> **
D** Generated on: 08/02/16 13:50 **
D** Build Level: L000000 **
D** Build Type: Production **
D** Pointer Size: 128 Bit **
D** Source File: **
D** CMQCN0G **
D** <END_BUILDINFO> **
D***** **
D* **
D* ..1....:....2....:....3....:....4....:....5....:....6....:....7..
D* **
D* **
D* MQCNO Structure
D* **
D* Structure identifier
D CNSID 1 4 INZ('CNO ')
D* Structure version number
D CNVER 5 8I 0 INZ(1)
D* Options that control the action of MQCONN
D CNOPT 9 12I 0 INZ(0)
D* Ver:1 **
D* Offset of MQCD structure for client connection
D CNCCO 13 16I 0 INZ(0)
D* Address of MQCD structure for client connection
D CNCCP 17 32* INZ(*NULL)
D* Ver:2 **
D* Queue managerconnection tag
D CNCT 33 160 INZ(X'0000000000000000-
D 000000000000000000000000-
D 000000000000000000000000-
D 000000000000000000000000-
D 000000000000000000000000-
D 000000000000000000000000-
D 000000000000000000000000-
D 000000000000000000000000-
D 000000000000000000000000-
D 0000000000000000')
D* Ver:3 **
D* Address of MQSCO structure for client connection
D CNSCP 161 176* INZ(*NULL)
D* Offset of MQSCO structure for client connection
D CNSCO 177 180I 0 INZ(0)
D* Ver:4 **
D* Unique Connection Identifier
D CNCONID 181 204 INZ(X'0000000000000000-
D 000000000000000000000000-
D 000000')
D* Offset of MQCSP structure
D CNSECP0 205 208I 0 INZ(0)
D* Address of MQCSP structure
D CNSECPP 209 224* INZ(*NULL)
D* Ver:5 **
D* Address of CCDT URL string
D CNCCDTUP 225 240* INZ(*NULL)
D* Offset of CCDT URL string
D CNCCDTUO 241 244I 0 INZ(0)
D* Length of CCDT URL
D CNCCDTUL 245 248I 0 INZ(0)
D* Ver:6 **
D* **
D***** **

```

## IBM i IBM i의 MQCSP(보안 매개변수)

IBM i에 대한 MQCSP 구조의 요약입니다.

### 개요

**목적:** MQCSP 구조는 권한 서비스가 사용자 ID 및 비밀번호를 인증할 수 있게 합니다. MQCONNX 호출에서 MQCSP 연결 보안 매개변수 구조를 지정합니다.

**문자 세트 및 인코딩:** MQCSP의 데이터는 **CodedCharSetId** 큐 관리자 속성에 의해 지정된 문자 세트 및 ENNAT에 의해 지정된 로컬 큐 관리자의 인코딩에 있어야 합니다.

- [1012 페이지의 『필드』](#)
- [1014 페이지의 『초기값』](#)
- [1014 페이지의 『RPG 선언』](#)

### 필드

MQCSP 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

#### CSAUTH(10자리의 부호 있는 정수)

이는 수행할 인증의 유형입니다.

올바른 값은 다음과 같습니다.

#### CSAN

사용자 ID 및 비밀번호 필드를 사용하지 마십시오.

#### CSAUIAP

인증 사용자 ID 및 비밀번호 필드.

입력 필드입니다. 이 필드의 초기값은 CSAN입니다.

#### CSCPPL(10자리의 부호 있는 정수)

이는 인증에 사용될 비밀번호의 길이입니다.

비밀번호의 최대 길이는 플랫폼에 의존하지 않습니다. 비밀번호의 길이가 허용된 길이보다 긴 경우 인증 요청은 RC2035로 실패합니다.

입력 필드입니다. 이 필드의 초기값은 0입니다.

#### CSCPPO(10자리의 부호 있는 정수)

이는 인증에서 사용되는 비밀번호의 오프셋(바이트)입니다.

오프셋은 양수 또는 음수일 수 있습니다.

입력 필드입니다. 이 필드의 초기값은 0입니다.

#### CSCPPP(pointer)

이는 인증에 사용될 비밀번호의 주소입니다.

입력 필드입니다. 이 필드의 초기값은 널 포인터입니다.

#### CSCSPUIL(10자리의 부호 있는 정수)

이는 인증에서 사용될 사용자 ID의 길이입니다.

사용자 ID의 최대 길이는 플랫폼에 의존하지 않습니다. 사용자 ID 길이가 허용된 길이보다 긴 경우 인증 요청은 RC2035로 실패합니다.

입력 필드입니다. 이 필드의 초기값은 0입니다.



**CSCSPUIO(10자리의 부호 있는 정수)**

이는 인증에서 사용되는 사용자 ID의 오프셋(바이트)입니다.  
오프셋은 양수 또는 음수일 수 있습니다.  
입력 필드입니다. 이 필드의 초기값은 0입니다.

**CSCSPUIP(pointer)**

이는 인증에 사용될 사용자 ID의 주소입니다.  
입력 필드입니다. 이 필드의 초기값은 널 포인터입니다. CSVER이 CSVER5 미만인 경우 이 필드는 무시됩니다.

**CSINITKL (10자리의 부호 있는 정수)**

이것은 암호 보호 시스템에 대한 초기 키의 길이입니다.  
입력 필드입니다. 이 필드의 초기값은 0입니다.

**CSINITKO (10자리의 부호 있는 정수)**

암호 보호 시스템에 대한 초기 키의 오프셋 (바이트) 입니다. 오프셋은 양수 또는 음수일 수 있습니다.  
CSINITKO 또는 CSINITKP 를 사용하여 초기 키를 지정할 수 있지만 둘 다 지정할 수는 없습니다. 자세한 정보는 CSINITKP 필드의 설명을 참조하십시오.  
입력 필드입니다. 이 필드의 초기값은 0입니다.

**CSINITKP (포인터)**

암호 보호 시스템에 대한 초기 키의 주소 (바이트) 입니다.  
입력 필드입니다. 이 필드의 초기값은 널 포인터입니다.  
IBM MQ MQI clients 는 일부 필드의 값을 IBM MQ 비밀번호 보호 시스템을 사용하여 암호화된 값으로 제공할 수 있습니다. 다음 필드는 암호화된 값을 포함할 수 있습니다.

- MQSCO 구조의 키 저장소 비밀번호입니다.

초기 키는 암호화 알고리즘에서 이러한 값을 암호화하고 복호화하는 데 사용됩니다. 이러한 필드의 값이 **runmqicred** 유틸리티를 사용하여 암호화될 때 초기 키가 제공되는 경우, 큐 관리자에 연결할 때 클라이언트가 동일한 초기 키를 지정해야 합니다.

이 필드를 사용하여 지정된 초기 키는 *MQS MQI KEYFILE* 환경 변수 또는 클라이언트 구성 파일의 보안 스탠자에 있는 *MQIInitialKeyFile* 특성을 사용하여 지정된 초기 키를 대체합니다.

CSINITKO 또는 CSINITKP 를 사용하여 초기 키를 지정할 수 있지만 둘 다 지정할 수는 없습니다.

**CSRE1(4바이트 문자열)**

IBM i의 포인터 정렬에 필요한 예약 필드입니다.  
입력 필드입니다. 이 필드의 초기값은 모두 널입니다.

**CSRS2(8바이트 문자열)**

IBM i의 포인터 정렬에 필요한 예약 필드입니다.  
입력 필드입니다. 이 필드의 초기값은 모두 널입니다.

**CSSID(4바이트 문자열)**

구조 ID입니다.  
값은 다음과 같아야 합니다.

**CSSIDV**

보안 매개변수 구조 ID.

**CSVER(10자리의 부호 있는 정수)**

구조 버전 번호입니다.

값은 다음과 같아야 합니다.

**CSVER1**

버전-1 보안 매개변수 구조.

**CSVER2**

Version-2 보안 매개변수 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

**CSVERC**

보안 매개변수 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 CSVER1입니다.

**초기값**

표 694. MQCNO의 필드		
필드 이름	상수의 이름	상수의 값
CSSID	CSSIDV	'CSP-'
CSVER	CSVER1	1
CSAUTHT	없음	0
CSRE1	없음	널
CSCSPUIP	없음	Null pointer
CSCSPUIO	없음	0
CSCSPUIL	없음	0
CSRS2	없음	널
CSCPPP	없음	Null pointer
CSCPPO	없음	0
CSCPPL	없음	0
CSINITKP	없음	Null pointer
CSINITKO	없음	0
CSINITKL	없음	0

**참고:**

1. - 기호는 단일 공백 문자를 나타냅니다.

**RPG 선언**

```
D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQCSP Structure
D*
D* Structure identifier
D  CSSID           1      4      INZ('CSP ')
D* Structure version number
D  CSVER           5      8I 0  INZ(1)
D* Type of authentication
D  CSAUTHT        9      12I 0  INZ(0)
D* Reserved
D  CSRE1          13     16      INZ(X'00000000')
D* Address of user ID
D  CSCSPUIP       17     32*    INZ(*NULL)
D* Offset of user ID
```

D	CSCSPUIO	33	36I 0 INZ(0)
D*	Length of user ID		
D	CSCSPUIL	37	40I 0 INZ(0)
D*	Reserved		
D	CSRS2	41	48 INZ(X'0000000000000000')
D*	Address of password		
D	CSCPPP	49	64* INZ(*NULL)
D*	Offset of password		
D	CSCPP0	65	68I 0 INZ(0)
D*	Length of password		
D	CSCPPL	69	72I 0 INZ(0)

## IBM i IBM i의 MQCTLO(제어 콜백 옵션 구조)

제어 콜백 함수를 지정하는 구조입니다.

### 개요

#### 목적

MQCTLO 구조는 제어 콜백 함수에 관하여 옵션을 지정하는 데 사용됩니다.

구조는 MQCTL 호출의 입력 및 출력 매개변수입니다.

#### 버전

MQCTLO의 현재 버전은 CTLV1입니다.

#### 문자 세트 및 인코딩

MQCTLO의 데이터는 **CodedCharSetId** 큐 관리자 속성에 의해 지정된 문자 세트와 ENNAT에 의해 지정된 로컬 큐 관리자의 인코딩에 있어야 합니다. 그러나 애플리케이션이 IBM MQ 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트와 인코딩으로 작성되어야 합니다.

- [1015 페이지의 『필드』](#)
- [1016 페이지의 『초기값』](#)
- [1016 페이지의 『RPG 선언』](#)

### 필드

MQCTLO 구조에는 다음과 같은 필드가 포함됩니다. 필드에 대해서 알파벳순으로 설명합니다.

#### COCONNAREA(10자리의 부호 있는 정수)

제어 옵션 구조 - ConnectionArea 필드.

이 필드는 콜백 함수에서 사용할 수 있는 필드입니다.

큐 관리자는 이 필드의 콘텐츠를 기반으로 결정하지 않고 MQCBD 구조의 CBCONNAREA 필드에서 변경하지 않은 채 전달되며 이는 MQCB 콜백의 매개변수입니다.

이 필드는 CTLRSW 및 CTLSSW 이외의 모든 조작에 대해 무시됩니다.

이는 콜백 함수의 입력 및 출력 필드입니다. 이 필드의 초기값은 널 포인터이거나 널 바이트입니다.

#### COOPT(10자리의 부호 있는 정수)

MQCTLO 조치를 제어하는 옵션입니다.

#### CTLFQ

큐 관리자 또는 연결이 정지 상태에 있는 경우 MQCTLO 호출 강제 실행에 실패합니다.

일시정지 상태인 경우에 메시지 이용자에게 알림이 발생하도록 하려면 MQCB 호출에 전달된 MQGMO 옵션에 GMFIQ를 지정하십시오.

#### CTLTHR

이 옵션은 애플리케이션이 동일 연결에 대해 모든 메시지 이용자가 동일한 스레드에서 호출되도록 요구함을 시스템에 알립니다.

**기본 옵션:** 설명한 옵션이 필요하지 않은 경우에는 다음 옵션을 사용하십시오.

**CTLNO**

기타 옵션이 지정되지 않았음을 표시하려면 이 값을 사용하십시오. 모든 옵션은 기본 값을 가정합니다. CTLNO는 프로그램 문서화를 보조하기 위해 정의됩니다. 이 옵션은 다른 옵션과 함께 사용하기 위한 용도가 아니며 값이 0이므로 이러한 사용을 감지할 수 없습니다.

입력 필드입니다. COOPT 필드의 초기값은 CTLNO입니다.

**CORSV(10자리의 부호 있는 정수)**

이 필드는 예약된 필드입니다. 이 필드의 초기값은 공백 문자입니다.

**COSID(10자리의 부호 있는 정수)**

제어 옵션 구조 - StrucId 필드.

구조 ID이며 값은 다음과 같아야 합니다.

**CTLSI**

제어 옵션 구조의 ID.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 CTLSI입니다.

**COVER(10자리의 부호 있는 정수)**

제어 옵션 구조 - Version 필드.

구조 버전 번호이며 값은 다음과 같아야 합니다.

**CTLV1**

버전 1 제어 옵션 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

**CTLCV**

제어 옵션 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 CTLV1입니다.

**초기값**

표 695. MQCTLO의 필드		
필드 이름	상수의 이름	상수의 값
COSID	CTLSI	'CTLO'
COVER	CTLV1	1
COOPT	CTLNO	널
CORSV	Reserved 필드	
COCONNAREA	없음	널 포인터 또는 널 바이트

**RPG 선언**

```

D* MQCTLO Structure
D*
D*
D* Structure identifier
D COSID          1      4  INZ('CTLO')
D*
D* Structure version number
D COVER          5      8I 0 INZ(1)
D*
D* Options that control the action of MQCTL
D COOPT          9     12I 0 INZ(0)
D*
D* Reserved
D CORSV         13     16I 0 INZ(-1)
D*

```

## IBM i IBM i의 MQDH(분배 헤더)

MQDH 구조는 해당 메시지가 전송 큐에 저장된 분배 목록 메시지일 때 메시지에 있는 추가 데이터를 설명합니다.

### 개요

**목적:** 분배 목록 메시지가 다중 목적지 큐에 전송하는 메시지입니다. 추가 데이터는 MQOR 레코드의 배열 및 MQPMR 레코드의 배열 다음에 오는 MQDH 구조로 구성됩니다.

이 구조는 메시지를 전송 큐에 직접 넣거나 전송 큐(예: 메시지 채널 에이전트)에서 메시지를 제거하는 특수화된 애플리케이션에서 사용하기 위한 것입니다.

이 구조는 메시지를 간단하게 분배 목록에 넣으려는 일반 애플리케이션에서 사용하지 않아야 합니다. 해당 애플리케이션은 MQOD 구조를 사용하여 목적지를 분배 목록에서 정의하고 MQPMO 구조를 사용하여 메시지 특성을 지정하거나 개별 목적지로 전송된 메시지에 대한 정보를 수신해야 합니다.

**문자 세트 및 인코딩:** MQDH의 데이터는 **CodedCharSetId** 큐 관리자 속성에 의해 지정된 문자 세트 및 C 프로그래밍 언어에 대한 ENNAT에 의해 지정된 로컬 큐 관리자의 인코딩에 있어야 합니다.

MQDH의 문자 세트 및 인코딩은 **MDCSI** 및 **MDENC** 필드에 설정되어야 합니다.

- MQMD(MQDH 구조가 메시지 데이터의 시작에 있는 경우) 또는
- MQDH 구조에 선행하는 헤더 구조(다른 모든 경우).

**사용법:** 애플리케이션이 메시지를 분배 목록에 넣을 때 목적지의 일부 또는 모두는 원격입니다. 큐 관리자는 MQXQH 및 MQDH 구조의 애플리케이션 메시지 데이터를 앞에 붙이고 관련 전송 큐에 메시지를 배치합니다. 따라서 데이터는 메시지가 전송 큐에 있을 때 다음 순서로 발생합니다.

- MQXQH 구조
- MQDH 구조와 MQOR 및 MQPMR 레코드의 배열
- 애플리케이션 메시지 데이터

목적지에 따라 이와 같은 둘 이상의 메시지는 큐 관리자에서 생성되고 다른 전송 큐에 배치할 수 있습니다. 이 경우 해당 메시지에서 MQDH 구조는 애플리케이션에서 연 분배 목록에 의해 정의된 목적지의 다른 서브세트를 식별합니다.

배포 목록 메시지를 전송 큐에 직접 넣는 애플리케이션은 이전에 설명한 순서를 준수해야 하며 MQDH 구조가 올바른지 확인해야 합니다. MQDH 구조가 올바르지 않은 경우 큐 관리자는 이유 코드 RC2135로 MQPUT 또는 MQPUT1 호출에 실패하도록 선택할 수 있습니다.

큐가 분배 목록 메시지를 지원할 수 있는 것으로 정의되는 경우에만 메시지는 분배 목록 양식을 큐에 저장할 수 있습니다(1305 페이지의 [『큐의 속성』](#)에 설명된 **DistLists** 큐 속성 참조). 애플리케이션이 분배 목록을 지원하지 않는 큐에 직접 분배 목록 메시지를 넣는 경우 큐 관리자는 분배 목록 메시지를 개별 메시지로 분할하고 대신 큐에 배치합니다.

- [1017 페이지의 『필드』](#)
- [1020 페이지의 『초기값』](#)
- [1021 페이지의 『RPG 선언』](#)

### 필드

MQDH 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

#### DHCNT(10자리의 부호 있는 정수)

존재하는 MQOR 레코드 수.

이는 목적지 수를 정의합니다. 분배 목록은 항상 하나 이상의 목적지를 포함해야 하므로 DHCNT은(는) 항상 0보다 커야 합니다.

이 필드의 초기값은 0입니다.

### **DHCSI(10자리의 부호 있는 정수)**

MQOR 및 MQPMR 레코드 뒤에 오는 데이터의 문자 세트 ID.

이는 MQOR 및 MQPMR 배열 구조 뒤에 오는 데이터의 문자 세트 ID를 식별합니다. MQDH 구조 자체의 문자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 다음 특수 값을 사용할 수 있습니다.

#### **CSINHT**

이 구조의 문자 세트 ID를 상속합니다.

이 구조 다음에 오는 데이터의 문자 데이터는 이 구조와 동일한 문자 세트로 되어 있습니다.

큐 관리자가 구조의 실제 문자 세트 ID에 메시지로 송신한 구조에서 이 값을 변경합니다. 오류가 발생하지 않으면 MQGET 호출에서 CSINHT 값을 리턴하지 않습니다.

MQMD의 MDPAT 필드 값이 ATBRKR이면 CSINHT를 사용할 수 없습니다.

이 필드의 초기값은 CSUNDF입니다.

### **DHENC(10자리의 부호 있는 정수)**

MQOR 및 MQPMR 레코드 뒤에 오는 데이터의 숫자 인코딩.

이는 MQOR 및 MQPMR 레코드 배열 뒤에 오는 데이터의 숫자 인코딩을 지정합니다. MQDH 구조 자체의 숫자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다.

이 필드의 초기값은 0입니다.

### **DHFLG(10자리의 부호 있는 정수)**

일반 플래그.

다음 플래그가 지정될 수 있습니다.

#### **DHFNEW**

새 메시지 ID를 생성합니다.

이 플래그는 새 메시지 ID가 분배 목록에서 각 목적지에 대해 생성됨을 표시합니다. 이는 넣기 메시지 레코드가 없을 때 또는 레코드가 있지만 *PRMID* 필드를 포함하지 않을 때만 설정할 수 있습니다.

이 플래그를 사용하면 마지막 가능한 순간까지 메시지 ID의 생성을 지연시킬 수 있습니다. 즉, 분배 목록 메시지가 최종적으로 개별 메시지로 분할되는 순간입니다. 이는 분배 목록 메시지로 플로우해야 하는 제어 정보의 양을 최소화합니다.

애플리케이션이 메시지를 분배 목록에 넣으면, 큐 관리자는 다음 다음 명령문이 모두 true인 경우 생성하는 MQDH에서 DHFNEW를 설정합니다.

- 애플리케이션에서 제공하는 넣기 메시지 레코드가 없거나 제공된 레코드가 *PRMID* 필드를 포함하지 않습니다.
- MQMD의 *MDMID* 필드는 MINONE이거나 MQPMO의 *PMOPT* 필드가 PMNMID를 포함합니다

플래그가 필요하지 않은 경우 다음을 지정할 수 있습니다.

#### **DHFNON**

플래그가 없습니다.

이 상수는 플래그가 지정되지 않았음을 표시합니다. DHFNON은 프로그램 문서화를 지원하기 위해 정의됩니다. 이 상수는 다른 상수와 함께 사용하기 위한 용도는 아니지만 값이 0이므로 이러한 사용을 감지할 수 없습니다.

이 필드의 초기값은 DHFNON입니다.

### **DHFMT(8바이트 문자열)**

MQOR 및 MQPMR 레코드 뒤에 오는 데이터의 형식 이름.

이는 MQOD 및 MQPMR 레코드(마지막에 발생하는 레코드) 뒤에 오는 데이터의 형식 이름을 지정합니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 이 필드의 코딩 규칙은 MQMD의 *MDFMT* 필드에 대한 규칙과 동일합니다.

이 필드의 초기값은 FMNONE입니다.

### **DHLEN(10자리의 부호 있는 정수)**

MQDH 구조 길이와 다음의 MQOR 및 MQPMR 레코드.

이는 MQDH 구조의 시작에서 MQOR 및 MQPMR 레코드의 배열 다음에 오는 메시지 데이터의 시작까지의 바이트 수입니다. 데이터는 다음 순서로 발생합니다.

- MQDH 구조
- MQOR 레코드의 배열
- MQPMR 레코드의 배열
- 메시지 데이터

MQOR 및 MQPMR 레코드의 배열은 MQDH 구조 내에 포함된 오프셋에서 처리됩니다. 이러한 오프셋으로 인해 하나 이상의 MQDH 구조, 레코드 배열 및 메시지 데이터 사이의 사용되지 않은 바이트가 생성되는 경우, 사용되지 않은 바이트는 *DHLEN*의 값에 포함되어야 하지만 해당 바이트의 콘텐츠는 큐 관리자에 의해 보존되지 않습니다. MQOR 레코드의 배열이 MQPMR 레코드의 배열보다 선행하는 것이 올바릅니다.

이 필드의 초기값은 0입니다.

### **DHORO(10자리의 부호 있는 정수)**

MQDH 시작 이후 첫 번째 MQOR 레코드의 오프셋.

이 필드는 목적지 큐의 이름을 포함하여 MQOR 오브젝트 레코드의 배열에서 첫 번째 레코드의 오프셋(바이트)을 제공합니다. 이 배열에 *DHCNT*개의 레코드가 있습니다. 이러한 레코드(첫 번째 오브젝트 레코드와 이전 필드 사이에 건너뛴 바이트 포함)는 *DHLEN* 필드에서 제공한 길이에 포함됩니다.

분배 목록은 항상 하나 이상의 목적지를 포함해야 하므로 *DHORO*은(는) 항상 0보다 커야 합니다.

이 필드의 초기값은 0입니다.

### **DHPRF(10자리의 부호 있는 정수)**

MQPMR 필드가 있음을 표시하는 플래그.

다음 플래그 중 하나 이상을 지정할 수 있습니다.

#### **PFMID**

메시지 ID 필드가 존재합니다.

#### **PFCID**

상관 ID 필드가 존재합니다.

#### **PFGID**

그룹 ID 필드가 존재합니다.

#### **PFFB**

피드백 필드가 존재합니다.

#### **PFACC**

Accounting-token 필드가 존재합니다.

MQPMR 필드가 없는 경우 다음을 지정할 수 있습니다.

#### **PFNONE**

넣기 메시지 레코드 필드가 없습니다.

PFNONE은 프로그램 문서화를 지원하기 위해 정의됩니다. 이 상수는 다른 상수와 함께 사용하기 위한 용도는 아니지만, 값이 0이므로 그러한 사용을 감지할 수 없습니다.

이 필드의 초기값은 PFNONE입니다.

### DHPRO(10자리의 부호 있는 정수)

MQDH 시작 이후 첫 번째 MQPMR 레코드의 오프셋.

이 필드는 메시지 특성을 포함하여 MQPMR 넣기 메시지 레코드의 배열에서 첫 번째 레코드의 오프셋(바이트)을 제공합니다. 존재하는 경우 이 배열에 DHCNT개의 레코드가 있습니다. 이러한 레코드(첫 번째 넣기 메시지 레코드와 이전 필드 사이에 건너뛴 바이트 포함)는 DHLEN 필드에서 제공한 길이에 포함됩니다.

넣기 메시지 레코드는 선택사항입니다. 레코드가 제공되지 않는 경우 DHPRO는 0이고 DHPRF에 값 PFNONE가 있습니다.

이 필드의 초기값은 0입니다.

### DHSID(4바이트 문자열)

구조 ID입니다.

값은 다음과 같아야 합니다.

#### DHSIDV

분배 헤더 구조의 ID.

이 필드의 초기값은 DHSIDV입니다.

### DHVER(10자리의 부호 있는 정수)

구조 버전 번호입니다.

값은 다음과 같아야 합니다.

#### DHVER1

분배 헤더 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### DHVERC

분배 헤더 구조의 현재 버전.

이 필드의 초기값은 DHVER1입니다.

## 초기값

표 696. MQDH의 필드		
필드 이름	상수의 이름	상수의 값
DHSID	DHSIDV	'DH??'
DHVER	DHVER1	1
DHLEN	없음	0
DHENC	없음	0
DHCSI	CSUNDF	0
DHFMT	FMNONE	공백
DHFLG	DHFNON	0
DHPRF	PFNONE	0
DHCNT	없음	0
DHORO	없음	0
DHPRO	없음	0

참고:



1. ~ 기호는 단일 공백 문자를 나타냅니다.

## RPG 선언

```
D*..1.....2.....3.....4.....5.....6.....7..
D* MQDH Structure
D*
D* Structure identifier
D DHSID 1 4 INZ('DH ')
D* Structure version number
D DHVER 5 8I 0 INZ(1)
D* Length of MQDH structure plus following MQOR and MQPMP records
D DHLEN 9 12I 0 INZ(0)
D* Numeric encoding of data that follows the MQOR and MQPMP records
D DHENC 13 16I 0 INZ(0)
D* Character set identifier of data that follows the MQOR and MQPMP
D* records
D DHCSI 17 20I 0 INZ(0)
D* Format name of data that follows the MQOR and MQPMP records
D DHFMT 21 28 INZ(' ')
D* General flags
D DHFLG 29 32I 0 INZ(0)
D* Flags indicating which MQPMP fields are present
D DHPRF 33 36I 0 INZ(0)
D* Number of MQOR records present
D DHCNT 37 40I 0 INZ(0)
D* Offset of first MQOR record from start of MQDH
D DHORO 41 44I 0 INZ(0)
D* Offset of first MQPMP record from start of MQDH
D DHPRO 45 48I 0 INZ(0)
```

## IBM i IBM i의 MQDLH (데드-레터 헤더)

### 개요

#### 목적

MQDLH 구조는 데드 레터(미발송 메시지) 큐에 메시지의 애플리케이션 메시지 데이터를 앞에 붙이는 정보를 설명합니다. 큐 관리자 또는 메시지 채널 에이전트가 큐로 경로 재지정되었기 때문에 메시지는 데드-레터 큐에 도착할 수 있습니다. 애플리케이션은 큐에 직접적으로 메시지를 넣을 수 있습니다.

#### 형식 이름

FMDLH

#### 문자 세트 및 인코딩

MQDLH는 애플리케이션 메시지 데이터의 시작 부분에 있을 수 있습니다. 이 경우 MQDLH 구조에서 필드는 MDCSI 및 MDENC 필드에 의해 지정된 문자 세트와 인코딩에 있습니다. 그렇지 않은 경우 문자 세트와 인코딩은 MQDLH에 선행하는 헤더 구조에서 MDCSI 및 MDENC 필드에 의해 설정됩니다.

문자 세트는 큐 이름에 유효한 문자에 사용되는 단일 바이트 문자가 있어야 합니다.

#### 사용법

데드-레터 큐에 직접 메시지를 넣은 애플리케이션은 메시지 데이터에 접두부로 MQDLH 구조를 지정해야 하며 적절한 값으로 필드를 초기화해야 합니다. 그러나 MQDLH 구조가 존재하거나 필드에 유효한 값이 지정되어 있지 않아도 됩니다.

메시지를 데드-레터 큐에 넣기에는 너무 긴 경우 애플리케이션이 다음 중 하나를 수행하는 것을 고려해야 합니다.

- 데드-레터 큐와 잘 맞도록 메시지 데이터를 자르십시오.
- 메시지를 보조 기억장치에 기록하고 메시지가 너무 길다는 것을 표시하기 위해 데드-레터 큐에 예외 보고서를 메시지를 배치하십시오.
- 메시지를 제거하고 오류를 해당 발신자로 리턴하십시오. 메시지가 중요 메시지인 경우입니다. 발신자에게 여전히 메시지의 사본이 있는 경우에만 메시지를 제거하십시오 (예: 통신 채널로부터 메시지 채널 에이전트에 의해 수신된 메시지).

어느 선택이 적절한지는 애플리케이션의 디자인에 따라 다릅니다.

세그먼트인 메시지를 정면에 MQDLH 구조로 넣을 때 큐 관리자는 특수 처리를 수행합니다. 추가 세부사항은 MQMDE 구조에 대한 설명을 참조하십시오.

- [1022 페이지의 『데드-레터 큐에 메시지 넣기』](#)
- [1022 페이지의 『데드-레터 큐에서 메시지 가져오기』](#)
- [1022 페이지의 『필드』](#)
- [1026 페이지의 『초기값』](#)
- [1026 페이지의 『RPG 선언』](#)

## 데드-레터 큐에 메시지 넣기

메시지를 데드-레터 큐에 넣은 경우 MQPUT 또는 MQPUT1 호출에 사용된 MQMD 구조는 메시지와 연관된 MQMD와 동일해야 합니다. MQMD는 다음 경우를 제외하고 일반적으로 MQGET 호출로 리턴된 것 중 하나입니다.

- MDCSI 및 MDENC 필드는 MQDLH 구조에서 필드에 사용되는 문자 세트 및 인코딩에 설정되어야 합니다.
- MDFMT 필드는 데이터가 MQDLH 구조로 시작함을 표시하기 위해 FMDLH로 설정되어야 합니다.
- 컨텍스트 필드, MDACC, MDAID, MDAOD, MDPAN, MDPAT, MDPD, MDPT 및 MDUID는 환경에 적절한 컨텍스트 옵션을 사용하여 설정되어야 합니다.
  - 데드-레터 큐에 선행 메시지와 관련되지 않은 메시지를 넣는 애플리케이션은 PMDEFC 옵션을 사용해야 합니다. PMDEFC 옵션으로 큐 관리자는 해당 기본값에 메시지 디스크립터의 모든 컨텍스트 필드를 설정할 수 있습니다.
  - 데드-레터 큐에 수신된 메시지를 넣은 서버 애플리케이션은 원래 컨텍스트 정보를 보존하기 위해 PMPASA 옵션을 사용해야 합니다.
  - 데드-레터 큐에 수신된 메시지에 대한 응답을 넣은 서버 애플리케이션은 PMPASI 옵션을 사용해야 합니다. PMPASI 옵션은 ID 정보를 보존하지만 원래 정보를 서버 애플리케이션의 정보로 설정합니다.
  - 데드-레터 큐에 통신 채널에서 수신한 메시지를 넣은 메시지 채널 에이전트는 PMSETA 옵션을 사용해야 합니다. PMSETA 옵션은 원래 컨텍스트 정보를 보존합니다.

MQDLH 구조 자체에서 이 필드는 다음과 같이 설정되어야 합니다.

- DLCSI, DLENC 및 DLFMT 필드는 MQDLH 구조 뒤에 오는 데이터를 설명하는 값으로 설정되어야 합니다. 이 값은 일반적으로 원래 메시지 디스크립터의 값입니다.
- 컨텍스트 필드 DLPAT, DLPAN, DLPD 및 DLPT는 데드-레터 큐에 메시지를 넣은 애플리케이션에 적절한 값으로 설정되어야 합니다. 이러한 값은 원래 메시지와 관련이 없습니다.
- 다른 필드는 적절하게 설정되어야 합니다.

애플리케이션은 모든 필드에 올바른 값이 있고 문자 필드가 필드의 정의된 길이에 공백으로 채워졌는지 확인해야 합니다. 문자 데이터는 널 특징을 사용하여 종료되지 않아야 합니다. 큐 관리자는 MQDLH 구조에서 널 및 후속 문자를 공백으로 변환하지 않습니다.

## 데드-레터 큐에서 메시지 가져오기

데드-레터 큐에서 메시지를 가져오는 애플리케이션은 메시지가 MQDLH 구조로 시작되는지 확인해야 합니다. MQDLH 구조가 MDFMT 필드를 메시지 디스크립터 MQMD에서 조사하여 존재하는지 여부를 애플리케이션은 판별할 수 있습니다. 필드에 값 FMDLH가 있는 경우 메시지 데이터는 MQDLH 구조로 시작합니다. 의도된 큐에 대해 원래 너무 긴 경우 데드-레터 큐의 메시지를 잘릴 수 있습니다.

## 필드

MQDLH 구조에는 다음과 같은 필드가 포함됩니다. 필드에 대해서 알파벳순으로 설명합니다.

### DLCSI(10자리의 부호 있는 정수)

MQDLH 뒤에 오는 데이터의 문자 세트 ID

DLCISI는 MQDLH 구조 뒤에 오는 데이터의 문자 세트 ID를 지정합니다. 데이터는 일반적으로 원래 메시지에서 가져옵니다. MQDLH 구조 자체에서 문자 데이터에 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적합한 값으로 설정해야 합니다. 다음 특수 값을 사용할 수 있습니다.

#### **CSINHT**

이 구조의 문자 세트 ID를 상속합니다.

이 구조 다음에 오는 데이터의 문자 데이터는 이 구조와 동일한 문자 세트로 되어 있습니다.

큐 관리자가 구조의 실제 문자 세트 ID에 메시지로 송신한 구조에서 이 값을 변경합니다. 오류가 발생하지 않으면 CSINHT 값이 MQGET 호출에 의해 리턴되지 않습니다.

MQMD의 MDPAT 필드 값이 ATBRKR이면 CSINHT를 사용할 수 없습니다.

이 필드의 초기값은 CSUNDF입니다.

#### **DLDM(48바이트 문자열)**

원래 목적지 큐 관리자의 이름.

이는 메시지의 원래 목적지였던 큐 관리자의 이름입니다.

이 필드의 길이는 LNQMN으로 제공됩니다. 이 필드의 초기값은 48개의 빈 문자입니다.

#### **DLDQ(48바이트 문자열)**

원래 목적지 큐의 이름.

이는 메시지의 원래 목적지였던 메시지 큐의 이름입니다.

이 필드의 길이는 LNQN으로 제공됩니다. 이 필드의 초기값은 48개의 빈 문자입니다.

#### **DLENC(10자리의 부호 있는 정수)**

MQDLH 뒤에 오는 데이터의 숫자 인코딩.

DLENC는 MQDLH 구조 뒤에 오는 데이터의 숫자 인코딩을 지정합니다. 데이터는 일반적으로 원래 메시지에서 가져옵니다. MQDLH 구조 자체에서 숫자 데이터에 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적합한 값으로 설정해야 합니다.

이 필드의 초기값은 0입니다.

#### **DLFMT(8바이트 문자열)**

MQDLH 뒤에 오는 데이터의 형식 이름.

이는 MQDLH 구조(일반적으로 원래 메시지에서 가져옴) 뒤에 오는 데이터의 형식 이름을 지정합니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적합한 값으로 설정해야 합니다. 이 필드를 코드화하기 위한 규칙은 MQMD의 MDFMT 필드의 규칙과 같습니다.

이 필드의 길이는 LNFMT으로 제공됩니다. 이 필드의 초기값은 FMNONE입니다.

#### **DLPAN(28바이트 문자열)**

데드 레터(미발송 메시지) 큐에 메시지를 넣은 애플리케이션의 이름.

이름의 형식은 DLPAT 필드에 따라 다릅니다. [1063 페이지의 『IBM i의 MQMD\(메시지 디스크립터\)』](#)의 MDPAN 필드에 대한 설명을 참조하십시오.

메시지를 데드-레터 큐로 경로 재지정하는 큐 관리자인 경우 DLPAN은 큐 관리자 이름의 첫 28자를 포함합니다. 필요한 경우 이름은 공백으로 채웁니다.

이 필드의 길이는 LNPAN으로 제공됩니다. 이 필드의 초기값은 28개의 공백 문자입니다.

#### **DLPAT(10자리의 부호 있는 정수)**

데드 레터(미발송 메시지) 큐에 메시지를 넣은 애플리케이션의 유형.

이 필드는 메시지 디스크립터 MQMD에서 MDPAT 필드와 같은 의미를 가지고 있습니다(세부사항은 [1063 페이지](#)의 『IBM i의 MQMD(메시지 디스크립터)』 참조).

메시지를 데드-레터 큐로 경로 재지정하는 큐 관리자인 경우 DLPAT에 값 ATQM이 있습니다.

이 필드의 초기값은 0입니다.

#### **DLPD(8바이트 문자열)**

메시지를 데드 레터(미발송 메시지) 큐에 넣은 날짜.

이 필드가 큐 관리자에 의해 생성될 때 날짜에 사용되는 형식입니다.

• YYYYMMDD

여기서 각 문자는 다음을 나타냅니다.

**YYYY**

연도(4자리 숫자)

**MM**

연 중 월(01에서 12)

**DD**

월 중 일(01에서 31)

GMT에 맞게 정확하게 설정된 시스템 시계에 따라 DLPD 및 DLPT 필드에 그리니치 표준시(GMT)가 사용됩니다.

이 필드의 길이는 LNPDAT으로 제공됩니다. 이 필드의 초기값은 8개의 빈 문자입니다.

#### **DLPT(8바이트 문자열)**

메시지를 데드 레터(미발송 메시지) 큐에 넣은 시간.

이 필드가 큐 관리자에 의해 생성될 때 시간에 사용되는 형식입니다.

• HHMMSSSTH

여기서 문자는 다음을 나타냅니다(순서대로).

**HH**

시간(00에서 23까지)

**MM**

분(00에서 59까지)

**SS**

초(0 - 59, 이 주제의 후자 부분 참조)

**T**

1/10초(0에서 9)

**H**

1/100초(0에서 9)

**참고:** 시스템 시계가 정확한 시간 표준에 동기화되는 경우 DLPT에서 초에 대해 60 또는 61이 리턴될 수 있습니다. 윤초가 글로벌 시간 표준에 삽입되는 경우에 추가 초가 발생합니다.

GMT에 맞게 정확하게 설정된 시스템 시계에 따라 DLPD 및 DLPT 필드에 그리니치 표준시(GMT)가 사용됩니다.

이 필드의 길이는 LNPTIM으로 제공됩니다. 이 필드의 초기값은 8개의 빈 문자입니다.

#### **DLREA(10자리의 부호 있는 정수)**

데드 레터(미발송 메시지) 큐에 도착한 이유 메시지.

이 메시지가 원래 목적지 큐 대신에 데드 레터에 큐에 배치된 이유를 식별합니다. FB\* 또는 RC\* 값(예: RC2053) 중 하나여야 합니다. 발생할 수 있는 공통 FB\* 값에 대한 세부사항은 [1063 페이지](#)의 『IBM i의 MQMD(메시지 디스크립터)』에서 *MDFB* 필드에 대한 설명을 참조하십시오.

값이 FBIFST - FBILST범위에 있는 경우, 실제 IMS 오류 코드는 DLREA 필드의 값에서 FBIERR 를 빼서 판별할 수 있습니다.

일부 FB\* 값은 이 필드에서만 발생합니다. 데드-레터 큐로 전송된 저장소 메시지, 트리거 메시지 또는 전송 큐 메시지와 관계가 있습니다. 값은 다음과 같습니다.

#### **FBABEG**

애플리케이션을 시작할 수 없습니다.

트리거 메시지를 처리하는 애플리케이션은 트리거 메시지의 TMAI 필드에서 이름 지정된 애플리케이션을 시작할 수 없습니다. [1180 페이지의 『MQTM - 트리거 메시지』](#)의 내용을 참조하십시오.

#### **FBATYP**

애플리케이션 유형 오류.

트리거 메시지를 처리하는 애플리케이션은 트리거 메시지의 TMAT 필드가 올바르지 않기 때문에 애플리케이션을 시작할 수 없습니다. [1180 페이지의 『MQTM - 트리거 메시지』](#)의 내용을 참조하십시오.

#### **FBBODC**

클러스터 수신자 채널이 삭제됩니다.

메시지는 FBIERR 옵션으로 연 클러스터 큐에 사용하는 클러스터 전송 큐에 있습니다. 메시지를 전송하기 전에 목적지 큐로 메시지를 전송하는 데 사용되는 원격 클러스터 수신자 채널이 삭제되었습니다. FBIERR이 지정되었기 때문에 큐가 열렸을 때 선택된 채널만 메시지를 전송하는 데 사용할 수 있습니다. 이 채널을 더 이상 사용할 수 없기 때문에 메시지를 데드-레터 큐에 배치했습니다.

#### **FBNARM**

메시지가 저장소 메시지가 아닙니다.

#### **FBSBCX**

채널 자동 정의 엑시트에 의해 중단된 메시지입니다.

#### **FBSBMX**

채널 메시지 엑시트가 중지한 메시지.

#### **FBTM**

MQTM 구조가 올바르지 않거나 누락되어 있습니다.

MQMD의 MDFMT 필드는 FMTM을 지정하지만 메시지는 올바른 MQTM 구조로 시작하지 않습니다. 예를 들어, TMSID 니모닉 아이캐치는 올바르지 않을 수 있습니다. TMVER은 인식되지 않을 수 있습니다. 트리거 메시지의 길이는 MQTM 구조를 포함하기에 불충분할 수 있습니다.

#### **FBXQME**

전송 큐의 메시지가 올바른 형식이 아닙니다.

메시지 채널 에이전트는 전송 큐의 메시지가 올바른 형식이 아님 발견했습니다. 메시지 채널 에이전트는 이 피드백 코드를 사용하여 데드-레터 큐에 메시지를 넣습니다.

이 필드의 초기값은 RCNONE입니다.

#### **DLSID(4바이트 문자열)**

구조 ID입니다.

값은 다음과 같아야 합니다.

#### **DLSIDV**

데드 레터 헤더 구조의 ID입니다.

이 필드의 초기값은 DLSIDV입니다.

#### **DLVER(10자리의 부호 있는 정수)**

구조 버전 번호입니다.

값은 다음과 같아야 합니다.

#### **DLVER1**

데드 레터 헤더 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

### DLVERC

데드 레터 헤더 구조의 현재 버전.

이 필드의 초기값은 DLVER1입니다.

### 초기값

표 697. 필드 위치 MQDLH		
필드 이름	상수의 이름	상수의 값
DLSID	DLSIDV	'DLH~'
DLVER	DLVER1	1
DLREA	RCNONE	0
DLDQ	없음	공백
DLDM	없음	공백
DLENC	없음	0
DLCSI	CSUNDF	0
DLFMT	FMNONE	공백
DLPAT	없음	0
DLPAN	없음	공백
DLPD	없음	공백
DLPT	없음	공백

### 참고:

1. ~ 기호는 단일 공백 문자를 나타냅니다.

### RPG 선언

```

D*..1.....2.....3.....4.....5.....6.....7..
D* MQDLH Structure
D*
D* Structure identifier
D DLSID          1          4    INZ('DLH ')
D* Structure version number
D DLVER          5          8I 0 INZ(1)
D* Reason message arrived on dead-letter(undelivered-message) queue
D DLREA          9          12I 0 INZ(0)
D* Name of original destination queue
D DLDQ          13         60    INZ
D* Name of original destination queue manager
D DLDM          61        108    INZ
D* Numeric encoding of data that followsMQDLH
D DLENC         109        112I 0 INZ(0)
D* Character set identifier of data thatfollows MQDLH
D DLCSI         113        116I 0 INZ(0)
D* Format name of data that followsMQDLH
D DLFMT         117        124    INZ(' ')
D* Type of application that put messageon dead-letter
D* (undelivered-message)queue
D DLPAT         125        128I 0 INZ(0)
D* Name of application that put messageon dead-letter
D* (undelivered-message)queue
D DLPAN         129        156    INZ
D* Date when message was put ondead-letter (undelivered-message)queue
D DLPD         157        164    INZ

```

## IBM i IBM i의 MQDMHO(메시지 핸들 삭제 옵션)

MQDMHO 구조를 사용하여 애플리케이션이 메시지 핸들을 삭제하는 방법을 제어하는 옵션을 지정할 수 있습니다.

### 개요

목적: 이 구조는 MQDLTMH 호출의 입력 매개변수입니다.

문자 세트 및 인코딩: MQDMHO의 데이터는 애플리케이션의 문자 세트 및 애플리케이션의 인코딩(ENNAT)에 있어야 합니다.

- [1027 페이지의 『필드』](#)
- [1027 페이지의 『초기값』](#)
- [1028 페이지의 『RPG 선언』](#)

### 필드

MQDMHO 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

#### DMOPT(10자리의 부호 있는 정수)

값은 다음과 같아야 합니다.

##### DMNONE

옵션이 지정되지 않았습니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 **DMNONE**입니다.

#### DMSID(10자리의 부호 있는 정수)

구조 ID이며 값은 다음과 같아야 합니다.

##### DMSIDV

삭제 메시지 핸들 옵션 구조의 ID.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 **DMSIDV**입니다.

#### DMVER(10자리의 부호 있는 정수)

구조 버전 번호이며 값은 다음과 같아야 합니다.

##### DMVER1

버전-1 삭제 메시지 핸들 옵션 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

##### DMVERC

삭제 메시지 핸들 옵션 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 **DMVER1**입니다.

### 초기값

표 698. MQDMHO의 필드		
필드 이름	상수의 이름	상수의 값
DMSID	DMSIDV	'DMHO'

표 698. MQDMHO의 필드 (계속)		
필드 이름	상수의 이름	상수의 값
DMVER	DMVER1	1
DMOPT	DMNONE	0

## RPG 선언

```

D* MQDMHO Structure
D*
D*
D* Structure identifier
D DMSID          1      4   INZ('DMHO')
D*
D* Structure version number
D DMVER          5      8I 0 INZ(1)
D*
D* Options that control the action of MQDLTMH
D DMOPT          9      12I 0 INZ(0)

```

## IBM i IBM i의 MQDMPO(메시지 특성 삭제 옵션)

삭제 메시지 특성 옵션을 정의하는 구조입니다.

### 개요

**목적:** MQDMPO 구조를 사용하여 애플리케이션이 메시지 특성이 삭제되는 방법을 제어하는 옵션을 지정할 수 있습니다. 구조는 MQDLTMP 호출의 입력 매개변수입니다.

**문자 세트 및 인코딩:** MQDMPO의 데이터는 애플리케이션의 문자 세트 및 애플리케이션의 인코딩(ENNAT)에 있어야 합니다.

- [1028 페이지의 『필드』](#)
- [1029 페이지의 『초기값』](#)
- [1029 페이지의 『RPG 선언』](#)

### 필드

MQDMPO 구조에는 다음과 같은 필드가 포함됩니다. 필드에 대해서 알파벳순으로 설명합니다.

#### DPOPT(10자리의 부호 있는 정수)

삭제 메시지 특성 옵션 구조 - DPOPT 필드.

**위치 옵션:** 다음 옵션은 특성 커서와 비교하여 특성의 상대 위치와 관련이 있습니다.

#### DPDELF

지정된 이름과 일치하는 첫 번째 특성을 삭제합니다.

#### DPDELC

특성 커서로 가리킨 특성(즉, IPINQF 또는 IPINQN 옵션을 사용하여 마지막으로 조회된 특성)을 삭제합니다.

메시지 핸들이 재사용될 때 특성 커서가 재설정됩니다. 또한 메시지 핸들이 MQGET 호출의 MQGMO 또는 MQPUT 호출의 MQPMO 구조에서 HMSG 필드에 지정될 때도 재설정됩니다.

메시지 핸들이 재사용되거나 메시지 핸들이 MQGET 호출의 MQGET 구조 또는 MQPUT 호출의 MQPMO 구조에서 MQGMO 구조의 HMSG 필드에 지정될 때 재설정됩니다.

이 옵션이 특성 커서가 아직 설정되지 않았을 때 사용되는 경우 완료 코드 CCFAIL 및 이유 RC2471로 호출에 실패합니다. 또한 특성 커서가 가리킨 특성이 이미 삭제된 경우에도 해당 코드로 실패합니다.



해당 옵션중 필요한 옵션이 없는 경우 다음 옵션을 사용할 수 있습니다.

**DPNONE**

옵션이 지정되지 않았습니다.

이 필드의 초기값은 DPDELF입니다.

**DPSID(10자리의 부호 있는 정수)**

삭제 메시지 특성 옵션 구조 - DPSID 필드.

구조 ID입니다. 값은 다음과 같아야 합니다.

**DPSIDV**

삭제 메시지 특성 옵션 구조의 ID입니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 DPSIDV입니다.

**DPVER(10자리의 부호 있는 정수)**

삭제 메시지 특성 옵션 구조 - DPVER 필드.

구조 버전 번호입니다. 값은 다음과 같아야 합니다.

**DPVER1**

삭제 메시지 특성 옵션 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

**DPVERC**

삭제 메시지 특성 옵션 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 DPVER1입니다.

**초기값**

표 699. MQDPMO의 필드		
필드 이름	상수의 이름	상수의 값
DPSID	DPSIDV	'DMPO'
DPVER	DPVER1	1
DPOPT	MQDLTMP 조치를 제어하는 옵션	DPNONE

**RPG 선언**

```

D* MQDPMO Structure
D*
D*
D* Structure identifier
D  DPSID          1      4    INZ('DMPO')
D*
D* Structure version number
D  DPVER          5      8I 0  INZ(1)
D*
** Options that control the action of
D* MQDLTMP
D  DPOPT          9     12I 0  INZ(0)
    
```

**IBM i IBM i의 MQEPH(임베드된 PCF 헤더)**

## 개요

### 목적

MQEPH 구조는 해당 메시지가 프로그래밍 가능 명령 형식(PCF) 메시지일 때 메시지에 있는 추가 데이터를 설명합니다. *EPPFH* 필드는 이 구조 뒤에 오는 PCF 매개변수를 정의하고 이로 인해 다른 헤더가 있는 PCF 메시지 데이터를 따를 수 있습니다.

### 형식 이름

EPFMT

### 문자 세트 및 인코딩

MQEPH의 데이터는 로컬 큐 관리자의 문자 세트 및 인코딩에 있어야 합니다. 이는 **CCSID** 큐 관리자 속성으로 지정됩니다.

MQEPH의 문자 세트 및 인코딩을 *MDCSI* 및 *MDENC* 필드에 설정하십시오.

- MQMD(MQEPH 구조가 메시지 데이터의 시작에 있는 경우) 또는
- MQEPH 구조에 선행하는 헤더 구조(다른 모든 경우).

### 사용법

명령을 명령 서버 또는 기타 큐 관리자 PCF 수용 서버에 보내기 위해 MQEPH 구조를 사용할 수 없습니다.

마찬가지로 명령 서버 또는 기타 큐 관리자 PCF-수용 서버는 MQEPH 구조를 포함하여 응답 또는 이벤트를 생성하지 않습니다.

- [1030 페이지의 『필드』](#)
- [1032 페이지의 『초기값』](#)
- [1032 페이지의 『RPG 선언』](#)

## 필드

MQEPH 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

### EPCSI(10자리의 부호 있는 정수)

이는 MQEPH 구조 뒤에 오는 데이터의 문자 세트 ID 및 연관된 PCF 매개변수입니다. MQEPH 구조 자체의 문자 데이터에는 적용되지 않습니다.

이 필드의 초기값은 EPCUND입니다.

### EPENC(10자리의 부호 있는 정수)

MQEPH 구조 및 연관된 PCF 매개변수 다음에 오는 데이터의 숫자 인코딩입니다. MQEPH 구조 자체에 있는 문자 데이터에는 적용되지 않습니다.

이 필드의 초기값은 0입니다.

### EPFLG(10자리의 부호 있는 정수)

다음 값을 사용할 수 있습니다.

#### EPNONE

플래그가 지정되지 않았습니다. *MDCSI* EPNONE는 프로그램 문서화를 지원하기 위해 정의됩니다. 이 상수는 다른 상수와 함께 사용하기 위한 용도는 아니지만, 값이 0이므로 그러한 사용을 감지할 수 없습니다.

#### EPCSEM

문자 데이터가 포함된 매개변수의 문자 세트는 각 구조의 *CCSID* 필드에 개별적으로 지정됩니다. *EPSID* 및 *EPFMT* 필드의 문자 세트는 MQEPH 구조를 생성하는 헤더 구조의 *CCSID*에서 정의하거나 MQEPH 구조가 메시지의 시작 부분에 있는 경우 MQMD의 *MDCSI* 필드에서 정의됩니다.

이 필드의 초기값은 EPNONE입니다.

### EPFMT(8바이트 문자열)

이는 MQEPH 구조 및 연관된 PCF 매개변수 다음에 오는 데이터의 형식 이름입니다.

이 필드의 초기값은 EPFMNO입니다.

### **EPLEN(10자리의 부호 있는 정수)**

이는 다음 헤더 구조 앞에 오는 데이터의 양입니다. 여기에는 다음이 포함됩니다.

- MQEPH 헤더의 길이
- 헤더 다음에 오는 모든 PCF 매개변수의 길이
- 해당 매개변수 다음에 오는 공백 채우기

EPLEN은 4의 배수여야 합니다.

구조의 고정 길이 부분은 EPSTLF에 의해 정의됩니다.

이 필드의 초기값은 68입니다.

### **EPPCFH(MQCFH)**

이는 MQEPH 구조 다음에 오는 PCF 매개변수를 정의하는 PCF(Programmable Command Format) 헤더입니다. 이로 인해 다른 헤더가 있는 PCF 메시지 데이터를 따를 수 있습니다.

PCF 헤더는 처음에 다음 값으로 정의됩니다.

표 700. EPPCFH의 필드		
필드 이름	상수의 이름	상수의 값
EP3TYP	CFTNON	0
EP3LEN	FHLENV	36
EP3VER	FHVER3	3
EP3CMD	CMNONE	0
EP3SEQ	없음	1
EP3CTL	CFCLST	1
EEP3CC	CCOK	0
EP3REA	RCNONE	0
EP3CNT	없음	0

애플리케이션은 EP3TYP를 CFTNON에서 임베드된 PCF 헤더로 구성된 사용에 올바른 구조 유형으로 변경해야 합니다.

### **EPSID(4바이트 문자열)**

값은 다음과 같아야 합니다.

#### **EPSTID**

임베드된 PCF 헤더 구조 ID.

이 필드의 초기값은 EPSTID입니다.

### **EPVER(10자리의 부호 있는 정수)**

가능한 값은 다음과 같습니다.

#### **EPVER1**

임베드된 PCF 헤더 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### **EPVER3**

임베드된 PCF 헤더 구조의 현재 버전.

이 필드의 초기값은 EPVER3입니다.

## 초기값

표 701. MQEPH의 필드		
필드 이름	상수의 이름	상수의 값
EPSID	EPSTID	'EP- -'
EPVER	EPVER1	1
EPLEN	EPSTLF	68
EPENC	없음	0
EPCSI	EPCUND	0
EPFMT	EPFMNO	공백
EPFLG	EPNONE	0
EPPCFH	1031 페이지의 표 700에 정의된 이름 및 값	0

### 참고:

1. - 기호는 단일 공백 문자를 나타냅니다.

## RPG 선언

```

D*..1.....2.....3.....4.....5.....6.....7..
D* MQEPH Structure
D*
D* Structure identifier
D EPSID 1 4
D* Structure version number
D EPVER 5 8I 0
D* Total length of MQEPH including MQCFHand parameter structures
D* that follow
D EPLEN 9 12I 0
D* Numeric encoding of data that follows last PCF parameter structure
D EPENC 13 16I 0
D* Character set identifier of data that follows last PCF parameter
D* structure
D EPCSI 17 20I 0
D* Format name of data that follows last PCF parameter structure
D EPFMT 21 28
D* Flags
D EPFLG 29 32I 0
D* Programmable Command Format Header
D EP3TYP 33 36I 0
D EP3LEN 37 40I 0
D EP3VER 41 44I 0
D EP3CMD 45 48I 0
D EP3SEQ 49 52I 0
D EP3CTL 53 56I 0
D EP3CC 57 60I 0
D EP3REA 61 64I 0
D EP3CNT 65 68I 0

```

## IBM i IBM i의 MQGMO(메시지 가져오기 옵션)

MQGMO 구조로 애플리케이션이 메시지가 큐에서 제거되는 방법을 제어하는 옵션을 지정할 수 있습니다.

### 개요

#### 목적

구조는 MQGET 호출의 입출력(I/O) 매개변수입니다.

#### 버전

MQGMO의 현재 버전은 GMVER4입니다. 최신 버전의 구조에만 있는 필드는 다음에 오는 설명에서 최신 필드로 식별됩니다.

제공된 COPY 파일에는 환경에서 지원하지만 *GMVER* 필드의 초기값이 *GMVER1*로 설정된 *MQGMO*의 최신 버전이 포함되어 있습니다. *version-1* 구조에 없는 필드를 사용하려면 애플리케이션이 *GMVER* 필드를 필요한 버전의 버전 번호로 설정해야 합니다.

### 문자 세트 및 인코딩

*MQGMO*의 데이터는 **CodedCharSetId** 큐 관리자 속성에 의해 지정된 문자 세트 및 *ENNAT*에 의해 제공된 로컬 큐 관리자의 인코딩으로 되어 있어야 합니다. 그러나 애플리케이션이 *IBM MQ* 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트와 인코딩으로 작성되어야 합니다.

- [1033 페이지의 『필드』](#)
- [1050 페이지의 『초기값』](#)
- [1051 페이지의 『RPG 선언』](#)

### 필드

*MQGMO* 구조에는 다음과 같은 필드가 포함됩니다. 필드에 대해서 알파벳순으로 설명합니다.

#### GMGST(1바이트 문자 문자열)

검색된 메시지가 그룹에 있는지 여부를 표시하는 플래그.

값은 다음 중 하나입니다.

##### GSNIG

메시지는 그룹에 없습니다.

##### GSMIG

메시지가 그룹에 있지만 그룹의 마지막이 아닙니다.

##### GSLMIG

메시지가 그룹의 마지막에 있습니다.

또한 이 값은 그룹이 하나의 메시지로만 구성된 경우에 리턴되는 값입니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 *GSNIG*입니다. *GMVER*이(가) *GMVER2*보다 작으면 이 필드는 무시됩니다.

#### GMMH(10자리의 부호 있는 정수)

메시지 핸들

*GMPRAQ* 옵션이 지정되고 *PRPCTL* 큐 속성이 *PRPRFH*로 설정되지 않으면 이는 메시지의 특성이 큐에서 검색되는 메시지에 대한 핸들로 채워집니다. 핸들은 *MQCRTMH* 호출에 의해 작성됩니다. 이미 핸들과 연관된 특성은 메시지를 검색하기 전에 지워집니다.

다음 값을 지정할 수도 있습니다.

##### MQHM\_NONE

제공된 메시지 핸들이 없습니다.

유효한 메시지 핸들이 제공되고 메시지 특성, 입력 필드에 사용된 메시지 핸들과 연관된 메시지 디스크립터를 포함하기 위해 출력에 사용된 경우 메시지 디스크립터는 *MQGET* 호출에 필요합니다.

메시지 디스크립터가 *MQGET* 호출에 지정되면 항상 메시지 핸들과 연관된 메시지 디스크립터에 우선합니다.

*GMPRRF*가 지정되거나 *GMPRAQ*가 지정되고 *PRPCTL* 큐 속성이 *PRPRFH*이면 메시지 디스크립터 매개변수가 지정되는 않은 경우 이유 코드 *RC2026*으로 호출에 실패합니다.

*MQGET* 호출에서 리턴 시 이 메시지 핸들과 연관된 특성 및 메시지 디스크립터는 검색된(하나의 *MQGET* 호출에 제공되는 경우 메시지 디스크립터 또한) 메시지의 상태를 반영하기 위해 업데이트됩니다. 그런 다음 메시지의 특성은 *MQINQMP* 호출을 사용하여 조회할 수 있습니다.

메시지 디스크립터 확장자를 제외하고(존재할 경우) *MQINQMP* 호출로 조회할 수 있는 특성은 메시지 데이터에 포함되지 않습니다. 메시지 데이터의 특성에 포함된 큐의 메시지는 데이터가 애플리케이션으로 리턴되기 전에 메시지 데이터에서 제거됩니다.

메시지 핸들이 제공되지 않거나 버전이 GMVER4 미만인 경우에는 MQGET 호출에 유효한 메시지 디스크립터를 제공해야 합니다. 메시지 특성(메시지 디스크립터에 포함된 특성 제외)은 MQGMO 구조 및 PRPCTL 큐 속성의 특성 옵션 값에 따라 메시지 데이터에 리턴됩니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 HMNONE입니다. GMVER이(가) GMVER4보다 작으면 이 필드는 무시됩니다.

### GMMO(10자리의 부호 있는 정수)

MQGET에 대해 사용된 선택 기준을 제어하는 옵션입니다.

이러한 옵션을 사용하면 애플리케이션이 **MSGDSC** 매개변수에서 MQGET 호출로 리턴된 메시지를 선택하는데 사용되는 필드를 선택할 수 있습니다. 애플리케이션은 이 필드에 필수 옵션을 설정한 후 **MSGDSC** 매개변수의 해당 필드를 해당 필드에 필요한 값으로 설정합니다. 메시지의 MQMD에 해당 값이 있는 메시지만 MQGET 호출에서 해당 **MSGDSC** 매개변수를 사용하는 검색 후보입니다. 리턴할 메시지를 선택할 때 해당하는 일치 옵션이 지정되지 않는 필드는 무시됩니다. MQGET 호출에 사용할 선택 기준이 없는 경우(즉, 메시지가 허용 가능함), **GMMO**을(를) **MONONE**으로 설정해야 합니다.

GMLOGO가 지정되는 경우 특정 메시지만 다음 MQGET 호출에 의해 리턴할 수 있습니다.

- 현재 그룹 또는 논리 메시지가 없는 경우 **MDSEQ**이(가) 1이고 **MDOFF**이(가) 0인 메시지만 리턴할 수 있습니다. 이 경우 다음 옵션 중 하나 이상을 사용하여 적합한 메시지 중 하나가 리턴된 메시지를 선택할 수 있습니다.
  - MOMSGI
  - MOCORI
  - MOGRPI
- 현재 그룹이거나 논리 메시지인 경우 그룹의 다음 메시지 또는 논리 메시지의 다음 세그먼트만 리턴할 수 있고 이는 **MO\*** 옵션을 지정하여 변경될 수 없습니다.

두 경우 모두 적용 가능하지 않은 일치 옵션을 지정할 수 있지만 **MSGDSC** 매개변수에서 관련 필드의 값은 리턴될 메시지의 해당 필드 값과 일치해야 합니다. 호출은 실패하며 이유 코드는 RC2247이며 이 조건은 충족되지 않습니다.

GMMUC 또는 GMBRWC가 지정된 경우 **GMMO**은(는) 무시됩니다.

다음 옵션 중 하나 이상을 지정할 수 있습니다.

#### MOMSGI

지정된 메시지 ID로 메시지를 검색하십시오.

이 옵션은 검색할 메시지에 MQGET 호출의 **MSGDSC** 매개변수에 있는 **MDMID** 필드의 값과 일치하는 메시지 ID가 있어야 함을 지정합니다. 이 일치하는 적용할 수 있는 기타 일치에 추가됩니다(예: 상관 ID).

이 옵션을 지정하지 않으면 **MSGDSC** 매개변수의 **MDMID** 필드가 무시되고 모든 메시지 ID가 일치합니다.

**참고:** 메시지 ID **MINONE**는 메시지에 대한 MQMD의 메시지 ID와도 일치하는 특수 값입니다. 따라서 **MINONE**가 있는 **MOMSGI**를 지정하면 **MOMSGI**를 지정하지 않는 것과 동일합니다.

#### MOCORI

지정된 상관 ID로 메시지를 검색하십시오.

이 옵션은 검색할 메시지에 MQGET 호출의 **MSGDSC** 매개변수에 있는 **MDCID** 필드의 값과 일치하는 상관 ID가 있어야 함을 지정합니다. 이 일치하는 적용할 수 있는 기타 일치에 추가됩니다(예: 메시지 ID).

이 옵션을 지정하지 않으면 **MSGDSC** 매개변수의 **MDCID** 필드가 무시되고 상관 ID가 일치합니다.

**참고:** 상관 ID **CINONE**는 메시지에 대한 MQMD의 상관 ID와도 일치하는 특수 값입니다. 따라서 **CINONE**가 있는 **MOMSGI**를 지정하면 **MOCORI**를 지정하지 않는 것과 동일합니다.

#### MOGRPI

지정된 그룹 ID로 메시지를 검색하십시오.

이 옵션은 검색할 메시지에 MQGET 호출의 **MSGDSC** 매개변수에 있는 **MDGID** 필드의 값과 일치하는 그룹 ID가 있어야 함을 지정합니다. 이 일치하는 적용할 수 있는 기타 일치에 추가됩니다(예: 상관 ID).

이 옵션을 지정하지 않으면 **MSGDSC** 매개변수의 **MDGID** 필드가 무시되고 그룹 ID가 일치합니다.

**참고:** 그룹 ID GINONE는 메시지에 대한 MQMD의 그룹 ID와도 일치하는 특수 값입니다. 따라서 GINONE가 있는 MOGRPI를 지정하면 MOGRPI를 지정하지 않는 것과 동일합니다.

#### **MOSEQN**

지정된 메시지 순서 번호로 메시지를 검색하십시오.

이 옵션은 검색할 메시지에 MQGET 호출의 **MSGDSC** 매개변수에 있는 **MDSEQ** 필드 값과 일치하는 메시지 순서 번호가 있어야 함을 지정합니다. 이 일치는 적용할 수 있는 기타 일치에 추가됩니다(예: 그룹 ID).

이 옵션을 지정하지 않으면 **MSGDSC** 매개변수의 **MDSEQ** 필드가 무시되고 메시지 순서 번호가 일치합니다.

#### **MOOFFS**

지정된 오프셋으로 메시지를 검색하십시오.

이 옵션은 검색할 메시지에 MQGET 호출의 **MSGDSC** 매개변수에 있는 **MDOFF** 필드의 값과 일치하는 오프셋이 있어야 함을 지정합니다. 이 일치는 적용할 수 있는 기타 일치에 추가됩니다(예: 메시지 순서 번호).

이 옵션을 지정하지 않으면 **MSGDSC** 매개변수의 **MDOFF** 필드가 무시되고 모든 오프셋 일치가 무시됩니다.

설명된 옵션 중 필요한 사항이 없으면 다음 옵션을 사용할 수 있습니다.

#### **MONONE**

일치 항목이 없습니다.

이 옵션은 리턴할 메시지를 선택하는 데 사용할 일치 항목이 없음을 지정합니다. 따라서 큐의 모든 메시지는 검색할 수 있습니다(GMAMSA, GMASGA 및 GMCMPM 옵션에 의해 제어할 수 있음).

MONONE는 프로그램 문서를 지원하기 위해 정의됩니다. 이 옵션은 다른 MO\* 옵션과 함께 사용되기 위한 용도는 아니지만 값이 0이므로 이러한 사용을 감지할 수 없습니다.

이 필드는 입력 필드입니다. 이 필드의 초기값은 MOCORI이 있는 MOMSGI입니다. **GMVER**이(가) **GMVER2**보다 작으면 이 필드는 무시됩니다.

**참고:** **GMMO** 필드의 초기값은 이전 버전 큐 관리자와의 호환성을 위해 정의됩니다. 그러나 선택 기준을 사용하지 않고 큐에서 일련의 메시지를 읽을 때 이 초기값은 애플리케이션이 각 MQGET 호출 전에 **MDMID** 및 **MDCID** 필드를 MINONE 및 CINONE으로 재설정해야 합니다. **GMVER**을(를) **GMVER2**로 설정하고 **GMMO**을(를) **MONONE**으로 설정하면 **MDMID** 및 **MDCID** 재설정을 피할 수 있습니다.

#### **GMOPT(10자리의 부호 있는 정수)**

MQGET의 조치를 제어하는 옵션.

설명된 다음 옵션 0 이상을 지정할 수 있습니다. 둘 이상이 필요한 경우 값을 추가할 수 있습니다(동일한 상수를 두 번 이상 추가하지 않음). 올바르게 않은 옵션의 결합이 설명되어 있습니다. 다른 모든 결합은 유효합니다.

**대기 옵션:** 다음 옵션은 메시지가 큐에 도착하기까지 대기하는 것과 관련됩니다.

#### **GMWT**

메시지 도착까지 대기하십시오.

적당한 메시지가 도착할 때까지 애플리케이션이 대기합니다. 애플리케이션이 대기하는 최대 시간은 **GMWI**에 지정됩니다.

MQGET 요청이 금지되거나 대기하는 동안 MQGET 요청이 금지되는 경우 큐에 적절한 메시지가 있는지 여부와 관계없이 CCFAIL 및 이유 코드 RC2016으로 완료합니다.

이 옵션은 GMBRWF 또는 GMBRWN 옵션으로 사용할 수 있습니다.

여러 애플리케이션이 동일한 공유 큐에서 대기 중인 경우 적절한 메시지가 도착할 때 활성화되는 애플리케이션이 이 섹션에 나중에 설명됩니다.

**참고:** 다음 설명에서 찾아보기 MQGET 호출은 찾아보기 옵션 중 하나를 지정하지만 **GMLK**는 지정하지 않습니다. **GMLK** 옵션을 지정하는 MQGET 호출은 비열람 호출로 처리됩니다.

- 하나 이상의 비열람 MQGET 호출이 대기 중이지만 비열람 MQGET 호출이 대기 중이 아닌 경우 하나가 활성화됩니다.
- 하나 이상의 열람 MQGET 호출이 대기 중이지만 비열람 MQGET 호출이 대기 중이 아닌 경우 모두 활성화됩니다.
- 하나 이상의 비열람 MQGET 호출 및 하나 이상의 열람 MQGET 호출이 대기 중인 경우 비열람 MQGET 호출은 활성화되고 일부 또는 모든 열람 MQGET 호출이 활성화되거나 활성화되는 열람 MQGET 호출이 없습니다 (운영 체제의 스케줄링 고려사항 및 기타 요소에 의존하기 때문에 활성화된 열람 MQGET 호출 수를 예측할 수 없습니다.).

둘 이상의 비열람 MQGET 호출이 동일 큐에서 대기 중인 경우 하나의 호출만 활성화됩니다. 이 경우 큐 관리자는 다음 순서에서 대기 중인 비열람 호출에 우선순위를 두려고 시도합니다.

1. 특정 메시지(예: 특정 *MDMID* 또는 *MDCID*(또는 둘 다)가 있는 메시지)에 의해서만 충족될 수 있는 특정 가져오기-대기 요청.
2. 어떤 메시지에 의해서도 충족될 수 있는 일반 가져오기-대기 요청

다음 사항을 참고해야 합니다.

- 첫 번째 범주 내에서 더 구체적인 가져오기-대기 요청(예를 들어, *MDMID* 및 *MDCID* 둘 다 지정하는 요청)에는 추가 우선순위가 주어지지 않습니다.
- 각 범주 내에서 어떤 애플리케이션이 선택되었는지 예측할 수 없습니다. 특히 가장 오래 대기 중인 애플리케이션이 반드시 선택되지는 않습니다.
- 경로 길이 및 운영 체제의 우선 순위 스케줄링 고려사항은 예상보다 하위 운영 체제 우선순위의 대기 중인 애플리케이션이 메시지를 검색함을 의미할 수 있습니다.
- 대기 중이 아닌 애플리케이션이 대기 중인 애플리케이션보다 우선하여 메시지를 검색할 수도 있습니다.

GMBRWC 또는 GMMUC로 지정된 경우 GMWT는 무시됩니다. 오류는 발생하지 않습니다.

#### **GMNWT**

적절한 메시지가 없는 경우 바로 리턴하십시오.

적절한 메시지가 사용 가능하지 않은 경우 애플리케이션은 대기하지 않습니다. 이는 GMWT 옵션과 반대이며 프로그램 문서화를 지원하기 위해 정의됩니다. 지정된 것이 없으면 이 값이 기본값입니다.

#### **GMFIQ**

큐 관리자가 정지 중인 경우 실패합니다.

이 옵션은 큐 관리자가 정지 중인 경우 MQGET 호출에 실패하게 합니다.

이 옵션이 GMWT로 지정된 경우 큐 관리자가 정지 상태로 들어갈 때 대기는 미해결 상태가 됩니다.

- 대기가 취소되고 호출은 이유 코드 RC2161과 함께 완료 코드 CCFAIL을 리턴합니다.

GMFIQ가 지정되지 않고 큐 관리자가 정지 상태로 들어가는 경우 대기는 취소되지 않습니다.

**동기점 옵션:** 다음 옵션은 작업 단위 내에서 MQGET 호출의 참여와 관련됩니다.

#### **GMSYP**

동기점 제어를 가진 메시지를 가져오십시오.

요청이 일반 작업 단위 프로토콜 내에서 조작됩니다. 이 메시지는 다른 애플리케이션에 사용 불가능한 것으로 표시되지만, 작업 단위가 커밋되는 경우에만 큐에서 삭제됩니다. 작업 단위가 백아웃되는 경우 메시지가 다시 사용 가능하게 됩니다.

이 옵션 또는 GMNSYP가 지정되지 않은 경우 가져오기 요청은 작업 단위 내에 포함되지 않습니다.

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

- GMBRWF
- GMBRWC
- GMBRWN
- GMLK



- GMNSYP
- GMPSYP
- GMUNLK

#### GMPSYP

메시지가 지속적인 경우 동기점 제어를 가진 메시지를 가져오십시오.

이 요청은 정상 작업 단위 프로토콜 내에서 조작하기 위한 요청입니다. 단, 검색된 메시지가 지속적인 경우에만 해당됩니다. 지속 메시지에는 MQMD의 *MDPER* 필드에 *PEPER* 값이 있습니다.

- 메시지가 지속적인 경우 큐 관리자는 애플리케이션이 *GMSYP*를 지정했다라도 호출을 처리합니다.
- 메시지가 지속적이지 않은 경우 큐 관리자는 애플리케이션이 *GMSYP*를 지정했다라도 호출을 처리합니다(세부사항은 다음 절 참조).

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

- GMBRWF
- GMBRWC
- GMBRWN
- GMCMPM
- GMNSYP
- GMSYP
- GMUNLK

#### GMNSYP

동기점 제어가 없는 메시지를 가져오십시오.

요청은 일반 작업 단위 프로토콜의 외부에서 조작하는 것입니다. 메시지는 바로 큐에서 삭제됩니다(찾아보기 요청이 아닌 한). 작업 단위를 백아웃하여 메시지를 다시 사용 가능하게 만들 수 없습니다.

*GMBRWF* 또는 *GMBRWN*이 지정되면 이 옵션이 가정됩니다.

이 옵션 또는 *GMSYP*가 지정되지 않은 경우 가져오기 요청은 작업 단위 내에 포함되지 않습니다.

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

- GMSYP
- GMPSYP

**찾아보기 옵션:** 다음은 큐에서 메시지 찾아보기와 관련된 옵션입니다.

#### GMBRWF

큐의 시작로부터 찾아보기.

*OOBRW* 옵션을 사용하여 큐를 연 경우 찾아보기 커서가 설정되어 큐의 첫 번째 메시지 앞에 논리적으로 위치 지정됩니다. *GMBRWF*, *GMBRWN* 또는 *GMBRWC* 옵션을 지정하는 후속적 *MQGET* 호출은 비소거식으로 메시지를 큐에서 검색할 때 사용할 수 있습니다. 찾아보기 커서는 큐에 있는 메시지 내에 해당 위치를 표시합니다. *GMBRWN*을 사용하는 다음 *MQGET* 호출은 그 위치부터 해당 메시지를 검색합니다.

*GMBRWF*가 있는 *MQGET* 호출은 찾아보기 커서의 이전 위치를 무시합니다. 메시지 디스크립터에 지정된 조건을 충족하는 큐의 첫 번째 메시지가 검색됩니다. 메시지는 큐에 남아 있고 찾아보기 커서가 이 메시지에 위치 지정됩니다.

이 호출 후에 찾아보기 커서는 리턴된 메시지에 위치 지정됩니다. *GMBRWN*이 있는 다음 *MQGET* 호출이 발행되기 전에 메시지가 큐에서 제거되는 경우 해당 위치가 현재 비어 있더라도 찾아보기 커서는 메시지가 차지되는 큐의 위치에 남아 있습니다.

*GMMUC* 옵션은 필요한 경우 비열람 *MQGET* 호출과 함께 사용하여 큐에서 메시지를 제거하는 데 사용할 수 있습니다.

찾아보기 커서는 동일한 *HOB* 핸들을 사용하는 비탐색 *MQGET* 호출에 의해 이동되지 않습니다. 또한 *CCFAIL* 완료 코드 또는 *RC2080* 이유 코드를 리턴하는 찾아보기 *MQGET* 호출을 통해 이동되지도 않습니다.

이 옵션과 함께 GMLK 옵션을 지정하면 검색할 메시지를 잠글 수 있습니다.

GMBRWF는 논리 메시지의 세그먼트 및 그룹의 메시지 처리를 제어하는 GM\* 및 MO\* 옵션과 올바르게 결합하여 지정할 수 있습니다.

GMLOGO가 지정되면 메시지가 논리 순서에서 발견됩니다. 이 옵션을 생략하면 메시지를 실제 순서로 찾아봅니다. GMBRWF를 지정하는 경우 논리 순서 및 실제 순서 사이에 전환하는 것은 가능하나 GMBRWN을 사용하는 후속적 MQGET 호출이 큐 핸들을 위해 GMBRWF를 지정한 최근 호출과 동일한 순서로 큐를 찾아야 합니다.

큐 관리자가 큐의 메시지를 찾는 MQGET 호출을 위해 유지하는 그룹 및 세그먼트 정보는 큐 관리자가 큐에서 메시지를 제거하는 MQGET 호출을 위해 유지하는 그룹 및 세그먼트 정보와 별도입니다. GMBRWF를 지정하는 경우 큐 관리자는 찾아보기를 위한 그룹과 세그먼트 정보를 무시하고 현재 그룹이 없고 현재 논리 메시지가 없는 것처럼 큐를 검색합니다. MQGET 호출이 완료되면(완료 코드 CCOK 또는 CCWARN) 찾아볼 그룹 및 세그먼트 정보가 리턴된 메시지의 해당 정보로 설정됩니다. 호출에 실패하면 호출 전에 있었던 그룹 및 세그먼트 정보가 유지됩니다.

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

- GMBRWC
- GMBRWN
- GMMUC
- GMSYP
- GMPSYP
- GMUNLK

큐가 찾아보기에 대해 열려지지 않은 경우에도 오류가 발생합니다.

#### **GMBRWN**

큐의 현재 위치로부터의 찾아보기.

찾아보기 커서를 MQGET 호출에 지정된 선택 기준을 충족하는 큐의 다음 메시지로 이동합니다. 메시지는 애플리케이션으로 리턴되지만 큐에 남아 있습니다.

큐가 읽기 전용으로 열린 후 핸들을 사용하는 첫 번째 찾아보기 호출은 GMBRWF 또는 GMBRWN 옵션 지정과 동일한 효과를 가집니다.

GMBRWN이 있는 다음 MQGET 호출이 발행되기 전에 메시지가 큐에서 제거되는 경우 해당 위치가 현재 비어 있더라도 찾아보기 커서는 논리적으로 메시지가 차지되는 큐의 위치에 남아 있습니다.

메시지는 다음 두 가지 방법 중 하나로 큐에 저장됩니다.

- 우선순위(MSPRIO) 내 FIFO, 또는
- 우선순위(MSFIFO)와 관계없는 FIFO

**MsgDeliverySequence** 큐 속성은 적용되는 메소드를 표시합니다(자세한 내용은 [1305 페이지의 『큐의 속성』](#) 참조).

큐에 MSPRIO의 *MsgDeliverySequence*이(가) 있고 찾아보기 커서가 현재 가리키는 큐보다 우선순위가 높은 큐에 메시지가 도달하면, GMBRWN을 사용하여 큐의 현재 스왑 동안 해당 메시지를 찾을 수 없습니다. GMBRWF를 사용하거나 큐를 다시 열어 찾아보기 커서를 재설정 한 후에만 발견될 수 있습니다.

GMMUC 옵션은 필요한 경우 비열람 MQGET 호출과 함께 사용하여 나중에 큐에서 메시지를 제거하는 데 사용할 수 있습니다.

찾아보기 커서는 동일한 HOBJ 핸들을 사용하는 비탐색 MQGET 호출에 의해 이동되지 않습니다.

이 옵션과 함께 GMLK 옵션을 지정하면 검색할 메시지를 잠글 수 있습니다.

GMBRWN은 논리 메시지의 세그먼트 및 그룹의 메시지 처리를 제어하는 GM\* 및 MO\* 옵션과 올바르게 결합하여 지정할 수 있습니다.

GMLOGO가 지정되면 메시지가 논리 순서에서 발견됩니다. 이 옵션을 생략하면 메시지를 실제 순서로 찾아봅니다. GMBRWF를 지정하는 경우 논리 순서 및 실제 순서 사이에 전환하는 것은 가능하나 GMBRWN

을 사용하는 후속적 MQGET 호출이 큐 핸들을 위해 GMBRWF를 지정한 최근 호출과 동일한 순서로 큐를 찾아야 합니다. 이 조건이 충족되지 않으면 이유 코드 RC2259와 함께 호출에 실패합니다.

**참고:** MQGET 호출이 GMLOGO가 지정되지 않을 때 메시지 그룹(또는 그룹에서 논리 메시지)의 끝을 넘어서 찾아보는 데 사용되면 특별한 주의가 필요합니다. 예를 들어, 그룹의 마지막 메시지가 큐에 있는 그룹의 첫 번째 메시지보다 먼저 발생하는 경우, GMBRWN을 사용하여 그룹 끝을 넘어 탐색하고 1(으)로 설정(다음 그룹의 첫 번째 메시지를 찾기 위해)된 MDSEQ을(를) 갖는 MOSEQN은 이미 탐색한 그룹의 첫 번째 메시지를 다시 리턴합니다. 이는 즉시 발생하거나 (중간 그룹이 있는 경우) 나중에 다수의 MQGET 호출이 발생할 수 있습니다.

무한 루프의 가능성은 열람을 위해 큐를 두 번 열어 피할 수 있습니다.

- 각 그룹에서 첫 번째 메시지만 찾아보려면 첫 번째 핸들을 사용하십시오.
- 특정 그룹 내의 메시지만 찾아보려면 두 번째 핸들을 사용하십시오.
- 그룹에서 메시지를 찾기 전에 두 번째 찾아보기 커서를 첫 번째 찾아보기 커서의 위치로 이동하기 위해 MO\* 옵션을 사용하십시오.
- 그룹의 끝을 넘어서 찾아보기 위해 GMBRWN을 사용하지 마십시오.

큐 관리자가 큐의 메시지를 찾는 MQGET 호출을 위해 유지하는 그룹 및 세그먼트 정보는 큐 관리자가 큐에서 메시지를 제거하는 MQGET 호출을 위해 유지하는 그룹 및 세그먼트 정보와 별도입니다.

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

- GMBRWF
- GMBRWC
- GMMUC
- GMSYP
- GMPSYP
- GMUNLK

큐가 찾아보기에 대해 열리지 않은 경우에도 오류가 발생합니다.

## GMBRWC

찾아보기 커서 아래의 메시지를 찾으십시오.

이 옵션을 사용하면 MQGMO의 GMMO 필드에 지정된 MO\* 옵션에 관계없이 찾아보기 커서가 가리키는 메시지가 비파괴적으로 검색됩니다.

찾아보기 커서로 지정된 메시지는 GMBRWF 또는 GMBRWN 옵션을 사용하여 마지막으로 검색된 메시지입니다. 큐가 열려진 후에 이 큐에 대해 이들 호출 중 어느 것도 발행되지 않은 경우 또는 찾아보기 커서 아래에 있던 메시지가 그 후 파괴적으로 검색된 경우 호출에 실패합니다.

찾아보기 커서의 위치는 이 호출로 변경되지 않습니다.

GMMUC 옵션은 필요한 경우 비열람 MQGET 호출과 함께 사용하여 큐에서 메시지를 제거하는 데 사용할 수 있습니다.

찾아보기 커서는 동일한 HOBJ 핸들을 사용하는 비탐색 MQGET 호출에 의해 이동되지 않습니다.

CCFAIL 완료 코드 또는 RC2080 이유 코드를 리턴하는 MQGET 찾아보기 호출을 통해 이동되기도 않습니다.

GMBRWC가 GMLK로 지정되는 경우:

- 이미 잠겨진 메시지가 있는 경우 커서 아래에 있는 메시지이므로 잠금 해제한 후 다시 잠글 필요없이 리턴됩니다. 메시지는 잠긴 상태로 유지됩니다.
- 잠긴 메시지가 없는 경우 찾아보기 커서 아래의 메시지는(하나가 있는 경우) 잠겨지고 애플리케이션으로 리턴됩니다. 찾아보기 커서 아래의 메시지가 없는 경우 호출에 실패합니다.

GMBRWC가 GMLK 없이 지정되는 경우:

- 이미 잠겨진 메시지가 있는 경우 메시지는 커서 아래에 있어야 합니다. 이 메시지는 애플리케이션으로 리턴된 다음 잠금 해제됩니다. 메시지가 이제 잠금 해제된 상태이므로 다시 찾아볼 수 있거나 파괴적

로 검색될 수 있습니다(큐에서 다른 애플리케이션 메시지 가져오기에 의해 파괴적으로 검색할 수 있음).

- 잠긴 메시지가 없는 경우 찾아보기 커서 아래의 메시지는(하나가 있는 경우) 애플리케이션으로 리턴됩니다. 찾아보기 커서 아래의 메시지가 없는 경우 호출에 실패합니다.

GMCMPM이 GMBRWC와 함께 지정된 경우, 찾아보기 커서는 MQMD의 *MDOFF* 필드가 0인 메시지를 식별해야 합니다. 이 조건이 충족되지 않으면 호출은 이유 코드 RC2246으로 실패합니다.

큐 관리자가 큐의 메시지를 찾는 MQGET 호출을 위해 유지하는 그룹 및 세그먼트 정보는 큐 관리자가 큐에서 메시지를 제거하는 MQGET 호출을 위해 유지하는 그룹 및 세그먼트 정보와 별도입니다.

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

- GMBRWF
- GMBRWN
- GMMUC
- GMSYP
- GMPSYP
- GMUNLK

큐가 찾아보기에 대해 열려지지 않은 경우에도 오류가 발생합니다.

### **GMMUC**

찾아보기 커서 아래의 메시지를 가져오십시오.

이 옵션을 사용하면 MQGMO의 *GMMO* 필드에 지정된 MO\* 옵션에 관계없이 찾아보기 커서가 가리키는 메시지가 검색됩니다. 메시지가 큐에서 제거됩니다.

찾아보기 커서로 지정된 메시지는 GMBRWF 또는 GMBRWN 옵션을 사용하여 마지막으로 검색된 메시지만입니다.

GMCMPM이 GMMUC와 함께 지정된 경우, 찾아보기 커서는 MQMD의 *MDOFF* 필드가 0인 메시지를 식별해야 합니다. 이 조건이 충족되지 않으면 호출은 이유 코드 RC2246으로 실패합니다.

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

- GMBRWF
- GMBRWC
- GMBRWN
- GMUNLK

큐가 열림 및 입력 모두에 대해 열려지지 않은 경우에도 오류가 발생합니다. 찾아보기 커서가 현재 검색 가능한 메시지를 지정하고 있지 않은 경우 MQGET 호출이 오류를 리턴합니다.

**잠금 옵션:** 다음은 큐에서 메시지 잠금과 관련된 옵션입니다.

### **GMLK**

잠금 메시지입니다.

이 옵션은 찾아본 메시지를 잠금 설정하여 메시지가 큐에 대해 열려진 다른 핸들에 감춰지게 합니다. 이 옵션은 다음 옵션 중 하나가 지정된 경우에만 지정될 수 있습니다.

- GMBRWF
- GMBRWN
- GMBRWC

하나의 메시지는 큐 핸들당 잠글 수 있지만 이는 논리 메시지 또는 실제 메시지일 수 있습니다.

- GMCMPM가 지정된 경우 논리 메시지를 구성하는 모든 메시지 세그먼트는 큐 핸들에 잠깁니다(큐에 모두 있고 검색 가능한 경우).

- GMCMPM을 지정하지 않은 경우 하나의 실제 메시지만 큐 핸들에 잠겨집니다. 이 메시지가 논리 메시지의 세그먼트가 될 경우 잠겨진 세그먼트는 GMCMPM을 사용하는 다른 애플리케이션이 이 논리 메시지를 검색하거나 찾아보지 못하게 합니다.

잠긴 메시지는 찾아보기 커서 아래 항상 하나가 있고 메시지가 GMMUC 옵션을 지정하는 후속 MQGET 호출에 의해 큐에서 제거될 수 있습니다. 큐 핸들을 사용하는 다른 MQGET 호출도 메시지를 제거할 수 있습니다(예: 잠긴 메시지의 메시지 ID를 지정하는 호출).

호출이 완료 코드 CCFAIL 또는 이유 코드 RC2080의 CCWARN을 리턴하는 경우 메시지가 잠겨지지 않습니다.

애플리케이션이 큐에서 메시지를 제거하지 않을 경우 잠금은 다음으로 해제될 수 있습니다.

- 각 GMBRWF 또는 GMBRWN이(GMLK 포함 또는 미포함) 지정된 채 이 핸들에 다른 MQGET 호출을 발행합니다. 메시지는 호출이 CCOK 또는 CCWARN으로 완료되면 잠금 해제되지만 호출이 CCFAIL로 완료되면 잠겨진 채로 있습니다. 그러나 다음 예외가 적용됩니다.

- CCWARN이 RC2080으로 리턴되면 메시지는 잠금 해제되지 않습니다.

- CCFAIL이 RC2033으로 리턴되면 메시지는 잠금 해제됩니다.

또한 GMLK가 지정되면 리턴된 메시지는 잠깁니다. GMLK가 지정되지 않으면 호출 후 잠긴 메시지가 없습니다.

GMWT가 지정되고 바로 사용 가능한 메시지가 없는 경우 원래 메시지의 잠금 해제는 대기의 시작 전에 발생합니다(그렇지 않은 경우 호출에 오류가 없음을 제공).

- GMBRWC가(GMLK 미포함) 있는 이 핸들에 다른 MQGET 호출을 발행합니다. 메시지는 호출이 CCOK 또는 CCWARN으로 완료되면 잠금 해제되지만 호출이 CCFAIL로 완료되면 잠겨진 채로 있습니다. 그러나 다음 예외가 적용됩니다.

- CCWARN이 RC2080으로 리턴되면 메시지는 잠금 해제되지 않습니다.

- 이 핸들에 GMUNLK를 지정하여 다른 MQGET 호출을 발행합니다.

- 이 핸들에 MQCLOSE 호출을 발행합니다(애플리케이션 종료에 의해 명시적으로 또는 내재적으로).

부수적인 찾아보기 옵션을 지정하는 데 필요한 OOBW 이외에는 이 옵션을 지정하는 데 특별한 열기 옵션이 필요하지 않습니다.

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

- GMSYP
- GMPSYP
- GMUNLK

## GMUNLK

잠금 해제 메시지입니다.

잠금 해제될 메시지는 GMLK 옵션을 사용하는 MQGET 호출에 의해 이미 잠겨져 있어야 합니다. 이 핸들에 대해 잠겨진 메시지가 없는 경우 호출은 CCWARN 및 RC2209로 완료됩니다.

GMUNLK가 지정된 경우 **MSGDSC**, **BUFLN**, **BUFFER** 및 **DATLEN** 매개변수는 검사되지 않거나 변경되지 않습니다. **BUFFER**에 메시지가 리턴되지 않습니다.

(우선 잠금 요청을 실행하기 위해 OOBW가 필요하지만) 이 옵션을 지정하기 위해 특별한 열기 옵션이 필요한 것은 아닙니다.

이 옵션은 다음을 제외한 옵션과 함께 사용할 수 없습니다.

- GMNWT
- GMNSYP

해당 두 옵션이 지정되었는지 여부와 관계없이 가정됩니다.

**메시지 데이터 옵션:** 다음 옵션은 큐에서 메시지를 읽을 때 메시지 데이터의 처리와 관련됩니다.

## GMATM

메시지 데이터의 잘림을 허용하십시오.

메세지 버퍼가 너무 작아서 전체 메세지를 보관할 수없는 경우 이 옵션은 MQGET 호출이 버퍼가 보유할 수 있는 만큼의 메시지로 버퍼를 채우고 경고 완료 코드를 발행하며 처리를 완료하도록 허용합니다. 이는 다음을 의미합니다.

- 메시지를 찾아볼 때 찾아보기 커서가 리턴된 메시지로 이동합니다.
- 메시지를 제거할 때 리턴된 메시지가 큐에서 제거됩니다.
- 다른 오류가 발생하지 않으면 이유 코드 RC2079이(가) 리턴됩니다.

이 옵션을 사용하지 않으면 버퍼는 보유할 수 있는 만큼 많은 메시지로 계속 채워지고 경고 완료 코드가 발행되지만 처리가 완료되지 않습니다. 이는 다음을 의미합니다.

- 메시지를 찾아볼 때 찾아보기 커서가 이동되지 않습니다.
- 메시지를 제거할 때 메시지가 큐에서 제거되지 않습니다.
- 다른 오류가 발생하지 않으면 이유 코드 RC2080이(가) 리턴됩니다.

## GMCONV

변환 메시지 데이터.

이 옵션은 데이터가 **BUFFER** 매개변수로 복사되기 전에 MQGET 호출의 **MSGDSC** 매개변수에 지정된 **MDCSI** 및 **MDENC** 값을 준수하도록 메시지의 애플리케이션 데이터가 변환되도록 요청합니다.

메시지를 넣을 때 지정된 **MDFMT** 필드는 메시지에서 데이터의 특성을 식별하기 위해 변환 프로세스에 의해 지정됩니다. 메시지 데이터는 내장 형식의 경우 큐 관리자의 의해 변환되고 기타 형식의 경우 사용자 작성 엑시트에 의해 변환됩니다.

- 변환이 성공적으로 수행되면 **MSGDSC** 매개변수에 지정된 **MDCSI** 및 **MDENC** 필드는 MQGET 호출의 리턴 시 변경되지 않습니다.
- 변환을 성공적으로 수행할 수 없는 경우(그러나 MQGET 호출은 오류 없이 완료됨), 메시지 데이터는 변환되지 않고 리턴되며 **MSGDSC**의 **MDCSI** 및 **MDENC** 필드는 변환되지 않은 메시지의 값으로 설정됩니다. 이 경우 완료 코드는 CCWARN입니다.

따라서 이러한 필드는 **BUFFER** 매개변수에 리턴되는 메시지 데이터의 문자 세트 ID 및 인코딩을 설명합니다.

큐 관리자가 변환을 수행하는 형식 이름의 목록은 [1063 페이지의 『IBM i의 MQMD\(메시지 디스크립터\)』](#)에 설명된 **MDFMT** 필드를 참조하십시오.

**그룹 및 세그먼트 옵션:** 다음은 논리 메시지의 세그먼트 및 그룹의 메시지 처리에 관련된 옵션입니다. 이러한 정의는 옵션 이해에 도움이 될 수 있습니다.

### 실제 메시지

큐에 배치하거나 큐에서 제거할 수 있는 가장 작은 정보 단위입니다. 대개 단일 MQPUT, MQPUT1 또는 MQGET 호출에서 지정되거나 검색되는 정보에 해당합니다. 모든 실제 메시지에는 자신의 메시지 디스크립터(MQMD)가 있습니다. 일반적으로 실제 메시지는 메시지 ID(MQMD의 **MDMID** 필드)에 대한 값을 서로 다르게 하여 구별되지만, 큐 관리자에 의해 강제 적용되지는 않습니다.

### 논리 메시지

이는 애플리케이션 정보의 한 단위입니다. 시스템 제한 조건이 없는 경우 논리 메시지는 실제 메시지와 동일합니다. 논리 메시지가 큰 경우 시스템 제한조건이 논리 메시지를 두 개 이상의 물리적 메시지(세그먼트라고 함)로 나누도록 권장하거나 요청할 수 있습니다.

하나의 세그먼트화된 논리 메시지는 널이 아닌 그룹 ID(MQMD의 **MDGID** 필드)와 메시지 순서 번호(MQMD의 **MDSEQ** 필드)가 동일한 둘 이상의 실제 메시지로 구성됩니다. 세그먼트는 세그먼트 오프셋(MQMD의 **MDOFF** 필드)에 대한 값을 서로 다르게 하여 구별되며, 이 값은 논리 메시지의 데이터 시작부터의 실제 메시지의 데이터 오프셋을 제공합니다. 각 세그먼트가 물리적 메시지이므로 논리 메시지에 있는 세그먼트는 일반적으로 서로 다른 메시지 ID를 가집니다.

세그먼트화되지 않았지만 전송하는 애플리케이션이 세그먼트화할 수 있는 논리 메시지에도 널이 아닌 그룹 ID가 있습니다. 다만 이 경우 논리 메시지가 메시지 그룹에 속하지 않으면 해당 그룹 ID의 실제 메시지가 하나뿐입니다. 논리 메시지가 메시지 그룹에 속하지 않는 경우 전송하는 애플리케이션이 세그먼트화할 수 없는 논리 메시지에는 널 그룹 ID(GINONE)가 있습니다.

## 메시지 그룹

이는 널이 아닌 동일한 그룹 ID를 갖는 하나 이상의 논리 메시지 세트입니다. 그룹의 논리 메시지는 1과 n 사이의 정수인 메시지 순서 번호에 대한 값을 다르게 하여 구별됩니다. 여기서 n은 그룹의 논리 메시지 수입니다. 하나 이상의 논리 메시지가 세그먼트화되면 그룹에는 n보다 많은 실제 메시지가 있습니다.

## GMLOGO

그룹의 메시지 및 논리 메시지의 세그먼트는 논리적인 순서로 리턴됩니다.

이 옵션은 큐 핸들에 대한 연속적인 MQGET 호출에 의해 메시지가 리턴되는 순서를 제어합니다. 이 옵션은 효과를 가지기 위해 각각의 그 호출에 지정되어야 합니다.

GMLOGO가 큐 핸들에 대한 연속적 MQGET 호출을 위해 지정되면 그룹의 메시지는 메시지 순서 번호에 의해 지정된 순서대로 리턴되고 논리 메시지에서 세그먼트가 해당 세그먼트 오프셋에 의해 지정된 순서대로 리턴됩니다. 이 순서는 해당 메시지 및 세그먼트가 큐에서 발생하는 순서와 차이가 있을 수 있습니다.

**참고:** GMLOGO를 지정하면 그룹에 속하지 않은 메시지 및 세그먼트가 아닌 메시지에 부정적인 영향을 주지 않습니다. 사실상 이러한 메시지는 각기 하나의 메시지로만 구성된 메시지 그룹에 속하는 것처럼 처리됩니다. 따라서 그룹에 있는 메시지, 메시지 세그먼트 및 그룹에 속하지 않은 세그먼트화되지 않은 메시지가 혼합된 큐에서 메시지를 검색할 때 GMLOGO를 지정하는 것이 안전합니다.

필요한 순서로 메시지를 리턴하기 위해 큐 관리자는 연속적인 MQGET 호출 사이에 그룹 및 세그먼트 정보를 보유합니다. 이 정보는 큐 핸들에 대한 현재 메시지 그룹 및 현재 논리 메시지, 그룹 및 논리 메시지 내의 현재 위치 및 메시지가 작업 단위 내에서 검색되는지 여부를 식별합니다. 큐 관리자가 이 정보를 보유하고 있으므로 애플리케이션은 각 MQGET 호출을 발행하기 전에 그룹 및 세그먼트 정보를 설정할 필요가 없습니다. 특히, 이는 애플리케이션이 MQMD에 *MDGID*, *MDSEQ* 및 *MDOFF* 필드를 설정할 필요가 없음을 의미합니다. 그러나 애플리케이션은 각 호출에서 *GMSYP* 또는 *GMNSYP* 옵션을 정확히 설정해야 합니다.

큐가 열려졌을 때는 현재 메시지 그룹 및 현재 논리 메시지가 없습니다. 메시지 그룹은 MQGET 호출에 의해 *MFMIG* 플래그를 가진 메시지가 리턴될 때 현재 메시지 그룹이 됩니다. 연속 호출에서 GMLOGO가 지정된 경우 이 그룹은 다음을 가진 메시지가 리턴될 때까지 현재 그룹으로 남아 있습니다.

- MFSEG가 없는 MFLMIG(즉, 그룹의 마지막 논리 메시지가 세그먼트화되지 않음) 또는
- MFLSEG가 있는 MFLMIG(즉, 리턴된 메시지가 그룹의 마지막 논리 메시지의 마지막 세그먼트임).

이러한 메시지가 리턴되면 메시지 그룹이 종료되고 MQGET 호출이 성공적으로 완료될 때 현재 그룹은 더 이상 존재하지 않습니다. 마찬가지로 논리 메시지는 MQGET 호출에 의해 MFSEG 플래그를 갖는 메시지가 리턴될 때 현재 논리 메시지가 됩니다. 해당 논리 메시지는 MFLSEG 플래그가 있는 메시지가 리턴될 때 종료됩니다.

선택 기준을 지정하지 않은 경우 연속 MQGET 호출은 더 이상 사용할 수 있는 메시지가 없을 때까지 큐의 첫 번째 메시지 그룹의 메시지를(올바른 순서로) 리턴한 후 두 번째 메시지 그룹의 메시지를 리턴합니다. *GMMO* 필드에 다음 옵션 중 하나 이상을 지정하여 리턴되는 특정 메시지 그룹을 선택할 수 있습니다.

- MOMSGI
- MOCORI
- MOGRPI

그러나 이 옵션은 현재 메시지 그룹 또는 논리 메시지가 없는 경우에만 유효합니다. 이 주제에 설명된 *GMMO* 필드를 참조하십시오.

1044 페이지의 표 702은(는) MQGET 호출에서 리턴할 메시지를 찾을 때 큐 관리자가 검색하는 *MDMID*, *MDCID*, *MDGID*, *MDSEQ* 및 *MDOFF* 필드의 값을 표시합니다. 이는 큐에서 메시지를 제거하고 큐에 있는 메시지를 찾아볼 때 모두 적용됩니다. 표의 열에는 다음 의미가 있습니다.

## LOG ORD

GMLOGO 옵션이 호출에 지정되었는지 여부를 표시합니다.

## Cur grp

현재 메시지 그룹이 호출 이전에 존재하는지 여부를 표시합니다.

## Cur log msg

현재 논리 메시지가 호출 이전에 존재하는지 여부를 표시합니다.



## 기타 열

큐 관리자가 찾는 값을 표시합니다. "이전"은 큐 핸들에 대한 이전 메시지에 있는 필드에 대해 리턴된 값을 표시합니다.

표 702. 논리 메시지의 세그먼트 및 그룹의 메시지와 관련된 MQGET 옵션							
지정하는 옵션	호출 이전의 그룹 및 로그 메시지 상태		큐 관리자가 찾는 값				
LOG ORD	Cur grp	Cur log msg	MDMID	MDCID	MDGID	MDSEQ	MDOFF
예	아니오	아니오	제어 주체: GMMO	제어 주체: GMMO	제어 주체: GMMO	1	0
예	아니오	예	모든 메시지 ID	모든 상관 ID	이전 그룹 ID	1	이전 오프셋 + 이전 세그먼트 길이
예	예	아니오	모든 메시지 ID	모든 상관 ID	이전 그룹 ID	이전 순서 번호 + 1	0
예	예	예	모든 메시지 ID	모든 상관 ID	이전 그룹 ID	이전 순서 번호	이전 오프셋 + 이전 세그먼트 길이
아니오	둘 중 하나	둘 중 하나	제어 주체: GMMO	제어 주체: GMMO	제어 주체: GMMO	제어 주체: GMMO	제어 주체: GMMO

다중 메시지 그룹이 큐에 존재하고 리턴할 수 있을 때 그룹은 각 그룹의 첫 번째 논리 메시지의 첫 번째 세그먼트 큐의 위치에 의해 판별된 순서대로 리턴됩니다(즉, 메시지 순서 번호가 1이고 오프셋이 0인 실제 메시지는 적합한 그룹이 리턴되는 순서를 판별합니다).

GMLOGO 옵션은 작업 단위에 다음과 같은 영향을 줍니다.

- 그룹의 첫 번째 논리 메시지 또는 세그먼트가 작업 단위 내에서 검색되는 경우 동일한 큐 핸들이 사용되면 그룹의 다른 모든 논리 메시지 및 세그먼트가 작업 단위 내에서 검색되어야 합니다. 그러나 이를 동일한 작업 단위 내에서 검색할 필요는 없습니다. 이 방법은 여러 실제 메시지로 구성된 메시지 그룹이 큐 핸들에 대해 두 개 이상의 연속 작업 단위에서 나뉘질 수 있게 합니다.
- 그룹의 첫 번째 논리 메시지 또는 세그먼트가 작업 단위 내에서 검색되지 않는 경우, 동일한 큐 핸들이 사용되면 그룹의 다른 논리 메시지 및 세그먼트는 작업 단위 내에서 검색할 수 없습니다.

이러한 조건이 충족되지 않으면 이유 코드 RC2245와 함께 MQGET 호출에 실패합니다.

GMLOGO를 지정하는 경우 MQGET 호출에 제공된 MQGMO는 GMVER2 미만이어야 하고 MQMD는 MDVER2 미만이어야 합니다. 이 조건이 충족되지 않으면 호출은 이유 코드 RC2256 또는 RC2257로 실패합니다.

큐 핸들에 대한 연속 MQGET 호출에 GMLOGO가 지정되지 않은 메시지는 메시지 그룹에 속하는지 또는 논리 메시지의 세그먼트인지 여부와 관계없이 리턴됩니다. 이는 특정 그룹 또는 논리 메시지의 메시지 또는 세그먼트가 순서 없이 리턴되거나 다른 그룹 또는 논리 메시지의 메시지 또는 세그먼트와 섞이거나 세그먼트가 아니거나 그룹에 속하지 않은 메시지와 섞일 수 있다는 것을 의미합니다. 이러한 상황에서 연속적인 MQGET 호출에 의해 리턴되는 특정 메시지는 이러한 호출에 지정된 MO\* 옵션으로 제어됩니다(이러한 옵션의 세부사항은 1032 페이지의 『IBM i의 MQGMO(메시지 가져오기 옵션)』에 설명된 GMMO 필드 참조).

이 방법은 시스템 실패가 발생한 후 중간에 있는 메시지 그룹 또는 논리 메시지를 재시작하는 데 사용될 수 있습니다. 시스템이 다시 시작되면 애플리케이션은 MDGID, MDSEQ, MDOFF 및 GMMO 필드를 적절한 값으로 설정한 다음, GMLOGO를 지정하지 않고 필요에 따라 GMSYP 또는 GMNSYP를 설정하여 MQGET 호출을 발행할 수 있습니다. 이 호출에 성공하면 큐 관리자가 그룹 및 세그먼트 정보를 보유할 수 있으며 해당 큐 핸들을 사용하는 후속 MQGET 호출이 정상적으로 GMLOGO를 지정할 수 있습니다.



큐 관리자가 MQGET 호출에 대해 보유하는 그룹 및 세그먼트 정보는 MQPUT 호출에 대해 보유하는 그룹 및 세그먼트 정보와 분리됩니다. 또한 큐 관리자는 다음에 대해 별도 정보를 보유합니다.

- 큐에서 메시지를 제거하는 MQGET 호출.
- 큐에 있는 메시지를 찾아보는 MQGET 호출.

지정된 큐 핸들의 경우 애플리케이션은 MQGET 호출이 있는 GMLOGO를 지정하는 MQGET 호출과 자유롭게 혼합할 수 있지만 다음 사항을 참고해야 합니다.

- GMLOGO가 지정되지 않으면 MQGET 호출을 성공할 때마다 큐 관리자는 저장된 그룹 및 세그먼트 정보를 리턴된 메시지에 해당하는 값으로 설정하게 됩니다. 이는 큐 관리자가 큐 핸들에 대해 보유하는 기존의 그룹 및 세그먼트 정보를 대체합니다. 호출의 조치에 적절한 정보(찾아보기 또는 제거)만 수정됩니다.
- GMLOGO를 지정하지 않은 경우 현재 메시지 그룹 또는 논리 메시지가 있으면 호출이 실패하지 않습니다. 호출은 CCWARN 완료 코드로 성공할 수 있습니다. 1045 페이지의 표 703에서는 발생할 수 있는 다양한 사례를 보여줍니다. 이 경우 완료 코드가 CCOK가 아니면 이유 코드는 다음 중 하나가 됩니다.
  - RC2241
  - RC2242
  - RC2245

**참고:** 큐 관리자는 큐를 찾아보거나 입력이 아닌 찾아보기용으로 열려졌던 큐를 닫을 때 그룹 및 세그먼트 정보를 검사하지 않습니다. 이 경우 완료 코드는 항상 CCOK입니다(다른 오류가 없다고 간주할 때).

표 703. MQGET 또는 MQCLOSE 호출이 그룹 및 세그먼트 정보와 일치하지 않은 경우 결과		
현재 호출	이전 호출이 GMLOGO를 가진 MQGE였음	이전 호출이 GMLOGO가 없는 MQGE였음
GMLOGO를 가진 MQGET	CCFAIL	CCFAIL
GMLOGO가 없는 MQGET	CCWARN	CCOK
종료되지 않은 그룹 또는 논리 메시지의 MQCLOSE	CCWARN	CCOK

메시지 및 세그먼트를 논리 순서로 검색하려는 애플리케이션은 GMLOGO를 지정하는 것이 좋습니다. 이 옵션은 가장 간단하게 사용할 수 있는 옵션입니다. 이 옵션은 큐 관리자가 이들 정보를 관리하기 때문에 애플리케이션이 그룹 및 세그먼트 정보를 관리할 필요가 없습니다. 그러나 특수화된 애플리케이션은 GMLOGO 옵션이 제공하는 것보다 더 많은 제어를 할 필요가 있으므로 이를 위해 이 옵션을 지정하지 않습니다. 이를 수행하는 경우, MQMD의 MDMID, MDCID, MDGID, MDSEQ 및 MDOFF 필드와 MQGMO의 GMMO에 있는 MO\* 옵션이 각 MQGET 호출 전에 올바르게 설정되어 있는지 확인해야 합니다.

예를 들어, 수신한 실제 메시지가 논리 메시지의 세그먼트 또는 그룹에 있는 메시지인지 여부와 관계없이 이들 메시지를 전달하려는 애플리케이션은 GMLOGO를 지정해서는 안 됩니다. 이는 송신 및 수신 큐 관리자 사이에 다중 경로가 포함된 복잡한 네트워크에서는 실제 메시지가 순서없이 도착할 수 있기 때문입니다. MQPUT 호출 시 GMLOGO 및 해당 PMLOGO를 지정하지 않으면 전달 애플리케이션이 논리적 순서로 다음에 도착하는 메시지를 기다릴 필요가 없이 각 실제 메시지를 도착하자마자 검색하여 전달할 수 있습니다.

GMLOGO는 GM\* 옵션 및 다른 적절한 환경에서 다양한 MO\* 옵션을 사용하여 지정될 수 있습니다.

### GMCMPM

전체 논리 메시지만 다시 검색할 수 있습니다.

이 옵션은 전체 논리 메시지만 MQGET 호출로 리턴될 수 있음을 지정합니다. 논리 메시지가 세그먼트화된 경우 큐 관리자는 이 세그먼트를 리어셈블링하여 완전한 논리 메시지를 애플리케이션으로 리턴합니다. 논리 메시지가 세그먼트화되었다는 사실은 이 메시지를 검색하는 애플리케이션에게 명시되지 않습니다.

**참고:** 이 옵션을 통해서만 큐 관리자가 메시지 세그먼트를 리어셈블링할 수 있습니다. 이 옵션을 지정하지 않으면 큐에 세그먼트가 있는 경우(및 MQGET 호출에 지정된 기타 선택 기준을 충족하는 경우) 이들

세그먼트가 개별적으로 애플리케이션으로 리턴됩니다. 개별 세그먼트를 수신하지 않으려는 애플리케이션은 항상 GMCMPM을 지정해야 합니다.

이 옵션을 사용하려면 애플리케이션은 전체 메시지를 수용할 수 있는 큰 버퍼를 제공하거나 GMATM 옵션을 지정해야 합니다.

큐에 세그먼트화된 메시지가 들어 있고 그 세그먼트 중 일부가 누락된 경우(네트워크에서 지연되었거나 아직 도착하지 않았기 때문에) GMATM을 지정하면 불완전한 논리 메시지에 속한 세그먼트가 검색되지 않습니다. 그러나 이러한 메시지 세그먼트는 여전히 **CurrentQDepth** 큐 속성의 값에 기여합니다. 이는 **CurrentQDepth**이(가) 0보다 큰 경우에도 검색 가능한 논리 메시지가 없을 수 있음을 의미합니다.

지속 메시지의 경우 큐 관리자는 작업 단위 내에서만 세그먼트를 리어셈블링할 수 있습니다.

- MQGET 호출이 사용자 정의 작업 단위 내에서 작동하는 경우 이 작업 단위가 사용됩니다. 리어셈블링 프로세스를 통한 호출이 실패하는 경우 큐 관리자는 리어셈블링 동안 제거된 모든 세그먼트를 큐에 재인스턴스화합니다. 그러나 이 실패와 상관없이 작업 단위는 성공적으로 커밋됩니다.
- 호출이 사용자 정의 작업 단위 외부에서 작동 중이고 사용자 정의 작업 단위가 없는 경우 큐 관리자는 호출이 지속되는 동안 작업 단위를 작성합니다. 호출이 성공하면 큐 관리자가 자동으로 작업 단위를 커밋합니다(애플리케이션이 이를 수행할 필요가 없습니다). 호출이 실패하면 큐 관리자가 작업 단위를 백아웃합니다.
- 호출이 사용자 정의 작업 단위 밖에서 작동하나 사용자 정의 작업 단위가 있는 경우에는 큐 관리자가 리어셈블링을 수행할 수 없습니다. 메시지를 리어셈블링할 필요가 없는 경우 호출은 계속 성공할 수 있습니다. 그러나 메시지를 리어셈블링해야 하는 경우 호출은 이유 코드 RC2255로 실패합니다.

비지속 메시지의 경우 리어셈블링을 수행하기 위해 큐 관리자가 작업 단위를 사용할 필요가 없습니다.

세그먼트인 각 실제 메시지에는 자체 메시지 디스크립터가 있습니다. 단일 논리적 메시지를 구성하는 세그먼트의 경우, 메시지 디스크립터에 있는 대부분의 필드는 논리 메시지의 모든 세그먼트에 대해 동일합니다. 일반적으로 논리 메시지의 세그먼트 간에는 **MDMID**, **MDOFF** 및 **MDMFL** 필드만 다릅니다. 그러나 세그먼트가 중간 큐 관리자에 있는 데드-레터 큐에 위치한 경우 DLQ 핸들러는 GMCONV 옵션을 지정하여 메시지를 검색하고 이는 세그먼트의 문자 세트 또는 인코딩이 변경되는 결과를 만들 수 있습니다. DLQ 핸들러가 세그먼트를 성공적으로 송신한 경우 세그먼트는 목적지 큐 관리자에 도착할 때 논리 메시지에 있는 다른 세그먼트와 다른 문자 세트 또는 인코딩을 가질 수 있습니다.

**MDCSI**, **MDENC** 또는 두 필드 모두가 서로 다른 세그먼트로 구성된 논리 메시지는 큐 관리자가 단일 논리 메시지로 리어셈블링할 수 없습니다. 대신 큐 관리자는 동일한 문자 세트 ID 및 인코딩을 갖는 논리 메시지의 시작 부분에 있는 처음 몇 개의 연속 세그먼트를 리어셈블링하여 리턴하고 MQGET 호출은 완료 코드 CCWARN 및 해당되는 이유 코드 RC2243 또는 RC2244와 함께 완료됩니다. 이는 GMCONV가 지정되었는지 여부와 관계없이 수행됩니다. 나머지 세그먼트를 검색하려면 애플리케이션이 GMCONV 옵션을 지정하지 않고 MQGET 호출을 다시 발행하여 세그먼트를 하나씩 검색해야 합니다. GMLOGO를 사용하여 나머지 세그먼트를 순서대로 검색할 수 있습니다.

또한 메시지 디스크립터의 다른 값을 세그먼트 간에 다른 값으로 설정하기 위해 애플리케이션에 세그먼트를 넣을 수 있습니다. 그러나 수신 애플리케이션이 GMCMPM을 사용하여 논리 메시지를 검색하는 경우에는 이 방법이 유리하지 않습니다. 큐 관리자는 논리 메시지를 리어셈블링할 때 메시지 디스크립터에서 첫 번째 세그먼트에 대한 메시지 디스크립터의 값을 리턴합니다. 유일한 예외는 **MDMFL** 필드이며, 이는 큐 관리자가 리어셈블링된 메시지가 유일한 세그먼트임을 표시하기 위해 설정합니다.

GMCMPM가 보고 메시지에 대해 지정된 경우 큐 관리자는 특별한 처리를 수행합니다. 큐 관리자는 큐를 검사하여 논리 메시지에 있는 서로 다른 세그먼트와 관련된 해당 보고 유형의 모든 보고 메시지가 큐에 있는지 확인합니다. 이들 메시지가 큐에 있으면 GMCMPM을 지정하여 이들 단일 메시지로 검색할 수 있습니다. 이렇게 하려면 보고 메시지가 세그먼트화를 지원하는 큐 관리자 또는 MCA에 의해 생성되어야 하거나 시작하는 애플리케이션이 최소 100바이트의 메시지 데이터를 요청해야 합니다(즉, 적절한 RO\*D 또는 RO\*F 옵션이 지정되어야 합니다). 세그먼트에 대해 전체 애플리케이션 데이터보다 적은 양이 존재하는 경우 누락된 바이트는 리턴된 보고 메시지에서 널로 대체됩니다.

GMCMPM이 GMMUC 또는 GMBRWC와 함께 지정된 경우, 찾아보기 커서는 MQMD의 **MDOFF** 필드 값이 0인 메시지에 위치해야 합니다. 이 조건이 충족되지 않으면 호출은 이유 코드 RC2246으로 실패합니다.

GMCMPM은 GMASGA를 지정할 필요가 없음을 의미합니다.

GMCMPM은 GMPSYP 이외의 GM\* 옵션 및 MOOFFS 이외의 MO\* 옵션을 사용하여 지정될 수 있습니다.

## GMAMSA

그룹에서 모든 메시지는 사용 가능해야 합니다.

이 옵션은 그룹에 있는 모든 메시지가 사용 가능할 때만 검색이 가능합니다. 큐에 메시지 그룹이 포함되어 있고 메시지 중 일부가 누락된 경우(네트워크에서 지연되었거나 아직 도착하지 않았기 때문에) GMAMSA를 지정하면 불완전한 그룹에 속한 메시지가 검색되지 않습니다. 그러나 이러한 메시지는 여전히 **CurrentQDepth** 큐 속성의 값에 기여합니다. 이는 **CurrentQDepth**이(가) 0보다 큰 경우에도 검색 가능한 메시지 그룹이 없을 수 있음을 의미합니다. 검색 가능한 다른 메시지가 없는 경우 지정된 대기 간격(있는 경우)이 만료된 후 이유 코드 RC2033이 리턴됩니다.

GMLOGO가 또한 지정되었는지 여부에 따라 GMAMSA의 처리가 달라집니다.

- 두 옵션 모두 지정된 경우 GMAMSA는 현재 그룹 또는 논리 메시지가 없는 경우에만 유효합니다. 현재 그룹 또는 논리 메시지가 있는 경우, GMAMSA는 무시됩니다. 이는 GMAMSA가 메시지를 논리 순서로 처리할 때 유지될 수 있음을 의미합니다.
- GMAMSA가 GMLOGO 없이 지정되는 경우 GMAMSA는 항상 효과를 가집니다. 이는 그룹의 나머지 메시지를 제거할 수 있도록 하려면 그룹의 첫 번째 메시지를 큐에서 제거한 후에 이 옵션을 꺼야 하는 것을 의미합니다.

GMAMSA를 지정하는 MQGET 호출을 성공적으로 완료했다는 것은 MQGET 호출이 발행되었을 때 그룹의 모든 메시지가 큐에 있었다는 것을 의미합니다. 그러나 다른 애플리케이션은 여전히 그룹에서 메시지를 제거할 수 있다는 점을 유념하십시오(그룹은 해당 그룹의 첫 번째 메시지를 검색하는 애플리케이션에 대해 잠겨지지 않습니다).

이 옵션이 지정되지 않은 경우 그룹이 불완전한 때에도 그룹에 속한 메시지가 검색될 수 있습니다.

GMAMSA는 GMASGA를 지정할 필요가 없음을 의미합니다.

GMAMSA는 GM\* 옵션 및 다른 MO\* 옵션을 사용하여 지정될 수 있습니다.

## GMASGA

논리 메시지의 모든 세그먼트는 사용 가능해야 합니다.

이 옵션은 논리 메시지에 있는 세그먼트가 사용 가능할 때만 논리 메시지의 세그먼트를 검색할 수 있도록 지정합니다. 큐에 세그먼트화된 메시지가 들어 있고 그 세그먼트 중 일부가 누락된 경우(네트워크에서 지연되었거나 아직 도착하지 않았기 때문에) GMASGA를 지정하면 불완전한 논리 메시지에 속한 세그먼트가 검색되지 않습니다. 그러나 이러한 세그먼트는 여전히 **CurrentQDepth** 큐 속성의 값에 기여합니다. 이는 **CurrentQDepth**이(가) 0보다 큰 경우에도 검색 가능한 논리 메시지가 없을 수 있음을 의미합니다. 검색 가능한 다른 메시지가 없는 경우 지정된 대기 간격(있는 경우)이 만료된 후 이유 코드 RC2033이 리턴됩니다.

GMLOGO가 또한 지정되었는지 여부에 따라 GMASGA의 처리가 달라집니다.

- 두 옵션 모두 지정된 경우 GMASGA는 현재 논리 메시지가 없는 경우에만 유효합니다. 현재 논리 메시지가 있는 경우 GMASGA는 무시됩니다. 이는 GMASGA가 메시지를 논리 순서로 처리할 때 유지될 수 있음을 의미합니다.
- GMASGA가 GMLOGO 없이 지정되는 경우 GMASGA는 항상 효과를 가집니다. 이는 논리 메시지의 나머지 세그먼트를 제거할 수 있도록 하려면 논리 메시지의 첫 번째 세그먼트를 큐에서 제거한 후에 이 옵션을 꺼야 하는 것을 의미합니다.

이 옵션이 지정되지 않은 경우 논리 메시지가 불완전한 때에도 메시지 세그먼트가 검색될 수 있습니다.

GMCMPM 및 GMASGA는 둘 다 세그먼트를 검색하려면 먼저 모든 세그먼트가 사용 가능해야 하지만 전자는 전체 메시지를 리턴하는 반면 후자는 세그먼트를 하나씩 검색할 수 있도록 합니다.

GMASGA가 보고 메시지에 대해 지정된 경우 큐 관리자는 특별한 처리를 수행합니다. 큐 관리자는 큐를 검사하여 전체 논리 메시지를 구성하는 각 세그먼트에 대해 최소한 하나의 보고 메시지가 있는지 확인합니다. 있는 경우 GMASGA 조건이 충족됩니다. 그러나 큐 관리자는 존재하는 보고 메시지의 유형을 검사하지 않으므로 논리 메시지의 세그먼트와 관련된 보고 메시지에서 보고 유형이 혼합될 수 있습니다. 따라서 GMASGA의 성공은 GMCMPM이 성공할 것을 의미하지 않습니다. 특정 논리 메시지의 세그먼트에 대해 혼합된 보고 유형이 있는 경우 이러한 보고 메시지는 하나씩 검색되어야 합니다.

GMASGA는 GM\* 옵션 및 다른 MO\* 옵션을 사용하여 지정될 수 있습니다.

**기본 옵션:** 설명된 옵션 중 필요한 옵션이 없는 경우 다음 옵션을 사용할 수 있습니다.

#### **GMNONE**

옵션이 지정되지 않았습니다.

이 값은 기타 옵션이 지정되지 않았음을 표시하는데 사용할 수 있습니다. 모든 옵션은 기본 값을 가집니다. GMNONE는 프로그램 문서화를 보조하기 위해 정의됩니다. 이 옵션은 다른 옵션과 함께 사용하기 위한 용도가 아니며 값이 0이므로 이러한 사용을 감지할 수 없습니다.

GMOPT 필드의 초기값은 GMNWT입니다.

#### **GMRE1(1바이트 문자 문자열)**

예약됨

이 필드는 예약된 필드입니다. 이 필드의 초기값은 공백 문자입니다. GMVER이(가) GMVER2보다 작으면 이 필드는 무시됩니다.

#### **GMRL(10자리의 부호 있는 정수)**

리턴된 메시지 데이터의 길이(바이트).

이는 **BUFFER** 매개변수에서 MQGET 호출에 의해 리턴된 메시지 데이터의 길이(바이트)로 큐 관리자가 설정하는 출력 필드입니다. 큐 관리자가 이 기능을 지원하지 않는 경우, GMRL은(는) RLUNDF 값으로 설정됩니다.

메시지가 인코딩 또는 문자 세트 간에 변환할 때 메시지 데이터는 종종 크기를 변경할 수 있습니다. MQGET 호출에서 리턴하는 경우:

- GMRL이(가) RLUNDF가 아닌 경우, 리턴된 메시지 데이터의 바이트 수는 GMRL(으)로 제공됩니다.
- GMRL에 값 RLUNDF가 있는 경우 일반적으로 **BUFLen** 및 **DATLEN**의 더 작은 수로 지정되지만 MQGET 호출이 이유 코드 RC2079로 완료되는 경우 리턴된 메시지 데이터의 바이트 수는 이 수의 미만일 수 있습니다. 이 경우, **BUFFER** 매개변수의 중요하지 않은 바이트는 널로 설정됩니다.

다음 특수 값이 정의됩니다.

#### **RLUNDF**

리턴된 데이터의 길이가 정의되지 않습니다.

이 필드의 초기값은 RLUNDF입니다. GMVER이(가) GMVER3보다 작으면 이 필드는 무시됩니다.

#### **GMRQN(48바이트 문자 문자열)**

해석된 목적지 큐의 이름.

로컬 큐 관리자에 정의된 것처럼 이는 메시지가 검색된 큐의 로컬 이름에 큐 관리자에 의해 설정되는 출력 필드입니다. 이는 큐를 여는 데 사용된 이름과 다릅니다.

- 알리어스 큐가 열렸음(이 경우 알리어스가 해석된 로컬 큐의 이름이 리턴됨), 또는
- 모델 큐가 열렸음(이 경우 동적 로컬 큐의 이름이 리턴됨)

이 필드의 길이는 LNQN으로 제공됩니다. 이 필드의 초기값은 48개의 빈 문자입니다.

#### **GMRS2(1바이트 문자 문자열)**

예약됨

이 필드는 예약된 필드입니다. 이 필드의 초기값은 공백 문자입니다. GMVER이(가) GMVER4보다 작으면 이 필드는 무시됩니다.

#### **GMSEG(1바이트 문자 문자열)**

검색된 메시지에 추가 세그먼트화가 허용되는지 표시하는 플래그.

값은 다음 중 하나입니다.

#### **SEGIHB**

허용되지 않은 세그먼트화.

#### **SEGALW**

세그먼트화가 허용됩니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 SEGIHB입니다. *GMVER*이(가) *GMVER2*보다 작으면 이 필드는 무시됩니다.

#### **GMSG1(10자리의 부호 있는 정수)**

신호.

이 필드는 예약 필드이며 값은 중요하지 않습니다. 이 필드의 초기값은 0입니다.

#### **GMSG2(10자리의 부호 있는 정수)**

신호 ID.

이 필드는 예약 필드이며 값은 중요하지 않습니다.

#### **GMSID(4바이트 문자 문자열)**

구조 ID입니다.

값은 다음과 같아야 합니다.

##### **GMSIDV**

get-message 옵션 구조의 ID입니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 GMSIDV입니다.

#### **GMSST(1바이트 문자 문자열)**

검색된 메시지가 논리 메시지의 세그먼트인지 표시하는 플래그.

값은 다음 중 하나입니다.

##### **SSNSEG**

메시지가 세그먼트가 아닙니다.

##### **SSSEG**

메시지가 세그먼트이지만 논리 메시지의 마지막 세그먼트가 아닙니다.

##### **SSLSEG**

메시지는 논리적 메시지의 마지막 세그먼트입니다.

또한 이는 논리적 메시지가 하나의 세그먼트로만 구성된 경우에 리턴되는 값입니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 SSNSEG입니다. *GMVER*이(가) *GMVER2*보다 작으면 이 필드는 무시됩니다.

#### **GMTOK(16바이트 비트 문자열)**

메시지 토큰.

이 필드는 예약 필드이며 값은 중요하지 않습니다. 다음 특수 값이 정의됩니다.

##### **MTKNON**

메시지 토큰이 없음.

이 값은 필드 길이만큼의 2진 0입니다.

이 필드의 길이는 LNMTOK로 제공됩니다. 이 필드의 초기값은 MTKNON입니다. *GMVER*이(가) *GMVER3*보다 작으면 이 필드는 무시됩니다.

#### **GMVER(10자리의 부호 있는 정수)**

구조 버전 번호입니다.

값은 다음 중 하나여야 합니다.

##### **GMVER1**

버전-1 메시지 가져오기 옵션 구조입니다.

##### **GMVER2**

버전-2 메시지 가져오기 옵션 구조입니다.

##### **GMVER3**

버전-3 메시지 가져오기 옵션 구조입니다.

**GMVER4**

버전-4 메시지 가져오기 옵션 구조입니다.

구조의 최신 버전에만 있는 필드는 필드 설명에 그렇게 식별되어 있습니다. 다음 상수는 현재 버전의 버전 번호를 지정합니다.

**GMVERC**

현재 버전의 가져오기 메시지 옵션 구조.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 GMVER1입니다.

**GMVER(10자리의 부호 있는 정수)**

구조 버전 번호입니다.

값은 다음 중 하나여야 합니다.

**GMVER1**

버전-1 메시지 가져오기 옵션 구조입니다.

**GMVER2**

버전-2 메시지 가져오기 옵션 구조입니다.

**GMVER3**

버전-3 메시지 가져오기 옵션 구조입니다.

**GMVER4**

버전-4 메시지 가져오기 옵션 구조입니다.

구조의 최신 버전에만 있는 필드는 필드 설명에 그렇게 식별되어 있습니다. 다음 상수는 현재 버전의 버전 번호를 지정합니다.

**GMVERC**

현재 버전의 가져오기 메시지 옵션 구조.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 GMVER1입니다.

**GMWI(10자리의 부호 있는 정수)**

간격 대기.

MQGET 호출이 도달하기에 적합한 메시지(즉, MQGET 호출의 **MSGDSC** 매개변수에 지정된 선택 기준을 충족하는 메시지)를 대기하는 대략적인 시간(밀리초)입니다. 자세한 내용은 1063 페이지의 『IBM i의 MQMD(메시지 디스크립터)』에 설명된 **MDMID** 필드를 참조하십시오. 이 시간이 경과한 후 적절한 메시지가 도달하지 않은 경우 호출은 CCFAIL 및 이유 코드 RC2033으로 완료합니다.

**GMWI**은(는) **GMWT** 옵션과 함께 사용됩니다. 이 옵션이 지정되지 않으면 무시됩니다. 지정된 경우, **GMWI**은(는) 0보다 크거나 같거나 다음 특수 값이어야 합니다.

**WIULIM**

무제한 대기 간격.

이 필드의 초기값은 0입니다.

**초기값**

표 704. MQGMO의 필드		
필드 이름	상수의 이름	상수의 값
<i>GMSID</i>	GMSIDV	'GMO-'
<i>GMVER</i>	GMVER1	1
<i>GMOPT</i>	GMNWT	0
<i>GMWI</i>	없음	0
<i>GMSG1</i>	없음	0

표 704. MQGMO의 필드 (계속)

필드 이름	상수의 이름	상수의 값
GMSG2	없음	0
GMRQN	없음	공백
GMMO	MOMSGI + MOCORI	3
GMGST	GSNIG	' '
GMSST	SSNSEG	' '
GMSEG	SEGIHB	' '
GMRE1	없음	' '
GMTOK	MTKNON	널
GMRL	RLUNDF	-1
GMRS2	없음	' '
GMMH	HMNONE	0

**참고:**

1. - 기호는 단일 공백 문자를 나타냅니다.

**RPG 선언**

```

D*.1.....2.....3.....4.....5.....6.....7..
D*
D* MQGMO Structure
D*
D* Structure identifier
D GMSID          1      4  INZ('GMO ')
D* Structure version number
D GMVER          5      8I 0 INZ(1)
D* Options that control the action ofMQGET
D GMOPT          9      12I 0 INZ(0)
D* Wait interval
D GMWI           13     16I 0 INZ(0)
D* Signal
D GMSG1          17     20I 0 INZ(0)
D* Signal identifier
D GMSG2          21     24I 0 INZ(0)
D* Resolved name of destination queue
D GMRQN          25     72  INZ
D* Options controlling selection criteriaused for MQGET
D GMMO           73     76I 0 INZ(3)
D* Flag indicating whether messageretrieved is in a group
D GMGST          77     77  INZ(' ')
D* Flag indicating whether messageretrieved is a segment of a
D* logicalmessage
D GMSST          78     78  INZ(' ')
D* Flag indicating whether furthersegmentation is allowed for themessage
D* retrieved
D GMSEG          79     79  INZ(' ')
D* Reserved
D GMRE1          80     80  INZ
D* Message token
D GMTOK          81     96  INZ(X'0000000000000000-
D                      0000000000000000')
D* Length of message data returned(bytes)
D GMRL           97     100I 0 INZ(-1)
D* Reserved
D GMRS2          101    104I 0 INZ(0)
D* Message handle
D GMMH           105    112I 0 INZ(0)
    
```

MQIIH 구조는 IBM MQ for z/OS를 통한 IMS 브릿지에 발송된 메시지의 시작 부분에 존재해야 하는 정보를 설명합니다.

## 개요

**형식 이름:** FMIMS입니다.

**문자 세트 및 인코딩:** 특별 조건이 MQIIH 구조 및 애플리케이션 메시지 데이터에 사용된 문자 세트 및 인코딩에 적용됩니다.

- IMS 브릿지 큐를 소유하는 큐 관리자에 연결되는 애플리케이션은 큐 관리자의 문자 세트 및 인코딩에 있는 MQIIH 구조를 제공해야 합니다. 이는 MQIIH 구조의 데이터 변환이 이 경우에 수행되지 않기 때문입니다.
- 다른 큐 관리자에 연결되는 애플리케이션은 지원되는 문자 세트 및 인코딩에 있는 MQIIH 구조를 제공할 수 있습니다. MQIIH의 변환은 IMS 브릿지 큐를 소유하는 큐 관리자에 연결된 수신되는 메시지 채널 에이전트에 의해 수행됩니다.

**참고:** 이에 대해 한 가지 예외가 있습니다. IMS 브릿지 큐를 소유하는 큐 관리자가 분산된 큐잉에 대해 CICS를 사용 중인 경우 MQIIH는 IMS 브릿지 큐를 소유하는 큐 관리자의 문자 세트 및 인코딩에 있어야 합니다.

- MQIIH 구조 뒤에 오는 애플리케이션 메시지 데이터는 MQIIH 구조와 동일한 문자 세트 및 인코딩에 있어야 합니다. MQIIH 구조의 *IICSI* 및 *IIENC* 필드는 애플리케이션 메시지 데이터의 문자 세트 및 인코딩을 지정하는데 사용될 수 없습니다.

데이터 변환 엑시트는 데이터가 큐 관리자에서 지원되는 내장 형식 중 하나가 아닌 경우 애플리케이션 메시지 데이터를 변환하기 위해 사용자에게 의해 제공되어야 합니다.

- [1052 페이지의 『IMS 브릿지 애플리케이션을 위한 인증 패스티켓』](#)
- [1052 페이지의 『필드』](#)
- [1055 페이지의 『초기값』](#)
- [1056 페이지의 『RPG 선언』](#)

## IMS 브릿지 애플리케이션을 위한 인증 패스티켓

이제 IBM MQ 관리자가 IMS 브릿지 애플리케이션을 위한 인증 패스티켓에 대해 사용될 애플리케이션 이름을 지정할 수 있습니다. 이를 수행하려면 애플리케이션 이름은 1 - 8자 영숫자 문자열로 STGCLASS 오브젝트 정의를 위해 새 속성 PTKTAPPL로 지정됩니다.

공백 값은 인증이 IBM MQ의 이전 릴리스 사용으로 발생함을 의미합니다. 즉, 인증 요청에 애플리케이션 이름 플로우가 없고 대신 MVSxxxx 값이 사용됩니다.

1 - 8 영숫자 문자의 값은 RACF 발행에 설명된 대로 패스티켓 애플리케이션 이름 규칙을 따라야 합니다.

IBM MQ 관리자 및 RACF 관리자는 모두 사용될 유효한 애플리케이션 이름에 합의해야 합니다. RACF관리자는 액세스를 부여하는 모든 애플리케이션의 사용자 ID에 읽기 액세스를 부여하여 PTKTDATA 클래스에 프로파일을 작성해야 합니다. IBM MQ 관리자는 패스티켓 인증에 사용될 애플리케이션 이름을 지정하는 필요한 STGCLASS 정의를 작성하거나 변경해야 합니다.

관련 정보는 스크립트 (MQSC) 명령 참조를 참조하십시오.

## 필드

MQIIH 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 [알파벳순](#)으로 설명합니다.

### IIAUT(8바이트 문자열)

RACF 비밀번호 또는 패스티켓.

이는 선택사항입니다. 지정되는 경우 보안 컨텍스트를 제공하기 위해 IMS에 전송한 UTOKEN을 빌드하기 위해 MQMD 보안 컨텍스트에서 사용자 ID와 함께 사용됩니다. 지정되지 않으면, 사용자 ID가 검증 없이 사용됩니다. 이는 RACF 스위치 설정에 따라 달라지며 인증자가 존재하도록 요구할 수 있습니다.

이는 첫 번째 바이트가 공백 또는 널이면 무시됩니다. 다음 특수 값을 사용할 수 있습니다.



**IAUNON**

인증이 없습니다.

이 필드의 길이는 LNAUTH로 제공됩니다. 이 필드의 초기값은 IAUNON입니다.

**IICMT(1바이트 문자열)**

커미트 모드.

IMS 커미트 모드에 대한 자세한 정보는 *OTMA* 참조를 참조하십시오. 값은 다음 중 하나여야 합니다.

**ICMCTS**

커미트 후 전송합니다.

이 모드는 출력의 이중 큐잉이지만 부족한 영역 점유 시간을 의미합니다. 빠른 경로 및 대화 트랜잭션은 이 모드로 실행할 수 없습니다.

**ICMSTC**

전송 후 커미트합니다.

이 필드의 초기값은 ICMCTS입니다.

**IICSI(10자리의 부호 있는 정수)**

예약됨

이 필드는 예약 필드이며 값은 중요하지 않습니다. 이 필드의 초기값은 0입니다.

**IIENC(10자리의 부호 있는 정수)**

예약됨

이 필드는 예약 필드이며 값은 중요하지 않습니다. 이 필드의 초기값은 0입니다.

**IIFLG(10자리의 부호 있는 정수)**

플래그입니다.

값은 다음과 같아야 합니다.

**IINONE**

플래그가 없습니다.

이 필드의 초기값은 IINONE입니다.

**IIFMT(8바이트 문자열)**

MQCIH 뒤에 오는 데이터의 IBM MQ 형식 이름.

MQCIH 구조 뒤에 오는 데이터의 IBM MQ 형식 이름을 지정합니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 이 필드의 코딩 규칙은 MQMD의 *MDFMT* 필드에 대한 규칙과 동일합니다.

이 필드의 길이는 LNFMT로 제공됩니다. 이 필드의 초기값은 FMNONE입니다.

**IILEN(10자리의 부호 있는 정수)**

MQIIH 구조 길이.

값은 다음과 같아야 합니다.

**IILEN1**

IMS 정보 헤더 구조의 길이입니다.

이 필드의 초기값은 IILEN1입니다.

**IILTO(8바이트 문자열)**

논리 터미널 대체.

이는 IO PCB 필드에 위치합니다. 이는 선택사항입니다. 지정되지 않는 경우 TPIPE 이름이 사용됩니다. 이는 첫 번째 바이트가 공백 또는 널이면 무시됩니다.

이 필드의 길이는 LNLTOV로 제공됩니다. 이 필드의 초기값은 8개의 빈 문자입니다.

#### **IIMMN(8바이트 문자열)**

메시지 형식 서비스 맵 이름.

이는 IO PCB 필드에 위치합니다. 이는 선택사항입니다. 입력 시에는 MID를 나타내며 출력 시에는 MOD를 나타냅니다. 이는 첫 번째 바이트가 공백 또는 널이면 무시됩니다.

이 필드의 길이는 LNMFMN으로 제공됩니다. 이 필드의 초기값은 8개의 빈 문자입니다.

#### **IIRFM(8바이트 문자열)**

응답 메시지의 IBM MQ 형식 이름.

현재 메시지에 응답하여 전송되는 응답 메시지의 IBM MQ 형식 이름입니다. 이 필드를 코딩하는 규칙은 MQMD의 *MDFMT* 필드를 코딩하는 규칙과 같습니다.

이 필드의 길이는 LNFMT로 제공됩니다. 이 필드의 초기값은 FMNONE입니다.

#### **IIRSV(1바이트 문자열)**

예약됨

이는 예약된 필드이며 비어 있어야 합니다.

#### **IISEC(1바이트 문자열)**

보안 범위.

이는 필요한 IMS 보안 처리를 표시합니다. 다음 값이 정의됩니다.

##### **ISSCHK**

보안 범위 확인.

ACEE는 종속 영역이 아니라 제어 영역에 빌드됩니다.

##### **ISSFUL**

전체 보안 범위.

캐시된 ACEE는 제어 영역에 빌드되고 캐시되지 않은 ACEE가 종속 영역에 빌드됩니다. ISSFUL을 사용하는 경우 ACEE가 빌드되는 사용자 ID가 종속 영역에 사용된 자원에 액세스하는지 확인하십시오.

ISSCHK 및 ISSFUL이 이 필드에 지정되지 않은 경우 ISSCHK는 가정됩니다.

이 필드의 초기값은 ISSCHK입니다.

#### **IISID(4바이트 문자열)**

구조 ID입니다.

값은 다음과 같아야 합니다.

##### **IISIDV**

IMS 정보 헤더 구조의 ID.

이 필드의 초기값은 IISIDV입니다.

#### **IITID(16바이트 비트 문자열)**

트랜잭션 인스턴스 ID

이 필드는 IMS에서 출력 메시지에 의해 사용되어 첫 번째 입력 시 무시됩니다. *IITST*가 *ITSIC*로 설정되는 경우 IMS가 메시지를 올바른 대화와 연관시킬 수 있도록 다음 입력 및 후속 입력에 이 값을 제공해야 합니다. 다음 특수 값을 사용할 수 있습니다.

##### **ITINON**

트랜잭션 인스턴스 ID가 없습니다.

이 필드의 길이는 LNTIID로 제공됩니다. 이 필드의 초기값은 ITINON입니다.

#### **IITST(1바이트 문자열)**

트랜잭션 상태입니다.

이는 IMS 대화상태를 표시합니다. 대화가 존재하지 않으므로 이는 첫 번째 입력에서 무시됩니다. 후속 입력 시 대화가 활성화인지 여부를 표시합니다. 출력 시 IMS에 의해 설정됩니다. 값은 다음 중 하나여야 합니다.

**ITSIC**

대화 중입니다.

**ITSNIC**

대화 중이 아닙니다.

**ITSARC**

아키텍처 양식으로 트랜잭션 상태 데이터를 리턴하십시오.

이 값은 IMS /DISPLAY TRAN 명령으로만 사용됩니다. 트랜잭션 상태 데이터가 문자 양식 대신 IMS 아키텍처 양식으로 리턴하게 합니다. 자세한 정보는 IBM MQ 를 통해 IMS 트랜잭션 프로그램 작성 의 내용을 참조하십시오.

이 필드의 초기값은 ITSNIC입니다.

**IIVER(10자리의 부호 있는 정수)**

구조 버전 번호입니다.

값은 다음과 같아야 합니다.

**IIVER1**

IMS 정보 헤더 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

**IIVERC**

IMS 정보 헤더 구조의 현재 버전.

이 필드의 초기값은 IIVER1입니다.

**초기값**

표 705. MQIIH의 필드		
필드 이름	상수의 이름	상수의 값
IISID	IISIDV	' IIH '
IIVER	IIVER1	1
IILEN	IILEN1	84
IIENC	없음	0
IICSI	없음	0
IIFMT	FMNONE	공백
IIFLG	IINONE	0
IILTO	없음	공백
IIMMN	없음	공백
IIRFM	FMNONE	공백
IIAUT	IAUNON	공백
IITID	ITINON	널
IITST	ITSNIC	' '
IICMT	ICMCTS	'0'
IISEC	ISSCHK	'C'

표 705. MQIIH의 필드 (계속)		
필드 이름	상수의 이름	상수의 값
IIRSV	없음	' '

**참고:**

1. ' ' 기호는 단일 공백 문자를 나타냅니다.

**RPG 선언**

```

D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQIIH Structure
D*
D* Structure identifier
D IISID 1 4 INZ('IIH ')
D* Structure version number
D IIVER 5 8I 0 INZ(1)
D* Length of MQIIH structure
D IILEN 9 12I 0 INZ(84)
D* Reserved
D IIENC 13 16I 0 INZ(0)
D* Reserved
D IICSI 17 20I 0 INZ(0)
D* MQ format name of data that followsMQIIH
D IIFMT 21 28 INZ(' ')
D* Flags
D IIFLG 29 32I 0 INZ(0)
D* Logical terminal override
D IILTO 33 40 INZ
D* Message format services map name
D IIMMN 41 48 INZ
D* MQ format name of reply message
D IIRFM 49 56 INZ(' ')
D* RACF password or passticket
D IIAUT 57 64 INZ(' ')
D* Transaction instance identifier
D IITID 65 80 INZ(X'0000000000000000-
0000000000000000')
D
D* Transaction state
D IITST 81 81 INZ(' ')
D* Commit mode
D IICMT 82 82 INZ('0')
D* Security scope
D IISEC 83 83 INZ('C')
D* Reserved
D IIRSV 84 84 INZ

```

**IBM i IBM i의 MQIMPO(메시지 특성 조회 옵션)**

MQIMPO 구조를 사용하여 애플리케이션이 메시지 특성이 조회되는 방법을 제어하는 옵션을 지정할 수 있습니다.

**개요**

**목적:** 이 구조는 MQINQMP 호출의 입력 매개변수입니다.

**문자 세트 및 인코딩:** MQIMPO의 데이터는 애플리케이션의 문자 세트 및 애플리케이션의 인코딩(ENNAT)에 있어야 합니다.

- [1057 페이지의 『필드』](#)
- [1062 페이지의 『초기값』](#)
- [1062 페이지의 『RPG 선언』](#)

## 필드

MQIMPO 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

### IPOPT(10자리의 부호 있는 정수)

다음 옵션은 MQINQMP의 조치를 제어합니다. 이러한 옵션을 하나 이상 지정할 수 있습니다. 둘 이상의 옵션을 지정하려면 값을 함께 추가하거나(동일한 상수를 두 번 이상 추가하지 않음), 비트 단위 OR 연산을 사용하여 값을 결합하십시오(프로그래밍 언어가 비트 연산을 지원하는 경우). 올바르지 않은 옵션 결합이 언급되어 있습니다. 기타 모든 결합은 올바릅니다.

**값 데이터 옵션:** 다음 옵션은 특성이 메시지에서 검색될 때 값 데이터의 처리와 관련됩니다.

#### IPCVAL

이 옵션은 MQINQMP 호출이 **Value** 영역의 특성 값을 리턴하기 전에 지정된 *IPREQCSI* 및 *IPREQENC* 값을 준수하도록 특성 값을 변환하도록 요청합니다.

- 변환이 성공한 경우 *IPRETCSI* 및 *IPRETENC* 필드는 MQINQMP 호출에서 리턴할 때 *IPREQCSI* 및 *IPREQENC*와 동일하게 설정됩니다.
- 변환에 실패해도 MQINQMP 호출이 오류 없이 완료되면 특성 값은 변환되지 않은 상태로 리턴됩니다. 특성이 문자열인 경우 *IPRETCSI* 및 *IPRETENC* 필드는 변환되지 않는 문자열의 문자 세트와 인코딩으로 설정됩니다.

이 경우 완료 코드는 이유 코드 RC2466과 함께 CCWARN입니다. 특성 커서는 리턴된 특성으로 이동됩니다.

특성 값이 변환 중 확장되고 **Value** 매개변수의 크기를 초과하는 경우 값은 완료 코드 CCFAIL로 변환되지 않고 리턴되며 이유 코드는 RC2469로 설정됩니다.

MQINQMP 호출의 **DataLength** 매개변수는 애플리케이션이 변환된 특성 값을 수용하는 데 필요한 버퍼 크기를 판별할 수 있도록 특성 값이 변환된 길이를 리턴합니다. 특성 커서가 변경되지 않습니다.

이 옵션은 또한 다음을 요청합니다.

- 특성 이름에 와일드카드가 포함된 경우 및
- *IPRETNAMECHRP* 필드는 리턴된 이름의 주소 또는 오프셋으로 초기화됩니다.

그런 다음 리턴된 이름은 *IPREQCSI* 및 *IPREQENC* 값을 준수하기 위해 변환됩니다.

- 변환에 성공한 경우 *IPRETNAMECHRP*의 *VSCCSID* 필드 및 리턴된 이름의 인코딩은 *IPREQCSI* 및 *IPREQENC*의 입력 값으로 설정됩니다.
- 변환에 실패해도 MQINQMP 호출이 오류 또는 경고 없이 완료되면 리턴된 이름은 변환되지 않습니다. 이 경우 완료 코드는 이유 코드 RC2492와 함께 CCWARN입니다.

특성 커서는 리턴된 특성으로 이동됩니다. 값 및 이름 모두 변환되지 않는 경우 RC2466이 리턴됩니다.

리턴된 이름이 변환 중 확장되고 *RequestedName*의 *VSBufsize* 필드 크기를 초과하는 경우 리턴된 문자열은 완료 코드 CCFAIL로 변환되지 않으며 이유 코드는 RC2465로 설정됩니다.

MQCHARV 구조의 *VSLength* 필드는 애플리케이션이 변환된 특성 값을 수용하는 데 필요한 버퍼 크기를 판별할 수 있도록 특성 값이 변환된 길이를 리턴합니다. 특성 커서가 변경되지 않습니다.

#### IPCTYP

이 옵션은 특성 값을 현재 데이터 유형에서 MQINQMP 호출의 **Type** 매개변수에 지정된 데이터 유형으로 변환하도록 요청합니다.

- 변환에 성공한 경우 **Type** 매개변수는 MQINQMP 호출의 리턴 시 변경되지 않습니다.
- 변환에 실패해도 MQINQMP 호출이 오류 없이 완료되면 이유 RC2470으로 호출에 실패합니다. 특성 커서가 변경되지 않습니다.

데이터 유형의 변환으로 변환 중에 값이 확장되고 변환된 값이 **Value** 매개변수의 크기를 초과하는 경우 값은 완료 코드 CCFAIL로 변환되지 않고 리턴되며 이유 코드는 RC2469로 설정됩니다.

MQINQMP 호출의 **DataLength** 매개변수는 애플리케이션이 변환된 특성 값을 수용하는 데 필요한 버퍼 크기를 판별할 수 있도록 특성 값이 변환된 길이를 리턴합니다. 특성 커서가 변경되지 않습니다.

MQINQMP 호출의 **Type** 매개변수 값이 유효하지 않은 경우 이유 RC2473으로 호출에 실패합니다.

요청한 데이터 유형 변환이 지원되지 않는 경우 이유 RC2470으로 호출에 실패합니다. 다음의 데이터 유형 변환이 지원됩니다.

표 706. 지원되는 데이터 유형 변환	
특성 데이터 유형	지원되는 대상 데이터 유형
TYPBOL	TYPSTR, TYPI8, TYPI16, TYPI32, TYPI64
TYPBST	TYPSTR
TYPI8	TYPSTR, TYPI16, TYPI32, TYPI64
TYPI16	TYPSTR, TYPI32, TYPI64
TYPI32	TYPSTR, TYPI64
TYPI64	TYPSTR
TYPF32	TYPSTR, TYPF64
TYPF64	TYPSTR
TYPSTR	TYPBOL, TYPI8, TYPI16, TYPI32, TYPI64, TYPF32, TYPF64
TYPNUL	없음

지원되는 변환에 적용되는 일반 규칙은 다음과 같습니다.

- 변환 중에 데이터가 손실되지 않는 경우 숫자 특성 값을 하나의 데이터 유형에서 다른 데이터 유형으로 변환할 수 없습니다.

예를 들어, 데이터 유형 TYPI32가 있는 특성 값을 데이터 유형 TYPI64가 있는 값으로 변환할 수 있지만 데이터 유형 TYPI16이 있는 값으로 변환할 수 없습니다.

- 데이터 유형의 특성 값은 문자열로 변환될 수 있습니다.
- 문자열이 변환에 맞게 형식화된 경우 문자열 특성 값은 다른 데이터 유형으로 변환될 수 있습니다. 애플리케이션이 올바르게 형식화되지 않은 문자열 특성 값을 변환시키는 경우 IBM MQ는 이유 코드 RC2472를 리턴합니다.
- 애플리케이션이 지원되지 않는 변환을 시도하는 경우 IBM MQ는 이유 코드 RC2470을 리턴합니다.

하나의 데이터 유형에서 다른 데이터 유형으로의 특성 값 변환에 대한 특정 규칙은 다음과 같습니다.

- TYPBOL 특성 값을 문자열로 변환할 때 값 TRUE는 문자열 "TRUE"로 변환되고 값 False는 문자열 "FALSE"로 변환됩니다.
- TYPBOL 특성 값을 숫자 데이터 유형으로 변환할 때 값 TRUE는 문자열 1로 변환되고 값 False는 0으로 변환됩니다.
- 문자열 특성 값을 TYPBOL 값으로 변환할 때 문자열 "TRUE" 또는 "1"은 TRUE로 변환되고 문자열 "FALSE" 또는 "0"은 FALSE로 변환됩니다.

용어 "TRUE" 및 "FALSE"는 대소문자가 구분되지 않습니다.

기타 문자열은 변환될 수 없습니다. IBM MQ는 이유 코드 RC2472를 리턴합니다.

- 문자열 특성 값을 데이터 유형 TYPI8, TYPI16, TYPI32 또는 TYPI64가 있는 값으로 변환할 때 문자열에 다음 형식이 있어야 합니다.

```
[blanks][sign]digits
```

문자열의 컴포넌트의 의미는 다음과 같습니다.

**blanks**

선택적 선두 공백 문자

**sign**

선택적 더하기 부호(+) 또는 빼기 부호(-) 문자.

**digits**

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

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

문자열이 올바르게 형식화되지 않은 경우 IBM MQ는 이유 코드 RC2472를 리턴합니다.

- 문자열 특성 값을 데이터 유형 TYPF32 또는 TYPF64가 있는 값으로 변환할 때 문자열에 다음 형식이 있어야 합니다.

```
[blanks][sign]digits[.digits][e_char[e_sign]e_digits]
```

문자열의 컴포넌트의 의미는 다음과 같습니다.

**blanks**

선택적 선두 공백 문자

**sign**

선택적 더하기 부호(+) 또는 빼기 부호(-) 문자.

**digits**

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

**e\_char**

지수 문자("E" 또는 "e" 중 하나).

**e\_sign**

지수에 대한 선택적 더하기 부호(+) 또는 빼기 부호(-) 문자.

**e\_digits**

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

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

문자열이 올바르게 형식화되지 않은 경우 IBM MQ는 이유 코드 RC2472를 리턴합니다.

- 숫자 특성 값을 문자열로 변환할 때 이 값은 해당 값의 ASCII 문자를 포함하는 문자열이 아니라 10진수로서의 값에 대한 문자열 표현으로 변환됩니다. 예를 들어, 정수 65는 "65" 문자열로 변환되며 "A" 문자열로 변환되지 않습니다.
- 바이트 문자열 특성 값을 문자열로 변환할 때 각 바이트는 바이트를 표시하는 두 개의 16진 문자로 변환됩니다. 예를 들어, 바이트 배열 {0xF1, 0x12, 0x00, 0xFF}는 "F11200FF" 문자열로 변환됩니다.

**IPQLEN**

특성 값의 유형 및 길이를 조회합니다. 길이는 MQINQMP 호출의 **DataLength** 매개변수에서 리턴됩니다. 특성 값은 리턴되지 않습니다.

*ReturnedName* 버퍼가 지정되는 경우 MQCHARV 구조의 *VSLength* 필드는 특성 이름의 길이로 채워집니다. 특성 이름은 리턴되지 않습니다.

**반복 옵션:** 다음 옵션은 와일드 카드 문자가 포함된 이름을 사용한 특성 반복과 관련이 있습니다.

**IPINQF**

지정된 이름과 일치하는 첫 번째 특성을 조회하십시오. 이 호출 이후 커서는 리턴된 특성에서 설정됩니다.

이는 기본값입니다.

필요한 경우 IPINQC 옵션은 다음에 MQINQMP 호출과 함께 사용하여 동일한 특성을 다시 조회할 수 있습니다.

하나의 특성 커서만 있으므로 특성 이름이 MQINQMP 호출에 지정된 경우 특성 이름이 변경하면 커서가 재설정됩니다.

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

IPINQN  
IPINQC

#### **IPINQN**

특성 커서로부터 검색을 계속하면서 지정된 이름과 일치하는 다음 특성을 조회합니다. 커서는 리턴된 특성으로 이동됩니다.

호출이 지정된 이름의 첫 번째 MQINQMP 호출인 경우, 지정된 이름과 일치하는 첫 번째 특성이 리턴됩니다.

필요한 경우 IPINQC 옵션은 다음에 MQINQMP 호출과 함께 사용하여 동일한 특성을 다시 조회할 수 있습니다.

커서 아래에 있는 특성이 삭제되는 경우 MQINQMP는 삭제된 특성 다음에 오는 다음 일치 특성을 리턴합니다.

반복이 진행 중인 동안 와일드 카드와 일치하는 특성이 추가되는 경우 특성은 반복이 완료되는 동안 리턴되거나 리턴되지 않을 수 있습니다. 반복이 IPINQF를 사용하여 재시작되면 특성이 리턴됩니다.

반복이 진행 중인 동안 삭제된 와일드 카드와 일치하는 특성은 해당 삭제 후에 리턴되지 않습니다.

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

IPINQF  
IPINQC

#### **IPINQC**

특성 커서가 지시하는 특성의 값을 검색합니다. 특성 커서로 지정된 메시지는 IPINQF 또는 IPINQN 옵션을 사용하여 마지막으로 조회된 메시지입니다.

메시지 핸들이 재사용될 때 메시지 핸들이 MQGET 호출에서 MQGMO의 *MsgHandle* 필드에 지정될 때 또는 메시지 핸들이 MQPUT 호출에서 MQPMO 구조의 *OriginalMsgHandle* 또는 *NewMsgHandle* 필드에 지정될 때 특성 커서가 재설정됩니다.

이 옵션은 특성 커서가 아직 설정되지 않았을 때 사용되거나 특성 커서가 가리키는 특성이 삭제된 경우 완료 코드 CCFAIL 및 이유 RC2471로 실패합니다.

이 옵션은 다음 옵션과 함께 사용할 수 없습니다.

IPINQF  
IPINQN

이전에 설명된 옵션 중 필요한 사항이 없는 경우 다음 옵션을 사용할 수 있습니다.

#### **IPNONE**

기타 옵션이 지정되지 않았음을 표시하려면 이 값을 사용하십시오. 모든 옵션은 기본 값을 가정합니다.

IPNONE는 프로그램 문서화를 지원합니다. 이 옵션은 다른 옵션과 함께 사용하기 위한 용도는 아니지만 값이 0이므로 이러한 사용을 감지할 수 없습니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 IPINQF입니다.

#### **IPREQCSI(10자리의 부호 있는 정수)**

값이 문자열인 경우 조회한 특성 값을 변환할 문자 세트입니다. 이는 또한 *ReturnedName*이 IPCVAL 또는 IPCTYP가 지정될 때 변환할 문자 세트입니다.

이 필드의 초기값은 CSAPL입니다.



### **IPREQENC(10자리의 부호 있는 정수)**

이는 조회한 특성 값이 IPCVAL 또는 IPCTYP가 지정될 때 변환할 인코딩입니다.

이 필드의 초기값은 ENNAT입니다.

### **IPRE1(10자리의 부호 있는 정수)**

이 필드는 예약된 필드입니다. 이 필드의 초기값은 공백 문자입니다.

### **IPRETCSI(10자리의 부호 있는 정수)**

출력에서 MQINQMP 호출의 **Type** 매개변수가 TYPSTR인 경우 리턴되는 값의 문자 세트입니다.

IPCVAL 옵션이 지정되고 변환에 성공하는 경우 리턴 시 *ReturnedCCSID* 필드는 전달된 값과 동일한 값입니다.

이 필드의 초기값은 0입니다.

### **IPRETENC(10자리의 부호 있는 정수)**

출력에서 이는 리턴된 값의 인코딩입니다.

IPCVAL 옵션이 지정되고 변환에 성공하는 경우 리턴 시 *ReturnedEncoding* 필드는 전달된 값과 동일한 값입니다.

이 필드의 초기값은 ENNAT입니다.

### **IPRETNAMCHRP(10자리의 부호 있는 정수)**

조회된 특성의 실제 이름.

입력 시 문자열 버퍼는 MQCHARV 구조의 *VSPtr* 또는 *VSOffset* 필드를 사용하여 전달할 수 있습니다. 문자열 버퍼의 길이는 MQCHARV 구조의 *VSBufsize* 필드를 사용하여 지정됩니다.

MQINQMP 호출에서 리턴 시 이름을 완전히 포함할 수 있을 만큼 긴 문자열 버퍼는 조회한 특성의 이름으로 완료됩니다. MQCHARV 구조의 *VSLength* 필드는 특성 이름의 길이로 채워집니다. MQCHARV 구조의 *VSCCSID* 필드는 이름 변환의 실패 여부와 관계없이 리턴된 이름의 문자 세트를 표시하기 위해 채워집니다.

이 필드는 입출력(I/O) 필드입니다. 이 필드의 초기값은 MQCHARV\_DEFAULT입니다.

### **IPSID(10자리의 부호 있는 정수)**

구조 ID입니다. 값은 다음과 같아야 합니다.

#### **IPSIDV**

메시지 특성 조회 옵션 구조의 ID입니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 IPSIDV입니다.

### **IPTYP(10자리의 부호 있는 정수)**

특성의 데이터 유형의 문자열 표현.

특성이 MQRFH2 헤더에 지정되었고 MQRFH2 dt 속성이 인식되지 않는 경우 이 필드는 특성의 데이터 유형을 판별하는 데 사용할 수 있습니다. *TypeString*이 코드화 문자 세트 1208(UTF-8)로 리턴되고 인식되지 않는 실패한 특성의 dt 속성 값의 첫 8바이트인임

이 필드는 항상 출력 필드입니다. 이 필드의 초기값은 C 프로그래밍 언어에서는 널 문자열이며 기타 프로그래밍 언어에서는 8자의 공백입니다.

## IPVER(10자리의 부호 있는 정수)

구조 버전 번호입니다. 값은 다음과 같아야 합니다.

### IPVER1

조회 메시지 특성 옵션 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

### IPVERC

조회 메시지 특성 옵션 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 IPVER1입니다.

## 초기값

표 707. MQIPMO의 필드		
필드 이름	상수의 이름	상수의 값
IPSID	IPSIDV	'IMPO'
IPVER	IPVER1	1
IPOPT	IPINQF	
IPREQENC	ENNAT	
IPREQCSI	CSAPL	
IPRETENC	ENNAT	
IPRETCSI	0	
IPRE1	0	
IPRETAMCHRP		
IPTYP		공백

## RPG 선언

```
D* MQIMPO Structure
D*
D*
D* Structure identifier
D IPSID          1   4  INZ('IMPO')
D*
D* Structure version number
D IPVER          5   8I 0  INZ(1)
D*
D* Options that control the action of
D* MQINQMP
D IPOPT          9  12I 0  INZ(0)
D*
D* Requested encoding of Value
D IPREQENC       13  16I 0  INZ(273)
D*
D* Requested character set identifier
D* of Value
D IPREQCSI       17  20I 0  INZ(-3)
D*
D* Returned encoding of Value
D IPRETENC       21  24I 0  INZ(273)
D*
D* Returned character set identifier of
D* Value
D IPRETCSI       25  28I 0  INZ(0)
D*
D* Reserved
D IPRE1          29  32I 0  INZ(0)
D*
```

```

D* Returned property name
D* Address of variable length string
D IPRETNAMCHRP      33  48* INZ(*NULL)
D* Offset of variable length string
D IPRETNAMCHRO     49  52I 0 INZ(0)
D* Size of buffer
D IPRETNAMVSBS     53  56I 0 INZ(-1)
D* Length of variable length string
D IPRETNAMCHRL     57  60I 0 INZ(0)
D* CCSID of variable length string
D IPRETNAMCHRC     61  64I 0 INZ(-3)
D*
D* Property data type as a string
D IPTYP            65  72  INZ

```

## IBM i IBM i의 MQMD(메시지 디스크립터)

### 개요

**목적:** MQMD 구조에는 송신 및 수신 애플리케이션 사이에서 메시지가 이동할 때 애플리케이션 데이터를 수반하는 제어 정보가 포함됩니다. 이 구조는 MQGET, MQPUT 및 MQPUT1 호출의 입력/출력 매개변수입니다.

**버전:** MQMD의 현재 버전은 MDVER2입니다. 최신 버전의 구조에만 있는 필드는 다음에 오는 설명에서 최신 필드로 식별됩니다.

제공된 COPY 파일에는 환경에서 지원되는 최신 버전의 MQMD가 포함되지만, MDVER 필드의 초기 값이 MDVER1로 설정됩니다. version-1 구조에 없는 필드를 사용하려면 애플리케이션이 MDVER 필드를 필요한 버전의 버전 번호로 설정해야 합니다.

버전-1 구조의 선언은 MQMD1 이름으로 사용 가능합니다.

**문자 세트 및 인코딩:** MQMD의 데이터는 ENNAT로 지정된 로컬 큐 관리자의 인코딩 및 **CodedCharSetId** 큐 관리자 속성으로 지정된 문자 세트에 있어야 합니다. 그러나 애플리케이션이 IBM MQ MQI client로 실행 중인 경우 구조는 클라이언트의 문자 세트 및 인코딩에 있어야 합니다.

송신 및 수신 큐 관리자가 서로 다른 문자 세트 또는 인코딩을 사용하는 경우, MQMD의 데이터가 자동으로 변환됩니다. 애플리케이션이 MQMD를 변환할 필요가 없습니다.

- [1063 페이지의 『MQMD의 다른 버전 사용』](#)
- [1064 페이지의 『메시지 컨텍스트』](#)
- [1064 페이지의 『메시지 만료』](#)
- [1064 페이지의 『필드』](#)
- [1101 페이지의 『초기값』](#)
- [1102 페이지의 『RPG 선언』](#)

### MQMD의 다른 버전 사용

버전-2 MQMD는 일반적으로 버전-1 MQMD 사용 및 MQMDE 구조로 메시지 데이터 접두부 지정과 동등합니다. 그러나, MQMDE 구조의 모든 필드에 해당 기본값이 있는 경우 MQMDE를 생략할 수 있습니다. MQMDE에 더하여 버전-1 MQMD가 이 섹션에서 나중에 설명된 대로 사용됩니다.

- 애플리케이션이 버전-1 MQMD를 제공하는 경우 애플리케이션은 MQPUT 및 MQPUT1 호출에서 선택적으로 메시지 데이터에 MQMDE로 접두부를 붙이고 MQMD의 MDFMT 필드를 FMMDE로 설정하여 MQMDE가 존재하고 있음을 표시할 수 있습니다. 애플리케이션이 MQMDE를 제공하지 않는 경우 큐 관리자는 MQMDE의 필드에 대해 기본값 가정합니다.

**참고:** 버전-1 MQMD가 아니라 버전-2 MQMD에 있는 여러 필드는 MQPUT 및 MQPUT1의 입출력(I/O) 필드입니다. 그러나 큐 관리자는 MQPUT 및 MQPUT1 호출의 출력에서 MQMDE의 동일 필드에는 아무 값도 리턴하지 않습니다. 애플리케이션은 이들 출력 값이 필요한 경우 버전-2 MQMD를 사용해야 합니다.

- MQGET 호출에서 애플리케이션이 버전-1 MQMD를 제공하는 경우 큐 관리자는 리턴된 메시지에 MQMDE로 접두부를 붙입니다. 단, MQMDE의 필드 중 하나 이상이 기본값이 아닌 값을 가진 경우에만 해당됩니다. MQMD의 MDFMT 필드에 FMMDE 값이 있어서 MQMDE가 있음을 나타냅니다.

MQMDE의 필드에 큐 관리자가 사용한 기본값은 1101 페이지의 표 709에 표시된 대로 해당 필드의 초기값과 동일합니다.

메시지가 전송 큐에 있으면 MQMD의 일부 필드가 특정한 값으로 설정됩니다. 세부사항은 1189 페이지의 『IBM의 MQXQH(전송 큐 헤더)』의 내용을 참조하십시오.

## 메시지 컨텍스트

MQMD의 특정 필드에 메시지 컨텍스트가 포함됩니다. 일반적으로, 다음과 같습니다.

- ID 컨텍스트는 원래 메시지를 넣은 애플리케이션과 관련됩니다.
- 원본 컨텍스트는 메시지를 가장 최근에 넣은 애플리케이션과 관련됩니다.
- 사용자 컨텍스트는 원래 메시지를 넣은 애플리케이션과 관련됩니다.

이러한 두 애플리케이션은 동일한 애플리케이션일 수 있지만, 서로 다른 애플리케이션일 수도 있습니다(예를 들어, 메시지가 한 애플리케이션에서 다른 애플리케이션으로 전달되는 경우).

일반적으로 ID 및 원본 컨텍스트가 이전에 설명된 의미를 가져도 MQMD의 컨텍스트 필드의 두 유형 모두의 콘텐츠는 실제로 메시지를 넣을 때 지정되는 PM\* 옵션에 따라 다릅니다. 결과적으로, ID 컨텍스트는 메시지를 원래 넣은 애플리케이션과 반드시 관련되지는 않으며, 원본 컨텍스트는 메시지를 가장 최근에 넣은 애플리케이션과 반드시 관련되지는 않습니다. 이는 애플리케이션 스위트의 디자인에 따라 다릅니다.

메시지 컨텍스트를 대체하지 않는 애플리케이션의 한 클래스가 있습니다(즉, 메시지 채널 에이전트(MCA)). 원격 큐 관리자에서 메시지를 수신하는 MCA는 MQPUT 또는 MQPUT1 호출에서 컨텍스트 옵션 PMSETA를 사용합니다. 이를 통해 수신 MCA는 송신 MCA에서 메시지와 함께 이동한 메시지 컨텍스트를 정확히 보존할 수 있습니다. 그러나, 결과적으로 원본 컨텍스트가 메시지를 가장 최근에 넣은 애플리케이션과 관련되지 않지만(수신 MCA) 대신에 메시지를 넣은 이전의 애플리케이션과 관련됩니다(시작 애플리케이션 자체일 수 있음).

자세한 정보는 메시지 컨텍스트를 참조하십시오.

## 메시지 만료

로드된 큐(열린 큐)에서 만기된 메시지는 만기 이후 적정 시간 이내에 큐에서 자동으로 제거됩니다. 이 릴리스의 IBM MQ의 일부 기타 새 기능으로 로드된 큐가 이전의 제품 버전에서보다 드물게 스캔될 수 있지만, 로드된 큐의 만기된 메시지는 만기된 적정 기간 이내에 항상 제거됩니다.

## 필드

MQMD 구조에는 다음 필드가 포함됩니다. 필드는 알파벳순으로 설명됩니다.


### MDACC(32바이트 비트 문자열)





계정 토큰.

메시지의 ID 컨텍스트의 부분입니다. 메시지 컨텍스트에 대한 자세한 정보는 메시지 컨텍스트 및 컨텍스트 정보 제어를 참조하십시오.

MDACC를 통해 애플리케이션은 메시지의 결과로 수행된 작업을 적절하게 맡길 수 있습니다. 큐 관리자는 이 정보를 비트 문자열로 처리하며 그 콘텐츠를 검사하지 않습니다.

큐 관리자가 이 정보를 생성하면 다음과 같이 설정됩니다.

- 필드의 첫 번째 바이트는 뒤따르는 바이트 단위로 제공된 회계 정보의 길이로 설정됩니다. 이 길이의 범위는 0부터 30까지이며 2진 정수로 첫 번째 바이트에 저장됩니다.
- 두 번째 및 후속 바이트(길이 필드에서 지정된 대로)는 환경에 적절한 회계 정보로 설정됩니다.
  -  z/OS에서 회계 정보는 다음으로 설정됩니다.
    - z/OS 배치의 경우, JES JOB 카드 또는 EXEC 카드의 JES ACCT 명령문에서 나온 회계 정보입니다(심표 구분 기호는 X'FF'로 변경됨). 이 정보는 필요 시 31바이트로 잘립니다.
    - TSO의 경우, 사용자의 회계 번호입니다.
    - CICS의 경우, LU 6.2 작업 단위 ID입니다(UEPUOWDS)(26바이트).

- IMS의 경우, 16자 IMS 복구 토큰과 함께 병합된 8자 PSB 이름입니다.
  -  IBM i에서, 회계 정보는 작업의 회계 코드로 설정됩니다.
  -   AIX and Linux에서 계정 정보는 ASCII 문자에서 숫자 사용자 ID로 설정됩니다.
  -  Windows에서, 회계 정보는 압축 형식으로 Windows NT 보안 ID(SID)로 설정됩니다. SID는 MDUID 필드에 저장된 사용자 ID를 고유하게 식별합니다. SID가 MDACC 필드에 저장되면 6바이트 ID 권한(SID의 세 번째 및 후속 바이트에 있음)이 생략됩니다. 예를 들어, Windows NT SID의 길이가 28바이트이면 SID 정보의 22바이트가 MDACC 필드에 저장됩니다.
- 마지막 바이트는 회계-토큰 유형으로 설정되며, 다음 값 중 하나입니다.

**ATTCIC**

CICS LUOW ID입니다.

**ATTDOS**

PC DOS 기본 회계 토큰입니다.

**ATTWNT**

Windows 보안 ID.

**ATT400**

IBM i 회계 토큰입니다.

**ATTUNX**

AIX and Linux 숫자 ID입니다.

**ATTUSR**

사용자 정의 회계 토큰입니다.

**ATTUNK**

알 수 없는 회계-토큰 유형입니다.

이 회계-토큰 유형은 다음 환경의 명시적 값으로만 설정됩니다.

-  AIX
-  IBM i
-  Windows

및 해당 시스템에 연결된 IBM MQ MQI clients

기타 환경에서는 회계-토큰 유형이 값 ATTUNK로 설정됩니다. 이러한 환경에서 MDPAT 필드를 사용하여 수신된 회계 토큰의 유형을 추론할 수 있습니다.

- 기타 모든 바이트는 2진 0으로 설정됩니다.

MQPUT 및 MQPUT1 호출의 경우, 이는 PMSETI 또는 PMSETA가 **PMO** 매개변수에 지정된 경우 입력/출력 필드입니다. PMSETI와 PMSETA가 모두 지정되지 않은 경우 이 필드는 입력에서 무시되며 출력 전용 필드입니다. 메시지 컨텍스트의 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어](#)를 참조하십시오.

MQPUT 또는 MQPUT1 호출이 성공적으로 완료되면, 이 필드에는 큐에 넣은 경우 메시지와 함께 전송된 MDACC가 포함됩니다. 이는 보유한 경우 메시지와 함께 보관되지만(보유된 발행물에 대한 자세한 내용은 [1122 페이지의 『IBM i의 MQPMO\(메시지 넣기 옵션\)』](#)에서 PMRET의 설명 참조), 구독자에게 송신된 모든 발행물에서 MDACC를 대체할 값을 구독자가 제공하므로 메시지가 구독자에게 발행물로 송신될 때 MDACC로 사용되지 않는 MDACC의 값입니다. 메시지에 컨텍스트가 없는 경우 필드 전체가 2진 0입니다.

MQGET 호출의 출력 필드입니다.

이 필드는 큐 관리자의 문자 세트에 기반한 변환 대상이 아닙니다. 이 필드는 문자의 문자열이 아니라 비트의 문자열로 처리됩니다.

큐 관리자는 이 필드의 정보로 수행하는 작업이 없습니다. 애플리케이션은 회계 용도로 정보를 사용하려는 경우 정보를 해석해야 합니다.

다음 특수 값을 MDACC 필드에 사용할 수도 있습니다.

## ACNONE

계정 토큰이 지정되지 않습니다.

이 값은 필드 길이 만큼의 2진 0입니다.

이 필드의 길이는 LNAACCT로 제공됩니다. 이 필드의 초기 값은 ACNONE입니다.

## MDAID(32바이트 문자 문자열)

ID와 관련된 애플리케이션 데이터.

메시지의 ID 컨텍스트의 부분입니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어](#)를 참조하십시오.

MDAID는 애플리케이션 스위트에서 정의된 정보이며, 메시지 또는 해당 지원지에 대한 추가 정보를 제공하는 데 사용될 수 있습니다. 큐 관리자는 이 정보를 문자 데이터로 처리하지만 그 형식은 정의하지 않습니다. 큐 관리자가 이 정보를 생성할 때 이는 전부 공백입니다.

MQPUT 및 MQPUT1 호출의 경우, 이는 PMSETI 또는 PMSETA가 **PMO** 매개변수에 지정된 경우 입력/출력 필드입니다. 널 문자가 있는 경우 널 및 다음에 오는 임의의 문자는 큐 관리자에 의해 공백으로 변환됩니다. PMSETI와 PMSETA가 모두 지정되지 않은 경우 이 필드는 입력에서 무시되며 출력 전용 필드입니다. 메시지 컨텍스트의 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어](#)를 참조하십시오.

MQPUT 또는 MQPUT1 호출이 성공적으로 완료되면, 이 필드에는 큐에 넣은 경우 메시지와 함께 전송된 MDAID가 포함됩니다. 이는 보유한 경우 메시지와 함께 보관되지만(보유된 발행물에 대한 자세한 내용은 PMRET의 설명 참조), 구독자에게 송신된 모든 발행물에서 MDAID를 대체할 값을 구독자가 제공하므로 메시지가 구독자에게 발행물로 송신될 때 MDAID로 사용되지 않는 MDAID의 값입니다. 메시지에 컨텍스트가 없는 경우 필드 전체가 공백입니다.

MQGET 호출의 출력 필드입니다. 이 필드의 길이는 LNAIDD로 제공됩니다. 이 필드의 초기값은 32개의 공백 문자입니다.

## MDAOD(4바이트 문자열)

원본과 관련된 애플리케이션 데이터입니다.

이는 메시지의 원본 컨텍스트의 일부입니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어](#)를 참조하십시오.

MDAOD는 메시지의 원본에 대한 추가 정보를 제공하는 데 사용될 수 있는 애플리케이션 스위트에서 정의된 정보입니다. 예를 들어, ID 데이터가 신뢰되는지 여부를 표시하기 위해 적절한 사용자 권한을 갖고 실행하는 애플리케이션이 이를 설정할 수 있습니다.

큐 관리자는 이 정보를 문자 데이터로 처리하지만 그 형식은 정의하지 않습니다. 큐 관리자가 이 정보를 생성할 때 이는 전부 공백입니다.

MQPUT 및 MQPUT1 호출의 경우, PMSETA가 **PMO** 매개변수에서 지정되면 이는 입/출력(I/O) 필드입니다. 필드 내에서 널 문자 이후의 정보는 제거됩니다. 널 문자 및 다음 문자는 큐 관리자에 의해 공백으로 변환됩니다. PMSETA를 지정하지 않으면 입력에서 이 필드가 무시되며 출력 전용 필드가 됩니다.

MQPUT 또는 MQPUT1 호출이 성공적으로 완료되면, 이 필드에는 큐에 넣은 경우 메시지와 함께 전송된 MDAOD가 포함됩니다. 이는 보유한 경우 메시지와 함께 보관되지만(보유된 발행물에 대한 자세한 내용은 PMRET의 설명 참조), 구독자에게 송신된 모든 발행물에서 MDAOD를 대체할 값을 구독자가 제공하므로 메시지가 구독자에게 발행물로 송신될 때 MDAOD로 사용되지 않는 MDAOD의 값입니다. 메시지에 컨텍스트가 없는 경우 필드 전체가 공백입니다.

MQGET 호출의 출력 필드입니다. 이 필드의 길이는 LNAORD에서 제공됩니다. 이 필드의 초기값은 4개의 공백 문자입니다.

## MDBOC(10자리 부호 표시 정수)

백아웃 카운터입니다.

이는 메시지가 작업 단위의 일부로 MQGET 호출에서 이전에 리턴되고 이후 백아웃된 횟수입니다. 이는 메시지 콘텐츠에 기반하는 처리 오류를 감지하는 데 애플리케이션에 보조로 제공됩니다. 수는 임의의 GMBRW\* 옵션을 지정한 MQGET 호출을 제외합니다.

이 수의 정확성은 **HardenGetBackout** 큐 속성의 영향을 받습니다. [1305 페이지의 『큐의 속성』](#)의 내용을 참조하십시오.

MQGET 호출의 출력 필드입니다. 이는 MQPUT 및 MQPUT1 호출에 대해 무시됩니다. 이 필드의 초기값은 0입니다.

### **MDCID(24바이트 비트 문자열)**

상관 ID.

이는 하나의 메시지를 다른 메시지에 관련시키거나 애플리케이션이 수행 중인 다른 작업에 메시지를 관련시키는 데 애플리케이션이 사용할 수 있는 바이트 문자열입니다. 상관 ID는 메시지의 지속적 특성이며 큐 관리자 재시작해도 지속됩니다. 상관 ID가 바이트 문자열이며 문자열이 아니므로, 메시지가 하나의 큐 관리자에서 다른 큐 관리자로 이동할 때 상관 ID는 문자 세트 간에 변환되지 않습니다.

MQPUT 및 MQPUT1 호출의 경우, 애플리케이션은 임의의 값을 지정할 수 있습니다. 큐 관리자가 메시지와 함께 이 값을 전송하며 메시지에 대해 **get** 요청을 실행하는 애플리케이션에 이를 전달합니다.

애플리케이션이 PMNCID를 지정하는 경우, 큐 관리자는 메시지와 함께 송신되고 MQPUT 또는 MQPUT1 호출 시 출력에서 송신 애플리케이션에 리턴되는 고유한 상관 ID를 생성합니다.

이 생성된 상관 ID는 보유된 경우 메시지와 함께 보관되고 MQSUB 호출 시 전달된 MQSD의 SDCID 필드에서 CINONE를 지정하는 구독자에게 메시지가 발행물로 전송되는 경우 상관 ID로 사용됩니다.

보유된 발행물에 대한 자세한 내용은 [1122 페이지의 『IBM i의 MQPMO\(메시지 넣기 옵션\)』](#)의 내용을 참조하십시오.

큐 관리자 또는 메시지 채널 에이전트가 보고 메시지를 생성하는 경우, 기존 메시지의 MDREP 필드에서 지정된 방식으로 MDCID 필드를 설정합니다(ROCMTC 또는 ROPCI). 보고 메시지를 생성하는 애플리케이션도 이를 수행해야 합니다.

MQGET 호출의 경우, MDCID는 큐에서 검색될 특정한 메시지를 선택하는 데 사용될 수 있는 5개의 필드 중 하나입니다. 이 필드의 값을 지정하는 방법에 대한 세부사항은 MDMID 필드의 설명을 참조하십시오.

상관 ID로 CINONE를 지정하면 MOCORI를 지정하지 않는 것과 동일한 영향을 미칩니다. 즉, 임의의 상관 ID가 일치합니다.

GMMUC 옵션이 MQGET 호출 시 **GMO** 매개변수에 지정된 경우 이 필드는 무시됩니다.

MQGET 호출에서 리턴 시, MDCID 필드는 리턴된 메시지의 상관 ID로 설정됩니다(있는 경우).

다음 특수 값이 사용될 수 있습니다.

#### **CINONE**

상관 ID가 지정되어 있지 않습니다.

이 값은 필드 길이만큼의 2진 0입니다.

#### **CINEWS**

메시지는 새 세션의 시작입니다.

이 값은 새 세션의 시작을 나타내는 것으로 CICS bridge에서 인식합니다. 즉, 메시지의 새 시퀀스의 시작입니다.

MQGET 호출의 경우 이는 입출력(I/O) 필드입니다. MQPUT 및 MQPUT1 호출의 경우, 이는 PMNCID가 지정되지 않으면 입력 필드이고, PMNCID가 지정되면 출력 필드입니다. 이 필드의 길이는 LNCID로 제공됩니다. 이 필드의 초기 값은 CINONE입니다.

### **MDCSI(10자리 부호 표시 정수)**

이는 메시지에서 문자 데이터의 문자 세트 ID를 지정합니다.

**참고:** 호출 시 매개변수인 MQMD 및 기타 IBM MQ 데이터 구조의 문자 데이터는 큐 관리자의 문자 세트에 있어야 합니다. 이는 큐 관리자의 **CodedCharSetId** 속성에서 정의됩니다. 이 속성의 세부사항은 [1334 페이지의 『IBM i의 큐 관리자에 대한 속성』](#)의 내용을 참조하십시오.

다음 특수 값을 사용할 수 있습니다.



## CSQM

큐 관리자의 문자 세트 ID입니다.

메시지의 문자 데이터가 큐 관리자의 문자 세트에 있습니다.

MQPUT 및 MQPUT1 호출 시, 큐 관리자는 메시지와 함께 송신된 MQMD의 이 값을 큐 관리자의 실제 문자 세트 ID로 변경합니다. 그 결과, 값 CSQM이 MQGET 호출에서 리턴되지 않습니다.

## CSINHT

이 구조의 문자 세트 ID를 상속합니다.

메시지의 문자 데이터가 이 구조와 동일한 문자 세트에 있습니다. 이는 큐 관리자의 문자 세트입니다. MQMD에 대해서만, CSINHT의 의미가 CSQM과 동일합니다.

큐 관리자는 메시지와 함께 송신된 MQMD의 이 값을 MQMD의 실제 문자 세트 ID로 변경합니다. 오류가 발생하지 않으면 MQGET 호출에서 CSINHT 값을 리턴하지 않습니다.

MQMD의 MDPAT 필드 값이 ATBRKR이면 CSINHT를 사용할 수 없습니다.

## CSEMBD

임베드된 문자 세트 ID입니다.

메시지의 문자 데이터는 메시지 데이터 자체 내에 포함된 ID와 함께 문자 세트에 있습니다. 메시지 데이터 내에 임베드된 임의의 수의 문자 세트 ID가 있어서, 데이터의 여러 파트에 적용될 수 있습니다. 이 값은 문자 세트의 혼합에서 데이터를 포함하는 PCF 메시지에 사용되어야 합니다. PCF 메시지의 형식 이름은 FMPCF입니다.

MQPUT 및 MQPUT1 호출에서만 이 값을 지정하십시오. MQGET 호출에서 지정된 경우, 이는 메시지의 변환을 막습니다.

MQPUT 및 MQPUT1 호출 시, 큐 관리자는 이전에 설명된 대로 메시지와 함께 송신된 MQMD의 값 CSQM 및 CSINHT를 변경하지만, MQPUT 또는 MQPUT1 호출 시 지정된 MQMD는 변경하지 않습니다. 지정된 값에 대해서는 다른 검사가 수행되지 않습니다.

메시지를 검색하는 애플리케이션은 애플리케이션이 예상하는 값과 이 필드를 대조해야 합니다. 값이 다르면 애플리케이션이 메시지의 문자 데이터를 변환해야 할 수 있습니다.

GMCONV 옵션이 MQGET 호출 시 지정된 경우 이 필드는 입력/출력 필드입니다. 애플리케이션에서 지정된 값은 필요 시 메시지 데이터가 변환되어야 하는 코딩된 문자 세트 ID입니다. 변환이 성공적이거나 불필요한 경우 값은 변경되지 않습니다(값 CSQM 또는 CSINHT가 실제 값으로 변환된 경우 제외). 변환이 실패하면 MQGET 호출 이후의 값은 애플리케이션에 리턴되는 변환되지 않은 메시지의 코딩된 문자 세트 ID를 나타냅니다.

그렇지 않으면, 이는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 초기 값은 CSQM입니다.

## MDENC(10자리 부호 표시 정수)

메시지 데이터의 숫자 인코딩입니다.

이는 메시지 데이터의 숫자 데이터의 숫자 인코딩을 지정합니다. 이는 MQMD 구조 자체에서 숫자 데이터에 적용되지 않습니다. 숫자 인코딩은 2진 정수, 팩형 10진수 정수 및 소수점수에 사용된 표시를 정의합니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 큐 관리자는 이 필드가 올바른지 검사하지 않습니다. 다음 특수 값이 정의됩니다.

## ENNAT

고유 시스템 인코딩입니다.

인코딩은 애플리케이션이 실행 중인 프로그래밍 언어 및 시스템의 기본값입니다.

**참고:** 이 상수 값은 프로그래밍 언어 및 환경에 따라 다릅니다. 따라서, 애플리케이션은 실행될 환경에 적절한 헤더, 매크로, COPY 또는 INCLUDE 파일을 통해 컴파일되어야 합니다.

메시지를 넣은 애플리케이션은 일반적으로 ENNAT를 지정해야 합니다. 메시지를 검색하는 애플리케이션은 값 ENNAT와 이 필드를 대조해야 합니다. 값이 다르면 애플리케이션이 메시지의 숫자 데이터를 변환해야 할 수 있습니다. GMCONV 옵션은 MQGET 호출의 처리 일부로 메시지를 변환하도록 큐 관리자에게 요청하는 데 사용할 수 있습니다.



GMCONV 옵션이 MQGET 호출 시 지정된 경우 이 필드는 입력/출력 필드입니다. 애플리케이션에서 지정된 값은 필요 시 메시지 데이터가 변환되어야 하는 인코딩입니다. 변환이 성공적이거나 불필요한 경우 값은 변경되지 않습니다. 변환이 실패하면 MQGET 호출 이후의 값은 애플리케이션에 리턴되는 변환되지 않은 메시지의 인코딩을 나타냅니다.

기타의 경우, 이는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 초기값은 ENNAT입니다.

### MDEXP(10자리 부호 표시 정수)

메시지 수명입니다.

이는 메시지를 넣는 애플리케이션이 설정한, 10분의 1초로 표시되는 일정 시간입니다. 이 기간이 경과하기 전에 목적지 큐에서 메시지를 제거하지 않은 경우, 이 메시지를 버릴 수 있습니다.

값은 감량되어 메시지가 목적지 큐에서 소비하는 시간을 반영하며, 원격 큐에 넣은 경우 임의의 중간 전송 큐에서 소비하는 시간도 반영합니다. 이는 전송 시간이 상당한 경우 이를 반영하도록 메시지 채널 에이전트에서 감량될 수도 있습니다. 마찬가지로 이 메시지를 다른 큐에 전달하는 애플리케이션은 상당한 시간 동안 메시지를 보유한 경우 필요하면 해당 값을 감소시킬 수 있습니다. 그러나, 만기 시간이 근사치로 처리되며 작은 시간 간격을 반영하는 데 값을 감량할 필요는 없습니다.

메시지가 MQGET 호출을 통해 애플리케이션에서 검색되면 MDEXP 필드가 계속 남아 있는 기존 만기 시간의 양을 나타냅니다.

메시지의 만기 시간이 경과된 후에는 큐 관리자가 이를 제거할 수 있습니다. 현재 구현에서, 아직 만기되지 않은 경우 메시지를 리턴했을 browse 또는 nonbrowse MQGET 호출이 발생하면 메시지가 제거됩니다. 예를 들어, MQGMO의 GMMO 필드가 FIFO 정렬 큐에서 읽는 MONONE으로 설정된 nonbrowse MQGET 호출로 인해 모든 만기된 메시지가 첫 번째 만기된 메시지까지 삭제됩니다. 우선순위 정렬 큐에서는 동일한 호출이 높은 우선순위의 만기된 메시지 및 첫 번째 만기되지 않은 메시지 이전에 큐에 도착한 동일한 우선순위의 메시지를 제거합니다.

만기된 메시지는 애플리케이션에 리턴되지 않아서(browse 또는 non-browse MQGET 호출에 의해), 성공적인 MQGET 호출 이후 메시지 디스크립터의 MDEXP 필드의 값이 0보다 크거나 특수 값 EIULIM입니다.

메시지를 원격 큐에 넣으면 목적지 큐에 도달하기 전에 중간 전송 큐에 있는 동안 만기될 수 있습니다(그리고 제거될 수 있음).

메시지가 ROEXP\* 보고서 옵션 중 하나를 지정한 경우 만기된 메시지가 제거되면 보고서가 생성됩니다. 이러한 옵션 중 지정된 옵션이 없으면, 해당 보고서가 생성되지 않습니다. 메시지는 이 기간 이후 더 이상 관련되지 않는 것으로 간주됩니다(이후 메시지가 이를 대체했기 때문일 수 있음).

만기 시간에 기준하여 메시지를 제거하는 기타 프로그램은 요청된 경우 적절한 보고 메시지도 전송해야 합니다.

#### 참고:

1. MDEXP 시간을 0으로 하여 메시지를 넣으면 MQPUT 또는 MQPUT1 호출이 이유 코드 RC2013으로 실패합니다. 이 경우 보고 메시지가 생성되지 않습니다.
2. 만기 시간이 경과된 메시지는 나중까지 실제로 제거되지 않을 수 있으므로, 해당 만기 시간이 지났고 따라서 검색에 적합하지 않은 메시지가 큐에 있을 수 있습니다. 그러나 이러한 메시지는 용량 트리거를 포함하여 큐의 모든 용도의 메시지 수에 가산됩니다.
3. 요청되는 경우 및 메시지가 실제로 제거되는 경우와 제거에 적합하게 되는 경우를 제외하고 만기 보고서가 생성됩니다.
4. 작업 단위 내에서 작동하는 MQGET 호출의 결과로 메시지가 제거를 위해 스케줄된 경우에도 만기된 메시지의 제거 및 필요 시 만기 보고서의 생성은 애플리케이션의 작업 단위 일부가 아닙니다.
5. 거의 만기된 메시지가 작업 단위 내의 MQGET 호출에서 검색되고 작업 단위가 이후 백아웃되는 경우, 메시지가 다시 검색되려면 제거될 수 있게 될 수 있습니다.
6. 거의 만기된 메시지가 GMLK와 함께 MQGET 호출에서 잠긴 경우, 메시지가 GMMUC와 함께 MQGET에서 검색되려면 제거될 수 있게 될 수 있습니다. 그러한 경우, 이유 코드 RC2034가 이 후속 MQGET 호출 시 리턴됩니다.
7. 만기 시간이 0보다 큰 요청 메시지가 검색된 경우 애플리케이션은 응답 메시지를 전송할 때 다음 조치 중 하나를 수행할 수 있습니다.

- 요청 메시지에서 응답 메시지로 남은 만기 시간을 복사합니다.
- 응답 메시지의 만기 시간을 0보다 큰 명확한 값으로 설정합니다.
- 응답 메시지의 만기 시간을 EIULIM으로 설정하십시오.

수행할 조치는 애플리케이션 스위트의 디자인에 따라 다릅니다. 그러나, 데드-레터(미발송 메시지) 큐에 메시지를 넣기 위한 기본 조치는 메시지의 남아 있는 만기 시간을 보존하고 계속해서 이를 감량하는 것이어야 합니다.

8. 트리거 메시지는 항상 EIULIM과 함께 생성됩니다.

9. MDFMT 이름이 FMXQH인 메시지(일반적으로 전송 큐에 있음)는 MQXQH 내에 두 번째 메시지 디스크립터가 있습니다. 따라서 이는 2개의 MDEXP 필드가 연관되어 있습니다. 이 경우 추가적으로 다음과 같은 사항을 참고해야 합니다.

- 애플리케이션이 리모트 큐에 메시지를 넣을 경우 큐 관리자는 이 메시지를 처음에 로컬 전송 큐에 넣고 애플리케이션 메시지 데이터의 접두부에 MQXQH 구조를 표시합니다. 큐 관리자는 2개의 MDEXP 필드의 값을 애플리케이션에서 지정된 것과 동일하게 설정합니다.

애플리케이션이 로컬 전송 큐에 직접 메시지를 넣는 경우, 메시지 데이터가 이미 MQXQH 구조로 시작되어야 하며 형식 이름이 FMXQH여야 합니다(그러나 큐 관리자는 이를 강제 실행하지 않음). 이 경우 애플리케이션은 이러한 두 개의 MDEXP 필드의 값을 동일하게 설정할 필요가 없습니다. 큐 관리자는 MQXQH 내의 MDEXP 필드에 올바른 값이 포함되거나 메시지 데이터가 이를 포함하도록 충분히 긴지도 확인하지 않습니다.

- MDFMT 이름이 FMXQH인 메시지가 큐에서 검색되는 경우(정상 또는 전송 큐인지 여부에 상관없이), 큐 관리자는 큐에서 대기하는 데 소요된 시간으로 이러한 MDEXP 필드를 둘 다 감량합니다. 메시지 데이터가 MQXQH에서 MDEXP 필드를 포함하도록 충분히 길지 않은 경우 오류가 제기되지 않습니다.
- 큐 관리자는 별도의 메시지 디스크립터(즉, MQXQH 구조 내에 임베드된 메시지 디스크립터의 항목 아님)에서 MDEXP 필드를 사용하여 메시지가 삭제될 수 있는지 여부를 테스트합니다.
- 두 MDEXP 필드의 초기값이 서로 다른 경우, MQXQH의 MDEXP 필드에 따른 시간이 경과하는 동안 메시지 검색 시 별도의 메시지 디스크립터의 MDEXP 시간이 0보다 클 수 있습니다(따라서 메시지가 삭제될 수 없음). 이 경우 MQXQH의 MDEXP 필드가 0으로 설정됩니다.

다음 특수 값이 인식됩니다.

#### **EIULIM**

무제한 수명입니다.

메시지에 무제한 만기 시간이 있습니다.

이 필드는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 초기값은 EIULIM입니다.

#### **MDFB(10자리 부호 표시 정수)**

피드백 또는 이유 코드.

이는 유형 MTRPRT의 메시지와 함께 사용되어 보고서의 네이처를 표시하고 해당 유형의 메시지에만 의미가 있습니다. 필드에는 FB\* 값 중 하나 또는 RC\* 값 중 하나가 포함될 수 있습니다. 다음과 같이 피드백 코드를 분류할 수 있습니다.

#### **FBNONE**

피드백이 제공되지 않습니다.

#### **FBSFST**

시스템에서 생성된 피드백에 대한 최저값입니다.

#### **FBSLST**

시스템에서 생성된 피드백에 대한 최고값입니다.

시스템 생성 피드백 코드 FBSFST에서 FBSLST의 범위는 이 섹션(FB\*)에서 나중에 나열된 일반 피드백 코드를 포함하며, 메시지를 목적지 큐에 넣을 수 없을 때 발생할 수 있는 이유 코드(RC\*)도 포함합니다.

#### **FBAFST**

애플리케이션에서 생성된 피드백에 대한 최저값입니다.

**FBALST**

애플리케이션에서 생성된 피드백에 대한 최고값입니다.

보고 메시지를 생성하는 애플리케이션은 큐 관리자 또는 메시지 채널 에이전트에서 생성된 보고 메시지를 시뮬레이션하려는 경우가 아니면 시스템 범위의 피드백 코드를 사용하지 않아야 합니다(FBQUIT 이외의).

MQPUT 또는 MQPUT1 호출 시, 지정된 값은 FBNONE이어야 하거나 시스템 범위 또는 애플리케이션 범위 내에 있어야 합니다. 이는 MDMT의 값에 상관없이 확인됩니다.

**일반 피드백 코드:****FBCOA**

목적지 큐의 도착 확인입니다(ROCOA 참조).

**FBCOD**

수신 애플리케이션에 대한 전달 확인입니다(ROCOD 참조).

**FBEXP**

메시지가 만기되었습니다.

만기 시간이 경과되기 전에 메시지가 목적지 큐에서 제거되지 않았으므로 제거되었습니다.

**FBPAN**

공정적 조치 알림입니다(ROPAN 참조).

**FBNAN**

부정적 조치 알림입니다(ROGAN 참조).

**FBQUIT**

애플리케이션이 종료되어야 합니다.

이 코드는 실행 중인 애플리케이션 프로그램의 인스턴스 수를 제어하기 위한 목적으로 워크로드 스케줄 프로그램에서 사용됩니다. 이 피드백 코드가 포함된 MTRPRT 메시지를 애플리케이션 프로그램의 인스턴스에 송신하는 경우 처리를 중지해야 함을 해당 인스턴스에 나타냅니다. 그러나, 이 규칙을 준수하는 것은 애플리케이션의 문제이며, 큐 관리자가 강제 실행하지 않습니다.

**IMS-브릿지 피드백 코드:** IMS 브릿지가 0이 아닌 IMS-OTMA 센스 코드를 수신하는 경우, IMS 브릿지가 센스 코드를 16진수에서 10진수로 변환하고 값 FBIERR(300)을 추가하며 응답 메시지의 MDFB 필드에 결과를 배치합니다. 그러면 IMS-OTMA 오류 발생 시 피드백 코드가 범위 FBIFST(301)에서 FBILST(399)의 값을 갖습니다.

다음 피드백 코드가 IMS 브릿지에 의해 생성될 수 있습니다.

**FBDLZ**

데이터 길이가 0입니다.

메시지의 애플리케이션 데이터에서 세그먼트 길이가 0입니다.

**FBDLN**

데이터 길이가 음수입니다.

메시지의 애플리케이션 데이터에서 세그먼트 길이가 음수입니다.

**FBDLTB**

데이터 길이가 너무 큼니다.

세그먼트 길이가 메시지의 애플리케이션 데이터에서 너무 컸습니다.

**FBBUFO**

버퍼 오버플로우입니다.

길이 필드 중 하나의 값으로 인해 데이터가 메시지 버퍼 오버플로우를 발생시킵니다.

**FBLOB1**

1로 오류가 발생한 길이입니다.

길이 필드 중 하나의 값이 1바이트 짧습니다.

**FBIIH**

MQIIH 구조가 올바르지 않거나 누락되었습니다.

MQMD의 MDFMT 필드는 FMIMS를 지정하지만 메시지가 올바른 MQIIH 구조로 시작되지 않습니다.

#### **FBNAFI**

IMS에서 사용하도록 권한 부여되지 않은 사용자 ID입니다.

메시지 디스크립터 MQMD에 포함된 사용자 ID 또는 MQIIH 구조의 IIAUT 필드에 포함된 비밀번호는 IMS 브릿지에서 수행된 유효성 검증에 실패했습니다. 그 결과, 메시지가 IMS에 전달되지 않았습니다.

#### **FBIERR**

예상치 못한 오류가 IMS에서 리턴되었습니다.

IMS에서 예상치 못한 오류가 리턴되었습니다. 오류에 대한 자세한 정보는 IMS 브릿지가 상주하는 시스템에서 IBM MQ 오류 로그를 참조하십시오.

#### **FBIFST**

IMS 생성 피드백의 최저값입니다.

IMS 생성 피드백 코드가 범위 FBIFST(300)에서 FBILST(399)를 차지합니다. IMS-OTMA 센스 코드 자체는 MDFB - FBIERR입니다.

#### **FBILST**

IMS 생성 피드백의 최고값입니다.

**CICS-브릿지 피드백 코드:** 다음 피드백 코드가 CICS bridge에서 생성될 수 있습니다.

#### **FBCAAB**

애플리케이션이 이상종료되었습니다.

메시지에 지정된 애플리케이션 프로그램이 이상종료되었습니다. 이 피드백 코드는 MQDLH 구조의 DLREA 필드에서만 발생합니다.

#### **FBCANS**

애플리케이션을 시작할 수 없습니다.

메시지에 지정된 애플리케이션 프로그램에 대한 EXEC CICS LINK가 실패했습니다. 이 피드백 코드는 MQDLH 구조의 DLREA 필드에서만 발생합니다.

#### **FBCBRF**

CICS bridge가 정상 오류 처리를 완료하지 않고 비정상적으로 종료되었습니다.

#### **FBCCSE**

문자 세트 ID가 올바르지 않습니다.

#### **FBCIHE**

CICS 정보 헤더 구조가 누락되었거나 올바르지 않습니다.

#### **FBCCAE**

CICS commarea의 길이가 올바르지 않습니다.

#### **FBCCIE**

상관 ID가 올바르지 않습니다.

#### **FBCDLQ**

데드-레터 큐가 사용 불가능합니다.

CICS bridge 태스크가 이 요청에 대한 응답을 데드-레터 큐에 복사할 수 없습니다. 요청이 백아웃되었습니다.

#### **FBCENE**

인코딩이 올바르지 않습니다.

#### **FBCINE**

CICS bridge에서 예상치 못한 오류가 발생했습니다.

이 피드백 코드는 MQDLH 구조의 DLREA 필드에서만 발생합니다.

#### **FBCNTA**

사용자 ID에 권한이 없거나 비밀번호가 올바르지 않습니다.

이 피드백 코드는 MQDLH 구조의 DLREA 필드에서만 발생합니다.

## **FBCUBO**

작업 단위가 백아웃되었습니다.

작업 단위가 다음 이유 중 하나로 백아웃되었습니다.

- 동일한 작업 단위 내에서 다른 요청을 처리하는 동안 실패가 감지되었습니다.
- 작업 단위가 진행 중인 동안 CICS 이상종료가 발생했습니다.

## **FBCUWE**

작업 단위 제어 필드 CIUOW가 올바르지 않습니다.

**MQ 이유 코드:** 예외 보고 메시지의 경우, MDFB에 MQ 이유 코드가 포함됩니다. 가능한 이유 코드는 다음과 같습니다.

### **RC2051**

(2051, X'803') 큐에 대해 금지된 Put 호출입니다.

### **RC2053**

(2053, X'805') 큐에 이미 최대 수의 메시지가 포함되어 있습니다.

### **RC2035**

(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

### **RC2056**

(2056, X'808') 큐에 사용할 수 있는 공간이 디스크에 없습니다.

### **RC2048**

(2048, X'800') 큐가 지속 메시지를 지원하지 않습니다.

### **RC2031**

(2031, X'7EF') 메시지 길이가 큐 관리자의 최대값보다 큽니다.

### **RC2030**

(2030, X'7EE') 메시지 길이가 큐의 최대 길이보다 깁니다.

이는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 초기값은 FBNONE입니다.

## **MDFMT(8바이트 문자열)**

메시지 데이터의 형식 이름입니다.

이는 메시지에서 데이터의 네이처를 수신자에게 표시하는 데 메시지의 송신자가 사용할 수 있는 이름입니다. 큐 관리자의 문자 세트에 있는 문자가 이름에 지정될 수 있지만, 이름을 다음으로 제한하도록 권장합니다.

- 대문자 A - Z
- 숫자 0 - 9

기타 문자가 사용되는 경우 송신 및 수신 큐 관리자의 문자 세트 간에 이름을 변환할 수 없을 수 있습니다.

이름은 필드의 길이까지 공백으로 채워지거나 널 문자가 필드의 종료 이전에 이름을 종료하는 데 사용됩니다. 널 및 임의의 후속 문자는 공백으로 처리됩니다. 선두 문자 또는 임베드된 공백으로 이름을 지정하지 마십시오. MQGET 호출의 경우, 큐 관리자는 필드의 길이까지 공백으로 채워진 이름을 리턴합니다.

큐 관리자는 이름이 이전에 설명된 권장사항을 준수하는지 확인하지 않습니다.

대문자, 소문자 및 둘 다로 "MQ"를 시작하는 이름은 큐 관리자에서 정의된 의미를 갖습니다. 사용자 고유의 형식에 이러한 문자로 시작하는 이름을 사용하지 않아야 합니다. 큐 관리자 내장 형식은 다음과 같습니다.

### **FMNONE**

형식 이름이 없습니다.

데이터의 네이처가 정의되지 않았습니다. 즉, GMCONV 옵션을 통해 큐에서 메시지가 검색될 때 데이터가 변환될 수 없습니다.

GMCONV가 MQGET 호출에 지정되고 메시지의 데이터 인코딩 또는 문자 세트가 **MSGDSC** 매개변수에 지정된 것과 다른 경우, 메시지가 다음 완료 및 이유 코드로 리턴됩니다(기타 오류가 없다고 가정).

- FMNONE 데이터가 메시지의 시작 부분에 있는 경우 완료 코드 CCWARN 및 이유 코드 RC2110입니다.

- FMNONE 데이터가 메시지의 종료 부분에 있는 경우 완료 코드 CCOK 및 이유 코드 RCNONE입니다 (즉, 하나 이상의 MQ 헤더 구조가 선행함). MQ 헤더 구조는 이 경우 요청된 문자 세트 및 인코딩으로 변환됩니다.

#### **FMADMN**

명령 서버 요청/응답 메시지입니다.

메시지는 프로그램 가능 명령 형식(PCF)의 명령-서버 요청 또는 응답 메시지입니다. GMCONV 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다. 프로그램 가능 명령 형식 메시지 사용에 대한 자세한 정보는 [프로그램 가능 명령 형식 사용](#)을 참조하십시오.

#### **FMCICS**

CICS 정보 헤더입니다.

메시지 데이터는 CICS 정보 헤더 MQCIH로 시작하며, 이는 애플리케이션 데이터가 뒤따릅니다. 애플리케이션 데이터의 형식 이름은 MQCIH 구조의 CIFMT 필드에서 제공됩니다.

#### **FMCMD1**

유형 1 명령 응답 메시지입니다.

메시지는 오브젝트 수, 완료 코드 및 이유 코드가 포함된 MQSC 명령-서버 응답 메시지입니다. GMCONV 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.

#### **FMCMD2**

유형 2 명령 응답 메시지입니다.

메시지는 요청된 오브젝트에 대한 정보가 포함된 MQSC 명령-서버 응답 메시지입니다. GMCONV 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.

#### **FMDLH**

데드-레터 헤더입니다.

메시지 데이터가 데드 레터 헤더 MQDLH로 시작합니다. 원래 메시지의 데이터 바로 뒤에 MQDLH 구조가 나옵니다. 원래 메시지 데이터의 형식 이름은 MQDLH 구조의 DLFMT 필드에서 제공됩니다. 이 구조에 대한 세부사항은 1021 페이지의 『IBM i의 MQDLH (데드-레터 헤더)』의 내용을 참조하십시오. GMCONV 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.

COA 및 COD 보고서는 FMDLH의 MDFMT가 있는 메시지에 생성되지 않습니다.

#### **FMDH**

분배-목록 헤더입니다.

메시지 데이터가 분배 목록 헤더 MQDH로 시작합니다. 여기에는 MQOR 및 MQPMR 레코드의 배열이 포함됩니다. 분배-목록 헤더는 추가 데이터가 뒤따를 수 있습니다. 추가 데이터(있는 경우)의 형식은 MQDH 구조의 DHFMT 필드에서 제공됩니다. 이 구조의 세부사항은 1017 페이지의 『IBM i의 MQDH(분배 헤더)』의 내용을 참조하십시오. GMCONV 옵션이 MQGET 호출에 지정된 경우 FMDH 형식의 메시지를 변환할 수 있습니다.

#### **FMEVNT**

이벤트 메시지입니다.

메시지가 발생한 이벤트를 보고하는 MQ 이벤트 메시지입니다. 이벤트 메시지의 구조는 프로그램 가능 명령과 동일합니다. 이 구조에 대한 자세한 정보는 [명령 및 응답의 구조](#)를 참조하십시오. 이벤트에 대한 정보는 [이벤트 모니터링](#)을 참조하십시오.

GMCONV 옵션이 MQGET 호출에 지정된 경우 버전-1 이벤트 메시지를 변환할 수 있습니다.

#### **FMIMS**

IMS 정보 헤더입니다.

메시지 데이터는 IMS 정보 헤더 MQIIH로 시작하며, 이는 애플리케이션 데이터가 뒤따릅니다. 애플리케이션 데이터의 형식 이름은 MQIIH 구조의 IIFMT 필드에서 제공됩니다. GMCONV 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.

#### **FMIMVS**

IMS 변수 문자열입니다.

메시지는 IMS 변수 문자열이며, 이는 양식 11zzccc의 문자열입니다. 여기서, 다음과 같습니다.

#### **11**

IMS 변수 문자열 항목의 총 길이를 지정하는 2바이트 길이 필드입니다. 이 길이는 11의 길이(2바이트) 더하기 zz의 길이(2바이트) 더하기 문자열 자체의 길이와 같습니다. 11은 MDENC 필드에서 지정된 인코딩의 2바이트 2진 정수입니다.

#### **zz**

IMS에 중요한 플래그가 포함된 2바이트 필드입니다. zz는 2개의 1바이트 비트 문자열 필드로 구성된 바이트 문자열이며 송신자부터 수신자까지 변경사항 없이 전송됩니다(즉, zz는 변환 대상이 아님).

#### **ccc**

11-4자가 포함된 변수-길이 문자열입니다. ccc는 MDCSI 필드에서 지정된 문자 세트에 있습니다. GMCONV 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.

#### **FMMDE**

메시지-디스크립터 확장자입니다.

메시지 데이터는 메시지-디스크립터 확장자 MQMDE로 시작하고 선택적으로 기타 데이터가 뒤따릅니다(일반적으로 애플리케이션 메시지 데이터). MQMDE를 뒤따르는 데이터의 형식 이름, 문자 세트 및 인코딩은 MQMDE의 MEFMT, MECSI 및 MEENC 필드에서 제공됩니다. 이 구조의 세부사항은 1103 페이지의 『IBM i의 MQMDE(메시지 디스크립터 확장자)』의 내용을 참조하십시오. GMCONV 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.

#### **FMPCF**

프로그램 가능 명령 형식(PCF)의 사용자 정의 메시지입니다.

메시지는 프로그램 가능 명령 형식(PCF) 메시지의 구조를 준수하는 사용자 정의 메시지입니다. GMCONV 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다. 프로그래밍 가능한 명령 형식 메시지 사용에 대한 자세한 정보는 [프로그램 가능한 명령 형식 사용](#)을 참조하십시오.

#### **FMRMH**

참조 메시지 헤더입니다.

메시지 데이터는 참조 메시지 헤더 MQRMH로 시작되며 선택적으로 기타 데이터가 뒤따릅니다. 데이터의 형식 이름, 문자 세트 및 인코딩은 MQRMH의 RMFMT, RMCSI 및 RMENC 필드에서 제공됩니다. 이 구조의 세부사항은 1146 페이지의 『IBM i의 MQRMH(참조 메시지 헤더)』의 내용을 참조하십시오. GMCONV 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.

#### **FMRFH**

규칙 및 형식화 헤더입니다.

메시지 데이터는 규칙 및 형식화 헤더 MQRFH로 시작되며 선택적으로 기타 데이터가 뒤따릅니다. 데이터(있는 경우)의 형식 이름, 문자 세트 및 인코딩은 MQRFH의 RFFMT, RFCSI 및 RFENC 필드에서 제공됩니다. GMCONV 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.

#### **FMRFH2**

규칙 및 형식화 헤더 버전 2입니다.

메시지 데이터는 버전-2 규칙 및 형식화 헤더 MQRFH2로 시작되며 선택적으로 기타 데이터가 뒤따릅니다. 선택적 데이터(있는 경우)의 형식 이름, 문자 세트 및 인코딩은 MQRFH2의 RF2FMT, RF2CSI 및 RF2ENC 필드에서 제공됩니다. GMCONV 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.

#### **FMSTR**

완전히 문자로 구성된 메시지입니다.

애플리케이션 메시지 데이터는 SBCS 문자열(1바이트 문자 세트) 또는 DBCS 문자열(2바이트 문자 세트)일 수 있습니다. GMCONV 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.

#### **FMTM**

트리거 메시지입니다.

메시지는 MQTM 구조에서 설명된 트리거 메시지입니다. 이 구조의 세부사항은 1180 페이지의 『MQTM - 트리거 메시지』의 내용을 참조하십시오. GMCONV 옵션이 MQGET 호출에 지정된 경우 이 형식의 메시지를 변환할 수 있습니다.

#### FMWIH

작업 정보 헤더입니다.

메시지 데이터는 작업 정보 헤더 MQWIH로 시작하며, 이는 애플리케이션 데이터가 뒤따릅니다. 애플리케이션 데이터의 형식 이름은 MQWIH 구조의 WIFMT 필드에서 제공됩니다.

#### FMXQH

전송 큐 헤더입니다.

메시지 데이터가 전송 큐 헤더 MQXQH로 시작합니다. 원래 메시지의 데이터 바로 뒤에 MQXQH 구조가 나옵니다. 기존 메시지 데이터의 형식 이름은 전송 큐 헤더 MQXQH의 일부인 MQMD 구조의 MDFMT 필드에서 제공됩니다. 이 구조의 세부사항은 1189 페이지의 『IBM i의 MQXQH(전송 큐 헤더)』의 내용을 참조하십시오.

COA 및 COD 보고서는 FMXQH의 MDFMT가 있는 메시지에 생성되지 않습니다.

이 필드는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 길이는 LNFMT로 제공됩니다. 이 필드의 초기값은 FMNONE입니다.

#### MDGID(24바이트 비트 문자열)

그룹 ID.

이는 실제 메시지가 속한 논리 메시지 또는 특정 메시지 그룹을 식별하는 데 사용되는 바이트 문자열입니다. 세그먼트화가 메시지에 허용된 경우 MDGID도 사용됩니다. 이러한 모든 경우, MDGID에는 널이 아닌 값이 있고 다음 플래그 중 하나 이상이 MDMFL 필드에 설정됩니다.

- MFMIG
- MFLMIG
- MFSEG
- MFLSEG
- MFSEGA

이러한 플래그 중 설정된 항목이 없는 경우 MDGID에 특수 널값 GINONE이 있습니다.

이 필드는 다음과 같은 경우 MQPUT 또는 MQGET 호출 시 애플리케이션에서 설정될 필요가 없습니다.

- MQPUT 호출 시, PMLOGO가 지정됩니다.
- MQGET 호출에서 MOGRPI이 지정되지 않습니다.

보고 메시지가 아닌 메시지에 대해 이러한 호출을 사용하도록 고려하십시오. 그러나, 애플리케이션에 추가 제어가 필요하거나 호출이 MQPUT1인 경우, 애플리케이션은 MDGID가 적절한 값으로 설정되는지 확인해야 합니다.

메시지 그룹 및 세그먼트는 그룹 ID가 고유한 경우에만 올바르게 처리될 수 있습니다. 이러한 이유로, 애플리케이션은 고유의 그룹 ID를 생성하지 않아야 하며, 대신에 다음 중 하나를 수행해야 합니다.

- PMLOGO가 지정되는 경우, 큐 관리자는 논리 메시지의 그룹 또는 세그먼트에서 첫 번째 메시지에 대해 고유한 그룹 ID를 자동으로 생성하고 논리 메시지의 그룹 또는 세그먼트에서 남아 있는 메시지에 대해 해당 그룹 ID를 사용하여, 애플리케이션이 특수 조치를 수행할 필요가 없습니다. 이 프로시저를 사용하도록 고려하십시오.
- PMLOGO가 지정되지 않는 경우, 애플리케이션은 논리 메시지의 그룹 또는 세그먼트에서 메시지에 대해 첫 번째 MQPUT 또는 MQPUT1 호출에서 MDGID를 GINONE으로 설정하여 그룹 ID를 생성하도록 큐 관리자에게 요청해야 합니다. 그런 다음 해당 호출의 출력에 큐 관리자가 리턴한 그룹 ID는 논리 메시지의 그룹 또는 세그먼트에서 남아 있는 메시지에 대해 사용되어야 합니다. 메시지 그룹이 세그먼트화된 메시지를 포함하는 경우 그룹의 모든 세그먼트 및 메시지에 대해 동일한 그룹 ID를 사용해야 합니다.



PMLOGO가 지정되지 않은 경우, 논리 메시지의 그룹 및 세그먼트에 있는 메시지를 임의의 순서대로(예를 들어, 반대 순서대로) 넣을 수 있지만, 그룹 ID가 임의의 해당 메시지에 대해 발행되는 첫 번째 MQPUT 또는 MQPUT1 호출에서 할당되어야 합니다.

MQPUT 및 MQPUT1 호출에 대한 입력에서, 큐 관리자는 PMOPT에서 자세히 설명된 값을 사용합니다. MQPUT 및 MQPUT1 호출에서 출력 시 큐 관리자는 열린 오브젝트가 단일 큐이지만 분배 목록이 아닌 경우 메시지와 함께 전송된 값으로 이 필드를 설정하지만 열린 오브젝트가 분배 목록인 경우 이를 변경되지 않은 상태로 둡니다. 후자의 경우, 애플리케이션이 생성된 그룹 ID를 알아야 하면 애플리케이션은 PRGID 필드가 포함된 MQPMR 레코드를 제공해야 합니다.

MQGET 호출에 대한 입력에서 큐 관리자는 표 1에 설명된 값을 사용합니다. MQGET 호출에서 출력 시 큐 관리자는 이 필드를 검색된 메시지의 값으로 설정합니다.

다음 특수 값이 정의됩니다.

#### **GINONE**

지정된 그룹 ID가 없습니다.

이 값은 필드 길이만큼의 2진 0입니다. 이는 그룹에 없고 논리 메시지의 세그먼트가 아니며 세그먼트가 허용되지 않는 메시지에 사용되는 값입니다.

이 필드의 길이는 LNGID에서 제공됩니다. 이 필드의 초기값은 GINONE입니다. MDVER가 MDVER2 미만이면 이 필드가 무시됩니다.

#### **MDMFL(10자리 부호 표시 정수)**

메시지 플래그.

이는 메시지의 속성을 지정하거나 해당 처리를 제어하는 플래그입니다. 플래그는 다음 범주로 나뉩니다.

- 세그먼트화 플래그
- 상태 플래그

이는 차례로 설명됩니다.

**세그먼트화 플래그:** 메시지가 큐에 너무 크면 큐에 메시지를 넣으려는 시도가 일반적으로 실패합니다. 세그먼트화는 큐 관리자 또는 애플리케이션이 메시지를 일명 세그먼트라는 더 작은 조각으로 분할하고 별도의 실제 메시지로 큐에 각 세그먼트를 배치하는 기술입니다. 메시지를 검색하는 애플리케이션은 세그먼트를 하나씩 검색하거나 MQGET 호출에서 리턴되는 단일 메시지로 세그먼트를 리어셈블링하도록 큐 관리자에게 요청할 수 있습니다. 후자는 MQGET 호출에서 GMCMPM 옵션을 지정하고 전체 메시지를 수용하도록 충분히 큰 버퍼를 제공하여 달성됩니다. GMCMPM 옵션의 세부사항은 [1032 페이지의 『IBM i의 MQGMO\(메시지가 저오기 옵션\)』](#)의 내용을 참조하십시오. 메시지의 세그먼트화는 송신 큐 관리자, 중간 큐 관리자 또는 목적지 큐 관리자에서 발생할 수 있습니다.

메시지의 세그먼트화를 제어하기 위해 다음 중 하나를 지정할 수 있습니다.

#### **MFSEGI**

세그먼트화가 금지되었습니다.

이 옵션은 메시지가 큐 관리자에 의해 세그먼트로 분할되지 않도록 합니다. 이미 세그먼트화된 메시지에 대해 이 옵션을 지정하면 이 옵션으로 인해 세그먼트가 더 작은 세그먼트로 구분되지 않습니다.

이 플래그의 값은 2진 0입니다. 기본값입니다.

#### **MFSEGA**

세그먼트화가 허용됩니다.

이 옵션은 메시지가 큐 관리자에 의해 세그먼트로 구분되는 것을 허용합니다. 이미 세그먼트화된 메시지에 대해 이 옵션을 지정하면 이 옵션으로 인해 세그먼트가 더 작은 세그먼트로 구분됩니다. MFSEGA는 MFSEG 또는 MFLSEG가 설정되지 않아도 설정될 수 있습니다.

큐 관리자가 메시지를 세그먼트화하는 경우, 큐 관리자는 각 세그먼트와 함께 송신되는 MQMD의 사본에서 MFSEG 플래그를 켜지만, MQPUT 또는 MQPUT1 호출 시 애플리케이션에서 제공된 MQMD에서 이러한 플래그의 설정을 대체하지 않습니다. 논리 메시지의 마지막 세그먼트의 경우, 큐 관리자는 세그먼트와 함께 송신되는 MQMD의 MFLSEG 플래그도 켭니다.

**참고:** PMLOGO 없이 MFSEGA와 함께 메시지를 넣을 때 주의가 필요합니다. 메시지가

- 세그먼트가 아니며
- 그룹에 없으며
- 전달되지 않는 경우

애플리케이션은 고유한 그룹 ID가 각 메시지마다 큐 관리자에 의해 생성되도록 하기 위해 각 MQPUT 또는 MQPUT1 호출 이전에 MDGID 필드를 GINONE로 재설정해야 합니다. 이를 수행하지 않으면 관련되지 않은 메시지가 의도치 않게 동일한 그룹 ID로 넘어갈 수 있으며, 이는 이후 올바르게 처리로 이어질 수 있습니다. MDGID 필드가 재설정되어야 하는 시기에 대한 자세한 정보는 MDGID 필드 및 PMLOGO 옵션의 설명을 참조하십시오.

큐 관리자는 세그먼트(필요할 수 있는 헤더 데이터 추가)가 큐에 맞도록 하기 위해 필요한 대로 세그먼트로 메시지를 분할합니다. 그러나, 큐 관리자가 생성한 세그먼트의 크기에 하한이 있으며 메시지에서 작성된 마지막 세그먼트만 이 한계보다 더 작을 수 있습니다. 애플리케이션 생성 세그먼트의 크기에 대한 하한은 1바이트입니다. 큐 관리자가 생성한 세그먼트는 길이가 같지 않을 수 있습니다. 큐 관리자는 다음과 같이 메시지를 처리합니다.

- 사용자 정의 형식은 16바이트의 배수인 경계에서 분할됩니다. 즉, 큐 관리자는 16바이트보다 더 작은 세그먼트를 생성하지 않습니다(마지막 세그먼트 이외에).
- FMSTR 이외의 내장 형식은 제공된 데이터의 네이처에 적절한 지점에서 분할됩니다. 그러나, 큐 관리자는 MQ 헤더 구조의 중간에서 메시지를 분할하지 않습니다. 즉, 단일 MQ 헤더 구조를 포함하는 세그먼트는 큐 관리자에 의해 추가적으로 분할될 수 없으며 결과적으로 해당 메시지에 대해 가능한 최소 크기는 16바이트보다 큼니다.

큐 관리자가 생성한 두 번째 세그먼트부터는 다음 중 하나로 시작합니다.

- MQ 헤더 구조
- 애플리케이션 메시지 데이터의 시작 부분
- 애플리케이션 메시지 데이터 도중
- FMSTR은 제공된 데이터의 네이처에 상관 없이 분할됩니다(SBCS, DBCS 또는 혼합된 SBCS/DBCS). 문자열이 DBCS 또는 혼합된 SBCS/DBCS인 경우, 이는 하나의 문자 세트에서 다른 문자 세트로 변환할 수 없는 세그먼트를 초래할 수 있습니다. 큐 관리자는 16바이트보다 더 작은 세그먼트로 FMSTR 메시지를 분할하지 않습니다(마지막 세그먼트 이외에).
- 각 세그먼트의 MQMD에 있는 MDFMT, MDCSI 및 MDENC 필드는 세그먼트의 시작 시 제공된 데이터를 올바르게 설명하기 위해 큐 관리자가 설정합니다. 형식 이름은 내장 형식의 이름 또는 사용자 정의 형식의 이름입니다.
- MDOFF가 0보다 큰 세그먼트의 MQMD에 있는 MDREP 필드는 다음과 같이 수정됩니다.
  - 각 보고서 유형마다, 보고서 옵션이 RO\*D이지만 세그먼트가 사용자 데이터의 첫 번째 100바이트를 포함할 수 없는 경우(즉, 제공될 수 있는 MQ 헤더 구조를 뒤따르는 데이터), 보고서 옵션이 RO\*로 변경됩니다.

큐 관리자는 이전 규칙을 따르지만, 그렇지 않으면 예측할 수 없게 메시지를 분할합니다. 메시지가 분할되는 위치에 대해 가정하지 마십시오.

지속 메시지의 경우, 큐 관리자는 작업 단위 내에서만 세그먼트화를 수행할 수 있습니다.

- MQPUT 또는 MQPUT1 호출이 사용자 정의 작업 단위 내에서 작동 중인 경우 해당 작업 단위가 사용됩니다. 호출이 세그먼트화 프로세스 도중 실패하는 경우, 큐 관리자는 실패 호출의 결과로 큐에 배치된 세그먼트를 제거합니다. 그러나 이 실패와 상관없이 작업 단위는 성공적으로 커밋됩니다.
- 호출이 사용자 정의 작업 단위 외부에서 작동 중이고 사용자 정의 작업 단위가 없는 경우 큐 관리자는 호출이 지속되는 동안 작업 단위를 작성합니다. 호출이 성공하면 큐 관리자가 자동으로 작업 단위를 커밋합니다(애플리케이션이 이를 수행할 필요가 없습니다). 호출이 실패하면 큐 관리자가 작업 단위를 백아웃합니다.
- 호출이 사용자 정의 작업 단위 외부에서 작동하지만 사용자 정의 작업 단위가 존재하는 경우, 큐 관리자는 세그먼트화를 수행할 수 없습니다. 메시지에서 세그먼트화가 필요하지 않은 경우, 호출은 여전히 성공할 수 있습니다. 그러나 메시지가 세그먼트화를 요구하는 경우, 호출은 이유 코드 RC2255로 실패합니다.

비지속 메시지의 경우, 세그먼트화를 수행하기 위해 큐 관리자가 작업 단위를 사용할 필요가 없습니다.

세그먼트화될 수 있는 메시지의 데이터를 변환하는 데 다음을 특히 고려해야 합니다.

- 데이터 변환이 MQGET 호출 시 수신 애플리케이션에서만 수행되고 애플리케이션이 GMCMPM 옵션을 지정하는 경우, 데이터-변환 엑시트에 전체 메시지가 전달되어 엑시트가 변환되도록 하며 메시지가 세그먼트화되었다는 사실이 엑시트에 대해 확실하지 않습니다.
- 수신 애플리케이션이 한 번에 하나의 세그먼트를 검색하는 경우 데이터-변환 엑시트가 호출되어 한 번에 하나의 세그먼트를 변환합니다. 따라서, 엑시트가 기타 세그먼트의 데이터와 별도로 세그먼트의 데이터를 변환할 수 있어야 합니다.

메시지에 있는 데이터의 네이처에서 16바이트 경계의 데이터에 대한 임의의 세그먼트화가 엑시트에서 변환될 수 없는 세그먼트를 초래할 수 있거나, 형식이 FMSTR이고 문자 세트가 DBCS 또는 혼합 SBCS/DBCS인 경우, 송신 애플리케이션은 자체적으로 세그먼트를 작성하고 넣어서, MFSEGI를 지정하여 추가 세그먼트화를 억제해야 합니다. 이런 방식으로, 송신 애플리케이션은 데이터-변환 엑시트가 성공적으로 세그먼트를 변환할 수 있도록 각 세그먼트에 충분한 정보가 포함되는지 확인할 수 있습니다.

- 송신자 변환이 송신 메시지 채널 에이전트(MCA)에 대해 지정된 경우, MCA가 논리 메시지의 세그먼트가 아닌 메시지만 변환합니다. MCA는 세그먼트인 메시지를 변환하려고 시도하지 않습니다.

이 플래그는 MQPUT 및 MQPUT1 호출에서는 입력 플래그이며 MQGET 호출에서는 출력 플래그입니다. 또한 후자의 호출 시, 큐 관리자는 MQGMO의 GMSEG 필드로 플래그의 값을 반향시킵니다.

이 플래그의 초기값은 MFSEGI입니다.

**상태 플래그:** 이는 실제 메시지가 메시지 그룹에 속하는지, 논리 메시지의 세그먼트인지, 둘 다이거나 둘 다 아닌지 여부를 나타내는 플래그입니다. 다음 중 하나 이상이 MQPUT 또는 MQPUT1 호출 시 지정되거나 MQGET 호출에서 리턴될 수 있습니다.

#### **MFMI**

메시지가 그룹의 멤버입니다.

#### **MFLMI**

메시지가 그룹의 마지막 논리적 메시지입니다.

이 플래그가 설정되는 경우, 큐 관리자는 메시지와 함께 송신되는 MQMD의 사본에서 MFMI를 켜지만, MQPUT 또는 MQPUT1 호출 시 애플리케이션에서 제공된 MQMD에서 이러한 플래그의 설정을 대체하지 않습니다.

이는 그룹이 하나의 논리 메시지로만 구성되는 데 유효합니다. 이러한 경우, MFLMI가 설정되지만 MDSEQ 필드의 값이 1입니다.

#### **MFSEG**

메시지가 논리적 메시지의 세그먼트입니다.

MFSEG가 MFLSEG 없이 지정되는 경우, 세그먼트의 애플리케이션 메시지 데이터 길이(제공될 수 있는 MQ 헤더 구조의 길이 제외)가 1 이상이어야 합니다. 길이가 0인 경우 MQPUT 또는 MQPUT1 호출은 이 유 코드 RC2253으로 실패합니다.

#### **MFLSEG**

메시지가 논리적 메시지의 마지막 세그먼트입니다.

이 플래그가 설정되는 경우, 큐 관리자는 메시지와 함께 송신되는 MQMD의 사본에서 MFSEG를 켜지만, MQPUT 또는 MQPUT1 호출 시 애플리케이션에서 제공된 MQMD에서 이러한 플래그의 설정을 대체하지 않습니다.

이는 논리 메시지가 하나의 세그먼트로만 구성되는 데 유효합니다. 이러한 경우, MFLSEG가 설정되지만 MDOFF 필드의 값이 0입니다.

MFLSEG가 지정되는 경우 세그먼트의 애플리케이션 메시지 데이터 길이(제공될 수 있는 헤더 구조의 길이 제외)가 0이 되도록 허용됩니다.

메시지를 넣을 때 애플리케이션이 이러한 플래그가 올바르게 설정되었는지 확인해야 합니다. PMLOGO가 지정되거나 큐 핸들에 대해 선행 MQPUT 호출에 지정된 경우, 플래그의 설정은 큐 핸들에 대해 큐 관리자가 보유한 그룹 및 세그먼트 정보와 일치해야 합니다. PMLOGO가 지정된 경우 큐 핸들에 대해 연속 MQPUT 호출에 다음 조건이 적용됩니다.

- 현재 그룹 또는 논리 메시지가 없는 경우 이러한 플래그 모두(및 해당 조합) 유효합니다.
- MFMI가 지정되었으면, MFLMI가 지정될 때까지 켜진 상태로 남아 있어야 합니다. 이 조건이 충족되지 않으면 호출이 이유 코드 RC2241로 실패합니다.
- MFSE가 지정되었으면, MFLSE가 지정될 때까지 켜진 상태로 남아 있어야 합니다. 이 조건이 충족되지 않으면 호출이 이유 코드 RC2242로 실패합니다.
- MFSE가 MFMI 없이 지정되었으면, MFLSE가 지정된 이후까지 MFMI가 켜진 상태로 남아 있어야 합니다. 이 조건이 충족되지 않으면 호출이 이유 코드 RC2242로 실패합니다.

표 1은 플래그의 유효한 조합 및 다양한 필드에 사용된 값을 표시합니다.

이러한 플래그는 MQPUT 및 MQPUT1 호출의 입력 플래그이며 MQGET 호출의 출력 플래그입니다. 또한 후자의 호출 시, 큐 관리자는 MQGMO의 GMGST 및 GMSST 필드로 플래그의 값을 반향시킵니다.

**기본 플래그:** 다음을 지정하여 메시지에 기본 속성이 있는지 나타낼 수 있습니다.

#### MFNONE

메시지 플래그가 없습니다(기본 메시지 속성).

이는 세그먼트화를 방지하며, 메시지가 그룹에 없고 논리적 메시지의 세그먼트가 아님을 표시합니다. MFNONE을 정의하여 프로그램 문서를 지원합니다. 이 플래그는 다른 플래그와 함께 사용하도록 의도되지 않았지만, 해당 값이 0이므로 그와 같은 사용을 감지할 수 없습니다.

MDMFL 필드는 서브필드로 파티션됩니다. 세부사항은 [1365 페이지의 『IBM i의 보고 옵션 및 메시지 플래그』](#)의 내용을 참조하십시오.

이 필드의 초기값은 MFNONE입니다. MDVER가 MDVER2 미만이면 이 필드가 무시됩니다.

#### MDMID(24바이트 비트 문자열)

메시지 ID.

한 메시지를 다른 메시지와 구분하기 위해 사용되는 바이트 문자열입니다. 일반적으로, 큐 관리자가 허용하지 않는 것은 아니지만 동일한 메시지 ID를 갖는 2개의 메시지가 없어야 합니다. 메시지 ID는 메시지의 영구적 특성이며 큐 관리자를 재시작해도 지속됩니다. 메시지 ID가 바이트 문자열이며 문자열이 아니므로, 메시지가 하나의 큐 관리자에서 다른 큐 관리자로 이동할 때 메시지 ID는 문자 세트 간에 변환되지 않습니다.

MQPUT 및 MQPUT1 호출의 경우, 애플리케이션에서 MINONE 또는 PMNMID를 지정하면 큐 관리자는 메시지를 넣을 때 고유한 메시지 ID를 생성하고 메시지와 함께 전송된 메시지 디스크립터에 이를 배치합니다. 또한 큐 관리자가 전송 중인 애플리케이션에 속하는 메시지 디스크립터에서 이 메시지 ID를 리턴합니다. 애플리케이션은 이 값을 사용하여 특정한 메시지에 대한 정보를 레코딩하고 애플리케이션의 기타 부분에서 조회에 응답할 수 있습니다.

큐 관리자가 생성하는 MDMID는 4바이트 제품 ID(ASCII 또는 EBCDIC로 된 AMQ- 또는 CSQ-, 여기서 -는 단일 공백 문자를 나타냄)와 그 뒤에 오는 고유 문자열의 제품 특정 구현으로 구성됩니다. IBM MQ에서 이는 큐 관리자 이름의 첫 번째 12자 및 시스템 클럭에서 나온 값을 포함합니다. 따라서 상호 통신할 수 있는 모든 큐 관리자의 이름이 첫 번째 12자에서 달라야 메시지 ID가 고유합니다. 또한 고유한 문자열을 생성할 기능은 역으로 변경되지 않는 시스템 클럭에 따라 다릅니다. 큐 관리자에서 생성된 메시지 ID가 애플리케이션에서 생성된 메시지 ID를 복제할 가능성을 제거하기 위해, 애플리케이션은 ASCII 또는 EBCDIC에서 A부터 I까지 범위의 초기 문자로 ID를 생성하지 않아야 합니다(X'41'부터 X'49' 및 X'C1'부터 X'C9'). 그러나 애플리케이션은 이러한 범위의 초기 문자를 사용하여 ID를 생성하는 것이 방지되지 않습니다.

메시지를 토크에 넣는 경우 큐 관리자가 발행되는 각 메시지에 필요한 대로 고유한 메시지 ID를 생성합니다. PMNMID가 애플리케이션에서 지정되는 경우, 큐 관리자는 고유한 메시지 ID를 생성하여 출력에 리턴합니다. MINONE이 애플리케이션에서 지정되는 경우, MQMD의 MDMID 필드 값이 호출에서 리턴 시 변경되지 않습니다.

보유된 발행에 대한 자세한 내용은 [PMOPT](#)에서 [PMRET](#)의 설명을 참조하십시오.

메시지를 분배 목록에 넣는 경우, 큐 관리자가 필요한 대로 고유한 메시지 ID를 생성하지만 MQMD의 MDMID 필드 값은 MINONE 또는 PMNMID가 지정되도 호출에서 리턴 시 변경되지 않습니다. 애플리케이션이 큐 관리자에서 생성된 메시지 ID를 알아야 하는 경우, 애플리케이션은 PRMID 필드가 포함된 MQPMR 레코드를 제공해야 합니다.

또한 송신 애플리케이션은 MINONE 이외에 메시지 ID의 특정한 값을 지정할 수 있습니다. 그러면 큐 관리자가 고유한 메시지 ID를 생성하는 것을 중지합니다. 메시지를 전달하는 애플리케이션은 이 기능을 사용하여 기존 메시지의 메시지 ID를 전파할 수 있습니다.

큐 관리자 자체는 다음을 제외하고 이 파일을 이용하지 않습니다.

- 이전에 설명된 대로 요청 시 고유한 값 생성
- 메시지에 대한 Get 요청을 실행하는 애플리케이션에 값 전달
- 이 메시지에 대해 생성하는 보고 메시지의 MDCID 필드에 값을 복사하십시오(MDREP 옵션에 따라).

큐 관리자 또는 메시지 채널 에이전트가 보고 메시지를 생성하는 경우, 기존 메시지의 MDREP 필드에서 지정된 방식으로 MDMID 필드를 설정합니다(ROMMI 또는 ROPMI). 보고 메시지를 생성하는 애플리케이션도 이를 수행해야 합니다.

MQGET 호출의 경우, MDMID는 큐에서 검색될 특정한 메시지를 선택하는 데 사용될 수 있는 5개의 필드 중 하나입니다. 일반적으로 MQGET 호출은 큐에 그 다음 메시지를 리턴하지만, 특정한 메시지가 필요한 경우 이는 임의의 조합에서 5개의 선택 기준 중 하나 이상을 지정하여 가져올 수 있습니다. 이러한 필드는 다음과 같습니다.

- MDMID
- MDCID
- MDGID
- MDSEQ
- MDOFF

애플리케이션은 이러한 필드 중 하나 이상을 요구된 값으로 설정한 후 MQGMO의 GMMO 필드에서 해당하는 MO\* 일치 옵션을 설정하여 해당 필드가 선택 기준으로 사용되어야 함을 나타냅니다. 해당 필드에 지정된 값이 있는 메시지만이 검색 후보입니다. GMMO 필드의 기본값은(애플리케이션에서 대체되지 않은 경우) 메시지 ID 및 상관 ID를 둘 다 일치시키는 것입니다.

일반적으로 리턴되는 메시지는 선택 기준을 만족시키는 큐의 첫 번째 메시지입니다. 그러나 GMBRWN이 지정된 경우 리턴된 메시지는 선택 기준을 충족하는 그 다음 메시지입니다. 이 메시지의 스캔은 현재 커서 위치를 다음 메시지로 시작합니다.

**참고:** 선택 기준을 충족하는 메시지에 대해 순차적으로 큐를 스캔하여, 선택 기준이 지정되지 않은 경우보다 검색 시간이 더 느리며, 적합한 항목을 찾기 전에 다수의 메시지를 스캔해야 하는 경우 특히 그렇습니다.

선택 기준이 다양한 상황에서 사용되는 방식에 대한 자세한 정보는 표 1을 참조하십시오.

메시지 ID로 MINONE을 지정하면 MOMSGI를 지정하지 않는 것과 동일한 영향을 미칩니다. 즉, 임의의 메시지 ID가 일치합니다.

GMMUC 옵션이 MQGET 호출 시 **GMO** 매개변수에 지정된 경우 이 필드는 무시됩니다.

MQGET 호출에서 리턴 시, MDMID 필드는 리턴된 메시지의 메시지 ID로 설정됩니다(있는 경우).

다음 특수 값을 사용할 수 있습니다.

#### **MINONE**

메시지 ID가 지정되지 않습니다.

이 값은 필드 길이 만큼의 2진 0입니다.

이는 MQGET, MQPUT 또는 MQPUT1 호출의 입출력(I/O) 필드입니다. 이 필드의 길이는 LNMID에서 제공됩니다. 이 필드의 초기값은 MINONE입니다.

#### **MDMT(10자리 부호 표시 정수)**

메시지 유형입니다.

이는 메시지의 유형을 표시합니다. 다음과 같이 메시지 유형을 분류할 수 있습니다.

#### **MTSFST**

시스템에서 정의된 메시지 유형에 대한 최저값입니다.

**MTSLST**

시스템에서 정의된 메시지 유형에 대한 최고값입니다.

현재 시스템 범위 내에서 정의된 값은 다음과 같습니다.

**MTDGRM**

응답을 요구하지 않는 메시지입니다.

응답을 요청하지 않는 메시지입니다.

**MTRQST**

응답을 요구하는 메시지입니다.

응답이 필요한 메시지입니다.

응답이 송신되어야 하는 큐의 이름이 MDRQ 필드에 지정되어야 합니다. MDREP 필드는 응답의 MDMID 및 MDCID가 설정되는 방식을 나타냅니다.

**MTRPLY**

이전의 요청 메시지에 대한 응답입니다.

메시지는 이전의 요청 메시지에 대한 응답입니다(MTRQST). 메시지는 요청 메시지의 MDRQ 필드에서 표시된 큐에 송신되어야 합니다. 요청의 MDREP 필드는 응답의 MDMID 및 MDCID가 설정되는 방식을 제어하는 데 사용되어야 합니다.

**참고:** 큐 관리자는 요청-응답 관계를 강요하지 않습니다. 이는 애플리케이션의 책임입니다.

**MTRPRT**

보고 메시지입니다.

메시지는 일반적으로 일부 기타 메시지에 관련되어 일부 예상되거나 예상치 못한 발생 시 보고합니다(예를 들어, 올바르지 않은 데이터를 포함한 요청 메시지가 수신됨). 메시지는 기존 메시지의 메시지 디스크립터의 MDRQ 필드에서 표시된 큐에 송신되어야 합니다. MDFB 필드는 보고서의 네이처를 표시하도록 설정되어야 합니다. 기존 메시지의 MDREP 필드는 보고 메시지의 MDMID 및 MDCID가 설정되어야 하는 방식을 제어하는 데 사용될 수 있습니다.

큐 관리자 또는 메시지 채널 에이전트에서 생성된 보고 메시지는 이전에 설명된 대로 MDFB 및 MDCID 필드가 설정되어 MDRQ 큐에 항상 송신됩니다.

시스템 범위 내의 기타 값은 MQI의 향후 버전에 정의될 수 있으며, 오류 없이 MQPUT 및 MQPUT1 호출에서 허용됩니다.

애플리케이션에서 정의된 값을 사용할 수도 있습니다. 값은 다음 범위 내에 있어야 합니다.

**MTAFST**

애플리케이션에서 정의된 메시지 유형에 대한 최저값입니다.

**MTALST**

애플리케이션에서 정의된 메시지 유형에 대한 최고값입니다.

MQPUT 및 MQPUT1 호출의 경우, MDMT 값은 시스템 정의 범위 또는 애플리케이션 정의 범위 내에 있어야 합니다. 그렇지 않은 경우, 호출은 이유 코드 RC2029로 실패합니다.

이는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 초기값은 MTDGRM입니다.

**MDOFF(10자리 부호 표시 정수)**

논리 메시지의 시작으로부터 실제 메시지의 데이터 오프셋입니다.

이는 데이터가 일부를 형성하는 논리 메시지의 시작으로부터 실제 메시지의 데이터 오프셋(바이트)입니다. 이 데이터를 세그먼트라고 합니다. 오프셋의 범위는 0 - 999,999,999입니다. 논리 메시지의 세그먼트가 아닌 실제 메시지는 0의 오프셋을 갖습니다.

이 필드는 다음과 같은 경우 MQPUT 또는 MQGET 호출 시 애플리케이션에서 설정될 필요가 없습니다.

- MQPUT 호출 시, PMLOGO가 지정됩니다.
- MQGET 호출에서 MOFFS가 지정되지 않습니다.



이는 보고 메시지가 아닌 메시지에 대해 이러한 호출을 사용하는 권장 방법입니다. 그러나 애플리케이션이 이러한 조건을 준수하지 않거나 호출이 MQPUT1인 경우, 애플리케이션은 MDOFF가 적절한 값으로 설정되는지 확인해야 합니다.

MQPUT 및 MQPUT1 호출에 대한 입력에서 큐 관리자는 표 1에 설명된 값을 사용합니다. MQPUT 및 MQPUT1 호출에서 출력 시 큐 관리자는 이 필드를 메시지와 함께 전송된 값으로 설정합니다.

논리 메시지의 세그먼트에 대해 보고하는 보고 메시지의 경우, MDOLN 필드(OLUNDF가 아니면)를 사용하여 큐 관리자가 보유한 세그먼트 정보의 오프셋을 업데이트합니다.

MQGET 호출에 대한 입력에서 큐 관리자는 표 1에 설명된 값을 사용합니다. MQGET 호출에서 출력 시 큐 관리자는 이 필드를 검색된 메시지의 값으로 설정합니다.

이 필드의 초기값은 0입니다. MDVER가 MDVER2 미만이면 이 필드가 무시됩니다.

### MDOLN(10자리 부호 표시 정수)

원래 메시지의 길이.

이 필드는 세그먼트인 보고 메시지에 대해서만 관련됩니다. 이는 보고 메시지가 관련된 메시지 세그먼트의 길이를 지정합니다. 이는 세그먼트가 일부를 형성하는 논리 메시지의 길이나 보고 메시지의 데이터 길이도 지정하지 않습니다.

**참고:** 세그먼트인 메시지에 대한 보고 메시지를 생성할 때 큐 관리자 및 메시지 채널 에이전트는 원래 메시지의 MDGID, MDSEQ, MDOFF 및 MDMFL 필드를 보고 메시지의 MQMD에 복사합니다. 결과적으로 보고 메시지 역시 세그먼트입니다. 보고 메시지를 생성하는 애플리케이션은 동일한 작업을 수행하고 MDOLN 필드가 올바르게 설정되는지 확인하도록 권장됩니다.

다음 특수 값이 정의됩니다.

#### OLUNDF

메시지의 원래 길이가 정의되지 않습니다.

MDOLN은 MQPUT 및 MQPUT1 호출의 입력 필드이지만, 애플리케이션에서 제공된 값은 특정한 상황에서만 허용됩니다.

- 넣는 메시지가 세그먼트이며 또한 보고 메시지인 경우, 큐 관리자가 지정된 값을 승인합니다. 값은 다음과 같아야 합니다.
  - 세그먼트가 마지막 세그먼트가 아닌 경우 0보다 큼
  - 세그먼트가 마지막 세그먼트인 경우 0 이상이어야 함
  - 메시지 내에 있는 데이터의 길이 이상이어야 함

이러한 조건이 충족되지 않으면 호출이 이유 코드 RC2252로 실패합니다.

- 넣는 메시지가 세그먼트이지만 보고 메시지가 아닌 경우, 큐 관리자는 필드를 무시하고 대신 애플리케이션 메시지 데이터의 길이를 사용합니다.
- 기타 모든 경우, 큐 관리자는 필드를 무시하고 대신 값 OLUNDF를 사용합니다.

이는 MQGET 호출의 출력 필드입니다.

이 필드의 초기값은 OLUNDF입니다. MDVER가 MDVER2 미만이면 이 필드가 무시됩니다.


### MDPAN(28바이트 문자열)

메시지를 넣는 애플리케이션의 이름입니다.

이는 메시지의 원본 컨텍스트의 일부입니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어](#)를 참조하십시오.

MDPAN의 형식은 MDPAT의 값에 따라 다릅니다.

이 필드가 큐 관리자에서 설정되는 경우(즉, PMSETA를 제외한 모든 옵션의 경우), 환경에서 판별되는 값으로 설정됩니다.

-  z/OS에서 큐 관리자는 다음을 사용합니다.
  - z/OS 배치의 경우, JES JOB 카드의 8자 작업 이름

- TSO의 경우, 7자 TSO 사용자 ID
- CICS의 경우, 4자 tranid가 뒤따르는 8자 applid
- IMS의 경우, 8자 PSB 이름이 뒤따르는 8자 IMS 시스템 ID
- XCF의 경우, 16자 XCF 멤버 이름이 뒤따르는 8자 XCF 그룹 이름
- 큐 관리자에서 생성된 메시지의 경우, 큐 관리자 이름의 첫 번째 28자
- CICS 없이 분산 큐잉의 경우, 8자 태스크 ID가 뒤따르는 데드-레터 큐에 넣는 모듈의 8자 이름이 뒤따르는 채널 시작기의 8자 작업 이름
- IBM MQ for z/OS을(를) 사용한 MQSeries Java 언어 바인딩 처리의 경우 z/OS UNIX System Services 환경에 맞게 작성된 주소 공간의 8자로 된 작업 이름입니다. 일반적으로, 이는 단일 숫자 문자가 추가된 TSO 사용자 ID가 됩니다.

각 이름은 필드의 나머지에 공백이 있는 것처럼 오른쪽까지 공백으로 채워집니다. 둘 이상의 이름이 있는 경우 해당 구분자가 없습니다.

- **Windows** PC DOS 및 Windows 시스템에서, 큐 관리자는 다음을 사용합니다.
  - CICS 애플리케이션의 경우, CICS 트랜잭션 이름
  - 비CICS 애플리케이션의 경우, 실행 파일의 완전한 이름 중 가장 오른쪽 28자
- **IBM i** IBM i에서, 큐 관리자는 완전한 작업 이름을 사용합니다.
- **Linux** **AIX** AIX and Linux에서 큐 관리자는 다음을 사용합니다.
  - CICS 애플리케이션의 경우, CICS 트랜잭션 이름
  - 비CICS 애플리케이션의 경우, 큐 관리자에게 사용 가능하면 실행 파일의 완전한 이름 중 가장 오른쪽 14자와 그렇지 않으면 공백(예를 들어, AIX에서)
- VSE/ESA에서, 큐 관리자는 4자 tranid가 뒤따르는 8자 applid를 사용합니다.

MQPUT 및 MQPUT1 호출의 경우, PMSETA가 **PMO** 매개변수에서 지정되면 이는 입/출력(I/O) 필드입니다. 필드 내에서 널 문자 이후의 정보는 제거됩니다. 널 문자 및 다음 문자는 큐 관리자에 의해 공백으로 변환됩니다. PMSETA를 지정하지 않으면 입력에서 이 필드가 무시되며 출력 전용 필드가 됩니다.

MQGET 호출의 출력 필드입니다. 이 필드의 길이는 LNPN에서 제공됩니다. 이 필드의 초기값은 28개의 공백 문자입니다.

### MDPAT(10자리 부호 표시 정수)

메시지를 넣는 애플리케이션의 유형입니다.

이는 메시지의 원본 컨텍스트의 일부입니다. 메시지 컨텍스트에 대한 자세한 정보는 메시지 컨텍스트 및 컨텍스트 정보 제어를 참조하십시오.

MDPAT에는 다음 표준 유형 중 하나가 있을 수 있습니다. 또한 사용자 정의 유형을 사용할 수 있지만 ATUFST에서 ATULST 범위의 값으로 제한되어야 합니다.

#### ATAIX

AIX 애플리케이션(ATUNIX와 동일한 값).

#### ATBRKR

브로커입니다.

#### ATCICS

CICS 트랜잭션.

#### ATCICB

CICS bridge.

#### ATVSE

CICS/VSE 트랜잭션.

#### ATDOS

PC DOS용 IBM MQ MQI client 애플리케이션입니다.



**ATDQM**

분산 큐 관리자 에이전트.

**ATGUAR**

Tandem Guardian 애플리케이션(ATNSK와 동일한 값)입니다.

**ATIMS**

IMS 애플리케이션입니다.

**ATIMSB**

IMS 브릿지.

**ATJAVA**

Java.

**ATMVS**

MVS 또는 TSO 애플리케이션(ATZOS와 동일한 값)입니다.

**ATNOTE**

Lotus Notes Agent 애플리케이션입니다.

**ATNSK**

Tandem NonStop Kernel 애플리케이션입니다.

**AT390**

OS/390 애플리케이션(ATZOS와 동일한 값)입니다.

**AT400**

IBM i 애플리케이션입니다.

**ATQM**

큐 관리자.

**ATUNIX**

UNIX 애플리케이션입니다.

**ATVOS**

Stratus VOS 애플리케이션입니다.

**ATWIN**

16비트 Windows 애플리케이션입니다.

**ATWINT**

32비트 Windows 애플리케이션입니다.

**ATXCF**

XCF.

**ATZOS**

z/OS 애플리케이션입니다.

**ATDEF**

기본 애플리케이션 유형입니다.

이는 애플리케이션이 실행되는 플랫폼의 기본 애플리케이션 유형입니다.

**참고:** 이 상수의 값은 환경에 따라 달라집니다.

**ATUNK**

알 수 없는 애플리케이션 유형입니다.

이 값은 기타 컨텍스트 정보가 있어도 애플리케이션 유형을 알 수 없음을 나타내는 데 사용할 수 있습니다.

**ATUFST**

사용자 정의된 애플리케이션 유형의 최저값입니다.

**ATULST**

사용자 정의된 애플리케이션 유형의 최고값입니다.

다음은 발생할 수 있는 특수 값입니다.

## ATNCON

메시지에 컨텍스트 정보가 없습니다.


이 값은 컨텍스트 없는 메시지를 넣을 때 큐 관리자가 설정합니다(즉, PMNOC 컨텍스트 옵션이 지정됨).

메시지가 검색될 때, MDPAT를 이 값에 대해 테스트하여 메시지에 컨텍스트가 있는지 여부를 결정할 수 있습니다(기타 컨텍스트 필드가 공백이 아닌 경우 PMSETA를 사용하는 애플리케이션에서 MDPAT가 ATNCON으로 설정되지 않도록 권장됨).

## ATSIB

메시지가 다른 IBM MQ 메시징 제품에서 시작되었고 SIB(Service Integration Bus) 브릿지를 통해 도달했음을 나타냅니다.

애플리케이션 넣기의 결과로 큐 관리자가 이 정보를 생성하는 경우 필드는 환경이 판별하는 값으로 설정됩니다.

 IBM i에서 필드가 AT400으로 설정됩니다. 큐 관리자는 IBM i에서 ATCICS를 사용하지 않습니다.

MQPUT 및 MQPUT1 호출의 경우, PMSETA가 **PMO** 매개변수에서 지정되면 이는 입/출력(I/O) 필드입니다. PMSETA를 지정하지 않으면 입력에서 이 필드가 무시되며 출력 전용 필드가 됩니다.

MQPUT 또는 MQPUT1 호출이 성공적으로 완료되면, 이 필드에는 큐에 넣은 경우 메시지와 함께 전송된 MDPAT가 포함됩니다. 이는 보유한 경우 메시지와 함께 보관되지만(보유된 발행물에 대한 자세한 내용은 PMRET의 설명 참조), 구독자에게 송신된 모든 발행물에서 MDPAT를 대체할 값을 구독자가 제공하므로 메시지가 구독자에게 발행물로 송신될 때 MDPAT로 사용되지 않는 MDPAT의 값입니다. 메시지에 컨텍스트가 없는 경우 필드가 ATNCON으로 설정됩니다.

MQGET 호출의 출력 필드입니다. 이 필드의 초기값은 ATNCON입니다.

## MDPD(8바이트 문자열)

메시지를 넣은 날짜입니다.

이는 메시지의 원본 컨텍스트의 일부입니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어](#)를 참조하십시오.

이 필드가 큐 관리자에 의해 생성될 때 날짜에 사용되는 형식입니다.

• YYYYMMDD

여기서 각 문자는 다음을 나타냅니다.

### YYYY

연도(4자리 숫자)

### MM

연 중 월(01에서 12)

### DD

월 중 일(01에서 31)

그리니치 표준시(GMT)가 MDPD 및 MDPT 필드에 사용되고, 시스템 클럭이 GMT로 정확히 설정됩니다.

메시지를 작업 단위의 일부로 넣은 경우, 날짜는 작업 단위가 커밋된 날짜가 아니라 메시지를 넣은 날짜입니다.

MQPUT 및 MQPUT1 호출의 경우, PMSETA가 **PMO** 매개변수에서 지정되면 이는 입/출력(I/O) 필드입니다. 필드의 콘텐츠는 필드 내에서 널 문자 다음 정보가 제거되는 것을 제외하고 큐 관리자에서 확인되지 않습니다. 널 문자 및 다음 문자는 큐 관리자에 의해 공백으로 변환됩니다. PMSETA를 지정하지 않으면 입력에서 이 필드가 무시되며 출력 전용 필드가 됩니다.

MQPUT 또는 MQPUT1 호출이 성공적으로 완료되면, 이 필드에는 큐에 넣은 경우 메시지와 함께 전송된 MDPD가 포함됩니다. 이는 보유한 경우 메시지와 함께 보관되지만(보유된 발행물에 대한 자세한 내용은 PMRET의 설명 참조), 구독자에게 송신된 모든 발행물에서 MDPD를 대체할 값을 구독자가 제공하므로 메시지가 구독자에게 발행물로 송신될 때 MDPD로 사용되지 않는 MDPD의 값입니다. 메시지에 컨텍스트가 없는 경우 필드 전체가 공백입니다.

MQGET 호출의 출력 필드입니다. 이 필드의 길이는 LNPDAT에서 제공됩니다. 이 필드의 초기값은 8개의 빈 문자입니다.

### MDPER(10자리 부호 표시 정수)

메시지 지속성.

이는 메시지가 시스템 실패 및 큐 관리자의 재시작 이후 남아 있는지 여부를 나타냅니다. MQPUT 및 MQPUT1 호출의 경우, 값은 다음 중 하나여야 합니다.

#### PEPER

메시지가 지속됩니다.

이는 시스템 장애 또는 큐 관리자 재시작 시 메시지가 지속된다는 의미입니다. 메시지를 넣었고 퍼터의 작업 단위가 커밋되었으면(메시지를 작업 단위의 일부로 넣는 경우), 메시지가 보조 기억장치에 보존됩니다. 이는 메시지가 큐에서 제거되고 게터의 작업 단위가 커밋될 때까지 해당 위치에 남아 있습니다(메시지를 작업 단위의 일부로 검색하는 경우).

지속 메시지가 리모트 큐로 송신되는 경우, 메시지가 그 다음 큐 관리자에서 도달했다고 알려질 때까지 목적지에 대한 라우트를 따라 각 큐 관리자에서 메시지를 보유하는 데 저장 후 전달 메커니즘이 사용됩니다.

다음 큐에는 지속 메시지를 넣을 수 없습니다.

- 임시 동적 큐
- 커플링 기능 구조 레벨이 3 미만이거나 커플링 기능 구조가 복구 가능하지 않은 공유 큐

지속 메시지는 커플링 기능 구조 레벨이 3이고 커플링 기능이 복구 가능한 공유 큐와 사전정의된 큐, 영구적 동적 큐에 배치될 수 있습니다.

#### PENPER

메시지가 지속되지 않습니다.

이는 일반적으로 시스템 장애 또는 큐 관리자 재시작 시 메시지가 지속되지 않는다는 의미입니다. 이것은 큐 관리자를 재시작하는 동안 원본 그대로의 메시지 사본이 보조 기억장치에서 발견되는 경우에도 적용됩니다.

공유 큐라는 특별한 사례에서는 큐 공유 그룹에서 큐 관리자를 재시작해도 비지속 메시지가 지속되지만, 공유 큐에 메시지를 저장하는 데 사용하는 커플링 기능에 장애가 발생하면 지속되지 않습니다.

#### PEQDEF

메시지에 기본 지속성이 있습니다.

- 큐가 클러스터 큐인 경우, 메시지의 지속성은 메시지가 배치된 큐의 특정 인스턴스를 소유하는 목적지 큐 관리자에서 정의된 **DefPersistence** 속성에서 가져옵니다. 일반적으로 클러스터 큐의 모든 인스턴스는 **DefPersistence** 속성에 대해 동일한 값을 가지지만, 이는 필수가 아닙니다.

메시지가 목적지 큐에 배치될 때 **DefPersistence** 값이 **MDPER** 필드에 복사됩니다.

**DefPersistence**이(가) 나중에 변경되면 큐에 이미 있는 메시지는 영향을 받지 않습니다.

- 큐가 클러스터 큐가 아닌 경우, 메시지의 지속성은 목적지 큐 관리자가 원격인 경우에도 로컬 큐 관리자에서 정의된 **DefPersistence** 속성에서 가져옵니다.

큐 이름 해석 경로에 정의가 둘 이상인 경우, 기본 지속성은 경로에서 첫 번째 정의에 있는 이 속성의 값에서 가져옵니다. 값은 다음이 될 수 있습니다.

- 알리어스 큐
- 로컬 큐
- 리모트 큐의 로컬 정의
- 큐 관리자 알리어스
- 전송 큐(예: DefXmitQName 큐)

메시지를 넣을 때 **DefPersistence** 값이 **MDPER** 필드에 복사됩니다. **DefPersistence**이(가) 나중에 변경되면 이미 넣은 메시지는 영향을 받지 않습니다.

지속 및 비지속 메시지 둘 다 동일한 큐에 존재할 수 있습니다.

메시지에 응답할 때, 애플리케이션은 일반적으로 요청 메시지의 지속성을 응답 메시지에 대해 사용해야 합니다.

MQGET 호출의 경우, 리턴된 값은 PEPER 또는 PENPER입니다.

이 필드는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 초기값은 PEQDEF입니다.

### MDPRI(10자리 부호 표시 정수)

메시지 우선순위입니다.

MQPUT 및 MQPUT1 호출의 경우, 값은 0 이상이어야 합니다. 0은 우선순위가 가장 낮습니다. 다음 특수 값을 사용할 수도 있습니다.

#### PRQDEF

큐의 기본 우선순위입니다.

- 큐가 클러스터 큐인 경우, 메시지의 우선순위는 메시지가 배치된 큐의 특정 인스턴스를 소유하는 목적지 큐 관리자에서 정의된 대로 **DefPriority** 속성에서 가져옵니다. 일반적으로 클러스터 큐의 모든 인스턴스는 **DefPriority** 속성에 대해 동일한 값을 가지지만, 이는 필수가 아닙니다.

메시지가 목적지 큐에 배치될 때 **DefPriority** 값이 MDPRI 필드에 복사됩니다. **DefPriority**이 (가) 나중에 변경되면 큐에 이미 있는 메시지는 영향을 받지 않습니다.

- 큐가 클러스터 큐가 아닌 경우, 메시지의 우선순위는 목적지 큐 관리자가 원격인 경우에도 로컬 큐 관리자에서 정의된 대로 **DefPriority** 속성에서 가져옵니다.

큐 이름 해석 경로에 정의가 둘 이상인 경우, 기본 우선순위는 경로에서 첫 번째 정의에 있는 이 속성의 값에서 가져옵니다. 값은 다음이 될 수 있습니다.

- 알리어스 큐
- 로컬 큐
- 리모트 큐의 로컬 정의
- 큐 관리자 알리어스
- 전송 큐(예: DefXmitQName 큐)

메시지를 넣을 때 **DefPriority** 값이 MDPRI 필드에 복사됩니다. **DefPriority**이(가) 나중에 변경되면 이미 넣은 메시지는 영향을 받지 않습니다.

MQGET 호출에서 리턴된 값은 항상 0 이상입니다. 값 PRQDEF는 리턴되지 않습니다.

메시지를 로컬 큐 관리자에서 지원된 최대값보다 큰 우선순위로 넣는 경우(이 최대값은 **MaxPriority** 큐 관리자 속성에서 제공됨), 메시지가 큐 관리자에서 허용되지만 큐 관리자의 최대 우선순위에서 큐에 배치됩니다. MQPUT 또는 MQPUT1 호출은 CCWARN 및 이유 코드 RC2049로 완료됩니다. 그러나, MDPRI 필드는 메시지를 넣는 애플리케이션에서 지정된 값을 보유합니다.

메시지에 응답할 때, 애플리케이션은 일반적으로 요청 메시지의 우선순위를 응답 메시지에 대해 사용해야 합니다. 기타 상황에서, PRQDEF를 지정하면 애플리케이션을 변경하지 않고 우선순위 성능 조정이 수행될 수 있습니다.

이 필드는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 초기값은 PRQDEF입니다.

### MDPT(8바이트 문자열)

메시지를 넣은 시간입니다.

이는 메시지의 **원본 컨텍스트**의 일부입니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어](#)를 참조하십시오.

이 필드가 큐 관리자에 의해 생성될 때 시간에 사용되는 형식입니다.

- HHMMSSTH

여기서 문자는 다음을 나타냅니다(순서대로).

**HH**

시간(00에서 23까지)

**MM**

분(00에서 59까지)

**SS**

초(00에서 59까지, [참고 참조](#))

**T**

1/10초(0에서 9)

**H**

1/100초(0에서 9)

**참고:** 시스템 클럭이 매우 정확한 시간 표준에 동기화되는 경우, MDPT에서 해당 초에 대해 60 또는 60이 드물게 리턴될 수 있습니다. 이는 윤초가 글로벌 시간 표준으로 삽입되는 경우 발생합니다.

그리니치 표준시(GMT)가 MDPD 및 MDPT 필드에 사용되고, 시스템 클럭이 GMT로 정확히 설정됩니다.

메시지를 작업 단위의 일부로 넣은 경우, 시간은 작업 단위가 커미트된 시간이 아니라 메시지를 넣은 시간입니다.

MQPUT 및 MQPUT1 호출의 경우, PMSETA가 **PMO** 매개변수에서 지정되면 이는 입/출력(I/O) 필드입니다. 필드의 콘텐츠는 필드 내에서 널 문자 다음 정보가 제거되는 것을 제외하고 큐 관리자에서 확인되지 않습니다. 널 문자 및 다음 문자는 큐 관리자에 의해 공백으로 변환됩니다. PMSETA를 지정하지 않으면 입력에서 이 필드가 무시되며 출력 전용 필드가 됩니다.

MQPUT 또는 MQPUT1 호출이 성공적으로 완료되면, 이 필드에는 큐에 넣은 경우 메시지와 함께 전송된 MDPT 값이 포함됩니다. 이는 보유한 경우 메시지와 함께 보관되지만(보유된 발행물에 대한 자세한 내용은 PMRET의 설명 참조), 구독자에게 송신된 모든 발행물에서 MDPT를 대체할 값을 구독자가 제공하므로 메시지가 구독자에게 발행물로 송신될 때 MDPT로 사용되지 않는 MDPT의 값입니다. 메시지에 컨텍스트가 없는 경우 필드 전체가 공백입니다.

MQGET 호출의 출력 필드입니다. 이 필드의 길이는 LNPTIM에서 제공됩니다. 이 필드의 초기값은 8개의 빈 문자입니다.

**MDREP(10자리 부호 표시 정수)**

보고 메시지의 옵션입니다.

보고 메시지는 다른 메시지에 대한 메시지이며 원래 메시지와 관련된 예상 이벤트 또는 예상치 못한 이벤트에 대해 애플리케이션에 알리는 데 사용됩니다. MDREP 필드에서는 기존 메시지를 송신하는 애플리케이션이 어느 보고 메시지가 필요한지, 애플리케이션 메시지 데이터가 이에 포함되는지 여부 및 (보고서 및 응답 둘 다에 대해) 보고 또는 응답 메시지의 메시지 및 상관 ID가 설정되는 방식을 지정할 수 있습니다. 다음 유형의 보고 메시지 중 일부 또는 모두(또는 없음) 요청될 수 있습니다.

- 예외
- 만기
- 도착 시 확인(COA)
- 전달 시 확인(COD)
- 긍정적 조치 알림(PAN)
- 부정적 조치 알림(NAN)

둘 이상의 유형의 보고 메시지가 필요하거나 기타 보고 옵션이 필요한 경우, 값이 함께 추가될 수 있습니다(두 번 이상 동일한 상수 추가 금지).

보고 메시지를 수신하는 애플리케이션은 MQMD의 MDFB 필드를 검사하여 보고서가 생성된 이유를 판별할 수 있습니다. 자세한 내용은 MDFB 필드를 참조하십시오.

메시지를 토크에 넣을 때 보고 옵션을 사용하면 0, 하나 또는 다수의 보고 메시지가 생성되어 애플리케이션에 송신될 수 있습니다. 이는 발행 메시지가 0, 하나 또는 다수의 구독 애플리케이션에 송신될 수 있기 때문입니다.

**예외 옵션:** 다음 옵션 중 하나를 지정하여 예외 보고 메시지를 요청할 수 있습니다.

## ROACTIVITY

활동 보고서 필수

이 보고서 옵션이 설정된 메시지가 지원 애플리케이션에서 처리될 때마다, 이 보고서 옵션을 통해 활동 보고서가 생성될 수 있습니다.

이 보고서 옵션이 설정된 메시지는 옵션을 '이해'하지 않은 경우에도 임의의 큐 관리자에서 허용되어야 합니다. 이전의 큐 관리자에서 처리되는 경우에도, 이를 통해 보고서 옵션이 임의의 사용자 메시지에 설정될 수 있습니다. 이를 달성하기 위해 보고서 옵션이 ROAUM 하위 필드에 배치됩니다.

프로세스(큐 관리자 또는 사용자 프로세스)가 ROACT가 설정된 메시지에서 활동을 수행하는 경우, 활동 보고서를 생성하고 넣도록 선택할 수 있습니다.

활동 보고서 옵션을 통해 메시지의 라우트가 큐 관리자 네트워크를 통해 추적될 수 있습니다. 보고서 옵션은 현재 사용자 메시지에서 지정될 수 있으며 즉시 네트워크를 통해 메시지의 라우트를 계산하도록 시작할 수 있습니다. 메시지를 생성하는 애플리케이션이 활동 보고서를 사용 가능하게 할 수 없는 경우, 큐 관리자가 제공하는 API 교차 엑시트를 사용하여 사용 가능하게 할 수 있습니다.

몇몇 조건이 활동 보고서에 적용 가능합니다.

1. 활동 보고서를 생성할 수 있는 네트워크에 더 적은 큐 관리자가 있는 경우 라우트는 덜 세부적입니다.
2. 활동 보고서는 수행된 라우트를 판별하기 위해 쉽게 '정렬 가능'하지 않을 수 있습니다.
3. 활동 보고서는 요청된 해당 목적지에 대한 라우트를 찾을 수 없을 수 있습니다.

## ROEXC

필수 예외 보고서입니다.

이 유형의 보고서는 메시지가 다른 큐 관리자에 송신되고 지정된 목적지 큐에 전달될 수 없는 경우 메시지 채널 에이전트에서 생성될 수 있습니다. 예를 들어, 목적지 큐 또는 중간 전송 큐가 가득 찼거나 메시지가 큐에 비해 너무 큰 경우입니다.

예외 보고 메시지의 생성은 기존 메시지의 지속성 및 기존 메시지가 통과하는 메시지 채널의 속도(보통 또는 빠름)에 따라 다릅니다.

- 모든 지속 메시지의 경우 및 보통 메시지 채널을 통과하는 비지속 메시지의 경우, 오류 조건에 대해 송신 애플리케이션에서 지정된 조치가 성공적으로 완료될 수 있는 경우에 한해서만 예외 보고서가 생성됩니다. 송신 애플리케이션은 다음 조치 중 하나를 지정하여 오류 조건 발생 시 기존 메시지의 배치를 제어할 수 있습니다.

- RODLQ(이로 인해 기존 메시지가 데드-레터 큐에 배치됨)
- RODISC(이로 인해 기존 메시지가 제거됨)

송신 애플리케이션에서 지정된 조치가 성공적으로 완료될 수 없는 경우, 기존 메시지가 전송 큐에 남아 있으며 예외 보고 메시지가 생성되지 않습니다.

- 빠른 메시지 채널을 통해 이동하는 비지속 메시지의 경우 원래 메시지가 전송 큐에서 제거되고, 오류 조건에 대해 지정된 조치가 성공적으로 완료되지 못한 경우에도 예외 보고서가 생성됩니다. 예를 들어, RODLQ가 지정되었으나 해당 큐가 가득 차서 기존 메시지가 데드-레터 큐에 배치될 수 없는 경우, 예외 보고 메시지가 생성되고 기존 메시지가 제거됩니다.

보통 및 빠른 메시지 채널에 대한 자세한 정보는 [메시지 지속](#)을 참조하십시오.

기존 메시지를 넣은 애플리케이션이 MQPUT 또는 MQPUT1 호출에서 리턴된 이유 코드로 문제점에 대해 동기적으로 알림을 받을 수 있는 경우 예외 보고서는 생성되지 않습니다.

또한 수신한 메시지가 처리될 수 없음을 나타내기 위해 애플리케이션이 예외 보고서를 송신할 수 있습니다(예를 들어, 계정이 해당 신용 한도를 초과하도록 하는 차변 거래이기 때문).

원래 메시지의 메시지 데이터는 보고 메시지에 포함되지 않습니다.

ROEXC, ROEXCD 및 ROEXCF 중 둘 이상을 지정하지 마십시오.

## ROEXCD

데이터가 포함된 필수 예외 보고서입니다.

이는 기존 메시지에서 애플리케이션 메시지 데이터의 첫 번째 100바이트가 보고 메시지에 포함되는 것을 제외하고 ROEXC와 동일합니다. 원래 메시지에 하나 이상의 MQ 헤더 구조가 포함되면 100바이트의 애플리케이션 바이트 외에 해당 구조도 보고 메시지에 포함됩니다.

ROEXC, ROEXCD 및 ROEXCF 중 둘 이상을 지정하지 마십시오.

#### **ROEXCF**

전체 데이터가 포함된 필수 예외 보고서입니다.

이는 기존 메시지에서 모든 애플리케이션 메시지 데이터가 보고 메시지에 포함되는 것을 제외하고 ROEXC와 동일합니다.

ROEXC, ROEXCD 및 ROEXCF 중 둘 이상을 지정하지 마십시오.

**만기 옵션:** 다음 옵션 중 하나를 지정하여 만기 보고 메시지를 요청할 수 있습니다.

#### **ROEXP**

필수 만기 보고서입니다.

이 유형의 보고서는 만기 시간이 지나서 애플리케이션에 전달하기 전에 메시지가 제거되는 경우 큐 관리자에서 생성됩니다(MDEXP 필드 참조). 이 옵션이 설정되지 않은 경우, 메시지가 이 이유로 제거되면 보고 메시지가 생성되지 않습니다(ROEXC\* 옵션 중 하나가 지정된 경우에도).

원래 메시지의 메시지 데이터는 보고 메시지에 포함되지 않습니다.

ROEXP, ROEXPD 및 ROEXPF 중 둘 이상을 지정하지 마십시오.

#### **ROEXPD**

데이터가 포함된 필수 만기 보고서입니다.

이는 기존 메시지에서 애플리케이션 메시지 데이터의 첫 번째 100바이트가 보고 메시지에 포함되는 것을 제외하고 ROEXP와 동일합니다. 원래 메시지에 하나 이상의 MQ 헤더 구조가 포함되면 100바이트의 애플리케이션 바이트 외에 해당 구조도 보고 메시지에 포함됩니다.

ROEXP, ROEXPD 및 ROEXPF 중 둘 이상을 지정하지 마십시오.

#### **ROEXPF**

전체 데이터가 포함된 필수 만기 보고서입니다.

이는 기존 메시지에서 모든 애플리케이션 메시지 데이터가 보고 메시지에 포함되는 것을 제외하고 ROEXP와 동일합니다.

ROEXP, ROEXPD 및 ROEXPF 중 둘 이상을 지정하지 마십시오.

**COA(도달 시 확인) 옵션:** 다음 옵션 중 하나를 지정하여 COA(도달 시 확인) 보고 메시지를 요청할 수 있습니다.

#### **ROCOA**

필수 COA(도달 시 확인) 보고서입니다.

메시지가 목적지 큐에 배치될 때, 이 유형의 보고서는 목적지 큐를 소유하는 큐 관리자에서 생성됩니다. 원래 메시지의 메시지 데이터는 보고 메시지에 포함되지 않습니다.

메시지를 작업 단위의 일부로 넣고 목적지 큐가 로컬 큐이면, 큐 관리자에서 생성된 COA 보고 메시지는 작업 단위가 커밋된 경우에만 검색이 가능하게 됩니다.

COA 보고서는 메시지 디스크립터의 MDFMT 필드가 FMXQH 또는 FMDLH인 경우 생성되지 않습니다. 이로 인해 메시지를 전송 큐에 넣거나 메시지를 배달할 수 없어 데드-레터 큐에 넣는 경우 COA 보고서가 생성되지 않습니다.

ROCOA, ROCOAD 및 ROCOAF 중 둘 이상을 지정하지 마십시오.

#### **ROCOAD**

데이터가 포함된 필수 COA(도달 시 확인) 보고서입니다.

이는 기존 메시지에서 애플리케이션 메시지 데이터의 첫 번째 100바이트가 보고 메시지에 포함되는 것을 제외하고 ROCOA와 동일합니다. 원래 메시지에 하나 이상의 MQ 헤더 구조가 포함되면 100바이트의 애플리케이션 바이트 외에 해당 구조도 보고 메시지에 포함됩니다.

ROCOA, ROCOAD 및 ROCOAF 중 둘 이상을 지정하지 마십시오.

## **ROCOAF**

전체 데이터가 포함된 필수 COA(도달 시 확인) 보고서입니다.

이는 기존 메시지에서 모든 애플리케이션 메시지 데이터가 보고 메시지에 포함되는 것을 제외하고 ROCOA와 동일합니다.

ROCOA, ROCOAD 및 ROCOAF 중 둘 이상을 지정하지 마십시오.

**제거 및 만기 옵션:** 다음 옵션을 지정하여 보고 메시지의 만기 시간 및 제거 플래그를 설정할 수 있습니다.

## **ROPDAE**

보고 메시지 만기 시간 및 제거 플래그를 설정합니다.

이 옵션은 해당 기존 메시지에서 보고 메시지 및 응답 메시지가 만기 시간 및 제거 플래그(제거 여부)를 상속하도록 합니다. 이 옵션이 설정되어, 보고 및 응답 메시지가 다음과 같습니다.

1. RODISC 플래그를 상속하십시오(설정된 경우).
2. 메시지가 만기 보고서가 아닌 경우 메시지의 남아 있는 만기 시간을 상속하십시오. 메시지가 만기 보고서인 경우 만기 시간이 60초로 설정됩니다.

이 옵션이 설정되어 다음이 적용됩니다.

### **참고:**

1. 보고 및 응답 메시지는 삭제 플래그 및 만기 값과 함께 생성되고, 시스템 내에 남아 있을 수 없습니다.
2. 추적 라우트 메시지는 비추적 라우트 사용 큐 관리자의 목적지 큐에 도달하는 것이 금지됩니다.
3. 통신 링크가 중단된 경우, 큐는 전달될 수 없는 보고서로 채워지는 것이 금지됩니다.
4. 명령 서버 응답은 요청의 남아 있는 만기를 상속합니다.

**COD(전달 시 확인) 옵션:** 다음 옵션 중 하나를 지정하여 COD(전달 시 확인) 보고 메시지를 요청할 수 있습니다.

## **ROCOD**

필수 COD(전달 시 확인) 보고서입니다.

이 유형의 보고서는 메시지가 큐에서 삭제되도록 하는 방식으로 애플리케이션이 목적지 큐에서 메시지를 검색할 때 큐 관리자에서 생성됩니다. 원래 메시지의 메시지 데이터는 보고 메시지에 포함되지 않습니다.

메시지를 작업 단위의 일부로 검색하는 경우 작업 단위가 커밋될 때까지 보고서가 사용 불가능하도록 동일한 작업 단위 내에서 보고 메시지가 생성됩니다. 작업 단위가 백아웃되면 보고서가 송신되지 않습니다.

COD 보고서는 메시지 디스크립터의 MDFMT 필드가 FMDLH인 경우 생성되지 않습니다. 이로 인해 메시지를 배달할 수 없어 데드-레터 큐에 넣는 경우 COD 보고서가 생성되지 않습니다.

ROCOD는 목적지 큐가 XCF 큐인 경우 유효하지 않습니다.

ROCOD, ROCODD 및 ROCODF 중 둘 이상을 지정하지 마십시오.

## **ROCODD**

데이터가 포함된 필수 COD(전달 시 확인) 보고서입니다.

이는 기존 메시지에서 애플리케이션 메시지 데이터의 첫 번째 100바이트가 보고 메시지에 포함되는 것을 제외하고 ROCOD와 동일합니다. 원래 메시지에 하나 이상의 MQ 헤더 구조가 포함되면 100바이트의 애플리케이션 바이트 외에 해당 구조도 보고 메시지에 포함됩니다.

GMATM이 기존 메시지에 대해 MQGET 호출 시 지정되고 검색된 메시지가 잘리는 경우, 보고 메시지에 배치된 애플리케이션 메시지 데이터의 양이 최소 다음입니다.

- 기존 메시지의 길이
- 100바이트

ROCODD는 목적지 큐가 XCF 큐인 경우 유효하지 않습니다.

ROCOD, ROCODD 및 ROCODF 중 둘 이상을 지정하지 마십시오.



## **ROCODF**

전체 데이터가 포함된 필수 COD(전달 시 확인) 보고서입니다.

이는 기존 메시지에서 모든 애플리케이션 메시지 데이터가 보고 메시지에 포함되는 것을 제외하고 ROCOD와 동일합니다.

ROCODF는 목적지 큐가 XCF 큐인 경우 유효하지 않습니다.

ROCOD, ROCODD 및 ROCODF 중 둘 이상을 지정하지 마십시오.

**조치-알림 옵션:** 다음 옵션 중 하나 또는 둘 다를 지정하여 수신 애플리케이션이 긍정-조치 또는 부정-조치 보고 메시지를 송신하도록 요청할 수 있습니다.

## **ROPAN**

필수 긍정 조치 알림 보고서입니다.

이 유형의 보고서는 메시지를 검색하고 이에 대해 조치를 취하는 애플리케이션에 의해 생성됩니다. 이는 메시지에 요청된 조치가 성공적으로 수행되었음을 나타냅니다. 보고서를 생성하는 애플리케이션은 데이터가 보고서에 포함되는지 여부를 판별합니다.

메시지를 검색하는 애플리케이션에 이 요청을 전달하는 것 이외에, 큐 관리자는 이 옵션에 기반하여 조치를 수행하지 않습니다. 적절한 경우 보고서를 생성하는 것은 검색 애플리케이션이 담당합니다.

## **RONAN**

필수 부정 조치 알림 보고서입니다.

이 유형의 보고서는 메시지를 검색하고 이에 대해 조치를 취하는 애플리케이션에 의해 생성됩니다. 이는 메시지에 요청된 조치가 성공적으로 수행되지 않았음을 나타냅니다. 보고서를 생성하는 애플리케이션은 데이터가 보고서에 포함되는지 여부를 판별합니다. 예를 들어, 요청이 수행될 수 없는 이유를 나타내는 일부 데이터를 포함하는 것이 바람직할 수 있습니다.

메시지를 검색하는 애플리케이션에 이 요청을 전달하는 것 이외에, 큐 관리자는 이 옵션에 기반하여 조치를 수행하지 않습니다. 적절한 경우 보고서를 생성하는 것은 검색 애플리케이션이 담당합니다.

긍정 조치에 해당하는 조건 및 부정 조치에 해당하는 조건을 판별하는 것은 애플리케이션이 담당합니다. 그러나, 요청이 부분적으로만 수행된 경우 PAN 보고서가 아니라 NAN 보고서가 요청 시 생성되도록 권장됩니다. 또한 모든 가능한 조건이 긍정 조치나 부정 조치에 해당하지만 둘 다는 아니도록 권장됩니다.

**메시지-ID 옵션:** 다음 옵션 중 하나를 지정하여 보고 메시지(또는 응답 메시지)의 MDMID가 설정되는 방식을 제어할 수 있습니다.

## **RONMI**

새 메시지 ID입니다.

이는 기본 조치이며, 보고서 또는 응답이 이 메시지의 결과로 생성되는 경우 새 MDMID가 보고 또는 응답 메시지에 대해 생성됨을 나타냅니다.

## **ROPMI**

메시지 ID를 전달합니다.

보고서 또는 응답이 이 메시지의 결과로 생성되는 경우, 이 메시지의 MDMID가 보고 또는 응답 메시지의 MDMID에 복사됩니다.

발행 메시지의 MsgId는 발행물의 사본을 수신하는 각 구독자마다 다르므로, 보고 또는 응답 메시지에 복사된 MsgId가 각각 다릅니다.

이 옵션이 지정되지 않은 경우 RONMI가 가정됩니다.

**상관-ID 옵션:** 다음 옵션 중 하나를 지정하여 보고 메시지(또는 응답 메시지)의 MDCID가 설정되는 방식을 제어할 수 있습니다.

## **ROCMTC**

상관 ID에 메시지 ID를 복사합니다.

이는 기본 조치이며, 보고서 또는 응답이 이 메시지의 결과로 생성되는 경우 이 메시지의 MDMID가 보고 또는 응답 메시지의 MDCID에 복사됨을 나타냅니다.

발행 메시지의 MsgId는 발행물의 사본을 수신하는 각 구독자마다 다르므로 보고 또는 응답 메시지의 CorrelId에 복사된 MsgId가 각각 다릅니다.

#### **ROPCI**

상관 ID를 전달합니다.

보고서 또는 응답이 이 메시지의 결과로 생성되는 경우, 이 메시지의 MDCID가 보고 또는 응답 메시지의 MDCID에 복사됩니다.

발행 메시지의 MDCID는 SOSCID 옵션을 사용하고 MQSD의 SCDIC 필드를 CINONE로 설정할 때까지 구독자에 특정합니다. 따라서, 보고 또는 응답 메시지의 MDCID에 복사된 MDCID가 각각 다를 수 있습니다.

이 옵션이 지정되지 않은 경우 ROCMTC가 가정됩니다.

요청에 응답하거나 보고 메시지를 생성하는 서버는 ROPMI 또는 ROPCI 옵션이 기존 메시지에 설정되었는지 여부를 확인하도록 권장됩니다. 그러한 경우, 서버는 해당 옵션에 대해 설명된 조치를 수행해야 합니다. 둘 다 설정되지 않은 경우, 서버는 해당하는 기본 조치를 수행해야 합니다.

: 다음 옵션 중 하나를 지정하여 목적지 큐에 전달될 수 없을 때 기존 메시지의 배치를 제어할 수 있습니다. 이러한 옵션은 송신 애플리케이션에서 요청된 경우 예외 보고 메시지가 생성되도록 초래하는 해당 상황에만 적용됩니다. 애플리케이션은 예외 보고서 요청과 상관없이 배치 옵션을 설정할 수 있습니다.

#### **RODLQ**

데드-레터 큐에 메시지를 배치합니다.

이는 기본 조치이며, 메시지가 목적지 큐에 전달될 수 없는 경우 메시지가 데드-레터 큐에 배치되어야 함을 나타냅니다. 이는 다음 상황에서 발생합니다.

- 기존 메시지를 넣은 애플리케이션이 MQPUT 또는 MQPUT1 호출에서 리턴된 이유 코드로 문제점에 대해 동기적으로 알림을 받을 수 없는 경우입니다. 송신자에 의해 요청된 경우, 예외 보고 메시지가 생성됩니다.
- 기존 메시지를 넣은 애플리케이션을 토픽에 넣는 경우  
송신자가 요청한 경우 예외 보고 메시지가 생성됩니다.

#### **RODISC**

메시지 제거입니다.

이는 목적지 큐에 전달될 수 없는 경우 메시지가 제거되어야 함을 나타냅니다. 이는 다음 상황에서 발생합니다.

- 기존 메시지를 넣은 애플리케이션이 MQPUT 또는 MQPUT1 호출에서 리턴된 이유 코드로 문제점에 대해 동기적으로 알림을 받을 수 없는 경우입니다. 송신자에 의해 요청된 경우, 예외 보고 메시지가 생성됩니다.
- 기존 메시지를 넣은 애플리케이션을 토픽에 넣는 경우  
송신자가 요청한 경우 예외 보고 메시지가 생성됩니다.

기존 메시지가 데드-레터 큐에 배치되지 않고서 송신자에게 기존 메시지를 리턴해야 하는 경우, 송신자는 ROEXCF가 포함된 RODISC를 지정해야 합니다.

**기본 옵션:** 보고서 옵션이 필요하지 않은 경우 다음을 지정할 수 있습니다.

#### **RONONE**

보고서가 필요하지 않습니다.

이 값은 기타 옵션이 지정되지 않았음을 나타내는 데 사용될 수 있습니다. RONONE은 프로그램 문서를 지원하기 위해 정의됩니다. 이 옵션은 기타와 함께 사용하기 위한 용도가 아니지만, 해당 값이 0이므로 해당 사용을 감지할 수 없습니다.

#### **일반 정보:**

1. 필요한 모든 보고서 유형은 원래 메시지를 전송하는 애플리케이션에 의해 특별히 요청되어야 합니다. 예를 들어, COA 보고서는 요청되었으나 예외 보고서는 요청되지 않은 경우 메시지가 목적지 큐에 배치되면 COA 보고서가 생성되지만 메시지가 도달할 때 목적지 큐가 가득 찬 경우 예외 보고서가 생성되지 않습니다.

다. MDREP 옵션이 설정되지 않은 경우, 보고 메시지가 큐 관리자 또는 메시지 채널 에이전트(MCA)에서 생성되지 않습니다.

로컬 큐 관리자가 보고서 옵션을 인식하지 못하는 경우에도 일부 보고서 옵션을 지정할 수 있습니다. 이는 옵션을 목적지 큐 관리자가 처리하는 경우 유용합니다. 자세한 내용은 [1365 페이지의 『IBM i의 보고 옵션 및 메시지 플래그』](#)의 내용을 참조하십시오.

보고 메시지가 요청되는 경우, 보고서가 송신되어야 하는 큐의 이름이 MDRQ 필드에 지정되어야 합니다. 보고 메시지가 수신될 때, 보고서의 네이치는 메시지 디스크립터에서 MDFB 필드를 검사하여 판별할 수 있습니다.

2. 보고 메시지를 생성하는 큐 관리자 또는 MCA가 응답 큐에 보고 메시지를 넣을 수 없는 경우(예를 들어, 응답 큐 또는 전송 큐가 가득 찼으므로), 보고 메시지는 데드-레터 큐에 대신 배치됩니다. 이 또한 실패하거나 데드-레터 큐가 없는 경우 수행되는 조치는 보고 메시지의 유형에 따라 다릅니다.

- 보고 메시지가 예외 보고서인 경우, 예외 보고서가 생성되도록 하는 메시지가 전송 큐에 남아 있습니다. 그러면 메시지가 유실되지 않습니다.
- 모든 기타 보고서 유형의 경우, 보고 메시지가 제거되고 정상적으로 계속 처리됩니다. 이는 기존 메시지가 이미 안전하게 전달되었거나(COA 또는 COD 보고 메시지의 경우) 더 이상 관련되지 않으므로(만기 보고 메시지의 경우) 수행됩니다.

보고 메시지가 큐에 성공적으로 배치되었으면(목적지 큐 또는 중간 전송 큐), 메시지는 더 이상 특수 처리 대상이 아닙니다. 이는 다른 메시지처럼 처리됩니다.

3. 보고서가 생성될 때, MDRQ 큐가 열리며 다음 경우를 제외하고 보고서의 원인이 되는 메시지의 MQMD의 MDUID 권한을 통해 보고 메시지를 넣습니다.

- 보고서의 원인이 되는 메시지를 넣으려고 시도했을 때 MCA가 사용한 권한에 상관없이, 수신 MCA에서 생성된 예외 보고서를 넣습니다. CDPA 채널 속성은 사용된 사용자 ID를 판별합니다.
- 큐 관리자가 생성한 COA 보고서는 보고서 발생 원인이 되는 메시지를 보고서를 생성한 큐 관리자에 넣을 때 사용한 권한을 사용하여 넣습니다. 예를 들어, MCA의 사용자 ID를 통해 수신 MCA에서 메시지를 넣은 경우, 큐 관리자가 MCA의 사용자 ID를 통해 COA 보고서를 넣습니다.

보고서를 생성하는 애플리케이션은 일반적으로 응답을 생성하는 데 사용했을 것과 동일한 권한을 사용해야 합니다. 이는 일반적으로 기존 메시지에서 사용자 ID의 권한이어야 합니다.

보고서가 원격 목적지로 이동해야 하는 경우, 송신자 및 수신자는 기타 메시지에 대해 수행하는 동일한 방식으로 이를 허용할지 여부를 결정할 수 있습니다.

4. 데이터가 있는 보고 메시지가 요청된 경우:

- 보고 메시지는 기존 메시지의 송신자가 요청한 데이터의 양으로 항상 생성됩니다. 보고 메시지가 응답 큐에 너무 큰 경우, 이전에 설명된 처리가 발생합니다. 보고 메시지는 응답 큐에 맞게 잘리지 않습니다.
- 기존 메시지의 MDFMT가 FMXQH인 경우, 보고서에 포함된 데이터가 MQXQH를 포함하지 않습니다. 보고서 데이터는 원래 메시지의 MQXQH 위에 있는 데이터의 첫 번째 바이트부터 시작됩니다. 이는 큐가 전송 큐인이 여부에 상관없이 발생합니다.

5. COA, COD 또는 만기 보고 메시지가 응답 큐에 수신되는 경우 기존 메시지가 적절한 대로 도달했고 전달되었거나 만기되었음이 확실합니다. 그러나 이러한 보고 메시지 중 하나 이상이 요청되었으나 수신되지 않은 경우 다음 중 하나가 발생했을 수 있으므로 반대로 가정할 수 없습니다.

- a. 링크가 끊어져 보고 메시지가 보류됩니다.
- b. 블로킹 조건이 중간 전송 큐 또는 응답 큐에 있으므로 보고 메시지가 보류됩니다(예를 들어, 큐가 가득 차거나 Put에 대해 상속됨).
- c. 보고 메시지가 데드-레터 큐에 있습니다.
- d. 큐 관리자가 보고 메시지를 생성하려고 시도할 때, 적절한 큐에 이를 넣을 수 없고 데드-레터 큐에도 이를 넣을 수 없어서, 보고 메시지가 생성될 수 없습니다.
- e. 보고되는 조치(도달, 전달 또는 만기) 및 해당하는 보고 메시지 사이에 큐 관리자의 실패가 발생했습니다. COD 보고 메시지가 동일한 작업 단위 내에 생성되어, 애플리케이션이 작업 단위 내에서 기존 메시지를 검색하는 경우, 이는 COD 보고 메시지에 대해 발생하지 않습니다.

예외 보고 메시지는 이전의 이유 1, 2 및 3으로 동일한 방식으로 보류될 수 있습니다. 그러나 MCA가 예외 보고 메시지를 생성할 수 없는 경우(보고 메시지를 응답 큐 또는 데드-레터 큐에 넣을 수 없음), 기존 메시지가 송신자의 전송 큐에 남아 있고 채널이 닫힙니다. 이는 보고 메시지가 채널의 송신 또는 수신 끝에서 생성되는지 여부에 상관없이 발생합니다.

6. 기존 메시지가 임시로 차단되지만(결과적으로, 예외 보고 메시지가 생성되고 기존 메시지를 데드-레터 큐에 넣음) 장애물이 정리된 후 애플리케이션이 데드-레터 큐에서 기존 메시지를 읽어서 해당 목적지에 다시 이를 넣는 경우, 다음이 발생할 수 있습니다.

- 예외 보고 메시지가 생성된 경우에도, 기존 메시지가 결국 해당 목적지에 성공적으로 도달합니다.
- 기존 메시지가 나중에 다른 장애물을 발견할 수 있으므로, 둘 이상의 예외 보고 메시지가 하나의 기존 메시지에 관해 생성됩니다.

#### 토픽에 넣을 때 보고 메시지:

1. 메시지를 토픽에 넣는 경우, 보고서가 생성될 수 있습니다. 이 메시지는 토픽에 모든 구독자에게 송신되며, 이는 0, 하나 또는 다수일 수 있습니다. 이는 결과적으로 다수의 보고 메시지가 생성될 수 있어서 보고서 옵션을 사용하도록 선택할 때 고려되어야 합니다.
2. 토픽에 메시지를 넣을 때, 메시지의 사본이 제공되는 다수의 목적지 큐가 있을 수 있습니다. 이러한 목적지 큐 중 일부에 큐가 가득 차는 등의 문제점이 있는 경우, MQPUT의 성공적인 완료는 NPMMSGDLV 또는 PMSGDLV의 설정에 따라 다릅니다(메시지의 지속성에 따라). 목적지 큐에 대한 메시지 전달이 성공적이어야 하도록 설정된 경우(예를 들어, 지속형 구독자에 대한 지속 메시지가이며 PMSGDLV가 ALL 또는 ALLDUR로 설정됨), 다음 기준 중 하나가 충족되면 성공으로 정의됩니다.

- 구독자 큐에 성공적으로 넣음
- 구독자 큐가 메시지를 수용할 수 없는 경우 RODLQ의 사용 및 데드-레터 큐에 대한 성공적 넣기
- 구독자 큐가 메시지를 수용할 수 없는 경우 RODISC의 사용입니다.

#### 메시지 세그먼트에 대한 보고 메시지:

1. 세그먼트화가 허용되어 있는 메시지에 대해 보고 메시지를 요청할 수 있습니다(MFSEGA 플래그의 설명 참조). 큐 관리자가 메시지를 세그먼트화하는 것이 필요하다는 것을 발견하면 이후에 관련 조건이 발생하는 각 세그먼트에 대해 보고 메시지가 생성될 수 있습니다. 따라서 애플리케이션은 요청된 보고 메시지의 각 유형마다 여러 보고 메시지를 수신할 준비가 되어 있어야 합니다. 보고 메시지의 MDGID 필드를 사용하여 기존 메시지의 그룹 ID와 여러 보고서를 상관시키고, MDFB 필드를 사용하여 각 보고 메시지의 유형을 식별할 수 있습니다.
2. GMLOGO를 사용하여 세그먼트에 대한 보고 메시지를 검색하는 경우, 여러 유형의 보고서가 연속 MQGET 호출에서 리턴될 수 있음을 인지하십시오. 예를 들어, COA 및 COD 보고서 둘 다 큐 관리자에서 세그먼트화된 메시지에 대해 요청되는 경우, 보고 메시지에 대한 MQGET 호출이 예상치 못한 방식으로 인터리브된 COA 및 COD 보고 메시지를 리턴할 수 있습니다. 이는 GMCMPM 옵션을 사용하여 막을 수 있습니다(선택적으로 GMATM 사용). GMCMPM으로 인해 큐 관리자는 동일한 보고 유형이 있는 보고 메시지를 리어셈블링합니다. 예를 들어, 첫 번째 MQGET 호출은 기존 메시지와 관련된 모든 COA 메시지를 리어셈블링하고, 두 번째 MQGET 호출은 모든 COD 메시지를 리어셈블링할 수 있습니다. 처음 리어셈블링되는 것은 큐에서 처음 발생하는 보고 메시지의 유형에 따라 다릅니다.
3. 직접 세그먼트를 넣는 애플리케이션은 각 세그먼트에 대해 서로 다른 보고서 옵션을 지정할 수 있습니다. 그러나, 다음을 참고해야 합니다.
  - 세그먼트가 GMCMPM 옵션을 통해 검색되는 경우, 첫 번째 세그먼트의 보고서 옵션만 큐 관리자에서 인정됩니다.
  - 세그먼트가 하나씩 검색되고 대부분 ROCOD\* 옵션 중 하나가 있지만 하나 이상의 세그먼트는 아닌 경우, 단일 MQGET 호출로 보고 메시지를 검색하는 데 GMCMPM 옵션을 사용하거나 모든 보고 메시지 도달 시기를 감지하는 데 GMASGA 옵션을 사용할 수 없게 됩니다.
4. MQ 네트워크에서, 큐 관리자에 다양한 기능이 있을 수 있습니다. 세그먼트의 보고 메시지가 세그먼트화를 지원하지 않는 MCA 또는 큐 관리자에서 생성되는 경우, 큐 관리자 또는 MCA는 기본적으로 보고 메시지에서 필수 세그먼트 정보를 포함하지 않으며 이로 인해 보고서가 생성되도록 한 기존 메시지를 식별하는 것이 어려울 수 있습니다. 이는 보고 메시지와 함께 데이터를 요청하여 막을 수 있습니다. 즉, 적절한 RO\*D 또는 RO\*F 옵션을 지정하여 막을 수 있습니다. 그러나 RO\*D가 지정된 경우, 보고 메시지가 세그먼트화를 지원하지 않는 MCA 또는 큐 관리자에서 생성되면, 100바이트 미만인 애플리케이션 메시지 데이터가 보고 메시지를 검색하는 애플리케이션에 리턴될 수 있습니다.

보고 메시지에 대한 메시지 디스크립터의 콘텐츠: 큐 관리자 또는 메시지 채널 에이전트(MCA)가 보고 메시지를 생성하는 경우, 메시지 디스크립터의 필드를 다음 값으로 설정한 후 정상적인 방법으로 메시지를 넣습니다.

표 708. 보고 메시지가 시스템에서 생성되는 경우에 MQMD 필드에 대해 사용되는 값

MQMD의 필드	사용된 값
MDSID	MDSIDV
MDVER	MDVER2
MDREP	RONONE
MDMT	MTRPRT
MDEXP	EIULIM
MDFB	보고서의 네이처에 적절한 대로(FBCOA, FBCOD, FBEXP 또는 RC* 값)
MDENC	원래 메시지 디스크립터로부터 복사됩니다.
MDCSI	원래 메시지 디스크립터로부터 복사됩니다.
MDFMT	원래 메시지 디스크립터로부터 복사됩니다.
MDPRI	원래 메시지 디스크립터로부터 복사됩니다.
MDPER	원래 메시지 디스크립터로부터 복사됩니다.
MDMID	원래 메시지 디스크립터의 보고서 옵션에 의해 지정됨
MDCID	원래 메시지 디스크립터의 보고서 옵션에 의해 지정됨
MDBOC	0
MDRQ	공백
MDRM	큐 관리자의 이름
MDUID	PMPASI 옵션에서 설정된 대로
MDACC	PMPASI 옵션에서 설정된 대로
MDAID	PMPASI 옵션에서 설정된 대로
MDPAT	ATQM 또는 메시지 채널 에이전트에 대해 적절하게 사용됨
MDPAN	큐 관리자 이름 또는 메시지 채널 에이전트 이름의 첫 번째 28바이트입니다. IMS 브릿지에서 생성된 보고 메시지의 경우, 이 필드에는 메시지와 관련된 IMS 시스템의 XCF 그룹 이름 및 XCF 멤버 이름이 포함됩니다.
MDPD	보고 메시지가 송신된 날짜
MDPT	보고 메시지를 송신한 시간
MDAOD	공백
MDGID	원래 메시지 디스크립터로부터 복사됩니다.
MDSEQ	원래 메시지 디스크립터로부터 복사됩니다.
MDOFF	원래 메시지 디스크립터로부터 복사됩니다.
MDMFL	원래 메시지 디스크립터로부터 복사됩니다.
MDOLN	OLUNDF가 아닌 경우 기존 메시지 디스크립터에서 복사되고, 그렇지 않으면 기존 메시지 데이터의 길이로 설정됨

보고서를 생성하는 애플리케이션은 다음을 제외하고 유사한 값을 설정하도록 권장됩니다.

- MDRM 필드는 공백으로 설정될 수 있습니다(메시지를 넣을 때 큐 관리자가 로컬 큐 관리자의 이름으로 이를 변경함).

• 컨텍스트 필드는 응답에 사용되었을 옵션을 통해 설정되어야 합니다(일반적으로 PMPASI).

**보고서 필드 분석:** MDREP 필드에 하위 필드가 포함됩니다. 이로 인해, 메시지의 송신자가 특정한 보고서를 요청했는지 여부를 확인해야 하는 애플리케이션이 [1367 페이지의 『IBM i에서 보고 필드 분석』](#)에 설명된 기술 중 하나를 사용해야 합니다.

이 필드는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 초기값은 RONONE입니다.

### MDRM(48바이트 문자열)

응답 큐 관리자의 이름입니다.

이는 응답 메시지 또는 보고 메시지가 송신되어야 하는 큐 관리자의 이름입니다. MDRQ는 이 큐 관리자에 정의되는 큐의 로컬 이름입니다.

MDRM 필드가 공백인 경우, 로컬 큐 관리자는 해당 큐 정의에서 **MDRQ** 이름을 검색합니다. 원격 큐의 로컬 정의가 이 이름으로 존재하는 경우, 전송된 메시지의 **MDRM** 값이 원격 큐의 정의에서 **RemoteQMgrName** 속성의 값으로 바뀌고, 이 값은 수신 애플리케이션이 메시지에 대한 MQGET 호출을 실행할 때 메시지 디스크립터에 리턴됩니다. 원격 큐의 로컬 정의가 없는 경우, 메시지와 함께 전송되는 MDRM이 로컬 큐 관리자의 이름입니다.

이름이 지정된 경우, 후미 문자 공백을 포함할 수 있습니다. 첫 번째 널 문자 및 뒤따르는 문자는 공백으로 처리됩니다. 그러나 그렇지 않으면 이름이 큐 관리자의 이름 지정 규칙을 충족하거나 이 이름이 송신 큐 관리자에 알려져 있는지 확인되지 않습니다. **MDRM**이 전송된 메시지에서 바뀌는 경우, 이는 전송된 이름에 대해서도 적용됩니다.

응답 대상 큐가 필요하지 않은 경우, MDRM 필드가 공백으로 설정되도록 권장됩니다(선택되지 않아도). 필드는 초기화 해제되어 남지 않아야 합니다.

MQGET 호출의 경우, 큐 관리자는 항상 필드 길이까지 공백으로 채워진 이름을 리턴합니다.

이 필드는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 길이는 LNQMNM에서 제공합니다. 이 필드의 초기값은 48개의 빈 문자입니다.

### MDRQ(48바이트 문자열)

응답 큐의 이름입니다.

이는 메시지에 대해 Get 요청을 실행한 애플리케이션이 MTRPLY 및 MTRPRT 메시지를 송신해야 하는 메시지 큐의 이름입니다. 이름은 MDRM에서 식별된 큐 관리자에 정의되는 큐의 로컬 이름입니다. 메시지를 넣을 때 송신 큐 관리자가 이를 확인하지 않아도 이 큐는 모델 큐가 아니어야 합니다.

MQPUT 및 MQPUT1 호출의 경우, MDMT 필드에 값 MTRQST가 있거나 보고 메시지가 MDREP 필드에서 요청되면 이 필드가 공백이 아니어야 합니다. 그러나 지정된(또는 대체된) 값은 메시지 유형에 상관없이 메시지에 대해 Get 요청을 실행하는 애플리케이션으로 전달됩니다.

MDRM 필드가 공백이면 로컬 큐 관리자가 소유하고 있는 큐 정의에서 MDRQ 이름을 검색합니다. 리모트 큐의 로컬 정의가 이 이름으로 존재하는 경우, 전송 메시지의 MDRQ 값이 리모트 큐의 정의에서 **RemoteQName** 속성의 값으로 바뀌고, 이 값은 수신 애플리케이션이 메시지에 대해 MQGET 호출을 실행할 때 메시지 디스크립터에 리턴됩니다. 리모트 큐의 로컬 정의가 없는 경우, MDRQ가 변경되지 않습니다.

이름이 지정된 경우, 후미 문자 공백을 포함할 수 있습니다. 첫 번째 널 문자 및 뒤따르는 문자는 공백으로 처리됩니다. 그러나 그렇지 않으면 이름이 큐의 이름 지정 규칙을 충족하는지 확인되지 않습니다. MDRQ가 전송된 메시지에서 바뀌는 경우, 이는 전송된 이름에 대해서도 적용됩니다. 상황에 따라, 이름이 지정되었지만 확인됩니다.

응답 대상 큐가 필요하지 않은 경우, MDRQ 필드가 공백으로 설정되도록 권장됩니다(선택되지 않아도). 필드는 초기화 해제되어 남지 않아야 합니다.

MQGET 호출의 경우, 큐 관리자는 항상 필드 길이까지 공백으로 채워진 이름을 리턴합니다.

보고 메시지를 요구하는 메시지를 전달할 수 없고 보고 메시지도 지정된 큐에 전달할 수 없는 경우, 기존 메시지 및 보고 메시지 둘 다 데드-레터(미발송-메시지) 큐로 이동합니다. [1334 페이지의 『IBM i의 큐 관리자에 대한 속성』](#)에 설명된 **DeadLetterQName** 속성을 참조하십시오.

이 필드는 MQGET 호출의 출력 필드이며 MQPUT 및 MQPUT1 호출의 입력 필드입니다. 이 필드의 길이는 LNQN으로 제공됩니다. 이 필드의 초기값은 48개의 빈 문자입니다.

### MDSEQ(10자리 부호 표시 정수)

그룹 내의 논리 메시지의 순서 번호.

순서 번호는 1에서 시작하여, 그룹의 각 새 논리 메시지마다 1씩 최대 999 999 999까지 증가합니다. 그룹에 없는 물리적 메시지는 순서 번호가 1입니다.

이 필드는 다음과 같은 경우 MQPUT 또는 MQGET 호출 시 애플리케이션에서 설정될 필요가 없습니다.

- MQPUT 호출 시, PMLOGO가 지정됩니다.
- MQGET 호출에서 MOSEQN이 지정되지 않습니다.

이는 보고 메시지가 아닌 메시지에 대해 이러한 호출을 사용하는 권장 방법입니다. 그러나, 애플리케이션에 추가 제어가 필요하거나 호출이 MQPUT1인 경우, 애플리케이션은 MDSEQ가 적절한 값으로 설정되는지 확인해야 합니다.

MQPUT 및 MQPUT1 호출에 대한 입력에서 큐 관리자는 표 1에 설명된 값을 사용합니다. MQPUT 및 MQPUT1 호출에서 출력 시 큐 관리자는 이 필드를 메시지와 함께 전송된 값으로 설정합니다.

MQGET 호출에 대한 입력에서 큐 관리자는 표 1에 설명된 값을 사용합니다. MQGET 호출에서 출력 시 큐 관리자는 이 필드를 검색된 메시지의 값으로 설정합니다.

이 필드의 초기값은 1입니다. MDVER가 MDVER2 미만이면 이 필드가 무시됩니다.

### MDSID(4바이트 문자열)

구조 ID입니다.

값은 다음과 같아야 합니다.

#### MDSIDV

메시지 디스크립터 구조의 ID.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MDSIDV입니다.

### MDUID(12바이트 문자열)

사용자 ID.


메시지의 ID 컨텍스트의 부분입니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어](#)를 참조하십시오.

MDUID는 메시지를 시작한 애플리케이션의 사용자 ID를 지정합니다. 큐 관리자는 이 정보를 문자 데이터로 처리하지만 그 형식은 정의하지 않습니다.

메시지를 수신한 후 후속 MQOPEN 또는 MQPUT1 호출의 **OBJDSC** 매개변수에 있는 ODAU 필드에서 MDUID를 사용할 수 있으므로 열기를 수행하는 애플리케이션 대신 MDUID 사용자에 대해 권한 검사가 수행됩니다.

큐 관리자가 MQPUT 또는 MQPUT1 호출에 대해 이 정보를 생성하는 경우 환경에서 판별된 사용자 ID를 사용합니다.

사용자 ID가 환경에서 결정되는 경우:

-  z/OS에서 큐 관리자는 다음을 사용합니다.
  - 배치의 경우, JES JOB 카드 또는 시작 태스크에서 사용자 ID
  - TSO의 경우, 로그인 사용자 ID
  - CICS의 경우, 태스크와 연관된 사용자 ID
  - IMS의 경우 사용자 ID는 애플리케이션 유형에 따라 다릅니다.
  - 다음의 경우:
    - Nonmessage BMP 리전
    - Nonmessage IFP 리전



- 성공적인 GU 호출을 발행하지 않은 메시지 BMP 및 메시지 IFP 영역


큐 관리자가 JES JOB 카드 영역 또는 TSO 사용자 ID로부터 사용자 ID를 사용합니다. 공백이거나 널인 경우, 이는 프로그램 스펙 블록(PSB)의 이름을 사용합니다.

- 다음의 경우:

- 성공적인 GU 호출을 발행한 메시지 BMP 및 메시지 IFP 영역
- MPP 리전

큐 관리자가 다음 중 하나를 사용합니다.


- 메시지와 연관된 사인온한 사용자 ID
- 논리 터미널(LTERM) 이름
- 영역 JES JOB 카드의 사용자 ID
- TSO 사용자 ID
- PSB 이름

-  IBM i에서, 큐 관리자는 애플리케이션 작업과 연관된 사용자 프로파일의 이름을 사용합니다.

-  Linux and AIX에서 큐 관리자는 다음을 사용합니다.

- 애플리케이션의 로그인 이름
- 사용 가능한 로그인이 없는 경우, 프로세스의 유효한 사용자 ID
- 애플리케이션이 CICS 트랜잭션인 경우, 트랜잭션과 연관된 사용자 ID

- VSE/ESA에서, 이는 예약된 필드입니다.

-  Windows에서, 큐 관리자는 로그인된 사용자 이름의 첫 번째 12자를 사용합니다.

MQPUT 및 MQPUT1 호출의 경우, 이는 PMSETI 또는 PMSETA가 **PMO** 매개변수에 지정된 경우 입력/출력 필드입니다. 필드 내에서 널 문자 이후의 정보는 제거됩니다. 널 문자 및 다음 문자는 큐 관리자에 의해 공백으로 변환됩니다. PMSETI 또는 PMSETA가 지정되지 않은 경우, 이 필드는 입력에서 무시되고 출력 전용 필드입니다.

MQPUT 또는 MQPUT1 호출이 성공적으로 완료되면, 이 필드에는 큐에 넣은 경우 메시지와 함께 전송된 MDUID가 포함됩니다. 이는 보유한 경우 메시지와 함께 보관되지만(보유된 발행물에 대한 자세한 내용은 PMRET의 설명 참조), 구독자에게 송신된 모든 발행물에서 MDUID를 대체할 값을 구독자가 제공하므로 메시지가 구독자에게 발행물로 송신될 때 MDUID로 사용되지 않는 MDUID의 값입니다. 메시지에 컨텍스트가 없는 경우 필드 전체가 공백입니다.

MQGET 호출의 출력 필드입니다. 이 필드의 길이는 LNUID에서 제공됩니다. 이 필드의 초기값은 12개의 공백 문자입니다.

### MDVER(10자리 부호 표시 정수)

구조 버전 번호입니다.

값은 다음 중 하나여야 합니다.

#### MDVER1

버전-1 메시지 디스크립터 구조입니다.

#### MDVER2

버전-2 메시지 디스크립터 구조입니다.

**참고:** 버전-2 MQMD가 사용되는 경우, 큐 관리자는 애플리케이션 메시지 데이터의 시작 시 제공될 수 있는 임의의 MQ 헤더 구조에서 추가 검사를 수행합니다. 자세한 내용은 MQPUT 호출의 사용 참고를 참조하십시오.

최신 버전의 구조에만 있는 필드는 필드의 설명에서 최신 필드로 식별됩니다. 다음 상수는 현재 버전의 버전 번호를 지정합니다.



**MDVERC**

메시지 디스크립터 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MDVER1입니다.

**초기값**

표 709. MQMD의 필드		
필드 이름	상수의 이름	상수의 값
MDSID	MDSIDV	'MD--'
MDVER	MDVER1	1
MDREP	RONONE	0
MDMT	MTDGRM	8
MDEXP	EIULIM	-1
MDFB	FBNONE	0
MDENC	ENNAT	환경에 따라 다름
MDCSI	CSQM	0
MDFMT	FMNONE	공백
MDPRI	PRQDEF	-1
MDPER	PEQDEF	2
MDMID	MINONE	널
MDCID	CINONE	널
MDBOC	없음	0
MDRQ	없음	공백
MDRM	없음	공백
MDUID	없음	공백
MDACC	ACNONE	널
MDAID	없음	공백
MDPAT	ATNCON	0
MDPAN	없음	공백
MDPD	없음	공백
MDPT	없음	공백
MDAOD	없음	공백
MDGID	GINONE	널
MDSEQ	없음	1
MDOFF	없음	0
MDMFL	MFNONE	0
MDOLN	OLUNDF	-1

표 709. MQMD의 필드 (계속)

필드 이름	상수의 이름	상수의 값
<b>참고:</b>		
1. ~ 기호는 단일 공백 문자를 나타냅니다.		

## RPG 선언

```

D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQMD Structure
D*
D* Structure identifier
D MDSID          1      4    INZ('MD ')
D* Structure version number
D MDVER          5      8I 0 INZ(1)
D* Options for report messages
D MDREP          9     12I 0 INZ(0)
D* Message type
D MDMT          13     16I 0 INZ(8)
D* Message lifetime
D MDEXP         17     20I 0 INZ(-1)
D* Feedback or reason code
D MDFB          21     24I 0 INZ(0)
D* Numeric encoding of message data
D MDENC         25     28I 0 INZ(273)
D* Character set identifier of messagedata
D MDCSI         29     32I 0 INZ(0)
D* Format name of message data
D MDFMT         33     40    INZ(' ')
D* Message priority
D MDPRI         41     44I 0 INZ(-1)
D* Message persistence
D MDPER         45     48I 0 INZ(2)
D* Message identifier
D MDMID         49     72    INZ(X'00000000000000-
D                      00000000000000000000-
D                      000000000000')
D* Correlation identifier
D MDCID         73     96    INZ(X'00000000000000-
D                      00000000000000000000-
D                      000000000000')
D* Backout counter
D MDBOC         97    100I 0 INZ(0)
D* Name of reply queue
D MDRQ         101    148    INZ
D* Name of reply queue manager
D MDRM         149    196    INZ
D* User identifier
D MDUID         197    208    INZ
D* Accounting token
D MDACC        209    240    INZ(X'00000000000000-
D                      00000000000000000000-
D                      00000000000000000000-
D                      000000')
D* Application data relating to identity
D MDAID        241    272    INZ
D* Type of application that put the message
D MDPAT        273    276I 0 INZ(0)
D* Name of application that put the message
D MDPAN        277    304    INZ
D* Date when message was put
D MDPD        305    312    INZ
D* Time when message was put
D MDPT        313    320    INZ
D* Application data relating to origin
D MDAOD        321    324    INZ
D* Group identifier
D MDGID        325    348    INZ(X'00000000000000-
D                      00000000000000000000-
D                      000000000000')
D* Sequence number of logical message within group
D MDSEQ        349    352I 0 INZ(1)
D* Offset of data in physical message from start of logical message

```

```

D MDOFF 353 356I 0 INZ(0)
D* Message flags
D MDMFL 357 360I 0 INZ(0)
D* Length of original message
D MDOLN 361 364I 0 INZ(-1)

```

## IBM i IBM i의 MQMDE(메시지 디스크립터 확장자)

### 개요

**목적:** MQMDE 구조는 종종 애플리케이션 메시지 데이터에 선행하여 발생하는 데이터를 설명합니다. 구조는 버전-1 MQMD가 아닌 버전-2 MQMD에 있는 MQMD 필드를 포함합니다.

**형식 이름:** FMMDE.

**문자 세트 및 인코딩:** MQMDE의 데이터는 **CodedCharSetId** 큐 관리자 속성에 의해 지정된 문자 세트 및 C 프로그래밍 언어에 대한 ENNAT에 의해 지정된 로컬 큐 관리자의 인코딩에 있어야 합니다.

MQMDE의 문자 세트 및 인코딩은 *MDCSI* 및 *MDENC* 필드에 설정되어야 합니다.

- MQMD(MQMDE 구조가 메시지 데이터의 시작에 있는 경우) 또는
- MQMDE 구조에 선행하는 헤더 구조(다른 모든 경우).

MQMDE가 큐 관리자의 문자 세트 및 인코딩에 없는 경우 MQMDE는 승인되지만 적용되지 않습니다 즉, MQMDE는 메시지 데이터로 처리됩니다.

**사용법:** 일반 애플리케이션은 버전-2 MQMD를 사용해야 합니다. 이 경우 MQMDE 구조가 발생하지 않습니다. 그러나 특수 애플리케이션 및 버전-1 MQMD를 계속 사용하는 애플리케이션은 경우에 따라 MQMDE가 발생할 수 있습니다. MQMDE 구조는 다음 환경에서 발생할 수 있습니다.

- MQPUT 및 MQPUT1 호출에 지정됨
- MQGET 호출로 리턴됨
- 전송 큐의 메시지에서
- [1103 페이지의 『MQPUT 및 MQPUT1 호출에 지정된 MQMDE』](#)
- [1104 페이지의 『MQGET 호출로 리턴된 MQMDE』](#)
- [1104 페이지의 『전송 큐 메시지의 MQMDE』](#)
- [1104 페이지의 『필드』](#)
- [1106 페이지의 『초기값』](#)
- [1107 페이지의 『RPG 선언』](#)

### MQPUT 및 MQPUT1 호출에 지정된 MQMDE

애플리케이션이 버전-1 MQMD를 제공하는 경우 애플리케이션은 MQPUT 및 MQPUT1 호출에서 선택적으로 메시지 데이터에 MQMDE로 접두부를 붙이고 MQMD의 *MDFMT* 필드를 FMMDE로 설정하여 MQMDE가 존재하고 있음을 표시할 수 있습니다. 애플리케이션이 MQMDE를 제공하지 않는 경우 큐 관리자는 MQMDE의 필드에 대해 기본값 가정합니다. 큐 관리자가 사용하는 기본값은 구조의 초기값과 동일합니다([1106 페이지의 표 711 참조](#)).

애플리케이션이 버전-2 MQMD이고 MQMDE를 사용하는 애플리케이션 메시지 데이터를 앞에 붙이는 경우 구조는 [1103 페이지의 표 710](#)에 표시된 대로 처리됩니다.

표 710. MQMDE가 MQPUT 또는 MQPUT1에서 지정된 경우 큐 관리자 조치			
MQMD 버전	버전-2 필드의 값	MQMDE의 해당하는 필드의 값	큐 관리자에 의해 수행되는 조치
1	-	올바름	MQMDE가 적용됨
2	기본값	올바름	MQMDE가 적용됨
2	기본값이 아님	올바름	MQMDE가 메시지 데이터로 처리됨

표 710. MQMDE가 MQPUT 또는 MQPUT1에서 지정된 경우 큐 관리자 조치 (계속)			
MQMD 버전	버전-2 필드의 값	MQMDE의 해당하는 필드의 값	큐 관리자에 의해 수행되는 조치
1 또는 2	임의	올바르지 않음	호출이 적절한 이유 코드로 실패함
1 또는 2	임의	MQMDE가 잘못된 문자 세트 또는 인코딩이거나 지원되지 않는 버전 임	MQMDE가 메시지 데이터로 처리됨

특별한 경우가 하나 있습니다. 애플리케이션이 버전-2 MQMD를 사용하여 세그먼트인 넣기 메시지(즉, MFSEG 또는 MFLSEG 플래그가 설정됨) 및 MQMD의 형식 이름이 FMDLH인 경우 큐 관리자는 MQMDE 구조를 생성하고 MQDLH 구조 및 뒤에 오는 데이터 사이에 삽입합니다. 큐 관리자가 메시지로 유지하는 MQMD에서 버전-2 필드는 해당 기본값으로 설정됩니다.

버전 2 MQMD에 존재하지만 버전 1 MQMD에 존재하지 않는 몇몇 필드가 MQPUT 및 MQPUT1 호출에서 입/출력(I/O) 필드로 사용됩니다. 그러나 큐 관리자는 MQPUT 및 MQPUT1 호출의 출력에서 MQMDE의 동일 필드에는 아무 값도 리턴하지 않습니다. 애플리케이션은 이들 출력 값이 필요한 경우 버전-2 MQMD를 사용해야 합니다.

### MQGET 호출로 리턴된 MQMDE

MQGET 호출에서 애플리케이션이 버전-1 MQMD를 제공하는 경우 큐 관리자는 리턴된 메시지에 MQMDE로 접두부를 붙입니다. 단, MQMDE의 필드 중 하나 이상이 기본값이 아닌 값을 가진 경우에만 이를 수행합니다. MQMD의 *MDFMT* 필드를 값 FMMDE로 설정하여 MQMDE가 존재하고 있음을 표시합니다.

애플리케이션이 **BUFFER** 매개변수의 시작 부분에 MQMDE를 제공하는 경우 MQMDE는 무시됩니다. MQGET 호출에서 리턴 시 메시지의 MQMDE에 의해 대체되거나(하나가 필요한 경우) 애플리케이션 메시지 데이터에 의해 덮어씁니다(MQMDE가 필요하지 않은 경우).

MQMDE가 MQGET 호출로 리턴되는 경우 MQMDE의 데이터는 일반적으로 큐 관리자의 문자 세트 및 인코딩에 있습니다. 그러나 MQMDE는 다음의 경우 일부 다른 문자 세트 및 인코딩에 있을 수 있습니다.

- MQMDE는 MQPUT 또는 MQPUT1 호출의 데이터로 처리되었습니다(이를 발생시킬 수 있는 환경은 [1103 페이지](#)의 표 710 참조).
- 메시지는 TCP 연결로 연결된 리모트 큐 관리자로부터 수신되었고 수신되는 메시지 채널 에이전트(MCA)가 올바르게 설정되지 않았습(추가 정보는 [IBM MQ for IBM i 오브젝트의 보안](#) 참조).

### 전송 큐 메시지의 MQMDE

전송 큐의 메시지는 MQXQH 구조 앞에 붙이며 버전-1 MQMD 내에 포함합니다. 또한 MQMDE는 존재해야 하며 MQXQH 구조 및 애플리케이션 메시지 데이터 사이에 위치하지만 이는 MQMDE에 있는 하나 이상의 필드에 기본값이 아닌 값이 있는 경우에만 해당됩니다.

다른 IBM MQ 헤더 구조는 또한 MQXQH 구조 및 애플리케이션 메시지 데이터 사이에 발생할 수 있습니다. 예를 들어, 데드 레터 헤더 MQDLH가 존재하면 메시지는 세그먼트가 아닙니다. 순서는 다음과 같습니다.

- MQXQH(버전-1 MQMD 포함)
- MQMDE
- MQDLH
- 애플리케이션 메시지 데이터

### 필드

MQMDE 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 [알파벳순](#)으로 설명합니다.

#### MECSI(10자리의 부호 있는 정수)

MQMDE를 뒤에 오는 데이터의 문자 세트 ID.

MQMDE 구조 뒤에 오는 데이터의 문자 세트 ID를 지정합니다. MQMDE 구조 자체의 문자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 큐 관리자는 이 필드가 올바른지 검사하지 않습니다. 다음 특수 값을 사용할 수 있습니다.

#### **CSINHT**

이 구조의 문자 세트 ID를 상속합니다.

이 구조 다음에 오는 데이터의 문자 데이터는 이 구조와 동일한 문자 세트로 되어 있습니다.

큐 관리자가 구조의 실제 문자 세트 ID에 메시지로 송신한 구조에서 이 값을 변경합니다. 오류가 발생하지 않으면 MQGET 호출에서 CSINHT 값을 리턴하지 않습니다.

MQMD의 *MDPAT* 필드 값이 ATBRKR이면 CSINHT를 사용할 수 없습니다.

이 필드의 초기값은 CSUNDF입니다.

#### **MEENC(10자리의 부호 있는 정수)**

MEENC(10자리의 부호 있는 정수)

MQMDE 구조 뒤에 오는 데이터의 숫자 인코딩을 지정합니다. MQMDE 구조 자체의 숫자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 큐 관리자는 이 필드가 올바른지 검사하지 않습니다. 데이터 인코딩에 대한 자세한 정보는 [1063 페이지의 『IBM i의 MQMD\(메시지 디스크립터\)』](#)에 설명된 *MDENC* 필드를 참조하십시오.

이 필드의 초기값은 ENNAT입니다.

#### **MEFLG(10자리의 부호 있는 정수)**

일반 플래그.

다음 플래그가 지정될 수 있습니다.

#### **MEFNON**

플래그가 없습니다.

이 필드의 초기값은 MEFNON입니다.

#### **MEFMT(8바이트 문자열)**

MQMDE를 뒤에 오는 데이터의 형식 이름.

MQMDE 구조 뒤에 오는 데이터의 형식 이름을 지정합니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 큐 관리자는 이 필드가 올바른지 검사하지 않습니다. 형식 이름에 대한 자세한 정보는 [1063 페이지의 『IBM i의 MQMD\(메시지 디스크립터\)』](#)에 설명된 *MDFMT* 필드를 참조하십시오.

이 필드의 초기값은 FMNONE입니다.

#### **MEGID(24바이트 비트 문자열)**

그룹 ID.

[1063 페이지의 『IBM i의 MQMD\(메시지 디스크립터\)』](#)에 설명된 *MDGID* 필드를 참조하십시오. 이 필드의 초기값은 GINONE입니다.

#### **MELEN(10자리의 부호 있는 정수)**

MQMDE 구조의 길이.

다음 값이 정의됩니다.

#### **MELEN2**

버전-2 메시지 디스크립터 확장자 구조의 길이.

이 필드의 초기값은 MELEN2입니다.

### MEMFL(10자리의 부호 있는 정수)

메시지 플래그.

1063 페이지의 『IBM i의 MQMD(메시지 디스크립터)』에 설명된 *MDMFL* 필드를 참조하십시오. 이 필드의 초기값은 MFNONE입니다.

### MEOFF(10자리의 부호 있는 정수)

논리 메시지의 시작으로부터 실제 메시지의 데이터 오프셋입니다.

1063 페이지의 『IBM i의 MQMD(메시지 디스크립터)』에 설명된 *MDOFF* 필드를 참조하십시오. 이 필드의 초기값은 0입니다.

### MEOLN(10자리의 부호 있는 정수)

원래 메시지의 길이.

1063 페이지의 『IBM i의 MQMD(메시지 디스크립터)』에 설명된 *MDOLN* 필드를 참조하십시오. 이 필드의 초기값은 OLUNDF입니다.

### MESEQ(10자리의 부호 있는 정수)

그룹 내의 논리 메시지의 순서 번호.

1063 페이지의 『IBM i의 MQMD(메시지 디스크립터)』에 설명된 *MDSEQ* 필드를 참조하십시오. 이 필드의 초기값은 1입니다.

### MESID(4바이트 문자열)

구조 ID입니다.

값은 다음과 같아야 합니다.

#### MESIDV

메시지 디스크립터 확장 구조의 ID입니다.

이 필드의 초기값은 MESIDV입니다.

### MEVER(10자리의 부호 있는 정수)

구조 버전 번호입니다.

값은 다음과 같아야 합니다.

#### MEVER2

버전2 메시지 디스크립터 확장자 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### MEVERC

메시지 디스크립터 확장자 구조의 현재 버전.

이 필드의 초기값은 MEVER2입니다.

## 초기값

표 711. MQMDE의 필드		
필드 이름	상수의 이름	상수의 값
MESID	MESIDV	'MDE~'
MEVER	MEVER2	2
MELEN	MELEN2	72
MEENC	ENNAT	환경에 따라 다름
MECSI	CSUNDF	0

표 711. MQMDE의 필드 (계속)		
필드 이름	상수의 이름	상수의 값
MEFMT	FMNONE	공백
MEFLG	MEFNON	0
MEGID	GINONE	널
MESEQ	없음	1
MEOFF	없음	0
MEMFL	MFNONE	0
MEOLN	OLUNDF	-1

**참고:**  
1. ~ 기호는 단일 공백 문자를 나타냅니다.

### RPG 선언

```

D*.1.....2.....3.....4.....5.....6.....7..
D*
D* MQMDE Structure
D*
D* Structure identifier
D MESID 1 4 INZ('MDE ')
D* Structure version number
D MEVER 5 8I 0 INZ(2)
D* Length of MQMDE structure
D MELEN 9 12I 0 INZ(72)
D* Numeric encoding of data that followsMQMDE
D MEENC 13 16I 0 INZ(273)
D* Character-set identifier of data thatfollows MQMDE
D MECSI 17 20I 0 INZ(0)
D* Format name of data that followsMQMDE
D MEFMT 21 28 INZ(' ')
D* General flags
D MEFLG 29 32I 0 INZ(0)
D* Group identifier
D MEGID 33 56 INZ(X'0000000000000000-
D 0000000000000000000000-
D 000000000000')
D* Sequence number of logical messagewithin group
D MESEQ 57 60I 0 INZ(1)
D* Offset of data in physical messagefrom start of logical message
D MEOFF 61 64I 0 INZ(0)
D* Message flags
D MEMFL 65 68I 0 INZ(0)
D* Length of original message
D MEOLN 69 72I 0 INZ(-1)

```

### IBM i IBM i의 MQMHBO(버퍼 옵션에 대한 메시지 핸들)

버퍼 옵션에 메시지 핸들을 정의하는 구조입니다.

#### 개요

**목적:** MQMHBO 구조를 사용하여 애플리케이션이 버퍼가 메시지 핸들에서 생성되는 방법을 제어하는 옵션을 지정할 수 있습니다. 구조는 MQMHBUF 호출의 입력 매개변수입니다.

**문자 세트 및 인코딩:** MQMHBO의 데이터는 애플리케이션의 문자 세트 및 애플리케이션의 인코딩(ENNAT)에 있어야 합니다.

- [1108 페이지의 『필드』](#)
- [1108 페이지의 『초기값』](#)

- 1109 페이지의 『RPG 선언』

## 필드

MQMHBO 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

### MBOPT(10자리의 부호 있는 정수)

버퍼 옵션 구조에 대한 메시지 핸들 - MBOPT 필드.

이러한 옵션은 MQMHBUF의 조치를 제어합니다.

다음 옵션을 지정해야 합니다.

#### MBPRRF

메시지 핸들의 특성을 버퍼로 변환할 때 MQRFH2 형식으로 변환하십시오.

선택적으로 다음 옵션을 지정할 수도 있습니다. 둘 이상의 옵션을 지정하려면 값을 함께 추가하거나(동일한 상수를 두 번 이상 추가하지 않음), 비트 단위 OR 연산을 사용하여 값을 결합하십시오(프로그래밍 언어가 비트 연산을 지원하는 경우).

#### MBDLPR

버퍼에 추가되는 특성은 메시지 핸들에서 삭제됩니다. 호출이 실패하면 어떤 특성도 삭제되지 않습니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MBPRRF입니다.

### MBSID(10자리의 부호 있는 정수)

버퍼 옵션 구조에 대한 메시지 핸들 - MBSID 필드.

구조 ID입니다. 값은 다음과 같아야 합니다.

#### MBSIDV

버퍼에 대한 메시지 핸들 옵션 구조의 ID입니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 isMBSIDV입니다.

### MBVER(10자리의 부호 있는 정수)

구조 버전 번호입니다. 값은 다음과 같아야 합니다.

#### MBVER1

버퍼 옵션 구조에 대한 메시지 핸들의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### MBVERC

버퍼 옵션 구조에 대한 메시지 핸들의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 MBVER1입니다.

## 초기값

표 712. MQMHBO의 필드		
필드 이름	상수의 이름	상수의 값
MVSID	MBSIDV	'MHBO'
MBVER	MBVER1	1
MBOPT	MBPRRF	

### 참고:

1. 널 문자열 값 또는 공백은 공백 문자를 표시합니다.



## RPG 선언

```
D* MQMHBO Structure
D*
D*
D* Structure identifier
D MBSID          1      4      INZ('MHBO')
D*
D* Structure version number
D MBVER          5      8I 0  INZ(1)
D*
D* Options that control the action of MQMHBUF
D MBOPT          9      12I 0 INZ(1)
```

## IBM i IBM i의 MQOD(오브젝트 디스크립터)

MQOD 구조는 이름별로 오브젝트를 지정하는 데 사용됩니다.

### 개요

**목적:** 오브젝트의 다음 유형은 유효합니다.

- 큐 또는 분배 목록
- 이름 목록
- 프로세스 정의
- 큐 관리자
- 토픽

구조는 MQOPEN 및 MQPUT1 호출의 입/출력 매개변수입니다.

**버전:** MQOD의 현재 버전은 ODVER4입니다. 최신 버전의 구조에만 있는 필드는 다음에 오는 설명에서 최신 필드로 식별됩니다.

제공된 COPY 파일에는 환경에서 지원하지만 *ODVER* 필드의 초기값이 ODVER1로 설정된 MQOD에 대한 최신 버전이 포함됩니다. version-1 구조에 없는 필드를 사용하려면 애플리케이션이 *ODVER* 필드를 필요한 버전의 버전 번호로 설정해야 합니다.

분배 목록을 열기 위해 *ODVER*은 ODVER2 이상이어야 합니다.

**문자 세트 및 인코딩:** MQOD의 데이터는 ENNAT로 지정된 로컬 큐 관리자의 인코딩 및 **CodedCharSetId** 큐 관리자 속성으로 지정된 문자 세트에 있어야 합니다. 그러나 애플리케이션이 IBM MQ 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트와 인코딩으로 작성되어야 합니다.

- [1109 페이지의 『필드』](#)
- [1116 페이지의 『초기값』](#)
- [1116 페이지의 『RPG 선언』](#)

### 필드

MQOD 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

#### ODASI(40바이트 비트 문자열)

대체 보안 ID.

이는 *ODAU*와 함께 권한 서비스로 전달되는 보안 ID이며 이를 통해 적절한 권한 검사를 수행할 수 있습니다. *ODASI*는 다음의 경우에만 사용됩니다.

- OOALTU가 MQOPEN 호출에 지정됨, 또는
- PMALTU가 MQPUT1 호출에 지정됨

그리고 *ODAU* 필드는 첫 번째 널 문자 또는 필드의 끝까지 완전히 공백이 아닙니다.

ODASI 필드에 다음 구조가 있습니다.

- 첫 번째 바이트는 뒤에 오는 중요한 데이터의 길이를 포함하는 2진 정수이며 이 값은 길이 바이트 자체를 제외합니다. 보안 ID가 존재하지 않으면 길이는 0입니다.
- 두 번째 바이트는 존재하는 보안 ID의 유형을 표시합니다. 다음 값이 가능합니다.

**SITWNT**

Windows 보안 ID.

**SITNON**

보안 ID가 없음.

- 첫 번째 바이트에 의해 정의된 길이까지의 세 번째 및 후속 바이트는 보안 ID 자체를 포함합니다.
- 필드에 남아 있는 바이트는 2진수 영으로 설정됩니다.

다음 특수 값을 사용할 수 있습니다.

**SINONE**

보안 ID가 지정되지 않음.

이 값은 필드 길이 만큼의 2진 0입니다.

입력 필드입니다. 이 필드의 길이는 LNSCID로 제공됩니다. 이 필드의 초기값은 SINONE입니다. ODVER가 ODVER3 미만이면 이 필드가 무시됩니다.

**ODAU(12바이트 문자열)**

대체 사용자 ID.

OOALTU가 MQOPEN 호출 또는 MQPUT1 호출에 대한 PMALTU 호출에 지정되는 경우 이 필드는 애플리케이션이 현재 실행 중인 사용자 ID 대신 열려 있는 권한을 확인하는 데 사용되는 대체 사용자 ID를 포함합니다. 그러나 일부 검사는 현재 사용자 ID(예: 컨텍스트 검사)로 실행됩니다.

OOALTU 및 PMALTU가 지정되지 않고 이 필드가 첫 번째 널 문자 또는 필드의 끝까지 완전히 비어 있는 경우 지정된 옵션을 사용하여 이 오브젝트를 여는 데 필요한 사용자 권한이 없는 경우에만 성공할 수 있습니다.

OOALTU 또는 PMALTU 중 아무 것도 지정되지 않은 경우 이 필드는 무시됩니다.

입력 필드입니다. 이 필드의 길이는 LNUID에서 제공합니다. 이 필드의 초기값은 12개의 공백 문자입니다.

**ODDN(48바이트 문자열)**

동적 큐 이름.

이는 MQOPEN 호출에서 작성하는 동적 큐의 이름입니다. ODDN이 모델 큐의 이름을 지정할 때만 관련성이 있습니다. 다른 모든 경우에 ODDN은 무시됩니다.

별표도 유효하다는 점을 제외하면 이름에 유효한 문자는 ODDN과 동일합니다. ODDN이 모델 큐의 이름인 경우 비어 있는 이름(또는 첫 번째 널 문자 앞에 공백만 표시되는 문자)은 유효하지 않습니다.

이름에서 공백이 아닌 마지막 문자가 별표 (\*) 인 경우, 큐 관리자는 별표를 큐에 대해 생성된 이름이 로컬 큐 관리자에서 고유하도록 보장하는 문자열로 대체합니다. 이에 대한 충분한 문자 수를 허용하기 위해 별표는 위치 1 - 33에서만 유효합니다. 별표 뒤에는 공백 또는 널 문자 이외의 문자가 있을 수 없습니다.

이는 별표가 첫 번째 문자 위치에서 발생하는 경우 유효하며 이 이름은 큐 관리자가 생성한 문자로만 구성됩니다.

입력 필드입니다. 이 필드의 길이는 LNQN으로 제공됩니다. 이 필드의 초기값은 'AMQ.\*'이며 공백으로 채워집니다.

**ODIDC(10자리의 부호 있는 정수)**

열기에 실패한 큐 수.

이는 여는 데 실패한 분배 목록의 큐 수입니다. 존재하는 경우 분배 목록에 없는 단일 큐를 열 때 이 필드 또한 설정됩니다.

**참고:** 존재하는 경우, 이 필드는 MQOPEN 또는 MQPUT1 호출의 CMPCOD 매개변수가 CCOK 또는 CCWARN 인 경우에만 설정됩니다. CMPCOD 매개변수가 CCFAIL인 경우 설정되지 않습니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 0입니다. *ODVER*이 *ODVER2* 미만이면 이 필드가 무시됩니다.

### ODKDC(10자리의 부호 있는 정수)

열린 로컬 큐의 수.

분배 목록에서 로컬 큐로 해석되고 성공적으로 열린 큐의 수입니다. 이 수는 리모트 큐로 해석되는 큐는 포함하지 않습니다(초기에 메시지를 저장하는 데 로컬 전송 큐가 사용된 경우라도). 존재하는 경우 분배 목록에 없는 단일 큐를 열 때 이 필드 또한 설정됩니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 0입니다. *ODVER*이 *ODVER2* 미만이면 이 필드가 무시됩니다.

### ODMN(48바이트 문자열)

오브젝트 큐 관리자 이름.

다음은 *ODON* 오브젝트가 정의되어 있는 큐 관리자의 이름입니다. 이름에 유효한 문자는 *ODON*과 동일합니다(이전 참조). 첫 번째 널 문자 또는 필드의 마지막까지 완전히 비어 있는 이름은 애플리케이션이 연결된 큐 관리자를 나타냅니다(로컬 큐 관리자).

다음 지점은 표시된 오브젝트의 유형에 적용됩니다.

- *ODOT*가 *OTTOP*, *OTNLST*, *OTPRO* 또는 *OTQM*인 경우 *ODMN*은 비어 있거나 로컬 큐 관리자의 이름이어야 합니다.
- *ODON*이 모델 큐의 이름인 경우 큐 관리자는 모델 큐의 속성으로 동적 큐를 작성하고 *ODMN* 필드에서 큐가 작성되는 큐 관리자의 이름을 리턴합니다. 이는 로컬 큐 관리자의 이름입니다. 모델 큐는 *MQOPEN* 호출에서만 지정될 수 있습니다. 모델 큐는 *MQPUT1* 호출에서 유효하지 않습니다.
- *ODON*이 클러스터 큐의 이름이고 *ODMN*이 비어 있는 경우 *MQOPEN* 호출로 리턴된 큐 핸들을 사용하여 전송된 메시지의 실제 목적지는 다음과 같이 큐 관리자(또는 하나가 설치된 경우 클러스터 워크로드 엑시트)에 의해 선택됩니다.
  - *OOBND0*가 지정된 경우 큐 관리자는 *MQOPEN* 호출을 처리하는 동안 클러스터 큐의 인스턴스를 선택하고 이 큐 핸들을 사용하는 모든 메시지 넣기가 해당 인스턴스에 전송됩니다.
  - *OOBNDN*이 지정된 경우 큐 관리자는 이 큐 핸들을 사용하는 각 후속 *MQPUT* 호출에 대한 목적지 큐(클러스터의 다른 큐 관리자에 상주함)의 다른 인스턴스를 선택할 수 있습니다.

애플리케이션이 메시지를 클러스터 큐(즉, 클러스터의 특정 큐 관리자에 상주하는 큐 인스턴스)의 특정 인스턴스에 전송해야 하는 경우 애플리케이션은 *ODMN* 필드의 해당 큐 관리자의 이름을 지정해야 합니다. 이는 로컬 큐 관리자가 강제로 지정된 대상 큐 관리자에 메시지를 전송하도록 합니다.

- 열려 있는 오브젝트가 분배 목록(즉, *ODREC*가 0보다 큼)인 경우 *ODMN*은 비어 있거나 널 문자열이어야 합니다. 이 조건이 충족되지 않으면 호출은 이유 코드 *RC2153*으로 실패합니다.

이는 *ODON*이 모델 큐의 이름이고 다른 모든 경우의 입력 전용 필드일 때 *MQOPEN* 호출에 대한 입출력(I/O) 필드입니다. 이 필드의 길이는 *LNQM*에서 제공됩니다. 이 필드의 초기값은 48개의 빈 문자입니다.

### ODON(48바이트 문자열)

오브젝트 이름.

이는 *ODMN*으로 식별된 큐 관리자에 정의된 오브젝트의 로컬 이름입니다. 이름에는 다음 문자가 포함될 수 있습니다.

- 대문자 알파벳(A - Z)
- 소문자 알파벳(a - z)
- 숫자(0 - 9)
- 마침표(.), 슬래시(/), 밑줄(\_), 퍼센트(%)

이름에 선두 문자 또는 임베드된 공백을 사용할 수 없으나 후미 문자 공백은 사용할 수 있습니다. 널 문자는 이름에서 중요한 데이터의 끝을 표시하는 데 사용할 수 있습니다. 널 및 그 다음에 오는 문자는 공백으로 처리됩니다. 다음 제한사항이 표시된 환경에서 적용됩니다.

- EBCDIC 가타카나를 사용하는 시스템에서는 소문자를 사용할 수 없습니다.
- IBM i에서 소문자, 슬래시 또는 퍼센트를 포함하는 이름은 명령에 지정할 때 따옴표로 묶어야 합니다. 이 인 용부호는 구조의 필드로서 또는 호출의 매개변수로서 발생하는 이름에는 지정될 수 없습니다.

다음 지점은 표시된 오브젝트의 유형에 적용됩니다.

- *ODON*이 모델 큐의 이름인 경우 큐 관리자는 모델 큐의 속성으로 동적 큐를 작성하고 *ODON* 필드에서 큐가 작성되는 큐 관리자의 이름을 리턴합니다. 모델 큐는 *MQOPEN* 호출에서만 지정될 수 있습니다. 모델 큐는 *MQPUT1* 호출에서 유효하지 않습니다.
- 열려 있는 오브젝트가 분배 목록(즉, *ODREC*가 존재하고 0보다 큼)인 경우 *ODON*은 비어 있거나 널 문자열 이어야 합니다. 이 조건이 충족되지 않으면 호출은 이유 코드 *RC2152*로 실패합니다.
- *ODOT*가 *OTQM*인 경우 특수 규칙이 적용됩니다. 이 경우에 이름은 첫 번째 널 문자 또는 필드의 마지막까지 완전히 비어 있어야 합니다.
- *ODON*이 *TARGETYPE(TOPIC)*가 있는 알리어스 큐의 이름인 경우 알리어스 큐의 사용에 정상인 것처럼 이름 지정된 알리어스 큐에 보안 검사가 먼저 작성됩니다. 이 보안 검사에 성공하는 경우 이 *MQOPEN* 호출이 계속되며 관리 토픽 오브젝트에 대한 보안 검사 작성을 포함하여 *OTTOP*의 *MQOPEN*처럼 작동합니다.

이는 *ODON*이 모델 큐의 이름이고 다른 모든 경우의 입력 전용 필드일 때 *MQOPEN* 호출에 대한 입력력(I/O) 필드입니다. 이 필드의 길이는 *LNQN*으로 제공됩니다. 이 필드의 초기값은 48개의 빈 문자입니다.

전체 토픽 이름은 두 개의 서로 다른 필드 *ODON* 및 *ODOS*에서 빌드될 수 있습니다. 이 두 필드를 사용하는 방법에 대한 자세한 내용은 [토픽 문자열 결합](#)을 참조하십시오.

### ODORO(10자리의 부호 있는 정수)

.*MQOD*의 시작에서부터 첫 번째 오브젝트 레코드의 오프셋

*MQOD* 구조의 시작에서 처음 *MQOR* 오브젝트 레코드의 오프셋(바이트)입니다. 오프셋은 양수 또는 음수일 수 있습니다. 분배 목록이 열려 있는 경우에만 *ODORO*가 사용됩니다. *ODREC*이(가) 0인 경우 필드는 무시됩니다.

분배 목록이 열리고 있을 때, 분배 목록에 목적지 큐의 이름을 지정하기 위해 하나 이상의 *MQOR* 오브젝트 레코드의 배열을 제공해야 합니다. 다음 두 가지 방법 중 하나로 완료할 수 있습니다.

- 오프셋 필드 *ODORO* 사용

이 경우 애플리케이션은 *MQOR* 레코드의 배열이 뒤에 오는 *MQOD*를 포함하는 자체 구조를 선언하고(필요한 만큼 많은 배열 요소로) *MQOD*의 시작에서 배열의 첫 번째 요소의 오프셋에 *ODORO*를 설정해야 합니다. 이 오프셋이 올바른지 주의해서 확인해야 합니다.

- 포인터 필드 *ODORP* 사용

이 경우 애플리케이션은 *MQOD* 구조에서 별도로 *MQOR* 구조의 배열을 선언하고 배열의 주소에 *ODORP*를 설정할 수 있습니다.

어느 기술이 선택되든 *ODORO* 및 *ODORP* 중 하나가 사용되어야 합니다. 둘 다 0이거나 0이 아닌 경우 이유 코드 *RC2152*로 호출에 실패합니다.

입력 필드입니다. 이 필드의 초기값은 0입니다. *ODVER*이 *ODVER2* 미만이면 이 필드가 무시됩니다.

### ODORP(pointer)

첫 번째 오브젝트 레코드의 주소.

첫 번째 *MQOR* 오브젝트 레코드의 주소입니다. 분배 목록이 열려 있는 경우에만 *ODORP*가 사용됩니다. *ODREC*이(가) 0인 경우 필드는 무시됩니다.

입력 필드입니다. 이 필드의 초기값은 널 포인터입니다. *ODORP* 또는 *ODORO*는 오브젝트 레코드를 지정하는데 사용할 수 있지만 둘 다 지정할 수 없습니다. 이전 세부사항은 *ODORO* 필드에 대한 설명을 참조하십시오. *ODORP*을(를) 사용하지 않으면 널 포인터 또는 널 바이트로 설정해야 합니다. *ODVER*이 *ODVER2* 미만이면 이 필드가 무시됩니다.

### ODOS(MQCHARV)

*ODOS*는 사용될 긴 오브젝트 이름을 지정합니다.

이 필드는 *ODOT*의 특정 값에만 참조됩니다. 이 필드가 사용됨을 표시하는 값의 세부사항은 *ODOT*에 대한 설명을 참조하십시오.

*MQCHARV* 구조의 사용 방법 설명에 의하면 *ODOS*가 올바르게 지정되었거나 최대 길이를 초과하는 경우 *RC2441* 이유 코드로 호출에 실패합니다.

입력 필드입니다. 이 구조에 있는 필드의 초기값은 *MQCHARV* 구조의 값과 동일합니다.

전체 토픽 이름은 두 개의 서로 다른 필드 *ODON* 및 *ODOS*에서 빌드될 수 있습니다. 이 두 필드를 사용하는 방법에 대한 자세한 내용은 [토픽 문자열 결합](#)을 참조하십시오. *ODVER*이 *ODVER4* 미만이면 이 필드가 무시됩니다.

### **ODOT(10자리의 부호 있는 정수)**

오브젝트 유형.

*ODON*에 이름 지정된 오브젝트의 유형입니다. 가능한 값은 다음과 같습니다.

#### **OTQ**

큐. 오브젝트의 이름은 *ODON* 필드에서 찾을 수 있습니다.

#### **OTNLST**

이름 목록. 오브젝트의 이름은 *ODON* 필드에서 찾을 수 있습니다.

#### **OTPRO**

process definition. 오브젝트의 이름은 *ODON* 필드에서 찾을 수 있습니다.

#### **OTQM**

큐 관리자. 오브젝트의 이름은 *ODON* 필드에서 찾을 수 있습니다.

#### **OTTOP**

있습니다. 전체 토픽 이름은 두 개의 서로 다른 필드 *ODON* 및 *ODOS*에서 빌드될 수 있습니다.

이러한 두 필드를 사용하는 방법에 대한 세부사항은 [토픽 문자열 결합](#)을 참조하십시오.

*ODON* 필드에서 식별된 오브젝트를 발견할 수 없는 경우 *ODOS*에 지정된 문자열이 있는 경우라도 이유 코드 *RC2425*로 호출에 실패합니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 *OTQ*입니다.

### **ODREC(10자리의 부호 있는 정수)**

존재하는 오브젝트 레코드 수.

이는 애플리케이션에서 제공한 *MQOR* 오브젝트 레코드의 수입니다. 이 수가 0보다 큰 경우 분배 목록이 목록에 있는 목적지 큐의 수가 되는 *ODREC*로 열려 있음을 표시합니다. 분배 목록은 단 하나의 목적지만 포함할 수 있습니다.

*ODREC*의 값은 0 미만이 아니어야 하고 0보다 큰 경우 *ODOT*는 *OTQ*이어야 합니다. 이러한 조건이 충족하지 않는 경우 이유 코드 *RC2154*로 호출에 실패합니다.

입력 필드입니다. 이 필드의 초기값은 0입니다. *ODVER*이 *ODVER2* 미만이면 이 필드가 무시됩니다.

### **ODRMN(48바이트 문자열)**

해석된 큐 관리자 이름.

이는 이름 해석이 로컬 큐 관리자에 의해 수행된 이후 목적지 큐 관리자의 이름입니다. 리턴되는 이름은 *ODRQN*(으)로 식별되는 큐를 소유하는 큐 관리자의 이름입니다. *ODRMN*은(는) 로컬 큐 관리자의 이름일 수 있습니다.

*ODRQN*이(가) 로컬 큐 관리자가 속한 큐 공유 그룹이 소유하는 공유 큐인 경우, *ODRMN*은(는) 큐 공유 그룹의 이름입니다. 다른 큐 공유 그룹이 큐를 소유하는 경우, *ODRQN*은(는) 큐 공유 그룹의 이름이거나 큐 공유 그룹의 구성원인 큐 관리자의 이름이 될 수 있습니다(리턴되는 값의 특성은 로컬 큐 관리자에 존재하는 큐 정의에 의해 결정됩니다).

오브젝트가 찾아보기, 입력 또는 출력(또는 결합)에 열린 단일 큐인 경우 비어 있는 값이 리턴됩니다. 열려 있는 오브젝트가 다음 중 하나인 경우 *ODRMN*은(는) 공백으로 설정됩니다.

- 큐가 아님

- 큐(단, 찾아보기, 입력 또는 출력을 위해 열려 있지 않음)
- 지정된 OOBNDN이 있는 클러스터 큐(또는 **DefBind** 큐 속성에 값 BNDNOT가 있을 때 OOBNDQ가 적용됨)
- 분배 목록

이 필드는 출력 필드입니다. 이 필드의 길이는 LNQN으로 제공됩니다. 이 필드의 초기값은 C에서는 널 문자 열이며 기타 프로그래밍 언어에서는 48자의 공백입니다. ODVER가 ODVER3 미만이면 이 필드가 무시됩니다.

### ODRO(MQCHARV)

ODRO는 큐 관리자가 ODOM에서 제공된 이름을 해석한 후의 긴 오브젝트 이름입니다.

이 필드는 토픽 오브젝트를 참조하는 오브젝트, 토픽 및 큐 알리어스의 특정 유형의 경우에만 리턴됩니다.

긴 오브젝트 이름을 ODOS에 제공하고 ODOM에는 아무 것도 입력하지 않은 경우 이 필드에 리턴되는 값은 ODOS에 제공한 값과 동일합니다.

이 필드가 생략되는 경우(즉, ODRO.VSBufSize가 0임) ODRO는 리턴되지 않지만 길이가 ODRO.VSLength에서 리턴됩니다. 길이가 전체 ODRO보다 짧으면 잘리고 제공된 길이에 맞을 수 있는 가장 오른쪽 문자를 리턴합니다.

MQCHARV 구조의 사용 방법 설명에 의하면 ODRO가 올바르게 지정되었거나 최대 길이를 초과하는 경우 RC2520 이유 코드로 호출에 실패합니다. ODVER이 ODVER4 미만이면 이 필드가 무시됩니다.

### ODRQN(48바이트 문자열)

해석된 큐 이름.

이는 이름 해석이 로컬 큐 관리자에 의해 수행된 이후 목적지 큐의 이름입니다. 리턴되는 이름은 ODRMN(으)로 식별되는 큐 관리자에 존재하는 큐의 이름입니다.

오브젝트가 찾아보기, 입력 또는 출력(또는 결합)에 열린 단일 큐인 경우 비어 있는 값이 리턴됩니다. 열려 있는 오브젝트가 다음 중 하나인 경우 ODRQN은(는) 공백으로 설정됩니다.

- 큐가 아님
- 큐(단, 찾아보기, 입력 또는 출력을 위해 열려 있지 않음)
- 분배 목록
- 토픽 오브젝트를 참조하는 알리어스 큐(대신 1114 페이지의 『ODRO(MQCHARV)』 참조)

이 필드는 출력 필드입니다. 이 필드의 길이는 LNQN으로 제공됩니다. 이 필드의 초기값은 C에서는 널 문자 열이며 기타 프로그래밍 언어에서는 48자의 공백입니다. ODVER가 ODVER3 미만이면 이 필드가 무시됩니다.

### ODRRO(10자리의 부호 있는 정수)

MQOD의 시작에서부터 첫 번째 응답 레코드의 오프셋.

MQOD 구조의 시작에서 처음 MQRR 응답 레코드의 오프셋(바이트)입니다. 오프셋은 양수 또는 음수일 수 있습니다. 분배 목록이 열려 있는 경우에만 ODRRO가 사용됩니다. ODRREC이(가) 0인 경우 필드는 무시됩니다.

분배 목록이 열려 있을 때 여는 데 실패한 큐(MQRR의 RRCC의 필드) 및 각 실패(MQRR의 RRREA 필드)의 이유를 식별하기 위해 MQRR 응답 레코드의 하나 이상의 배열을 제공할 수 있습니다. 큐 이름이 오브젝트 레코드 배열에 나오는 동일한 순서대로 응답 레코드 배열에 데이터가 리턴됩니다. 큐 관리자는 호출 결과가 혼합된 경우에만 응답 레코드를 설정합니다(즉, 일부 큐가 성공적으로 열렸거나 다른 메시지는 실패했거나 모두 실패했지만 다른 이유로 실패했습니다). 호출의 이유 코드 RC2136은 이 경우를 나타냅니다. 동일한 이유 코드가 모든 큐에 적용되는 경우 해당 이유는 MQOPEN 또는 MQPUT1 호출의 REASON 매개변수에 리턴되고 응답 레코드는 설정되지 않습니다. 응답 레코드는 선택사항이지만 제공되는 경우 ODRREC가 있어야 합니다.

응답 레코드는 ODRRO의 오프셋을 지정하거나 ODRRP의 주소를 지정하여 오브젝트 레코드와 동일한 방법으로 제공될 수 있습니다. 이를 수행하는 방법에 대한 이전의 세부사항은 ODRRO에 대한 설명을 참조하십시오. 그러나 둘 이상의 ODRRO 및 ODRRP는 사용할 수 없습니다. 모두 0이 아닌 경우 이유 코드 RC2156으로 호출에 실패합니다.

MQPUT1 호출의 경우 이러한 응답 레코드는 메시지가 분배 목록에 있는 큐에 전송될 때 발생하는 오류 및 큐가 열릴 때 발생하는 오류에 대한 정보를 리턴하는 데 사용됩니다. 후자의 완료 코드가 CCOK 또는 CCWARN 인 경우에만 해당 큐의 열기 조작에서 대체하는 큐에 대한 넣기 조작의 완료 코드 및 이유 코드입니다.

입력 필드입니다. 이 필드의 초기값은 0입니다. ODVER이 ODVER2 미만이면 이 필드가 무시됩니다.

#### **ODRRP(pointer)**

첫 번째 응답 레코드 주소.

이는 첫 번째 MQRR 응답 레코드의 주소입니다. 분배 목록이 열려 있는 경우에만 ODRRP가 사용됩니다. ODREC이(가) 0인 경우 필드는 무시됩니다.

ODRRP 또는 ODRRO는 응답 레코드를 지정하는 데 사용할 수 있지만 둘 다 지정할 수 없습니다. 세부사항은 ODRRO 필드에 대한 설명을 참조하십시오. ODRRP(를) 사용하지 않으면 널 포인터 또는 널 바이트로 설정해야 합니다.

입력 필드입니다. 이 필드의 초기값은 널 포인터입니다. ODVER이 ODVER2 미만이면 이 필드가 무시됩니다.

#### **ODSID(4바이트 문자열)**

구조 ID입니다.

값은 다음과 같아야 합니다.

##### **ODSIDV**

오브젝트 디스크립터 구조의 ID.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 ODSIDV입니다.

#### **ODSS(MQCHARV)**

ODSS는 큐에서 메시지를 검색할 때 사용된 선택 기준을 제공하는 데 사용된 문자열을 포함합니다.

ODSS는 다음 경우에 제공되지 않아야 합니다.

- ODOT가 OTQ가 아닌 경우
- 열려 있는 큐가 입력 옵션 중 하나인 OOINP\*를 사용하여 열려 있지 않은 경우

이 경우 ODSS가 제공되면 이유 코드 RC2516으로 호출에 실패합니다.

MQCHARV 구조의 사용 방법 설명에 의하면 ODSS가 올바르게 지정되었거나 최대 길이를 초과하는 경우 이유 코드 RC2519로 호출에 실패합니다. ODVER이 ODVER4 미만이면 이 필드가 무시됩니다.

#### **ODUDC(10자리의 부호 있는 정수)**

열린 리모트 큐의 수

분배 목록에서 리모트 큐로 해석되고 성공적으로 열린 큐의 수입니다. 존재하는 경우 분배 목록에 없는 단일 큐를 열 때 이 필드 또한 설정됩니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 0입니다. ODVER이 ODVER2 미만이면 이 필드가 무시됩니다.

#### **ODVER(10자리의 부호 있는 정수)**

구조 버전 번호입니다.

값은 다음 중 하나여야 합니다.

##### **ODVER1**

버전 1 오브젝트 디스크립터 구조.

##### **ODVER2**

버전 2 오브젝트 디스크립터 구조.

##### **ODVER3**

버전 3 오브젝트 디스크립터 구조.

##### **ODVER4**

버전 4 오브젝트 디스크립터 구조.



구조의 최신 버전에만 있는 필드는 필드 설명에 그렇게 식별되어 있습니다. 다음 상수는 현재 버전의 버전 번호를 지정합니다.

### ODVERC

현재 버전의 오브젝트 디스크립터 구조.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 ODVER1입니다.

## 초기값

표 713. MQOD의 필드		
필드 이름	상수의 이름	상수의 값
ODSID	ODSIDV	'OD↵↵'
ODVER	ODVER1	1
ODOT	OTQ	1
ODON	없음	공백
ODMN	없음	공백
ODDN	없음	'AMQ.*'
ODAU	없음	공백
ODREC	없음	0
ODKDC	없음	0
ODUDC	없음	0
ODIDC	없음	0
ODORO	없음	0
ODRRO	없음	0
ODORP	없음	널 포인터 또는 널 바이트
ODRRP	없음	널 포인터 또는 널 바이트
ODASI	SINONE	널
ODRQN	없음	공백
ODRMN	없음	공백
ODOS	MQCHARV에 대해 정의된 대로	MQCHARV에 대해 정의된 대로
ODRO	ODOS에 제공된 것처럼	ODOS에 제공된 것처럼
ODSS	없음	공백
<b>참고:</b>		
1. ↵ 기호는 단일 공백 문자를 나타냅니다.		

## RPG 선언

```

D*.1.....2.....3.....4.....5.....6.....7..
D*
D* MQOD Structure
D*
D*
D* Structure identifier
D  ODSID          1      4   INZ('OD ')

```



```

D*
D* Structure version number
D ODVER          5          8I 0 INZ(1)
D*
D* Object type
D ODOT           9          12I 0 INZ(1)
D*
D* Object name
D ODON           13         60      INZ
D*
D* Object queue manager name
D ODMN           61         108     INZ
D*
D* Dynamic queue name
D ODDN           109        156     INZ('AMQ.*')
D*
D* Alternate user identifier
D ODAU           157        168     INZ
D*
** Number of object records
D* present
D ODREC          169        172I 0 INZ(0)
D*
** Number of local queues opened
D* successfully
D ODKDC          173        176I 0 INZ(0)
D*
** Number of remote queues opened
D* successfully
D ODUDC          177        180I 0 INZ(0)
D*
** Number of queues that failed to
D* open
D ODIDC          181        184I 0 INZ(0)
D*
** Offset of first object record
D* from start of MQOD
D ODORO          185        188I 0 INZ(0)
D*
** Offset of first response record
D* from start of MQOD
D ODRRO          189        192I 0 INZ(0)
D*
D* Address of first object record
D ODORP          193        208*    INZ(*NULL)
D*
** Address of first response
D* record
D ODRRP          209        224*    INZ(*NULL)
D*
D* Alternate security identifier
D ODASI          225        264     INZ(X'0000000000000000-
D                               000000000000000000000000-
D                               000000000000000000000000-
D                               000000000000')
D*
D* Resolved queue name
D ODRQN          265        312     INZ
D*
D* Resolved queue manager name
D ODRMN          313        360     INZ
D*
D* reserved field
D ODRE1          361        364I 0 INZ(0)
D*
D* reserved field
D ODRS2          365        368I 0 INZ(0)
D*
D* Object long name
D* Address of variable length string
D ODOSCHRP       369        384*    INZ(*NULL)
D* Offset of variable length string
D ODOSCHRO       385        388I 0 INZ(0)
D* Size of buffer
D ODOSVSBS       389        392I 0 INZ(-1)
D* Length of variable length string
D ODOSCHRL       393        396I 0 INZ(0)
D* CCSID of variable length string
D ODOSCHRC       397        400I 0 INZ(-3)
D*
D* Message Selector
D* Address of variable length string

```

```

D ODSSCHRP          401    416*   INZ(*NULL)
D* Offset of variable length string
D ODSSCHRO          417    420I  0 INZ(0)
D* Size of buffer
D ODSSVSBS          421    424I  0 INZ(-1)
D* Length of variable length string
D ODSSCHRL          425    428I  0 INZ(0)
D* CCSID of variable length string
D ODSSCHRC          429    432I  0 INZ(-3)
D*
D* Resolved long object name
D* Address of variable length string
D ODRSOCHRP         433    448*   INZ(*NULL)
D* Offset of variable length string
D ODRSOCHRO         449    452I  0 INZ(0)
D* Size of buffer
D ODRSOVSBS         453    456I  0 INZ(-1)
D* Length of variable length string
D ODRSOCHRL         457    460I  0 INZ(0)
D* CCSID of variable length string
D ODRSOCHRC         461    464I  0 INZ(-3)
D*
D* Alias queue resolved object type
D ODRT              465    468I  0 INZ(0)

```

## IBM i IBM i의 MQOR(오브젝트 레코드)

MQOR 구조는 단일 목적지 큐의 큐 이름 및 큐 관리자 이름을 지정하는 데 사용됩니다.

### 개요

**목적:** MQOR은 MQOPEN 및 MQPUT1 호출에 대한 입력 구조입니다.

**문자 세트 및 인코딩:** MQOR의 데이터는 **CodedCharSetId** 큐 관리자 속성에 의해 지정된 문자 세트 및 ENNAT에 의해 지정된 로컬 큐 관리자의 인코딩에 있어야 합니다. 그러나 애플리케이션이 IBM MQ 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트와 인코딩으로 작성되어야 합니다.

**사용법:** MQOPEN 호출에 이러한 구조의 배열을 제공하여 큐의 목록을 여는 것이 가능합니다. 이 목록은 분배 목록이라고 부릅니다. 큐가 열린 경우 각 메시지는 목록에서 각 큐에 배치되는 MQOPEN 호출로 리턴된 큐 핸들을 사용하여 넣습니다.

- [1118 페이지의 『필드』](#)
- [1119 페이지의 『초기값』](#)
- [1119 페이지의 『RPG 선언』](#)

### 필드

MQOR 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

#### ORMN(48바이트 문자열)

오브젝트 큐 관리자 이름.

MQOD 구조의 *ODMN* 필드와 동일합니다(자세한 내용은 MQOD 참조).

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 48개의 빈 문자입니다.

#### ORON(48바이트 문자열)

오브젝트 이름.

이는 다음을 제외하고는 MQOD 구조의 *ODON* 필드와 동일합니다(자세한 내용은 MQOD 참조).

- 큐의 이름이어야 합니다.
- 모델 큐의 이름이 아니어야 합니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 48개의 빈 문자입니다.

## 초기값

표 714. MQOR의 필드		
필드 이름	상수의 이름	상수의 값
ORON	없음	공백
ORMN	없음	공백

## RPG 선언

```
D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQOR Structure
D*
D* Object name
D  ORON                1      48    INZ
D* Object queue manager name
D  ORMN                49     96    INZ
```

## MQPD - 특성 디스크립터

MQPD는 특성의 속성을 정의하는 데 사용됩니다.

### 개요

목적: 이 구조는 MQSETMP 호출의 입출력(I/O) 매개변수 및 MQINQMP 호출의 출력 매개변수입니다.

문자 세트 및 인코딩: MQPD의 데이터는 애플리케이션의 문자 세트 및 애플리케이션의 인코딩(ENNAT)에 있어야 합니다.

- [1119 페이지의 『필드』](#)
- [1121 페이지의 『초기값』](#)
- [1122 페이지의 『RPG 선언』](#)

### 필드

MQPD 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

#### PDCT(10자리의 부호 있는 정수)

특성이 속한 메시지 컨텍스트에 대해 설명합니다.

큐 관리자가 올바르게 않은 것으로 인식되는 IBM MQ정의 특성을 포함하는 메시지를 큐 관리자가 받을 때 큐 관리자는 PDCT 필드의 값을 수정합니다.

다음 옵션을 지정할 수 있습니다.

#### PDUSC

특성은 사용자 컨텍스트와 연관됩니다.

MQSETMP 호출을 사용하여 사용자 컨텍스트와 연관된 특성을 설정할 수 있는 특수 권한은 필요하지 않습니다.

이전에 설명된 옵션이 필요하지 않다면 다음 옵션을 사용할 수 있습니다.

#### PDNOC

특성은 메시지 컨텍스트와 연관되지 않습니다.

인식되지 않은 값은 RC2482의 PDREA 코드로 거부됩니다.

이는 MQSETMP 호출에 대한 입출력(I/O) 필드이며 MQINQMP 호출의 출력 필드입니다. 이 필드의 초기값은 PDNOC입니다.

## **PDCPYOPT(10자리의 부호 있는 정수)**

속성이 복사되어야 하는 메시지 유형에 대해 설명합니다.

이는 인식된 IBM MQ 정의 특성에 대한 유일한 출력 필드입니다. IBM MQ는 적절한 값을 설정합니다.

큐 관리자가 올바르게 않은 것으로 인식되는 IBM MQ정의 특성을 포함하는 메시지를 큐 관리자가 받을 때 큐 관리자는 *CopyOptions* 필드의 값을 수정합니다.

이 옵션 중 하나 이상을 지정할 수 있습니다. 둘 이상의 옵션을 지정하려면 값을 함께 추가하거나(동일한 상수를 두 번 이상 추가하지 않음), 비트 단위 OR 연산을 사용하여 값을 결합하십시오(프로그래밍 언어가 비트 연산을 지원하는 경우).

### **COPFOR**

이 특성은 전달되는 메시지에 복사됩니다.

### **COPPUB**

이 특성은 메시지가 발행되는 경우 구독자가 수신하는 메시지에 복사됩니다.

### **COPREP**

이 특성은 응답 메시지에 복사됩니다.

### **COPRP**

이 특성은 보고서 메시지에 복사됩니다.

### **COPALL**

이 특성은 모든 유형의 후속 메시지에 복사됩니다.

### **COPNON**

이 특성은 메시지에 복사되지 않습니다.

**기본 옵션:** 기본 복사 옵션 세트를 제공하기 위해 다음 옵션을 지정할 수 있습니다.

### **COPDEF**

이 특성은 전달되는 메시지, 보고서 메시지 또는 메시지가 발행되는 경우 구독자가 수신하는 메시지에 복사됩니다.

이는 옵션 COPFOR, COPRP, COPPUB 결합의 지정과 동등합니다.

설명된 옵션 중 이전에 필요한 옵션이 없는 경우 다음 옵션을 사용할 수 있습니다.

### **COPNON**

기타 복사 옵션이 지정되지 않았음을 표시하려면 이 값을 사용하십시오. 프로그래밍 방식으로 이 특성 및 후속 메시지 사이에 관계가 존재하지 않습니다. 이는 항상 메시지 디스크립터 특성에 대해 리턴됩니다.

이는 MQSETMP 호출에 대한 입출력(I/O) 필드이며 MQINQMP 호출의 출력 필드입니다. 이 필드의 초기값은 COPDEF입니다.

## **PDOPT(10자리의 부호 있는 정수)**

값은 다음과 같아야 합니다.

### **PDNONE**

옵션이 지정되지 않음

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 PDNONE입니다.

## **PDSID(10자리의 부호 있는 정수)**

구조 ID이며 값은 다음과 같아야 합니다.

### **PSIDV**

특성 디스크립터 구조의 ID.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 PSIDV입니다.

## **PDSUP(10자리의 부호 있는 정수)**

이 필드는 큐 관리자가 이 특성을 포함하는 메시지를 큐에 넣기 위해 메시지 특성에 대한 어떤 지원 레벨이 필요한지 설명합니다. 이는 IBM MQ 정의 특성에만 적용됩니다. 다른 모든 특성에 대한 지원은 선택사항입니다.

큐 관리자가 IBM MQ 정의 특성을 알고 있는 경우 필드는 올바른 값으로 자동 설정됩니다. 특성이 인식되지 않은 경우 PDSUPO가 지정됩니다. 큐 관리자가 올바르게 지정된 것으로 인식되는 IBM MQ 정의 특성을 포함하는 메시지를 큐 관리자가 받을 때 큐 관리자는 PDSUP 필드의 값을 수정합니다.

CMNOVA 옵션이 설정된 메시지 핸들에서 MQSETMP 호출을 사용하여 IBM MQ 정의 특성을 설정할 때 PDSUP는 입력 필드가 됩니다. 이를 통해 애플리케이션은 연결된 큐 관리자에서 특성이 지원되지 않지만 메시지가 다른 큐 관리자에서 처리되어야 하는 경우 IBM MQ 정의 특성을 올바른 값과 함께 넣을 수 있습니다.

값 PDSUPO는 IBM MQ 정의 특성이 아닌 특성에 지정됩니다.

CMNOVA 옵션이 설정된 메시지 핸들에서 MQSETMP 호출을 사용할 때 다음 값 중 하나가 MQINQMP 호출로 리턴되거나 다음 값 중 하나를 지정할 수 있습니다.

**PDSUPO**

지원되지 않는 경우라도 특성은 큐 관리자에 의해 허용됩니다. 특성은 메시지 특성을 지원하지 않는 큐 관리자로 메시지를 플로우하기 위해 특성을 제거할 수 있습니다. 또한 이 값은 IBM MQ에서 정의하지 않은 특성에 지정됩니다.

**PDSUPR**

특성에 대한 지원은 필수입니다. 메시지는 IBM MQ 정의 특성을 지원하지 않는 큐 관리자에 의해 거부됩니다. MQPUT 또는 MQPUT1 호출은 완료 코드 CCFAIL 이유 코드 및 RC2490으로 실패합니다.

**PDSUPL**

메시지가 로컬 큐로 예정되는 경우 메시지는 IBM MQ 정의 특성을 지원하지 않는 큐 관리자에 의해 거부됩니다. MQPUT 또는 MQPUT1 호출은 완료 코드 CCFAIL 이유 코드 및 RC2490으로 실패합니다.

MQPUT 또는 MQPUT1 호출은 메시지가 리모트 큐 관리자로 예정되는 경우 성공합니다.

메시지 핸들이 CMNOVA 옵션 세트에 작성되는 경우 이는 MQINQMP 호출의 출력 필드 및 MQSETMP 호출의 입력 필드입니다. 이 필드의 초기값은 PDSUPO입니다.

**PDVER(10자리의 부호 있는 정수)**

구조 버전 번호이며 값은 다음과 같아야 합니다.

**PDVER1**

버전 1 특성 디스크립터 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

**PDVERC**

특성 디스크립터 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 PDVER1입니다.

**초기값**

표 715. MQPD의 필드		
필드 이름	상수의 이름	상수의 값
PDSID	PDSIDV	'PD'
PDVER	PDVER1	1
PDOPT	PDNONE	0
PDSUP	PDSUPO	0
PDCT	PDNOC	0
PDCPYOPT	COPDEF	0

## RPG 선언

```
D* MQDMHO Structure
D*
D*
D* Structure identifier
D DMSID          1      4      INZ('DMHO')
D*
D* Structure version number
D DMVER          5      8I 0  INZ(1)
D*
D* Options that control the action of MQDLTMH
D DMOPT          9      12I 0 INZ(0)
```

## IBM i IBM i의 MQPMO(메시지 넣기 옵션)

MQPMO 구조로 애플리케이션이 메시지가 큐에 배치되거나 토픽에 발행되는 방법을 제어하는 옵션을 지정할 수 있습니다.

### 개요

#### 목적

구조는 MQPUT 및 MQPUT1 호출의 입/출력 매개변수입니다.

#### 버전

MQPMO에 대한 최신 버전은 PMVER2입니다. 최신 버전의 구조에만 있는 필드는 다음에 오는 설명에서 최신 필드로 식별됩니다.

제공된 COPY 파일에는 환경에서 지원하지만 *PMVER* 필드의 초기값이 *PMVER1*로 설정된 MQPMO의 최신 버전이 포함되어 있습니다. *version-1* 구조에 없는 필드를 사용하려면 애플리케이션이 *PMVER* 필드를 필요한 버전의 버전 번호로 설정해야 합니다.

#### 문자 세트 및 인코딩

MQPMO의 데이터는 **CodedCharSetId** 큐 관리자 속성에 의해 지정된 문자 세트 및 ENNAT에 의해 제공된 로컬 큐 관리자의 인코딩으로 되어 있어야 합니다. 그러나 애플리케이션이 IBM MQ 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트와 인코딩으로 작성되어야 합니다.

- [1122 페이지의 『필드』](#)
- [1134 페이지의 『초기값』](#)
- [1135 페이지의 『RPG 선언』](#)

### 필드

MQPMO 구조에는 다음과 같은 필드가 포함됩니다. 필드에 대해서 알파벳순으로 설명합니다.

#### PMCT(10자리의 부호 있는 정수)

입력 큐의 오브젝트 핸들.

PMPASI 또는 PMPASA가 지정되는 경우 이 필드는 넣을 메시지 및 관련된 컨텍스트 정보를 가져올 입력 큐 핸들을 포함해야 합니다.

PMPASI 및 PMPASA가 지정되지 않은 경우 이 필드는 무시됩니다.

입력 필드입니다. 이 필드의 초기값은 0입니다.

#### PMIDC(10자리의 부호 있는 정수)

전송할 수 없는 메시지 수.

이는 분배 목록의 큐에 송신할 수 없는 메시지의 수입니다. 이 개수에는 열리는데 실패한 큐 및 열렸지만 넣기 조작에 실패한 큐를 포함합니다. 또한 이 필드는 분배 목록에 없는 단일 큐에 메시지를 넣을 때에도 설정됩니다.

**참고:** 이 필드는 MQPUT 또는 MQPUT1 호출의 **CMPCOD** 매개변수가 CCOK 또는 CCWARN인 경우에만 설정됩니다. **CMPCOD** 매개변수가 CCFAIL인 경우에는 설정되지 않습니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 0입니다. **PMVER**이(가) **PMVER2**보다 작은 경우 이 필드는 설정되지 않습니다.

### **PMKDC(10자리의 부호 있는 정수)**

로컬 큐에 성공적으로 전송된 메시지 수.

이는 현재 MQPUT 또는 MQPUT1 호출에서 로컬 큐인 분배 목록의 큐에 송신한 메시지의 수입니다. 이 수는 리포트 큐로 분석되는 큐에 송신된 메시지는 포함하지 않습니다(초기에 메시지를 저장하는 데 로컬 전송 큐가 사용된 경우라도). 또한 이 필드는 분배 목록에 없는 단일 큐에 메시지를 넣을 때에도 설정됩니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 0입니다. **PMVER**이(가) **PMVER2**보다 작은 경우 이 필드는 설정되지 않습니다.

### **PMOPT(10자리의 부호 있는 정수)**

MQPUT 및 MQPUT1의 조치를 제어하는 옵션.

다음 중 모두를 지정하거나 아무것도 지정하지 않을 수 있습니다. 둘 이상이 필요한 경우 값을 추가할 수 있습니다(동일한 상수를 두 번 이상 추가하지 않음). 올바르게 않은 결합이 설명되어 있습니다. 다른 모든 결합은 유효합니다.

**발행 옵션:** 다음 옵션은 메시지가 토픽에 발행되는 이유를 제어합니다.

#### **PMSRTO**

이 발행의 MDRM의 MDRQ 및 MQMD 필드에 채워진 정보는 구독자에게 전달되지 않습니다. 이 옵션이 ReplyToQ가 필요한 보고서 옵션과 함께 사용되는 경우 RC2027로 호출에 실패합니다.

#### **PMRET**

송신되는 발행물을 큐 관리자가 보유하게 됩니다. 이렇게 보유되므로 구독자는 발행된 시간 이후에 MQSUBRQ 호출을 사용하여 이 발행물의 사본을 요청할 수 있습니다. 또한 발행된 시간 후에 구독을 작성하는 애플리케이션으로 발행물을 전송할 수도 있습니다. SONEWP 옵션을 사용하여 전송하도록 선택하지 않는 경우). 애플리케이션으로 보유된 발행물이 전송되는 경우 해당 발행물의 mq.IsRetained 메시지 특성으로 표시됩니다.

주제 트리의 각 노드는 단 하나의 발행물만 보유할 수 있습니다. 이는 이 주제에 대해 보유된 발행물이 이미 있는 경우에 다른 애플리케이션이 발행하면 이 발행물로 바뀔을 의미합니다. 그러므로 동일한 주제에 대해 둘 이상의 발행자가 메시지를 보유하지 않도록 하는 것이 좋습니다.

구독자가 보유된 발행물을 요청하는 경우 사용된 구독에서 토픽과 와일드카드가 포함될 수 있으며 이 경우 많은 보유된 발행물이 일치할 수 있고(주제 트리 내의 다양한 노드에서) 요청하는 애플리케이션으로 여러 발행물이 전송될 수 있습니다. 자세한 정보는 [765 페이지의 『MQSUBRQ - 구독 요청』](#) 호출에 대한 설명을 참조하십시오.

이 옵션을 사용해도 발행물을 보유할 수 없는 경우에는 메시지가 발행되지 않고 호출이 실패하며 RC2479가 발생합니다.

**동기점 옵션:** 다음 옵션은 작업 단위 내에서 MQPUT 또는 MQPUT1 호출의 참여와 관련됩니다.

#### **PMSYP**

동기점 제어를 가진 메시지를 넣으십시오.

요청이 일반 작업 단위 프로토콜 내에서 조작됩니다. 작업 단위가 커밋될 때까지 작업 단위의 외부에는 메시지가 표시되지 않습니다. 작업 단위가 백아웃되면 메시지가 삭제됩니다.

이 옵션 또는 PMNSYP가 지정되지 않은 경우 넣기 요청은 작업 단위 내에 포함되지 않습니다.

PMSYP는 PMNSYP로 지정되지 않아야 합니다.

#### **PMNSYP**

동기점 제어가 없는 메시지 넣기.

요청은 일반 작업 단위 프로토콜의 외부에서 조작하는 것입니다. 메시지는 즉시 사용 가능하며 작업 단위를 백아웃하여 삭제할 수는 없습니다.

이 옵션 또는 PMSYP가 지정되지 않은 경우 넣기 요청은 작업 단위 내에 포함되지 않습니다.

PMNSYP는 PMSYP로 지정되지 않아야 합니다.

**메시지 ID 및 상관 ID 옵션:** 다음 옵션은 큐 관리자가 새 메시지 ID 또는 상관 ID를 생성하도록 요청합니다.

#### **PMNMID**

새 메시지 ID 생성.

이 옵션을 사용하면 큐 관리자가 MQMD에 있는 *MDMID* 필드의 콘텐츠를 새 메시지 ID로 대체합니다. 이 메시지 ID는 메시지와 함께 송신되며 MQPUT 또는 MQPUT1 호출 출력 시 애플리케이션으로 리턴됩니다.

이 옵션은 메시지를 분배 목록에 넣을 때 지정할 수도 있습니다. 자세한 정보는 MQPMR 구조의 *PRMID* 필드에 대한 설명을 참조하십시오.

이 옵션을 사용하면 각 MQPUT 또는 MQPUT1 호출 전에 *MDMID* 필드를 MINONE으로 재설정해야 하는 애플리케이션을 제거합니다.

#### **PMNCID**

새 상관 ID 생성.

이 옵션을 사용하면 큐 관리자가 MQMD에 있는 *MDCID* 필드의 콘텐츠를 새 상관 ID로 대체합니다. 이 상관 ID는 메시지와 함께 송신되며 MQPUT 또는 MQPUT1 호출 출력 시 애플리케이션으로 리턴됩니다.

이 옵션은 메시지를 분배 목록에 넣을 때 지정할 수도 있습니다. 자세한 정보는 MQPMR 구조의 *PRCID* 필드에 대한 설명을 참조하십시오.

PMNCID는 애플리케이션에 고유한 상관 ID가 필요한 상황에서 유용합니다.

**그룹 및 세그먼트 옵션:** 다음은 논리 메시지의 세그먼트 및 그룹의 메시지 처리에 관련된 옵션입니다. 이러한 정의는 옵션 이해에 도움이 될 수 있습니다.

#### **실제 메시지**

큐에 배치하거나 큐에서 제거할 수 있는 가장 작은 정보 단위입니다. 대개 단일 MQPUT, MQPUT1 또는 MQGET 호출에서 지정되거나 검색되는 정보에 해당합니다. 모든 실제 메시지에는 자신의 메시지 디스크립터(MQMD)가 있습니다. 일반적으로 실제 메시지는 메시지 ID(MQMD의 *MDMID* 필드)에 대한 값을 서로 다르게 하여 구별되지만, 큐 관리자에 의해 강제 적용되지는 않습니다.

#### **논리 메시지**

이는 애플리케이션 정보의 한 단위입니다. 시스템 제한 조건이 없는 경우 논리 메시지는 실제 메시지와 동일합니다. 그러나 논리 메시지가 큰 경우 시스템 제한 조건에 따라 논리 메시지를 세그먼트라는 둘 이상의 실제 메시지로 분할해야 할 수 있습니다.

하나의 세그먼트화된 논리 메시지는 널이 아닌 그룹 ID(MQMD의 *MDGID* 필드)와 메시지 순서 번호(MQMD의 *MDSEQ* 필드)가 동일한 둘 이상의 실제 메시지로 구성됩니다. 세그먼트는 세그먼트 오프셋(MQMD의 *MDOFF* 필드)에 대한 값을 서로 다르게 하여 구별되며, 이 값은 논리 메시지의 데이터 시작부의 실제 메시지의 데이터 오프셋을 제공합니다. 각 세그먼트가 물리적 메시지이므로 논리 메시지에 있는 세그먼트는 일반적으로 서로 다른 메시지 ID를 가집니다.

세그먼트화되지 않았지만 전송하는 애플리케이션이 세그먼트화할 수 있는 논리 메시지에 널이 아닌 그룹 ID가 있습니다. 다만 이 경우 논리 메시지가 메시지 그룹에 속하지 않으면 해당 그룹 ID의 실제 메시지가 하나뿐입니다. 논리 메시지가 메시지 그룹에 속하지 않는 경우 전송하는 애플리케이션이 세그먼트화할 수 없는 논리 메시지에는 널 그룹 ID(GINONE)가 있습니다.

#### **메시지 그룹**

이는 널이 아닌 동일한 그룹 ID를 갖는 하나 이상의 논리 메시지 세트입니다. 그룹의 논리 메시지는 1과 n 사이의 정수인 메시지 순서 번호에 대한 값을 다르게 하여 구별됩니다. 여기서 n은 그룹의 논리 메시지 수입니다. 하나 이상의 논리 메시지가 세그먼트화되면 그룹에는 n보다 많은 실제 메시지가 있습니다.

#### **PMLOGO**

그룹의 메시지 및 논리 메시지의 세그먼트는 논리적인 순서로 넣습니다.



이 옵션은 큐 관리자에게 애플리케이션이 논리 메시지의 그룹 및 세그먼트에 메시지를 넣는 방법을 알려 줍니다. 이 옵션은 MQPUT 호출에만 지정할 수 있으며 MQPUT1 호출에서는 유효하지 않습니다.

PMLOGO가 지정되는 경우 이는 애플리케이션이 다음에 대해 연속적인 MQPIT 호출을 사용함을 표시합니다.

- 증가하는 세그먼트 오프셋(0부터 시작) 순으로 간격 없이 세그먼트를 각 논리 메시지에 넣습니다.
- 다음 논리 메시지에 세그먼트를 넣기 전에 하나의 논리 메시지에 모든 세그먼트를 넣습니다.
- 증가하는 세그먼트 순서 번호(1부터 시작) 순으로 간격 없이 논리 메시지를 각 메시지 그룹에 넣습니다.
- 다음 메시지 그룹에 논리 메시지를 넣기 전에 하나의 메시지 그룹에 논리 메시지 모두를 넣습니다.

이 순서는 호출된 "논리적인 순서"입니다.

애플리케이션이 메시지를 그룹 및 논리 메시지의 세그먼트에 넣는 방법을 알렸기 때문에 애플리케이션은 큐 관리자가 이를 수행한 것처럼 각 MQPUT 호출에 대한 그룹 및 세그먼트 정보를 유지보수하고 업데이트할 필요가 없습니다. 특히, 이는 큐 관리자가 이러한 값을 적절한 값으로 설정하므로 애플리케이션이 MQMD의 MDGID, MDSEQ 및 MDOFF 필드를 설정할 필요가 없음을 의미합니다. 애플리케이션은 메시지가 그룹에 속하거나 논리 메시지의 세그먼트인지를 표시하고 논리 메시지의 그룹 또는 마지막 세그먼트에 마지막 메시지를 표시하기 위해 MQMD의 MDMFL 필드만 설정해야 합니다.

메시지 그룹 또는 논리 메시지가 시작되면 후속 MQPUT 호출은 MQMD의 MDMFL에 적절한 MF\* 플래그를 지정해야 합니다. 종료되지 않은 메시지 그룹이 있는 경우 애플리케이션이 그룹에 메시지를 넣으려고 시도하거나 종료되지 않은 논리 메시지가 있을 때 세그먼트가 아닌 메시지를 넣으면 적절하게 이유 코드 RC2241 또는 이유 코드 RC2242로 호출에 실패합니다. 그러나 큐 관리자는 현재 메시지 그룹 또는 현재 논리 메시지에 정보를 보유하고 애플리케이션은 MFLMIG 또는 MFLSEG를 적절하게 지정하는 메시지(애플리케이션 메시지 데이터가 없을 수도 있음)를 보내 MQPUT 호출을 재실행하기 전에 그룹에 없거나 세그먼트가 아닌 메시지를 종료할 수 있습니다.

1126 페이지의 표 716은(는) 유효한 옵션 및 플래그의 조합과, 큐 관리자가 각 케이스에서 사용하는 MDGID, MDSEQ 및 MDOFF 필드의 값을 표시합니다. 표에 표시되지 않은 옵션과 플래그의 조합은 유효하지 않습니다. 표의 열에는 다음 의미가 있습니다.

#### **LOG ORD**

PMLOGO 옵션이 호출에 지정되었는지 여부를 표시합니다.

#### **MIG**

MFMIG 또는 MFLMIGT 옵션이 호출에 지정되었는지 여부를 표시합니다.

#### **SEG**

MFSEG 또는 MFLSEG 옵션이 호출에 지정되었는지 여부를 표시합니다.

#### **SEG OK**

MFSEGA 옵션이 호출에 지정되었는지 여부를 표시합니다.

#### **Cur grp**

현재 메시지 그룹이 호출 이전에 존재하는지 여부를 표시합니다.

#### **Cur log msg**

현재 논리 메시지가 호출 이전에 존재하는지 여부를 표시합니다.

#### **기타 열**

큐 관리자가 사용하는 값을 보여줍니다. "이전"은 큐 핸들에 대한 이전 메시지에 있는 필드에 대해 사용된 값을 표시합니다.

#### **PMRLOC**

MQPMO 구조에서 PMRQN이 메시지가 실제 넣는 로컬 메시지의 이름으로 완료되어야 함을 지정합니다. 이와 유사하게 ResolvedQMgrName은 로컬 큐를 호스트하는 로컬 큐 관리자의 이름으로 완료됩니다. 이것이 의미하는 바는 OORLOQ를 참조하십시오. 사용자에게 큐에 넣기 위한 권한이 부여되면 MQPUT 호출에 이 플래그를 지정하기 위해 필요한 권한이 있습니다. 특별한 권한이 필요하지는 않습니다.

표 716. 논리 메시지의 세그먼트 및 그룹의 메시지에 관한 MQPUT 옵션

지정하는 옵션				호출 이전의 그룹 및 로그 메시지 상태		큐 관리자가 사용하는 값		
LOG ORD	MIG	SEG	SEG OK	Cur grp	Cur log msg	MDGID	MDSEQ	MDOFF
예	아니오	아니오	아니오	아니오	아니오	GINONE	1	0
예	아니오	아니오	예	아니오	아니오	새 그룹 ID	1	0
예	아니오	예	예 또는 아니오	아니오	아니오	새 그룹 ID	1	0
예	아니오	예	예 또는 아니오	아니오	예	이전 그룹 ID	1	이전 오프셋 + 이전 세그먼트 길이
예	예	예 또는 아니오	예 또는 아니오	아니오	아니오	새 그룹 ID	1	0
예	예	예 또는 아니오	예 또는 아니오	예	아니오	이전 그룹 ID	이전 순서 번호 + 1	0
예	예	예	예 또는 아니오	예	예	이전 그룹 ID	이전 순서 번호	이전 오프셋 + 이전 세그먼트 길이
아니오	아니오	아니오	아니오	예 또는 아니오	예 또는 아니오	GINONE	1	0
아니오	아니오	아니오	예	예 또는 아니오	예 또는 아니오	GINONE이면 새 그룹 ID이고, 그렇지 않으면 필드 값	1	0
아니오	아니오	예	예 또는 아니오	예 또는 아니오	예 또는 아니오	GINONE이면 새 그룹 ID이고, 그렇지 않으면 필드 값	1	필드 값
아니오	예	아니오	예 또는 아니오	예 또는 아니오	예 또는 아니오	GINONE이면 새 그룹 ID이고, 그렇지 않으면 필드 값	필드 값	0
아니오	예	예	예 또는 아니오	예 또는 아니오	예 또는 아니오	GINONE이면 새 그룹 ID이고, 그렇지 않으면 필드 값	필드 값	필드 값

**참고:**

- PMLOGO는 MQPUT1 호출에 유효하지 않습니다.
- MDMID 필드의 경우, 큐 관리자가 PMNMID 또는 MINONE이 지정된 경우 새 메시지 ID를 생성하고 그렇지 않으면 필드의 값을 사용합니다.
- MDCID 필드의 경우, 큐 관리자가 PMNCID가 지정된 경우 새 상관 ID를 생성하고 그렇지 않으면 필드의 값을 사용합니다.

PMLOGO가 지정된 경우, 큐 관리자는 논리 메시지의 그룹 및 세그먼트에 있는 모든 메시지를 MQMD의 *MDPER* 필드에 동일한 값으로 넣어야 합니다. 즉, 모두 지속적이거나 모두 비지속적이어야 합니다. 이 조건이 충족되지 않은 경우 이유 코드 RC2185로 호출에 실패합니다.

PMLOGO 옵션은 작업 단위에 다음과 같은 영향을 줍니다.

- 그룹 또는 논리 메시지의 첫 번째 실제 메시지를 작업 단위 내에 넣은 경우 동일한 큐 핸들이 사용되면 그룹 또는 논리 메시지의 다른 모든 실제 메시지를 작업 단위 내에 넣어야 합니다. 그러나 이들이 동일한 작업 단위 내에서 넣을 필요는 없습니다. 이 방법은 여러 실제 메시지로 구성된 메시지 그룹 또는 논리 메시지가 큐 핸들에 대해 두 개 이상의 연속 작업 단위에서 나뉘질 수 있게 합니다.
- 그룹 또는 논리 메시지의 첫 번째 실제 메시지를 작업 단위 내에 넣지 않은 경우 동일한 큐 핸들이 사용되면 그룹 또는 논리 메시지의 다른 모든 실제 메시지를 작업 단위 내에 넣을 수 없어야 합니다.

이러한 조건이 충족되지 않으면 이유 코드 RC2245와 함께 MQPUT 호출에 실패합니다.

PMLOGO를 지정할 때 MQPUT 호출에 제공된 MQMD는 MDVER2 미만이어야 합니다. 이 조건이 충족되지 않으면 호출은 이유 코드 RC2257로 실패합니다.

PMLOGO가 지정되지 않은 경우 그룹의 메시지 및 논리 메시지의 세그먼트를 순서대로 넣을 수 있으며 전체 메시지 그룹을 넣거나 논리 메시지를 완료할 필요는 없습니다. *MDGID*, *MDSEQ*, *MDOFF* 및 *MDMFL* 필드에 적절한 값이 있는지 확인하는 것은 애플리케이션의 책임입니다.

이 방법은 시스템 실패가 발생한 후 중간에 있는 메시지 그룹 또는 논리 메시지를 재시작하는 데 사용될 수 있습니다. 시스템이 다시 시작되면 애플리케이션은 *MDGID*, *MDSEQ*, *MDOFF*, *MDMFL* 및 *MDPER* 필드를 적절한 값으로 설정한 다음, PMLOGO를 지정하지 않고 필요에 따라 PMSYP 또는 PMNSYP를 설정하여 MQPUT 호출을 발행할 수 있습니다. 이 호출에 성공하면 큐 관리자가 그룹 및 세그먼트 정보를 보유할 수 있으며 해당 큐 핸들을 사용하는 후속 MQPUT 호출이 정상적으로 PMLOGO를 지정할 수 있습니다.

큐 관리자가 MQPUT 호출에 대해 보유하는 그룹 및 세그먼트 정보는 MQGET 호출에 대해 보유하는 그룹 및 세그먼트 정보와 분리됩니다.

지정된 큐 핸들의 경우 애플리케이션은 PMLOGO를 지정하는 MQPUT 호출을 MQPUT 호출과 자유롭게 혼합할 수 있지만 다음 사항을 참고해야 합니다.

- PMLOGO가 지정되지 않은 경우 MQPUT 호출을 성공할 때마다 큐 관리자는 큐 핸들에 대한 그룹 및 세그먼트 정보를 애플리케이션에서 지정한 값으로 설정하게 됩니다. 이는 큐 관리자가 큐 핸들에 대해 보유하는 기존의 그룹 및 세그먼트 정보를 대체합니다.
- PMLOGO를 지정하지 않은 경우 현재 메시지 그룹 또는 논리 메시지가 있으면 호출이 실패하지 않습니다. 호출은 CCWARN 완료 코드로 성공할 수 있습니다. [1127 페이지의 표 717](#)에서는 발생할 수 있는 다양한 사례를 보여줍니다. 이 경우 완료 코드가 CCOK가 아닌 경우 이유 코드는 다음 중 하나가 됩니다 (해당되는 경우).

- RC2241
- RC2242
- RC2185
- RC2245

**참고:** 큐 관리자는 MQPUT1 호출을 위한 그룹 및 세그먼트 정보를 확인하지 않습니다.

표 717. MQPUT 또는 MQCLOSE 호출이 그룹 및 세그먼트 정보와 일치하지 않을 때의 결과		
현재 호출	이전 호출은 PMLOGO가 있는 MQPUT였음	이전 호출은 PMLOGO가 없는 MQPUT였음
PMLOGO가 있는 MQPUT	CCFAIL	CCFAIL
PMLOGO가 없는 MQPUT	CCWARN	CCOK
종료되지 않은 그룹 또는 논리 메시지의 MQCLOSE	CCWARN	CCOK

메시지 및 세그먼트를 논리 순서로 넣는 애플리케이션은 PMLOGO를 지정하는 것이 좋습니다. 이 옵션은 가장 간단하게 사용할 수 있는 옵션입니다. 이 옵션은 큐 관리자가 이들 정보를 관리하기 때문에 애플리케이션이 그룹 및 세그먼트 정보를 관리할 필요가 없습니다. 그러나 특수화된 애플리케이션은 PMLOGO 옵션이 제공하는 것보다 더 많은 제어를 할 필요가 있으므로 이를 위해 이 옵션을 지정하지 않습니다. 이를 수행하는 경우, 애플리케이션은 각 MQPUT 또는 MQPUT1 호출 전에 MQMD의 MDGID, MDSEQ, MDOFF 및 MDMFL 필드가 올바르게 설정되어 있는지 확인해야 합니다.

예를 들어, 수신한 실제 메시지가 논리 메시지의 세그먼트 또는 그룹에 있는 메시지인지 여부와 관계없이 이들 메시지를 전달하려는 애플리케이션은 PMLOGO를 지정해서는 안 됩니다. 여기에는 다음과 같은 두 가지 이유가 있습니다.

- 메시지를 검색하여 순서대로 넣은 경우 PMLOGO를 지정하면 새 그룹 ID가 메시지에 할당되며 이는 메시지 작성자가 메시지 그룹의 결과인 응답 또는 보고 메시지를 상관시키기 어렵거나 불가능할 수 있습니다.
- 송신 및 수신 큐 관리자 사이에 다중 경로가 포함된 복잡한 네트워크에서는 실제 메시지가 순서없이 도착합니다. MQGET 호출 시 PMLOGO 및 해당 GMLOGO를 지정하지 않으면 전달 애플리케이션이 논리적 순서로 다음에 도착하는 메시지를 기다릴 필요가 없이 각 실제 메시지를 도착하자마자 검색하여 전달할 수 있습니다.

보고 메시지를 넣을 때 그룹의 메시지 또는 논리 메시지의 세그먼트에 대한 보고 메시지를 생성하는 애플리케이션은 PMLOGO도 지정하지 않아야 합니다.

PMLOGO는 다른 PM\* 옵션으로 지정될 수 있습니다.

**컨텍스트 옵션:** 다음 옵션은 메시지 컨텍스트의 처리를 제어합니다.

#### PMNOC

메시지와 연관된 컨텍스트가 없습니다.

컨텍스트가 없음을 표시하기 위해 ID와 원본 컨텍스트를 둘 다 설정합니다. 즉, MQMD의 컨텍스트 필드가 다음과 같이 설정됩니다.

- 문자 필드의 경우 공백
- 바이트 필드의 경우 널
- 숫자 필드의 경우 0

#### PMDEFC

기본 환경을 사용하십시오.

메시지는 ID와 원본 둘 다에 대해 연관된 기본 컨텍스트 정보를 갖게 됩니다. 큐 관리자는 메시지 디스크립터의 컨텍스트 필드를 다음과 같이 설정합니다.

표 718. MQMD 필드에 대한 기본 컨텍스트 정보 값

MQMD의 필드	사용된 값
MDUID	가능한 경우 환경에 따라 결정됩니다. 그밖의 경우에는 공백으로 설정됩니다.
MDACC	가능한 경우 환경에 따라 결정됩니다. 그밖의 경우에는 ACNONE로 설정됩니다.
MDAID	공백으로 설정합니다.
MDPAT	환경에 따라 결정됩니다.
MDPAN	가능한 경우 환경에 따라 결정됩니다. 그밖의 경우에는 공백으로 설정됩니다.
MDPD	메시지를 넣은 날짜로 설정합니다.
MDPT	메시지를 넣은 시간으로 설정합니다.
MDAOD	공백으로 설정합니다.

메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어](#)를 참조하십시오. 컨텍스트 옵션을 지정하지 않은 경우 이는 기본 조치입니다.

## PMPASI

입력 큐 핸들에서 ID 컨텍스트를 전달하십시오.

메시지에는 연관된 컨텍스트 정보가 있습니다. ID 컨텍스트는 *PMCT* 필드에 지정된 큐 핸들에서 가져옵니다. 원래 컨텍스트 정보는 *PMDEF*C와 동일한 방법으로 큐 관리자가 생성합니다(값은 이전 표 참조). 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트](#) 및 [컨텍스트 정보 제어](#)를 참조하십시오.

MQPUT 호출의 경우 큐는 OOPASI 옵션(또는 이를 암시하는 옵션)을 사용하여 열었어야 합니다. MQPUT1 호출의 경우 OOPASI 옵션이 있는 MQOPEN 호출로 동일한 권한 확인이 수행됩니다.

## PMPASA

입력 큐 핸들에서 모든 컨텍스트를 전달하십시오.

메시지에는 연관된 컨텍스트 정보가 있습니다. ID 및 원본 컨텍스트는 모두 *PMCT* 필드에 지정된 큐 핸들에서 가져옵니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트](#) 및 [컨텍스트 정보 제어](#)를 참조하십시오.

MQPUT 호출의 경우 큐는 OOPASA 옵션(또는 이를 암시하는 옵션)을 사용하여 열었어야 합니다. MQPUT1 호출의 경우 OOPASA 옵션이 있는 MQOPEN 호출로 동일한 권한 확인이 수행됩니다.

## PMSETI

애플리케이션의 ID 컨텍스트를 설정하십시오.

메시지에는 연관된 컨텍스트 정보가 있습니다. 애플리케이션은 MQMD 구조에 ID 컨텍스트를 지정합니다. 원래 컨텍스트 정보는 *PMDEF*C와 동일한 방법으로 큐 관리자가 생성합니다(값은 이전 표 참조). 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트](#) 및 [컨텍스트 정보 제어](#)를 참조하십시오.

MQPUT 호출의 경우 큐는 OOSSETI 옵션(또는 이를 암시하는 옵션)을 사용하여 열었어야 합니다. MQPUT1 호출의 경우 OOSSETI 옵션이 있는 MQOPEN 호출로 동일한 권한 확인이 수행됩니다.

## PMSETA

애플리케이션의 모든 컨텍스트를 설정하십시오.

메시지에는 연관된 컨텍스트 정보가 있습니다. 애플리케이션은 MQMD 구조의 ID 및 원래 ID 컨텍스트를 지정합니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트](#) 및 [컨텍스트 정보 제어](#)를 참조하십시오.

MQPUT 호출의 경우 큐는 OOSSETA 옵션을 사용하여 열었어야 합니다. MQPUT1 호출의 경우 OOSSETA 옵션이 있는 MQOPEN 호출로 동일한 권한 확인이 수행됩니다.

PM\* 컨텍스트 옵션 중 하나만 지정할 수 있습니다. 옵션이 지정되지 않은 경우 *PMDEF*C가 가정됩니다.

**응답 넣기 유형.** 다음 옵션은 MQPUT 또는 MQPUT1 호출로 리턴되는 응답을 제어합니다. 이 옵션 중 하나만 지정할 수 있습니다. PMARES 및 PMSRES가 지정되지 않은 경우 PMRASQ 또는 PMRAST가 가정됩니다.

## PMARES

PMARES 옵션은 큐 관리자가 호출이 완료되기를 기다리지 않고 MQPUT 또는 MQPUT1 조치가 완료되도록 요청합니다. 이 옵션을 사용하여 특히, 클라이언트 바인딩을 사용하는 애플리케이션의 경우 메시지 성능을 향상시킬 수 있습니다. 애플리케이션은 MQSTAT verb를 사용하여 이전 비동기 호출 중에 오류가 발생했는지 주기적으로 확인할 수 있습니다.

이 옵션을 사용하면 MQMD에서 다음 필드만 완료됩니다.

- MDAID
- MDPAT
- MDPAN
- MDAOD

또한 PMNMID 또는 PMNCID 중 하나 또는 모두가 옵션으로 지정된 경우 리턴되는 MDMID 및 MDCID도 완료됩니다 (PMNMID는 공백 MDMID 필드를 지정하여 암시적으로 지정할 수 있습니다).

이전에 지정된 필드만 완료됩니다. 일반적으로 MQMD 또는 MQPMO 구조에서 리턴되는 기타 정보는 정의되지 않습니다.

MQPUT 또는 MQPUT1에 대해 비동기 넣기 응답을 요청하는 경우 CCOK와 RCNONE의 CMPCOD 및 REASON은 반드시 큐에 메시지를 성공적으로 넣었음을 의미하지는 않습니다. 비동기 넣기 응답을 사용

하는 MQI 애플리케이션을 개발 중이고 메시지를 큐에 넣었다는 확인이 필요한 경우에는 넣기 조작에서 발생한 완료 코드 및 이유 코드를 모두 확인하고 또한 MQSTAT를 사용하여 비동기 오류 정보도 조회해야 합니다.

각 개별 MQPUT/MQPUT1 호출의 성공 또는 실패는 즉시 리턴되지 않을 수 있지만 비동기 호출에서 발생한 첫 번째 오류는 나중에 MQSTAT 호출을 통해 판별할 수 있습니다.

동기점 아래에서 비동기 넣기 응답을 사용하는 지속적 메시지 전달이 실패하고 트랜잭션을 커밋하려고 시도하는 경우 커밋이 실패하며 트랜잭션은 완료 코드 CCFAIL 및 이유 코드 RC2003과 함께 취소됩니다. 애플리케이션은 이전 MQPUT 또는 MQPUT1 실패의 원인을 판별하기 위해 MQSTAT를 호출할 수 있습니다.

#### **PMSRES**

이 MQPMO 구조의 넣기 조작 값을 지정하면 MQPUT 또는 MQPUT1 조작이 항상 동기적으로 실행됩니다. 조작이 성공하는 경우 MQMD 및 MQPMO의 모든 필드가 완료됩니다. 이는 동기 응답이 큐 또는 주제 오브젝트에 정의된 기본 넣기 응답과 무관함을 확인하기 위해 제공됩니다.

#### **PMRASQ**

MQPUT 호출에 대해 이 값이 지정된 경우 사용되는 넣기 응답 유형은 애플리케이션에서 열 때 큐에 지정된 DEFPRESP 값에서 가져옵니다. 클라이언트 애플리케이션이 IBM WebSphere MQ 7.0 이전의 레벨에서 큐 관리자에 연결되는 경우 PMSRES가 지정된 것처럼 작동합니다.

이 옵션이 MQPUT1 호출에 지정되는 경우 큐 정의의 DEFPRESP 값은 사용되지 않습니다. MQPUT1 호출이 PMSYP를 사용 중인 경우에는 PMARES에 대한 것처럼 작동하고 PMNSYP를 사용 중인 경우에는 PMSRES에 대한 것처럼 작동합니다.

#### **PMRAST**

이는 토픽 오브젝트에 사용할 PMRASQ의 동의어입니다.

**기타 옵션:** 다음 옵션은 권한 검사 및 큐 관리자가 정지 중일 때 발생하는 현상을 제어합니다.

#### **PMALTU**

지정된 사용자 ID로 유효화합니다.

이는 MQPUT1 호출의 **OBJDSC** 매개변수에 있는 **ODAU** 필드가 큐에 메시지를 넣을 수 있는 권한을 유효성 검증하는 데 사용되는 사용자 ID를 포함함을 표시합니다. 애플리케이션이 실행 중인 사용자 ID에 권한이 부여되었는지 여부에 관계없이 이 **ODAU**(가) 지정된 옵션으로 큐를 열 수 있는 권한이 있는 경우에만 호출이 성공할 수 있습니다. (지정된 컨텍스트 옵션에는 적용되지 않지만 애플리케이션을 실행하는 사용자 ID에 대해서는 항상 확인됩니다.)

이 옵션은 MQPUT1 호출과 함께 사용하는 경우에만 유효합니다.

#### **PMFIQ**

큐 관리자가 정지 중인 경우 실패합니다.

이 옵션은 큐 관리자가 정지 중인 경우 MQPUT 또는 MQPUT1 호출을 실패하게 합니다.

호출은 이유 코드 RC2161과 함께 완료 코드 CCFAIL을 리턴합니다.

**기본 옵션:** 이전에 설명된 옵션 중 필요한 옵션이 없는 경우 다음 옵션을 사용할 수 있습니다.

#### **PMNONE**

옵션이 지정되지 않았습니니다.

이 값은 기타 옵션이 지정되지 않았음을 표시하는데 사용할 수 있습니다. 모든 옵션은 기본 값을 가정합니다. PMNONE는 프로그램 문서화를 보조하기 위해 정의됩니다. 이 옵션은 다른 옵션과 함께 사용하기 위한 용도가 아니며 값이 0이므로 이러한 사용을 감지할 수 없습니다.

입력 필드입니다. **PMOPT** 필드의 초기값은 PMNONE입니다.

#### **PMPRF(10자리의 부호 있는 정수)**

MQPMR 필드가 있음을 표시하는 플래그.

이 필드는 MQPMR 필드가 애플리케이션에서 제공한 넣기 메시지 레코드에 있음을 표시하기 위해 설정해야 하는 플래그를 포함합니다. **PMPRF**은(는) 메시지가 분배 목록에 놓이는 경우에만 사용됩니다. **PMREC**(가) 0이거나 **PMPRO** 및 **PMPRP** 둘 다 0인 경우 필드는 무시됩니다.

존재하는 필드의 경우 큐 관리자는 각 목적지에 대해 해당하는 Put 메시지 레코드에 있는 필드의 값을 사용합니다. 없는 필드의 경우 큐 관리자는 MQMD 구조의 값을 사용합니다.

다음 중 하나 이상의 플래그는 넣기 메시지 레코드에 있는 필드를 표시하도록 지정할 수 있습니다.

**PFMID**

메시지 ID 필드가 존재합니다.

**PFCID**

상관 ID 필드가 존재합니다.

**PFGID**

그룹 ID 필드가 존재합니다.

**PFFB**

피드백 필드가 존재합니다.

**PFACC**

Accounting-token 필드가 존재합니다.

이 플래그가 지정된 경우 PMSETI 또는 PMSETA는 *PMOPT* 필드에 지정되어야 합니다. 이 조건을 충족하지 않는 경우 이유 코드 RC2158로 호출에 실패합니다.

MQPMR 필드가 없는 경우 다음을 지정할 수 있습니다.

**PFNONE**

넣기 메시지 레코드 필드가 없습니다.

이 값이 지정된 경우, *PMREC*은(는) 0이거나, *PMPRO* 및 *PMPRP* 모두 0이어야 합니다.

*PFNONE*은 프로그램 문서화를 지원하기 위해 정의됩니다. 이 상수는 다른 상수와 함께 사용하기 위한 용도는 아니지만 값이 0이므로 이러한 사용을 감지할 수 없습니다.

*PMPRF*가 유효하지 않은 플래그를 포함하는 경우 넣기 메시지 레코드가 제공되지만 *PMPRF*에 값 *PFNONE*이 있어 이유 코드 RC2158로 호출에 실패합니다.

입력 필드입니다. 이 필드의 초기값은 *PFNONE*입니다. *PMVER*이(가) *PMVER2*보다 작으면 이 필드는 무시됩니다.

**PMPRO(10자리의 부호 있는 정수)**

MQPMO 시작 이후 첫 번째 넣기 레코드의 오프셋.

이는 MQPMO 구조의 시작에서 첫 번째 MQPMR put 메시지 레코드의 오프셋(바이트)입니다. 오프셋은 양수 또는 음수일 수 있습니다. *PMPRO*은(는) 메시지가 분배 목록에 놓이는 경우에만 사용됩니다. *PMREC*이(가) 0인 경우 필드는 무시됩니다.

메시지를 분배 목록에 넣을 때 개별적으로 각 목적지의 메시지에 대한 특정 특성을 지정하기 위해 하나 이상의 MQPMR 넣기 메시지 레코드의 배열을 제공할 수 있습니다. 이러한 특성은 다음과 같습니다.

- 메시지 ID
- 상관 ID
- 그룹 ID
- 피드백 값
- 계정 토큰

이러한 모든 특성을 지정할 필요는 어느 서브세트가 선택되든 필드는 올바른 순서로 지정되어야 합니다. 자세한 정보는 MQPMR 구조에 대한 설명을 참조하십시오.

일반적으로 분배 목록이 열릴 때 MQOD에서 지정한 오브젝트 레코드가 있는 만큼 많은 넣기 메시지 레코드를 넣어야 합니다. 각 넣기 메시지 레코드는 해당 오브젝트 레코드에 의해 식별된 큐에 대한 메시지 특성을 제공합니다. 이 경우 메시지 특성이 무시되지만 열지 못한 분배 목록의 큐에는 배열의 적절한 위치에 넣기 메시지 레코드가 할당되어야 합니다.

넣기 메시지 레코드의 수는 오브젝트 레코드의 수와 다를 수 있습니다. 오브젝트 레코드보다 넣기 메시지 레코드가 적을 경우 메시지 레코드를 넣지 않은 목적지의 메시지 특성은 메시지 디스크립터 MQMD의 해당 필드에서 가져옵니다. 오브젝트 레코드보다 Put 메시지 레코드가 많은 경우 초과분은 사용되지 않습니다(여전



히 액세스해야 하는 경우에도). 넣기 메시지 레코드는 선택사항이지만 제공되는 경우 *PMREC*이(가) 있어야 합니다.

넣기 메시지 레코드는 *PMPRO*에 오프셋을 지정하거나 *PMPRP*에 주소를 지정하여 MQOD의 오브젝트 레코드와 유사한 방법으로 제공될 수 있습니다. 이를 수행하는 방법에 대한 자세한 정보는 [1109 페이지의 『IBM i의 MQOD\(오브젝트 디스크립터\)』](#)에 설명된 *ODORO* 필드를 참조하십시오.

둘 이상의 *PMPRO* 및 *PMPRP*는 사용할 수 없습니다. 모두 0이 아닌 경우 이유 코드 RC2159로 호출에 실패합니다.

입력 필드입니다. 이 필드의 초기값은 0입니다. *PMVER*이(가) *PMVER2*보다 작으면 이 필드는 무시됩니다.

### **PMPRP(pointer)**

첫 번째 메시지 넣기 레코드의 주소.

첫 번째 MQPMR Put 메시지 레코드의 주소입니다. *PMPRP*은(는) 메시지가 분배 목록에 놓이는 경우에만 사용됩니다. *PMREC*이(가) 0인 경우 필드는 무시됩니다.

*PMPRP* 또는 *PMPRO*을(를) 사용하여 넣기 메시지 레코드를 지정할 수 있지만 둘 다 지정할 수는 없습니다. 자세한 정보는 *PMRRO* 필드의 설명을 참조하십시오. *PMPRP*을(를) 사용하지 않으면 널 포인터 또는 널 바이트로 설정해야 합니다.

입력 필드입니다. 이 필드의 초기값은 널 포인터입니다. *PMVER*이(가) *PMVER2*보다 작으면 이 필드는 무시됩니다.

### **PMREC(10자리의 부호 있는 정수)**

존재하는 넣기 메시지 레코드 또는 응답 레코드 수.

애플리케이션에서 제공하는 MQPMR Put 메시지 레코드 또는 MQRR 응답 레코드의 수입니다. 이 숫자는 메시지가 분배 목록에 넣기 되는 경우에만 0보다 클 수 있습니다. 넣기 메시지 레코드 및 응답 레코드는 선택사항입니다. 애플리케이션이 레코드를 제공할 필요가 없거나 하나의 레코드 유형만 제공하도록 선택할 수 있습니다. 그러나 애플리케이션이 두 유형 모두의 레코드를 제공하는 경우 각 유형의 *PMREC* 레코드를 제공해야 합니다.

*PMREC*의 값은 분배 목록의 목적지 수와 동일할 필요는 없습니다. 너무 많은 레코드가 제공되면 초과분은 사용되지 않습니다. 레코드가 너무 적으면 넣기 메시지 레코드가 없는 목적지의 메시지 특성에 기본값이 사용됩니다(이 주제의 *PMPRO* 참조).

*PMREC*가 0 미만이거나 0보다 크지만 메시지를 분배 목록에 넣지 않은 경우 RC2154로 호출에 실패합니다.

입력 필드입니다. 이 필드의 초기값은 0입니다. *PMVER*이(가) *PMVER2*보다 작으면 이 필드는 무시됩니다.

### **PMRMN(48바이트 문자 문자열)**

해석된 목적지 큐 관리자의 이름.

이는 이름 해석이 로컬 큐 관리자에 의해 수행된 이후 목적지 큐 관리자의 이름입니다. 리턴되는 이름은 *PMRQN*(으)로 식별되는 큐를 소유하는 큐 관리자의 이름이며 로컬 큐 관리자의 이름이 될 수 있습니다.

*PMRQN*이(가) 로컬 큐 관리자가 속한 큐 공유 그룹이 소유하는 공유 큐인 경우, *PMRMN*은(는) 큐 공유 그룹의 이름입니다. 다른 큐 공유 그룹이 큐를 소유하는 경우, *PMRQN*은(는) 큐 공유 그룹의 이름이거나 큐 공유 그룹의 구성원인 큐 관리자의 이름이 될 수 있습니다(리턴되는 값의 특성은 로컬 큐 관리자에 존재하는 큐 정의에 의해 결정됩니다).

오브젝트가 단일 큐인 경우에만 공백이 아닌 값이 리턴됩니다. 오브젝트가 분배 목록이거나 주제인 경우 리턴된 값이 정의되지 않습니다.

이 필드는 출력 필드입니다. 이 필드의 길이는 LNQMN에서 제공합니다. 이 필드의 초기값은 48개의 빈 문자입니다.

### **PMRQN(48바이트 문자 문자열)**

해석된 목적지 큐의 이름.



이는 이름 해석이 로컬 큐 관리자에 의해 수행된 이후 목적지 큐의 이름입니다. 리턴되는 이름은 *PMRMO*(으)로 식별되는 큐 관리자에 존재하는 큐의 이름입니다.

오브젝트가 단일 큐인 경우에만 공백이 아닌 값이 리턴됩니다. 오브젝트가 분배 목록이거나 주제인 경우 리턴된 값이 정의되지 않습니다.

이 필드는 출력 필드입니다. 이 필드의 길이는 LNQN으로 제공됩니다. 이 필드의 초기값은 48개의 빈 문자입니다.

### **PMRRO(10자리의 부호 있는 정수)**

MQPMO의 시작에서부터 첫 번째 응답 레코드의 오프셋.

이는 MQPMO 구조의 시작에서 첫 번째 MQRR 응답 레코드의 오프셋(바이트)입니다. 오프셋은 양수 또는 음수일 수 있습니다. *PMRRO*은(는) 메시지가 분배 목록에 놓이는 경우에만 사용됩니다. *PMREC*이(가) 0인 경우 필드는 무시됩니다.

메시지를 분배 목록에 넣을 때, 메시지가 성공적으로 전송되지 않은 큐(MQRR의 *RRCC* 필드) 및 각 실패의 이유(MQRR의 *RRREA* 필드)를 식별하기 위해 하나 이상의 MQRR 응답 레코드 배열을 제공할 수 있습니다. 큐를 여는 데 실패했거나 Put 조작이 실패했기 때문에 메시지가 송신되지 않았을 수 있습니다. 큐 관리자는 호출 결과가 혼합된 경우에만 응답 레코드를 설정합니다(즉, 일부 메시지는 성공적으로 송신되었거나 다른 메시지는 실패했거나 모두 실패했지만 다른 이유로 실패했습니다). 원인 코드 RC2136은 이 경우를 나타냅니다. 동일한 이유 코드가 모든 큐에 적용되면 해당 이유가 MQPUT 또는 MQPUT1 호출의 **REASON** 매개변수에 리턴되고 응답 레코드가 설정되지 않습니다.

일반적으로 분배 목록이 열릴 때 MQOD에서 지정한 오브젝트 레코드만큼 많은 응답 레코드가 있어야 합니다. 필요한 경우 각 응답 레코드는 해당 오브젝트 레코드에서 식별하는 큐에 넣기 위한 완료 코드 및 이유 코드로 설정됩니다. 분배 목록의 큐는 배열의 적절한 위치에 응답 레코드가 할당되어 있어야 하지만 넣기 조작 대신 열기 조작의 결과로 완료 코드 및 이유 코드로 설정됩니다.

응답 레코드의 수는 오브젝트 레코드의 수와 다를 수 있습니다. 오브젝트 레코드보다 응답 레코드 수가 적으면 애플리케이션이 넣기 조작이 실패한 모든 목적지 또는 실패 이유를 식별할 수 없습니다. 오브젝트 레코드보다 응답 레코드가 많은 경우 초과분은 사용되지 않습니다(여전히 액세스해야 하는 경우에도). 응답 레코드는 선택사항이지만 제공되는 경우 *PMREC*이어야 합니다.

응답 레코드는 *PMRRO*에서 오프셋을 지정하거나 *PMRRP*에 주소를 지정하여 MQOD의 오브젝트 레코드와 유사한 방법으로 제공될 수 있습니다. 이를 수행하는 방법에 대한 자세한 정보는 1109 페이지의 『IBM i의 MQOD(오브젝트 디스크립터)』에 설명된 *ODORO* 필드를 참조하십시오. 그러나 둘 이상의 *PMRRO* 및 *PMRRP*는 사용할 수 없습니다. 모두 0이 아닌 경우 이유 코드 RC2156으로 호출에 실패합니다.

MQPUT1 호출의 경우 이 필드는 0이어야 합니다. 이는 응답 정보(필요하면)가 오브젝트 디스크립터 MQOD에서 지정한 응답 레코드에서 리턴되기 때문입니다.

입력 필드입니다. 이 필드의 초기값은 0입니다. *PMVER*이(가) *PMVER2*보다 작으면 이 필드는 무시됩니다.

### **PMRRP(pointer)**

첫 번째 응답 레코드 주소.

이는 첫 번째 MQRR 응답 레코드의 주소입니다. *PMRRP*은(는) 메시지가 분배 목록에 놓이는 경우에만 사용됩니다. *PMREC*이(가) 0인 경우 필드는 무시됩니다.

*PMRRP* 또는 *PMRRO*을(를) 사용하여 응답 레코드를 지정할 수 있지만 둘 다 지정할 수는 없습니다. 자세한 정보는 *PMRRO* 필드의 설명을 참조하십시오. *PMRRP*을(를) 사용하지 않으면 널 포인터 또는 널 바이트로 설정해야 합니다.

MQPUT1 호출의 경우 이 필드는 널 포인터 또는 널 바이트여야 합니다. 이는 응답 정보(필요하면)가 오브젝트 디스크립터 MQOD에서 지정한 응답 레코드에서 리턴되기 때문입니다.

입력 필드입니다. 이 필드의 초기값은 널 포인터입니다. *PMVER*이(가) *PMVER2*보다 작으면 이 필드는 무시됩니다.

### **PMSID(4바이트 문자 문자열)**

구조 ID입니다.

값은 다음과 같아야 합니다.

**PMSIDV**

Put 메시지 옵션 구조의 ID.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 PMSIDV입니다.

**PMSL(MQLONG)**

이 발행물로 대상이 지정된 구독의 레벨입니다.

이 값 이하의 PMSL이(가) 가장 높은 구독만 이 발행물을 수신합니다. 이 값은 0 - 9 범위에 있어야 하며, 0이 가장 낮은 레벨입니다.

이 필드의 초기값은 9입니다.

**PMTO(10자리의 부호 있는 정수)**

예약됨

이 필드는 예약 필드이며 값은 중요하지 않습니다. 이 필드의 초기값은 -1입니다.

**PMUDC(10자리의 부호 있는 정수)**

리모트 큐로 전송된 메시지의 수.

이는 현재 MQPUT 또는 MQPUT1 호출에서 리모트 큐로 분석되는 분배 목록의 큐에 송신한 메시지의 수입니다. 큐 관리자가 분배 목록 양식에 임시로 보유하는 메시지는 해당 분배 목록에 포함되어 있는 개별 목적지의 수로 간주됩니다. 또한 이 필드는 분배 목록에 없는 단일 큐에 메시지를 넣을 때에도 설정됩니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 0입니다. PMVER이(가) PMVER2보다 작은 경우 이 필드는 설정되지 않습니다.

**PMVER(10자리의 부호 있는 정수)**

구조 버전 번호입니다.

값은 다음 중 하나여야 합니다.

**PMVER1**

버전 1 Put 메시지 옵션 구조.

**PMVER2**

버전 2 Put 메시지 옵션 구조.

최신 버전의 구조에만 있는 필드는 필드의 설명에서 최신 필드로 식별됩니다. 다음 상수는 현재 버전의 버전 번호를 지정합니다.

**PMVERC**

현재 버전의 Put 메시지 옵션 구조.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 PMVER1입니다.

**초기값**

표 719. MQPMO의 필드		
필드 이름	상수의 이름	상수의 값
PMSID	PMSIDV	'PMO-'
PMVER	PMVER1	1
PMOPT	PMNONE	0
PMTO	없음	-1
PMCT	없음	0
PMKDC	없음	0

표 719. MQPMO의 필드 (계속)

필드 이름	상수의 이름	상수의 값
PMUDC	없음	0
PMIDC	없음	0
PMRQN	없음	공백
PMRMN	없음	공백
PMREC	없음	0
PMPRF	PFNONE	0
PMPRO	없음	0
PMRRO	없음	0
PMPRP	없음	널 포인터 또는 널 바이트
PMRRP	없음	널 포인터 또는 널 바이트

**참고:**

1. ~ 기호는 단일 공백 문자를 나타냅니다.

**RPG 선언**

```

D*..1.....2.....3.....4.....5.....6.....7..
D* MQPMO Structure
D*
D* Structure identifier
D PMSID 1 4 INZ('PMO ')
D* Structure version number
D PMVER 5 8I 0 INZ(1)
D* Options that control the action of MQPUT and MQPUT1
D PMOPT 9 12I 0 INZ(0)
D* Reserved
D PMTO 13 16I 0 INZ(-1)
D* Object handle of input queue
D PMCT 17 20I 0 INZ(0)
D* Number of messages sent successfully to local queues
D PMKDC 21 24I 0 INZ(0)
D* Number of messages sent successfully to remote queues
D PMUDC 25 28I 0 INZ(0)
D* Number of messages that could not be sent
D PMIDC 29 32I 0 INZ(0)
D* Resolved name of destination queue
D PMRQN 33 80 INZ
D* Resolved name of destination queue manager
D PMRMN 81 128 INZ
D* Number of put message records or response records present
D PMREC 129 132I 0 INZ(0)
D* Flags indicating which MQPMR fields are present
D PMPRF 133 136I 0 INZ(0)
D* Offset of first put message record from start of MQPMO
D PMPRO 137 140I 0 INZ(0)
D* Offset of first response record from start of MQPMO
D PMRRO 141 144I 0 INZ(0)
D* Address of first put message record
D PMPRP 145 160* INZ(*NULL)
D* Address of first response record
D PMRRP 161 176* INZ(*NULL)
D* Original message handle
D PMOMH 177 184I 0
D* New message handle
D PMNMH 185 190I 0
D* The action being performed
D PMACT 191 194I 0
D* Reserved
D PMRE1 195 198I 0
    
```

MQPMR 구조는 메시지를 분배 목록에 넣을 때 단일 목적지의 다양한 메시지 특성을 지정하는 데 사용됩니다.

## 개요

**목적:** MQPMR은 MQPUT 및 MQPUT1 호출을 위한 입출력(I/O) 구조입니다.

**문자 세트 인코딩:** MQPMR의 데이터는 **CodedCharSetId** 큐 관리자 속성에 의해 지정된 문자 세트 및 ENNAT에 의해 지정된 로컬 큐 관리자의 인코딩에 있어야 합니다. 그러나 애플리케이션이 IBM MQ 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트와 인코딩으로 작성되어야 합니다.

**사용법:** MQPUT 또는 MQPUT1 호출에 이러한 구조의 배열을 제공하여 분배 목록에서 각 목적지 큐의 다양한 값을 지정할 수 있습니다. 일부 필드는 입력 전용이며 다른 필드는 입출력(I/O)용입니다.

**참고:** 이 구조는 고정 레이아웃이 없다는 점에서 혼치 않습니다. 이 구조의 필드는 선택적이고 각 필드의 존재 또는 부재가 MQPMO의 *PMPRF* 필드의 플래그에 의해 표시됩니다. 존재하는 필드는 **다음 순서대로 발생해야 합니다.**

- *PRMID*
- *PRCID*
- *PRGID*
- *PRFB*
- *PRACC*

없는 필드는 레코드에서 공백을 차지하지 않습니다.

MQPMR에 고정된 레이아웃이 없기 때문에 COPY 파일에 제공된 정의가 없습니다. 애플리케이션 프로그래머는 애플리케이션에서 필요한 필드를 포함하는 선언을 작성하고 존재하는 필드를 표시하기 위해 *PMPRF*에 플래그를 설정해야 합니다.

- [1136 페이지의 『필드』](#)
- [1137 페이지의 『초기값』](#)
- [1137 페이지의 『RPG 선언』](#)

## 필드

MQPMR 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

### PRACC(32바이트 비트 문자열)

계정 토큰.

MQOPEN 또는 MQPUT1 호출에 제공된 MQOR 구조 배열에서 해당하는 요소가 지정한 이름의 큐에 메시지를 송신하는 데 사용될 계정 토큰입니다. 단일 큐에 넣기 위해 MQMD에서 *MDACC* 필드와 동일한 방법으로 처리됩니다. 이 필드의 콘텐츠에 대한 정보는 [1063 페이지의 『IBM i의 MQMD\(메시지 디스크립터\)』](#)에서 *MDACC*의 설명을 참조하십시오.

이 필드가 없는 경우 MQMD의 값이 사용됩니다.

입력 필드입니다.

### PRCID(24바이트 비트 문자열)

상관 ID.

MQOPEN 또는 MQPUT1 호출에 제공된 MQOR 구조 배열에서 해당하는 요소가 지정한 이름의 큐에 메시지를 송신하는 데 사용될 상관 ID입니다. 단일 큐에 넣기 위해 MQMD에서 *MDCID* 필드와 동일한 방법으로 처리됩니다.

이 필드가 MQPMR 레코드에 존재하지 않거나 목적지보다 적은 MQPMR 레코드가 있는 경우 MQMD의 값은 *PRCID* 필드를 포함하는 MQPMR 레코드를 포함하지 않는 목적지에 사용됩니다.

PMNCID가 지정되는 경우 단일 새 상관 ID가 생성되고 MQPMR 레코드가 있는지 여부에 관계없이 분배 목록의 모든 목적지에 사용됩니다. 이는 PMNMID가 처리되는 방법과 다릅니다(*PRMID* 필드 참조).

이 필드는 입출력(I/O) 필드입니다.

### PRFB(10자리의 부호 있는 정수)

피드백 또는 이유 코드.

MQOPEN 또는 MQPUT1 호출에 제공된 MQOR 구조 배열에서 해당하는 요소가 지정한 이름의 큐에 메시지를 송신하는 데 사용될 피드백 코드입니다. 단일 큐에 넣기 위해 MQMD에서 *MDFB* 필드와 동일한 방법으로 처리됩니다.

이 필드가 없는 경우 MQMD의 값이 사용됩니다.

입력 필드입니다.

### PRGID(24바이트 비트 문자열)

그룹 ID.

MQOPEN 또는 MQPUT1 호출에 제공된 MQOR 구조 배열에서 해당하는 요소가 지정한 이름의 큐에 메시지를 송신하는 데 사용될 그룹 ID입니다. 단일 큐에 넣기 위해 MQMD에서 *MDGID* 필드와 동일한 방법으로 처리됩니다.

이 필드가 MQPMR 레코드에 존재하지 않거나 목적지보다 적은 MQPMR 레코드가 있는 경우 MQMD의 값은 *PRGID* 필드를 포함하는 MQPMR 레코드를 포함하지 않는 목적지에 사용됩니다. 이 값은 1126 페이지의 표 716에서 문서화된 대로 처리되지 않지만 다음 차이점을 포함합니다.

- 새 그룹 ID가 사용될 경우 큐 관리자는 각 목적지에 대해 서로 다른 그룹 ID를 생성합니다. (즉, 동일한 그룹 ID를 갖는 두 개의 목적지는 없습니다.)
- 필드 값이 사용되는 경우 이유 코드 RC2258로 호출에 실패합니다.

이 필드는 입출력(I/O) 필드입니다.

### PRMID(24바이트 비트 문자열)

메시지 ID.

MQOPEN 또는 MQPUT1 호출에 제공된 MQOR 구조 배열에서 해당하는 요소가 지정한 이름의 큐에 메시지를 송신하는 데 사용될 메시지 ID입니다. 단일 큐에 넣기 위해 MQMD에서 *MDMID* 필드와 동일한 방법으로 처리됩니다.

이 필드가 MQPMR 레코드에 존재하지 않거나 목적지보다 적은 MQPMR 레코드가 있는 경우 MQMD의 값은 *PRMID* 필드를 포함하는 MQPMR 레코드를 포함하지 않는 목적지에 사용됩니다. 해당 값이 MINONE인 경우 새 메시지 ID는 각 목적지에 대해 생성됩니다(즉, 두 목적지 중 동일한 메시지 ID를 포함한 목적지가 없음).

PMNMID가 지정되는 경우 새 메시지 ID는 MQPMR 레코드가 있는지 여부에 관계없이 분배 목록에서 모든 목적지에 생성됩니다. 이는 PMNCID가 처리되는 방법과 다릅니다(*PRCID* 필드 참조).

이 필드는 입출력(I/O) 필드입니다.

## 초기값

구조 선언도 제공되지 않기 때문에 이 구조에 대해 정의된 초기값이 없습니다. 다음 예제 선언은 모든 필드가 필요한 경우 구조가 애플리케이션 프로그래머에 의해 선언되어야 하는 방법을 표시합니다.

## RPG 선언

```
D*..1.....2.....3.....4.....5.....6.....7..
D* MQPMR Structure
D*
D* Message identifier
D PRMID 1 24
D* Correlation identifier
D PRCID 25 48
D* Group identifier
```

D PRGID	49	72
D* Feedback or reason code		
D PRFB	73	76I 0
D* Accounting token		
D PRACC	77	108

## IBM i IBM i의 MQRFH(규칙 및 형식화 헤더)

MQRFH 구조는 규칙 및 형식화 헤더의 레이아웃을 정의합니다.

### 개요

**목적:** 이 헤더는 이름-값 쌍의 형식으로 문자열 데이터를 전송하는 데 사용할 수 있습니다.

**형식 이름:** FMRFH.

**문자 세트 및 인코딩:** MQRFH 구조의 필드는(RFNVS 포함) MQRFH에 선행하는 헤더 구조의 MDCSI 및 MDENC 필드에 의해 제공된 문자 세트 및 인코딩에 있거나 MQRFH가 애플리케이션 메시지 데이터의 시작 부분에 있는 경우 MQMD 구조의 해당 필드에 의해 제공됩니다.

문자 세트는 큐 이름에 유효한 문자에 사용되는 단일 바이트 문자가 있어야 합니다.

- [1138 페이지의 『필드』](#)
- [1140 페이지의 『초기값』](#)
- [1140 페이지의 『RPG 선언』](#)

### 필드

MQRFH 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

#### RFCSI(10자리의 부호 있는 정수)

RFNVS를 뒤에 오는 데이터의 문자 세트 ID.

RFNVS 뒤에 오는 데이터의 문자 세트 ID를 지정합니다. MQRFH 구조 자체의 문자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 다음 특수 값을 사용할 수 있습니다.

#### CSINHT

이 구조의 문자 세트 ID를 상속합니다.

이 구조 다음에 오는 데이터의 문자 데이터는 이 구조와 동일한 문자 세트로 되어 있습니다.

큐 관리자가 구조의 실제 문자 세트 ID에 메시지로 송신한 구조에서 이 값을 변경합니다. 오류가 발생하지 않으면 MQGET 호출에서 CSINHT 값을 리턴하지 않습니다.

MQMD의 MDPAT 필드 값이 ATBRKR이면 CSINHT를 사용할 수 없습니다.

이 필드의 초기값은 CSUNDF입니다.

RFNVS 뒤에 오는 데이터의 숫자 인코딩.

RFNVS 뒤에 오는 데이터의 숫자 인코딩을 지정합니다. MQRFH 구조 자체의 숫자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다.

이 필드의 초기값은 ENNAT입니다.

#### RFFLG(10자리의 부호 있는 정수)

플래그입니다.

다음을 지정할 수 있습니다.

## RFNONE

플래그가 없습니다.

이 필드의 초기값은 RFNONE입니다.

## RFFMT(8바이트 문자열)

RFNVS 뒤에 오는 데이터의 형식 이름.

RFNVS 뒤에 오는 데이터의 형식명을 지정합니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 이 필드의 코딩 규칙은 MQMD의 MDFMT 필드에 대한 규칙과 동일합니다.

이 필드의 초기값은 FMNONE입니다.

## RFLEN(10자리의 부호 있는 정수)

RFNVS를 포함한 MQRFH의 전체 길이.

이는 구조의 끝에 있는 RFNVS 필드를 포함하여 MQRFH 구조의 길이(바이트)입니다. 길이에 RFNVS 필드 다음에 오는 사용자 데이터가 포함되지 않습니다.

일부 환경에서 사용자 데이터의 데이터 변환 문제를 방지하기 위해 RFLEN을 4의 배수로 사용하는 것을 고려하십시오.

다음 상수는 구조의 고정된 부분의 길이(즉, RFNVS 필드를 제외한 길이)를 제공합니다.

## RFLENV

MQRFH 구조에서 고정 부분의 길이.

이 필드의 초기값은 RFLENV입니다.

## RFNVS(n바이트 문자열)

이름-값 쌍을 포함하는 문자열.

다음 양식으로 이름-값 쌍을 포함하는 가변 길이 문자열입니다.

```
name1 value1 name2 value2 name3 value3 ...
```

각 이름 또는 값은 하나 이상의 공백 문자열을 사용하여 근접한 이름 또는 값과 분리해야 합니다. 이 때 사용된 공백은 중요하지 않습니다. 따옴표 문자가 포함된 이름 또는 값을 앞 및 뒤에 붙여 이름 또는 값에 중요한 공백을 포함할 수 있습니다. 여는 따옴표와 일치하는 닫는 따옴표 사이의 모든 문자는 중요한 것으로 처리됩니다. 다음 예에서 이름은 FAMOUS\_WORDS이고 값은 Hello World입니다.

```
FAMOUS_WORDS "Hello World"
```

이름 또는 값은 널 문자 이외의 문자를 포함할 수 있습니다(RFNVS의 구분 기호로 수행). 그러나 상호운용성을 지원하기 위해 애플리케이션은 다음 문자로 이름을 제한하는 것을 선호할 수 있습니다.

- 첫 번째 문자: 대문자 또는 소문자 알파벳(A - Z 또는 a - z) 또는 대문자.
- 후속 문자: 대문자 또는 소문자 영문자, 10진수(0 - 9), 밑줄, 하이픈 또는 점.

이름 또는 값에 하나 이상의 따옴표가 포함되면 이름 또는 값은 따옴표로 묶어야 하고 문자열 내의 각 따옴표는 두배가 되어야 합니다.

```
Famous_Words "The program displayed ""Hello World"""
```

이름과 값은 대소문자를 구분합니다. 즉, 소문자는 대문자와 동일한 것으로 간주되지 않습니다. 예를 들어, FAMOUS\_WORDS 및 Famous\_Words는 두 개의 다른 이름입니다.

RFNVS의 길이(바이트)는 RFLEN에서 RFLENV를 뺀 값과 동일합니다. 일부 환경에서 사용자 데이터의 데이터 변환 문제를 방지하기 위해 이 길이는 4의 배수인 것이 좋습니다. RFNVS는 이 길이에 공백으로 채워지거나

나 문자열의 마지막 중요 문자 뒤에 오는 널 문자를 배치하여 초기 종료됩니다. *RFNV*S의 지정된 길이까지 이 뒤에 오는 널 문자 및 바이트는 무시됩니다.

**참고:** 이 필드의 길이가 고정되어 있지 않기 때문에, 지원되는 프로그래밍 언어를 위해 제공되는 구조의 선언에서 필드는 생략됩니다.

**RFSID(4바이트 문자열)**

구조 ID입니다.

값은 다음과 같아야 합니다.

**RFSIDV**

규칙 및 형식화 헤더 구조의 ID.

이 필드의 초기값은 RFSIDV입니다.

**RFVER(10자리의 부호 있는 정수)**

구조 버전 번호입니다.

값은 다음과 같아야 합니다.

**RFVER1**

버전-1 규칙 및 형식화 헤더 구조 ID.

이 필드의 초기값은 RFVER1입니다.

**초기값**

표 720. MQRFH의 필드		
필드 이름	상수의 이름	상수의 값
RFSID	RFSIDV	'RFH~'
RFVER	RFVER1	1
RFLEN	RFLENV	32
RFENC	ENNAT	환경에 따라 다름
RFCSI	CSUNDF	0
RFFMT	FMNONE	공백
RFFLG	RFNONE	0

**참고:**  
1. ~ 기호는 단일 공백 문자를 나타냅니다.

**RPG 선언**

```
D*..1.....2.....3.....4.....5.....6.....7..
D* MQRFH Structure
D*
D* Structure identifier
D RFSID          1      4    INZ('RFH ')
D* Structure version number
D RFVER          5      8I 0 INZ(1)
D* Total length of MQRFH includingNameValueString
D RFLEN          9     12I 0 INZ(32)
D* Numeric encoding of data that followsNameValueString
D RFENC         13     16I 0 INZ(273)
D* Character set identifier of data thatfollows NameValueString
D RFCSI         17     20I 0 INZ(0)
D* Format name of data that followsNameValueString
D RFFMT         21     28    INZ(' ')
```



## IBM i IBM i의 MQRFH2(규칙 및 형식화 헤더 1)

MQRFH2 구조는 버전-2 규칙 및 형식화 헤더의 형식을 정의합니다.

### 개요

**목적:** 이 헤더는 XML-유사 구문을 사용하여 인코딩된 데이터를 전송하는 데 사용할 수 있습니다. 메시지는 두 개 이상의 MQRFH 구조를 연속적으로 포함할 수 있으며 사용자 데이터는 선택적으로 시리즈의 마지막 MQRFH 구조 뒤에 옵니다.

**형식 이름:** FMRFH2.

**문자 세트 및 인코딩:** 특별 규칙이 MQRFH2 구조에 사용된 문자 세트 및 인코딩에 적용합니다.

- *RF2NVD* 이외의 필드는 MQRFH2에 선행하는 헤더 구조의 *MDCSI* 및 *MDENC* 필드에 의해 제공된 문자 세트 및 인코딩에 있거나 MQRFH2가 애플리케이션 메시지 데이터의 시작 부분에 있는 경우 MQMD 구조의 해당 필드에 의해 제공됩니다.

문자 세트는 큐 이름에 유효한 문자에 사용되는 단일 바이트 문자가 있어야 합니다.

GMCONV가 MQGET 호출에 지정될 때 큐 관리자는 요청된 문자 세트 및 인코딩으로 이러한 필드를 변환합니다.

- *RF2NVD*는 *RF2NVC* 필드에서 제공하는 문자 세트입니다. 특정 유니코드 문자 세트만 *RF2NVC*에 유효합니다 (세부사항은 *RF2NVC*에 대한 설명 참조).

일부 문자 세트에는 인코딩에 종속한 표현이 있습니다. *RF2NVC*가 해당 문자 세트 중 하나인 경우 *RF2NVD*는 MQRFH2의 다른 필드와 동일한 인코딩에 있어야 합니다.

GMCONV가 MQGET 호출에 지정될 때 큐 관리자는 *RF2NVD*를 요청된 인코딩으로 변환하지만 해당 문자 세트를 변경하지 않습니다.

- [1141 페이지의 『필드』](#)
- [1146 페이지의 『초기값』](#)
- [1146 페이지의 『RPG 선언』](#)

### 필드

MQRFH2 구조에는 다음과 같은 필드가 포함됩니다. 필드에 대해서 알파벳순으로 설명합니다.

#### RF2CSI(10자리의 부호 있는 정수)

마지막 *RF2NVD* 필드 뒤에 오는 데이터의 문자 세트 ID.

이는 마지막 *RF2NVD* 필드 뒤에 오는 데이터의 문자 세트 ID를 지정합니다. MQRFH2 구조 자체에서 문자 데이터에 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 다음 특수 값을 사용할 수 있습니다.

#### CSINHT

이 구조의 문자 세트 ID를 상속합니다.

이 구조 다음에 오는 데이터의 문자 데이터는 이 구조와 동일한 문자 세트로 되어 있습니다.

큐 관리자가 구조의 실제 문자 세트 ID에 메시지로 송신한 구조에서 이 값을 변경합니다. 오류가 발생하지 않으면 MQGET 호출에서 CSINHT 값을 리턴하지 않습니다.

MQMD의 *MDPAT* 필드 값이 ATBRKR이면 CSINHT를 사용할 수 없습니다.

이 필드의 초기값은 CSINHT입니다.

### **RF2ENC(10자리의 부호 있는 정수)**

마지막 *RF2NVD* 필드 뒤에 오는 데이터의 숫자 인코딩.

마지막 *RF2NVD* 필드 뒤에 오는 데이터의 숫자 인코딩을 지정합니다. *MQRFH2* 구조 자체의 숫자 데이터에는 적용되지 않습니다.

*MQPUT* 또는 *MQPUT1* 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다.

이 필드의 초기값은 *ENNAT*입니다.

### **RF2FLG(10자리의 부호 있는 정수)**

플래그입니다.

다음 값을 지정할 수 있습니다.

#### **RFNONE**

플래그가 없습니다.

이 필드의 초기값은 *RFNONE*입니다.

### **RF2FMT(8바이트 문자열)**

마지막 *RF2NVD* 필드 뒤에 오는 데이터의 형식 이름.

마지막 *RF2NVD* 필드 뒤에 오는 데이터의 형식 이름을 지정합니다.

*MQPUT* 또는 *MQPUT1* 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 이 필드의 코딩 규칙은 *MQMD*의 *MDFMT* 필드에 대한 규칙과 동일합니다.

이 필드의 초기값은 *FMNONE*입니다.

### **RF2LEN(10자리의 부호 있는 정수)**

*RF2NVL* 및 *RF2NVD* 필드 모두를 포함하는 *MQRFH2*의 총 길이.

이는 구조의 마지막 부분에 있는 *RF2NVL* 및 *RF2NVD* 필드를 포함하는 *MQRFH2* 구조의 길이(바이트)입니다. 구조의 마지막 부분에 있는 *RF2NVL* 및 *RF2NVD* 필드의 여러 쌍이 순서대로 있는 것에 유효합니다.

```
length1, data1, length2, data2, ...
```

*RF2LEN*은 구조의 끝에서 마지막 *RF2NVD* 필드 다음에 올 수 있는 사용자 데이터를 포함하지 않습니다.

일부 환경에서 사용자 데이터의 데이터 변환 문제를 방지하기 위해 *RF2LEN*을 4의 배수로 사용하는 것을 고려하십시오.

다음 상수는 구조의 고정된 파트 길이(즉, *RF2NVL* 및 *RF2NVD* 필드를 제외한 길이)를 제공합니다.

#### **RFLEN2**

*MQRFH2* 구조의 고정된 부분 길이.

이 필드의 초기값은 *RFLEN2*입니다.

### **RF2NVC(10자리의 부호 있는 정수)**

*RF2NVD*의 문자 세트 ID.

*RF2NVD* 필드에 있는 데이터의 코드화된 문자 세트 ID를 지정합니다. 이는 *MQRFH2* 구조에 있는 다른 문자열의 문자 세트와 다르고 구조의 마지막 부분에 있는 마지막 *RF2NVD* 필드 뒤에 오는 데이터의 문자 세트(있는 경우)와 다를 수 있습니다.

*RF2NVC*에는 다음 *CCSID*값 중 하나가 있어야 합니다.

#### **1200**

UTF-16, 가장 최근 지원된 유니코드 버전

#### **13488**

UTF-16, 유니코드 버전 2.0 서브세트

## 17584

UTF-16, 유니코드 버전 3.0 서브세트(유로 기호 포함)

## 1208

UTF-8, 가장 최근 지원된 유니코드 버전

UTF-16 문자 세트의 경우 *RF2NVD*의 인코딩(바이트 순서)이 *MQRFH2* 구조에서 기타 필드의 인코딩과 동일해야 합니다. 대리 문자('X'D800' - 'X'DFFF')는 지원되지 않습니다.

**참고:** *RF2NVC*에 이전에 나열된 값 중 하나가 없고 *MQRFH2* 구조에 *MQGET* 호출에 대한 변환이 필요한 경우 이유 코드 RC2111로 호출이 완료되며 메시지는 변환되지 않고 리턴됩니다.

이 필드의 초기값은 1208입니다.

### RF2NVD(n-바이트 문자열)

이름/값 데이터.

이는 XML형 구문을 사용하여 인코딩되는 데이터를 포함하는 가변 길이 문자열입니다. 이 문자열의 길이(바이트)는 *RF2NVD* 필드에 선행하는 *RF2NVL* 필드에서 제공됩니다. 이 길이는 4의 배수여야 합니다.

*RF2NVL* 및 *RF2NVD* 필드는 선택사항이지만, 존재하는 경우 쌍으로 발생하고 인접해야 합니다. 필드의 쌍은 필요한 만큼 반복될 수 있습니다. 예:

```
length1 data1 length2 data2 length3 data3
```

이 필드는 선택사항이기 때문에 지원되는 프로그래밍 언어를 위해 제공되는 구조의 선언에서 필드는 생략됩니다.

GMCONV 옵션을 사용하여 메시지를 검색할 때 *MQGET* 호출에 지정된 문자 세트로 변환되지 않기 때문에 *RF2NVD*는 혼치 않습니다. *RF2NVD*는 원래 문자 세트에 유지됩니다. 그러나 *RF2NVD*는 *MQGET* 호출에서 지정된 인코딩으로 변환됩니다.

**이름/값 데이터의 구문:** 문자열은 0 또는 추가 특성이 포함된 단일 "폴더"로 구성됩니다. 폴더는 폴더와 동일한 이름으로 XML 시작 및 종료 태그로 구분됩니다.

```
<folder> property1 property2 ... </folder>
```

*RF2NVL*에 의해 정의된 길이까지 폴더 종료 태그 뒤에 오는 문자는 비어 있어야 합니다. 폴더 내에서 각 특성은 이름 및 값, 선택적으로 데이터 유형으로 구성됩니다.

```
<name dt="datatype">value</name>
```

이러한 예에서:

- 구분 문자(<, =, ", / 및 >)는 표시된 대로 정확하게 지정되어야 합니다.
- name은 특성의 사용자 지정 이름입니다. 이름에 대한 자세한 정보는 다음 예제를 참조하십시오.
- datatype은 특성의 선택적 사용자 지정 데이터 유형입니다. 올바른 데이터 유형은 다음 예제를 참조하십시오.
- value는 특성의 사용자 지정 값입니다. 이름에 대한 자세한 정보는 다음 단락을 참조하십시오.
- 공백은 값에 선행하는 > 문자, 값 뒤에 오는 < 문자 및 dt=에 선행해야 하는 하나 이상의 공백 사이에 주요합니다. 다른 공백은 태그 사이 또는 태그 앞이나 뒤에서 자유롭게 코드화할 수 있습니다(예: 가독성 향상을 위해). 이러한 공백은 중요하지 않습니다.

특성이 서로 관련되면 그룹과 동일한 이름으로 XML 시작 및 종료 태그 내에 둘러싸여 함께 그룹화할 수 있습니다.

```
<folder> <group> property1 property2 ... </group> </folder>
```

그룹은 한계없이 다른 그룹 내에 중첩될 수 있고 그룹이 폴더 내에 두 번 이상 발생할 수 있습니다. 또한 폴더에 그룹의 일부 속성과 그룹이 아닌 다른 속성을 포함하는 것이 유효합니다.

**특성, 그룹 및 폴더의 이름:** 특성, 그룹 및 폴더의 이름이 콜론 문자를 제외하고는 유효한 XML 태그 이름이어야 하며 특성, 그룹 또는 폴더 이름에서 허락되지 않습니다. 특히, 다음과 같습니다.

- 이름은 문자 또는 밑줄로 시작해야 합니다. 유효한 문자는 W3C XML 스펙에서 정의되고 유니코드 범주 Ll, Lu, Lo, Lt 및 N1로 기본적으로 구성됩니다.
- 이름에서 나머지 문자는 문자, 10진수 숫자, 밑줄, 하이픈 또는 도트일 수 있습니다. 이는 유니코드 범주 Ll, Lu, Lo, Lt, Nl, Mc, Mn, Lm 및 Nd에 해당됩니다.
- 유니코드 호환성 문자('F900' 이상)는 이름의 부분에서도 허락되지 않습니다.
- 이름은 대문자 또는 소문자의 혼합에서 XML로 시작하지 않아야 합니다.

또한 다음이 포함되어 있습니다.

- 이름은 대소문자를 구분합니다. 예를 들어, ABC, abc 및 Abc는 세가지 다른 이름입니다.
- 각 폴더에는 개별 네임스페이스가 있습니다. 따라서 한 폴더의 그룹 또는 특성은 다른 폴더에서 동일한 이름의 그룹 또는 특성과 충돌하지 않습니다.
- 그룹 및 특성은 폴더 내에 동일한 네임스페이스를 차지합니다. 따라서 특성은 해당 특성을 포함하는 폴더 내에 그룹과 동일한 이름을 가지고 있을 수 없습니다.

일반적으로 해당 특성 또는 그룹이 올바르게 형성되면 *RF2NVD* 필드를 분석하는 프로그램은 프로그램이 인식하지 않는 이름이 있는 특성 또는 그룹을 무시해야 합니다.

**특성의 데이터 유형:** 각 특성이 선택적 데이터 유형을 가질 수 있습니다. 지정되는 경우 데이터 유형은 상단, 하단 또는 혼합 문자로 다음 값 중 하나여야 합니다.

표 721. 데이터 유형 및 해당 사용법	
데이터 유형	사용됨
string	일련의 문자입니다. 특정 문자는 이스케이프 순서표를 사용하여 지정되어야 합니다.
boolean	특징 0 또는 1(1이 TRUE을 표시합니다).
bin.hex	옥텟을 나타내는 16진 숫자.
i1	-128에서 +127까지의 범위에 있는 정수가 10진수 숫자 및 선택적 부호만 사용하여 표시됩니다.
i2	-32에서에서 +127까지의 범위에 있는 정수가 10진수 숫자 및 선택적 부호만 사용하여 표시됩니다.
i4	-2 147 483 648에서 +2 147 483 647까지의 범위에 있는 정수가 10진수 숫자 및 선택적 부호만 사용하여 표시됩니다.
i8	-9 223 372 036 854 775에서 +9 223 372 2230 2231 2232 807까지의 범위에 있는 정수가 10진수 숫자 및 선택적 부호만 사용하여 표시됩니다.
int	-9 223 372 036 854 775에서 +9 223 372 2230 2231 2232 807까지의 범위에 있는 정수가 10진수 숫자 및 선택적 부호만 사용하여 표시됩니다. 송신자가 특정 정밀도를 암시하지 않으려는 경우 i1, i2, i4 또는 i8 대신 사용할 수 있습니다.
r4	1.175E-37에서 3.402 823 47E+38까지 범위 규모의 부동 소수점이 10진수 숫자, 선택적 부호, 선택적 분수 자리 및 선택적 지수를 사용하여 표시됩니다.

표 721. 데이터 유형 및 해당 사용법 (계속)	
데이터 유형	사용법
r8	2.225E-307에서 1.797 693 134+308까지 범위 규모의 부동 소수점이 10진수 숫자, 선택적 부호, 선택적 분수 자리 및 선택적 지수를 사용하여 표시됩니다.

**특성 값:** 특성 값은 연관된 필수 이스케이프 순서가 있는 특수 문자를 제외하고 임의 문자로 구성될 수 있습니다. 다음 표에서 "필수"로 표시되는 문자 값에서의 각 발생은 해당 이스케이프 순서로 대체되어야 합니다. 표에서도 연관되는 선택적 이스케이프 순서가 있는 문자를 표시합니다. "필수"로 표시된 문자 값의 각 발생은 해당하는 이스케이프 순서에 의해 대체되어야 하지만 이는 필요하지 않습니다.

표 722. 이스케이프 문자 및 해당 사용법		
문자	이스케이프 순서	사용법
&	&amp;	필수
<	<	필수
>	&gt;	선택사항
"	&quot;	선택사항
'	&apos;	선택사항

**참고:** 이스케이프 시퀀스 시작 부분의 & 문자는 &amp; 로 대체하지 않아야 합니다.

다음 예제에서 값의 공백은 중요지만 이스케이프 순서는 필요하지 않습니다.

```
<Famous_Words>The program displayed "Hello World"</Famous_Words>
```

### RF2NVL(10자리의 부호 있는 정수)

RF2NVD의 길이.

RF2NVD 필드 안에 있는 데이터의 길이(바이트)를 지정합니다. RF2NVD 필드 뒤에 오는 데이터의 데이터 변환 문제(있는 경우)를 피하기 위해 RF2NVL은 4의 배수여야 합니다.

**참고:** RF2NVL 및 RF2NVD 필드는 선택사항이지만, 존재하는 경우 쌍으로 발생하고 인접해야 합니다. 필드의 쌍은 필요한 만큼 반복될 수 있습니다. 예:

```
length1 data1 length2 data2 length3 data3
```

이 필드는 선택사항이기 때문에 지원되는 프로그래밍 언어를 위해 제공되는 구조의 선언에서 필드는 생략됩니다.

### RF2SID(4바이트 문자열)

구조 ID입니다.

값은 다음과 같아야 합니다.

#### RFSIDV

규칙 및 형식화 헤더 구조의 ID.

이 필드의 초기값은 RFSIDV입니다.

### RF2VER(10자리의 부호 있는 정수)

구조 버전 번호입니다.

값은 다음과 같아야 합니다.

## RFVER2

버전-2 규칙 및 형식화 헤더 구조.

이 필드의 초기값은 RFVER2입니다.

### 초기값

표 723. MQRFH2의 필드		
필드 이름	상수의 이름	상수의 값
RF2SID	RFSIDV	'RFH-'
RF2VER	RFVER2	2
RF2LEN	RFLEN2	36
RF2ENC	ENNAT	환경에 따라 다름
RF2CSI	CSINHT	-2
RF2FMT	FMNONE	공백
RF2FLG	RFNONE	0
RF2NVC	없음	1208

### 참고:

1. - 기호는 단일 공백 문자를 나타냅니다.

### RPG 선언

```
D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQRFH2 Structure
D*
D* Structure identifier
D RF2SID          1      4    INZ('RFH ')
D* Structure version number
D RF2VER          5      8I 0 INZ(2)
D* Total length of MQRFH2 including allNameValueLength and
D* NameValueDatafields
D RF2LEN          9      12I 0 INZ(36)
D* Numeric encoding of data that followslast NameValueData field
D RF2ENC          13     16I 0 INZ(273)
D* Character set identifier of data thatfollows last NameValueData field
D RF2CSI          17     20I 0 INZ(-2)
D* Format name of data that follows lastNameValueData field
D RF2FMT          21     28    INZ(' ')
D* Flags
D RF2FLG          29     32I 0 INZ(0)
D* Character set identifier ofNameValueData
D RF2NVC          33     36I 0 INZ(1208)
```

## IBM i IBM i의 MQRMH(참조 메시지 헤더)

MQRMH 구조는 참조 메시지 헤더의 형식을 정의합니다.

### 개요

**목적:** 이 헤더는 하나의 큐 관리자에서 다른 큐 관리자까지 다량의 데이터(호출된 "벌크 데이터")를 전송하기 위해 사용자 작성 메시지 채널 엑시트와 함께 사용됩니다. 일반 메시징과의 차이점은 벌크 데이터가 큐에 저장되지 않는다는 것입니다. 대신 벌크 데이터에 대한 참조만 큐에 저장됩니다. 이는 몇 개의 대량 메시지로 소비되고 있는 IBM MQ 자원의 가능성을 감소시킵니다.

**형식 이름:** FMRMH.

**문자 세트 및 인코딩:** MQRMH의 문자 데이터 및 오프셋 필드로 처리된 문자열은 로컬 큐 관리자의 문자 세트에 있어야 합니다. 이는 **CodedCharSetId** 큐 관리자 속성에서 제공됩니다. MQRMH의 숫자 데이터는 고유 시스템 인코딩에 있어야 합니다. 이는 C 프로그래밍 언어의 ENNAT 값으로 제공됩니다.

MQRMH의 문자 세트 및 인코딩은 *MDCSI* 및 *MDENC* 필드에 설정되어야 합니다.

- MQMD(MQRMH 구조가 메시지 데이터의 시작에 있는 경우) 또는
- MQRMH 구조에 선행하는 헤더 구조(다른 모든 경우).

**사용법:** 애플리케이션은 MQRMH로 구성되지만 벌크 데이터를 생략하는 메시지를 넣습니다. 메시지가 메시지 채널 에이전트(MCA)에 의해 전송 큐에서 읽혀질 때 사용자 제공 메시지 엑시트가 참조 메시지 헤더를 처리하기 위해 호출됩니다. MCA에서 채널을 통해 메시지를 다음 큐 관리자에 송신하기 전에, 엑시트는 MQRMH 구조에서 식별한 벌크 데이터를 참조 메시지에 추가할 수 있습니다.

수신 끝에서 참조 메시지를 대기하는 메시지 엑시트가 존재해야 합니다. 참조 메시지가 수신될 때 엑시트는 메시지에서 MQRMH 뒤에 오는 벌크 데이터로부터 오브젝트를 작성한 후 벌크 데이터 없이 참조 메시지에 전달해야 합니다. 나중에 참조 메시지를 읽는 애플리케이션에서 벌크 데이터 없이 큐에서 참조 메시지를 검색할 수 있습니다.

일반적으로, MQRMH 구조는 메시지에 있는 전부입니다. 그러나 메시지가 전송 큐에 있는 경우 하나 이상의 추가 헤더가 MQRMH 구조에 선행합니다.

참조 메시지도 분배 목록에 송신할 수 있습니다. 이런 경우 메시지가 전송 큐에 있을 때 MQDH 구조 및 관련 레코드는 MQRMH 구조에 우선합니다.

**참고:** 메시지 엑시트를 올바르게 처리할 수 없기 때문에 참조 메시지는 세그먼트화된 메시지로 전송되지 않아야 합니다.

- [1147 페이지의 『데이터 변환』](#)
- [1147 페이지의 『필드』](#)
- [1151 페이지의 『초기값』](#)
- [1152 페이지의 『RPG 선언』](#)

## 데이터 변환

데이터 변환 목적의 경우 MQRMH 구조의 변환은 소스 환경 데이터의 변환, 소스 오브젝트 이름, 목적지 환경 데이터 및 목적지 오브젝트 이름을 포함합니다. 구조 시작의 *RMLEN* 바이트 내에 있는 기타 바이트는 제거되거나 데이터 변환 후에 정의되지 않은 값을 포함합니다. 다음 명령문이 모두 true인 변환된 벌크 데이터가 제공됩니다.

- 데이터 변환이 수행될 때 벌크 데이터는 메시지에 존재합니다.
- MQRMH의 *RMFMT* 필드에 FMNONE 이외의 값이 있습니다.
- 사용자 작성 데이터 변환 엑시트는 지정된 형식 이름으로 존재합니다.

그러나 메시지가 큐에 있는 경우 일반적으로 벌크 데이터가 메시지에 있지 않고 결과적으로 GMCONV 옵션이 벌크 데이터를 변환하지 않음에 주의하십시오.

## 필드

MQRMH 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

### RMCSI(10자리의 부호 있는 정수)

벌크 데이터의 문자 세트 ID.

이는 벌크 데이터의 문자 세트 ID를 지정합니다. MQRMH 구조 자체의 문자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 다음 특수 값을 사용할 수 있습니다.

### CSINHT

이 구조의 문자 세트 ID를 상속합니다.

이 구조 다음에 오는 데이터의 문자 데이터는 이 구조와 동일한 문자 세트로 되어 있습니다.

큐 관리자가 구조의 실제 문자 세트 ID에 메시지로 송신한 구조에서 이 값을 변경합니다. 오류가 발생하지 않으면 MQGET 호출에서 CSINHT 값을 리턴하지 않습니다.

MQMD의 *MDPAT* 필드 값이 ATBRKR이면 CSINHT를 사용할 수 없습니다.

이 필드의 초기값은 CSUNDF입니다.

#### **RMDEL(10자리의 부호 있는 정수)**

목적지 환경 데이터 길이.

이 필드가 0이면 목적지 환경 데이터가 없으며 *RMDEO*이(가) 무시됩니다.

#### **RMDEO(10자리의 부호 있는 정수)**

목적지 환경 데이터의 오프셋.

이 필드는 MQRMH 구조의 시작에서 목적지 환경 데이터의 오프셋을 지정합니다. 해당 데이터가 작성자로 알려진 경우 참조 메시지의 작성자가 목적지 환경 데이터를 지정할 수 있습니다. 예를 들어, 목적지 환경 데이터는 벌크 데이터가 저장되는 오브젝트의 디렉토리 경로일 수 있습니다. 그러나 작성자가 목적지 환경 데이터를 알지 못하는 경우 사용자 제공 메시지 엑시트에서 필요한 환경 정보가 있는지 판별해야 합니다.

목적지 환경 데이터의 길이는 *RMDEL*에 의해 지정됩니다. 이 길이가 0인 경우 목적지 환경 데이터가 없고 *RMDEO*가 무시됩니다. 존재하는 경우 목적지 환경 데이터는 구조의 시작에서 *RMLen* 바이트 내에 완전히 상주해야 합니다.

애플리케이션은 목적지 환경 데이터가 *RMSEO*, *RMSNO* 및 *RMDNO* 필드에서 처리되는 데이터와 근접하다고 가정하지 않아야 합니다.

이 필드의 초기값은 0입니다.

#### **RMDL(10자리의 부호 있는 정수)**

벌크 데이터 길이.

*RMDL* 필드는 MQRMH 구조가 참조하는 벌크 데이터의 길이를 지정합니다.

벌크 데이터가 메시지에 존재하는 경우 데이터는 MQRMH 구조의 시작에서 *RMLen* 바이트의 오프셋에 시작합니다. 전체 메시지의 길이에서 *RMLen*을(를) 뺀 값이 존재하는 벌크 데이터의 길이를 제공합니다.

데이터가 메시지에 있는 경우 *RMDL*은(는) 관련된 데이터의 양을 지정합니다. 일반적인 경우는 *RMDL*이(가) 메시지에 있는 데이터 길이와 동일한 값을 갖는 경우입니다.

MQRMH 구조가 오브젝트의 나머지 데이터를 나타내는 경우(지정된 논리 오프셋에서 시작) 벌크 데이터가 메시지에 존재하지 않으면 0값은 *RMDL*에 대해 사용될 수 있습니다.

데이터가 존재하지 않는 경우 MQRMH의 끝은 메시지 끝과 일치합니다.

이 필드의 초기값은 0입니다.

#### **RMDNL(10자리의 부호 있는 정수)**

목적지 오브젝트 이름 길이.

이 필드가 0이면 목적지 오브젝트 이름이 없으며 *RMDNO*이(가) 무시됩니다.

#### **RMDNO(10자리의 부호 있는 정수)**

목적지 오브젝트 이름의 오프셋.

이 필드는 MQRMH 구조의 시작에서 목적지 오브젝트 이름의 오프셋을 지정합니다. 해당 데이터가 작성자로 알려진 경우 참조 메시지의 작성자가 목적지 오브젝트 이름을 지정할 수 있습니다. 그러나 작성자가 목적지 오브젝트 이름을 알지 못하는 경우 사용자 제공 메시지 엑시트가 작성되거나 수정된 오브젝트를 식별해야 합니다.

목적지 오브젝트 이름의 길이는 *RMDNL*에 의해 제공됩니다. 이 길이가 0이면 목적지 오브젝트 이름이 없으며 *RMDNO*이(가) 무시됩니다. 존재하는 경우, 목적지 오브젝트 이름은 구조의 시작으로부터 *RMLen* 바이트 내에 완전히 상주해야 합니다.



애플리케이션은 목적지 오브젝트 이름이 *RMSEO*, *RMSNO* 및 *RMDEO* 필드에서 처리되는 데이터와 근접하다고 가정하지 않아야 합니다.

이 필드의 초기값은 0입니다.

#### **RMDO(10자리의 부호 있는 정수)**

벌크 데이터의 낮은 오프셋.

이 필드는 벌크 데이터가 일부를 구성하는 오브젝트의 시작에서 벌크 데이터의 낮은 오프셋을 지정합니다. 오브젝트의 시작에서 벌크 데이터의 오프셋은 논리 오프셋이라고 합니다. 이는 MQRMH 구조의 시작에서 벌크 데이터의 실제 오프셋이 아닙니다. 해당 오프셋은 *RMLen*으로 지정됩니다.

대형 오브젝트가 참조 메시지를 사용하여 송신되도록 허용하기 위해, 논리 오프셋이 두 개 필드로 나뉘며 이러한 두 필드의 합으로 실제 논리 오프셋이 지정됩니다.

- *RMDO*은 논리 오프셋이 1 000 000 000으로 나뉘질 때 획득된 나머지를 나타냅니다. 따라서 값의 범위는 0 - 999,999,999입니다.
- *RMDO2*는 논리 오프셋이 1 000 000 000으로 나뉘질 때 획득된 결과를 나타냅니다. 따라서 논리 오프셋에 있는 1 000 000 000의 완전한 배수의 수입니다. 배수의 수는 0 - 999 999 999 범위에 있습니다.

이 필드의 초기값은 0입니다.

#### **RMDO2(10자리의 부호 있는 정수)**

벌크 데이터의 높은 오프셋.

이 필드는 벌크 데이터가 일부를 구성하는 오브젝트의 시작에서 벌크 데이터의 높은 오프셋을 지정합니다. 값은 0 - 999,999,999의 범위에 있습니다. 세부사항은 *RMDO*을(를) 참조하십시오.

이 필드의 초기값은 0입니다.

#### **RMENC(10자리의 부호 있는 정수)**

벌크 데이터의 숫자 인코딩.

이는 벌크 데이터의 숫자 인코딩을 지정합니다. MQRMH 구조 자체의 숫자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다.

이 필드의 초기값은 ENNAT입니다.

#### **RMFLG(10자리의 부호 있는 정수)**

참조 메시지 플래그.

다음의 플래그가 정의됩니다.

##### **RMLAST**

참조 메시지는 오브젝트의 마지막 부분을 포함하거나 나타냅니다.

이 플래그는 참조 메시지가 참조된 오브젝트의 마지막 부분을 나타내거나 포함한다는 것을 표시합니다.

##### **RMNLST**

참조 메시지는 오브젝트의 마지막 부분을 나타내거나 포함하지 않습니다.

RMNLST는 프로그램 문서화를 지원하기 위해 정의됩니다. 이 옵션은 기타와 함께 사용하기 위한 용도가 아니지만, 해당 값이 0이므로 해당 사용을 감지할 수 없습니다.

이 필드의 초기값은 RMNLST입니다.

#### **RMFMT(8바이트 문자열)**

벌크 데이터의 형식 이름.

벌크 데이터의 형식 이름을 지정합니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 이 필드의 코딩 규칙은 MQMD의 *MDFMT* 필드에 대한 규칙과 동일합니다.

이 필드의 초기값은 FMNONE입니다.

**RMLEN(10자리의 부호 있는 정수)**

고정 필드의 끝에 있는 문자열을 포함하지만 벌크 데이터를 포함하지 않는 MQRMH의 총 길이.  
이 필드의 초기값은 0입니다.

**RMOII(24바이트 비트 문자열)**

오브젝트 인스턴스 ID.

이 필드는 오브젝트의 특정 인스턴스를 식별하는 데 사용할 수 있습니다. 필요하지 않은 경우 다음 값으로 설정되어야 합니다.

**OIINON**

지정된 오브젝트 인스턴스 ID가 없습니다.

이 값은 필드 길이 만큼의 2진 0입니다.

이 필드의 길이는 LNOIID로 제공됩니다. 이 필드의 초기값은 OIINON입니다.

**RMOT(8바이트 문자열)**

오브젝트 유형.

이는 지원하는 참조 메시지의 유형을 인식하기 위해 메시지 엑시트에 의해 사용될 수 있는 이름입니다. 이름이 *RMFMT* 필드와 동일한 규칙을 준수하도록 고려하십시오.

이 필드의 초기값은 8개의 공백입니다.

**RMSEL(10자리의 부호 있는 정수)**

소스 환경 데이터 길이.

이 필드가 0이면 소스 환경 데이터가 없으며 *RMSEO*이(가) 무시됩니다.

이 필드의 초기값은 0입니다.

**RMSEO(10자리의 부호 있는 정수)**

소스 환경 데이터의 오프셋.

이 필드는 MQRMH 구조의 시작에서 소스 환경 데이터의 오프셋을 지정합니다. 해당 데이터가 작성자로 알려진 경우 참조 메시지의 작성자가 소스 환경 데이터를 지정할 수 있습니다. 예를 들어, 소스 환경 데이터는 벌크 데이터가 포함된 오브젝트의 디렉토리 경로일 수 있습니다. 그러나 작성자가 소스 환경 데이터를 알지 못하는 경우 사용자 제공 메시지 엑시트에서 필요한 환경 정보가 있는지 판별해야 합니다.

소스 환경 데이터의 길이는 *RMSEL*에 의해 제공됩니다. 이 길이가 0인 경우 목적지 환경 데이터가 없고 *RMSEO*가 무시됩니다. 존재하는 경우 소스 환경 데이터는 구조의 시작에서 *RMLEN* 바이트 내에 완전히 상주해야 합니다.

애플리케이션은 환경 데이터가 해당 구조에서 필드를 마지막으로 고정된 후 즉시 시작되거나 데이터가 *RMSNO*, *RMDEO* 및 *RMDNO* 필드에서 처리되는 데이터와 근접하다고 가정하지 않아야 합니다.

이 필드의 초기값은 0입니다.

**RMSID(4바이트 문자열)**

구조 ID입니다.

값은 다음과 같아야 합니다.

**RMSIDV**

참조 메시지 헤더 구조의 ID입니다.

이 필드의 초기값은 RMSIDV입니다.

**RMSNL(10자리의 부호 있는 정수)**

소스 오브젝트 이름 길이.

이 필드가 0이면 소스 오브젝트 이름이 없으며 *RMSNO*이(가) 무시됩니다.

이 필드의 초기값은 0입니다.

## RMSNO(10자리의 부호 있는 정수)

소스 오브젝트 이름의 오프셋.

이 필드는 MQRMH 구조의 시작에서 소스 오브젝트 이름의 오프셋을 지정합니다. 해당 데이터가 작성자로 알려진 경우 참조 메시지의 작성자가 소스 오브젝트 이름을 지정할 수 있습니다. 그러나 작성자가 소스 오브젝트 이름을 알지 못하는 경우 사용자 제공 메시지 엑시트가 처리될 오브젝트를 식별해야 합니다.

소스 오브젝트 이름의 길이는 RMSNL(으)로 제공됩니다. 이 길이가 0이면 소스 오브젝트 이름이 없으며 RMSNO이(가) 무시됩니다. 존재하는 경우, 소스 오브젝트 이름은 구조의 시작으로부터 RMLLEN 바이트 내에 완전히 상주해야 합니다.

애플리케이션은 소스 오브젝트 이름이 RMSEO, RMDEO 및 RMDNO 필드에서 처리되는 데이터와 근접하다고 가정하지 않아야 합니다.

이 필드의 초기값은 0입니다.

## RMVER(10자리의 부호 있는 정수)

구조 버전 번호입니다.

값은 다음과 같아야 합니다.

### RMVER1

버전1 참조 메시지 헤더 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

### RMVERC

참조 메시지 헤더 구조의 현재 버전.

이 필드의 초기값은 RMVER1입니다.

## 초기값

표 724. MQRMH의 필드		
필드 이름	상수의 이름	상수의 값
RMSID	RMSIDV	'RMH-'
RMVER	RMVER1	1
RMLLEN	없음	0
RMENC	ENNAT	환경에 따라 다름
RMCSI	CSUNDF	0
RMFMT	FMNONE	공백
RMFLG	RMNLST	0
RMOT	없음	공백
RMOII	OIINON	널
RMSEL	없음	0
RMSEO	없음	0
RMSNL	없음	0
RMSNO	없음	0
RMDEL	없음	0
RMDEO	없음	0
RMDNL	없음	0

표 724. MQRMH의 필드 (계속)

필드 이름	상수의 이름	상수의 값
RMDNO	없음	0
RMDL	없음	0
RMDO	없음	0
RMD02	없음	0

**참고:**  
 1. ~ 기호는 단일 공백 문자를 나타냅니다.

```

D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQRMH Structure
D*
D* Structure identifier
D RMSID          1      4    INZ('RMH ')
D* Structure version number
D RMVER          5      8I 0 INZ(1)
D* Total length of MQRMH, including strings at end of fixed fields, but not
D* the bulk data
D RMLEN          9      12I 0 INZ(0)
D* Numeric encoding of bulk data
D RMENC          13     16I 0 INZ(273)
D* Character set identifier of bulkdata
D RMCSI          17     20I 0 INZ(0)
D* Format name of bulk data
D RMFMT          21     28    INZ('      ')
D* Reference message flags
D RMFLG          29     32I 0 INZ(0)
D* Object type
D RMOT           33     40    INZ
D* Object instance identifier
D RMOII          41     64    INZ('X'00000000000000-
D                    0000000000000000000000-
D                    000000000000')
D* Length of source environmentdata
D RMSEL          65     68I 0 INZ(0)
D* Offset of source environmentdata
D RMSEO          69     72I 0 INZ(0)
D* Length of source object name
D RMSNL          73     76I 0 INZ(0)
D* Offset of source object name
D RMSNO          77     80I 0 INZ(0)
D* Length of destination environmentdata
D RMDL           81     84I 0 INZ(0)
D* Offset of destination environmentdata
D RMDEO          85     88I 0 INZ(0)
D* Length of destination objectname
D RMDNL          89     92I 0 INZ(0)
D* Offset of destination objectname
D RMDNO          93     96I 0 INZ(0)
D* Length of bulk data
D RMDL           97     100I 0 INZ(0)
D* Low offset of bulk data
D RMDO           101    104I 0 INZ(0)
D* High offset of bulk data
D RMD02          105    108I 0 INZ(0)
    
```

## RPG 선언

### IBM i의 MQR(응답 레코드)

목적지가 분배 목록일 때 MQR 구조는 단일 목적지 큐에 대한 열기 또는 넣기 조작에서 완료 코드 및 이유 코드를 수신하는 데 사용됩니다.

## 개요

**목적:** MQRR은 MQOPEN, MQPUT 및 MQPUT1 호출에 대한 출력 구조입니다.

**문자 세트 및 인코딩:** MQRR의 데이터는 **CodedCharSetId** 큐 관리자 속성에 의해 지정된 문자 세트 및 ENNAT에 의해 지정된 로컬 큐 관리자의 인코딩에 있어야 합니다. 그러나 애플리케이션이 IBM MQ 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트와 인코딩으로 작성되어야 합니다.

**사용법:** MQOPEN 및 MQPUT 호출 또는 MQPUT1 호출에 이러한 구조의 배열을 제공하여 호출의 결과가 혼합될 때 즉, 호출이 목록의 일부 큐에 대해 성공하지만 다른 큐에 대해 실패했을 때 분배 목록의 모든 큐에 대한 완료 코드 및 이유 코드를 판별할 수 있습니다. 호출의 이유 코드 RC2136은 응답 레코드(애플리케이션에서 제공됨)가 큐 관리자에 의해 설정됨을 표시합니다.

- [1153 페이지의 『필드』](#)
- [1153 페이지의 『초기값』](#)
- [1153 페이지의 『RPG 선언』](#)

## 필드

MQRR 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

### RRCC(10자리의 부호 있는 정수)

큐의 완료 코드.

MQOPEN 또는 MQPUT1 호출에 제공된 MQOR 구조 배열에서 해당하는 요소가 지정한 이름의 큐에 대한 열기 또는 넣기 조작의 결과인 완료 코드입니다.

이 필드는 항상 출력 필드입니다. 이 필드의 초기값은 CCOK입니다.

### RRREA(10자리의 부호 있는 정수)

큐의 이유 코드.

MQOPEN 또는 MQPUT1 호출에 제공된 MQOR 구조 배열에서 해당하는 요소가 지정한 이름의 큐에 대한 열기 또는 넣기 조작의 결과인 이유 코드입니다.

이 필드는 항상 출력 필드입니다. 이 필드의 초기값은 RCNONE입니다.

## 초기값

표 725. MQRR의 필드		
필드 이름	상수의 이름	상수의 값
RRCC	CCOK	0
RRREA	RCNONE	0

## RPG 선언

```
D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQRR Structure
D*
D* Completion code for queue
D RRCC 1 4I 0 INZ(0)
D* Reason code for queue
D RRREA 5 8I 0 INZ(0)
```

## IBM i의 MQSCO(TLS 구성 옵션)

MQSCO 구성(MQCD 구성에서 TLS 필드와 함께)을 사용하여 IBM MQ MQI client로 실행하는 애플리케이션이 채널 프로토콜이 TCP/IP일 때 클라이언트 연결을 위해 TLS의 사용을 제어하는 구성 옵션을 지정할 수 있습니다.

## 개요

**목적:** 이 구조는 MQCONNX 호출의 입력 매개변수입니다.

클라이언트 채널에 대한 채널 프로토콜이 TCP/IP가 아니면 MQSCO 구조가 무시됩니다.

**문자 세트 및 인코딩:** MQSCO의 데이터는 **CodedCharSetId** 큐 관리자 속성에 의해 지정된 문자 세트 및 ENNAT에 의해 지정된 로컬 큐 관리자의 인코딩에 있어야 합니다.

- [1154 페이지의 『필드』](#)
- [1158 페이지의 『초기값』](#)
- [1158 페이지의 『RPG 선언』](#)

## 필드

MQSCO 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

### SCAIC(10자리의 부호 있는 정수)

SCAIP 또는 SCAIO 필드에서 처리하는 인증 정보(MQAIR) 레코드의 수입니다. 자세한 정보는 [973 페이지의 『IBM i의 MQAIR\(인증 정보 레코드\)』](#)의 내용을 참조하십시오. 값은 0 이상이어야 합니다. 이 값이 올바르지 않은 경우 호출은 이유 코드 RC2383으로 실패합니다.

입력 필드입니다. 이 필드의 초기값은 0입니다.

### SCAIO(10자리의 부호 있는 정수)

MQSCO 구조의 시작에서 첫 번째 인증 정보 레코드의 오프셋(바이트)입니다. 오프셋은 양수 또는 음수일 수 있습니다. SCAIC이(가) 0인 경우 필드는 무시됩니다.

SCAIO 또는 SCAIP을(를) 사용하여 MQAIR 레코드를 지정할 수 있지만 둘 다 지정할 수는 없습니다. 자세한 내용은 SCAIP 필드의 설명을 참조하십시오.

입력 필드입니다. 이 필드의 초기값은 0입니다.

### SCAIP(10자리의 부호 있는 정수)

첫 번째 인증 정보 레코드의 주소입니다. SCAIC이(가) 0인 경우 필드는 무시됩니다.

다음 두 가지 방법 중 하나로 MQAIR 레코드의 배열을 제공할 수 있습니다.

- 포인터 필드 SCAIP 사용

이 경우 애플리케이션은 MQSCO 구조에서 별도로 MQAIR 레코드의 배열을 선언하고 배열의 주소에 SCAIP를 설정할 수 있습니다.

다른 환경(예를 들면, C 프로그래밍 언어)에 대해 현대 가능한 방법으로 포인터 데이터 유형을 지원하는 언어를 프로그래밍에 대한 SCAIP 사용을 고려하십시오.

- 오프셋 필드 SCAIO 사용

이 경우 애플리케이션은 MQAIR 레코드의 배열이 뒤에 오는 MQSCO를 포함하는 복합 구조를 선언하고 MQSCO 구조 시작의 배열에서 첫 번째 레코드의 오프셋에 SCAIO를 설정해야 합니다. 이 값이 정확하고 MQLONG 내에 수용할 수 있는 값이 있는지 확인하십시오(가장 제한적 프로그래밍 언어가 COBOL이며 올바른 범위는 -999 999 999 - +999 999 999임).

포인터 데이터 유형을 지원하지 않거나 다른 환경(예를 들면, COBOL 프로그래밍 언어)에 대해 현대가 가능하지 않을 수도 있는 방법으로 포인터 데이터 유형을 지원하지 않는 언어 프로그래밍에 SCAIO 사용을 고려하십시오.

어떤 기술을 선택하든 SCAIP 및 SCAIO 중 하나만 사용할 수 있습니다. 모두 0이 아닌 경우 이유 코드 RC2384로 호출에 실패합니다.

입력 필드입니다. 이 필드의 초기값은 포인터를 지원하는 프로그래밍 언어로 된 널 포인터이며, 그렇지 않은 경우 모두 널 바이트 문자열입니다.

**참고:** 프로그래밍 언어가 포인터 데이터 유형을 지원하지 않는 플랫폼에서 이 필드는 적절한 길이의 바이트 문자열로 선언됩니다.

### SCCERLBL(10자리의 부호 있는 정수)

이 필드는 사용 중인 인증서 레이블의 세부사항을 제공합니다.

IBM MQ는 공백으로 SCCERLBL 필드의 값을 초기화합니다. 필수 값을 입력하거나 기본값을 허용하십시오.

ibmwebspheremquser\_id는 제품의 모든 버전에 대한 이 필드에 유효한 값이고 MQSCO 버전이 5.0미만인 경우 유일하게 유효한 값입니다. 따라서 이 필드의 값은 런타임 시 해석되고 필요한 경우 변경됩니다. MQSCO 버전을 5.0 미만으로 지정하거나 SCCERLBL 필드에 공백의 기본값을 허용하는 경우 시스템은 값 ibmwebspheremquser\_id를 사용합니다.

입력 필드입니다.

### SCCERTVPOL(10자리의 부호 있는 정수)

이 필드는 사용되는 인증서 유효성 검증 정책의 유형을 지정합니다. 다음 값 중 하나로 필드를 설정할 수 있습니다.

#### MQ\_CERT\_VAL\_POLICY\_ANY

소스 소켓 라이브러리에서 지원하는 각 인증서 유효성 검증 정책을 적용합니다. 정책이 인증서 체인을 유효한 것으로 간주하는 경우 인증서 체인을 승인합니다.

#### MQ\_CERT\_VAL\_POLICY\_RFC5280

RFC5280 준수 인증서 유효성 검증 정책만 적용합니다. 이 설정은 임의(ANY) 설정보다 엄격한 유효성 검증을 제공하지만 일부 오래된 디지털 인증서는 거부합니다.

이 필드의 초기값은 MQ\_CERT\_VAL\_POLICY\_ANY입니다.

### SCCCH(10자리의 부호 있는 정수)

이 필드는 클라이언트 시스템에 연결된 암호화 하드웨어의 구성 세부사항을 제공합니다.

다음 형식에서 문자열에 필드를 설정하거나 공백 또는 널로 두십시오.

```
GSK_PKCS11=the PKCS #11 driver path and file name;the PKCS #11 token label;the PKCS #11 token password;symmetric cipher setting>;
```

PKCS11 인터페이스에 준수하는 암호화 하드웨어를 사용하려면(예:IBM 4960 또는 IBM 4963) 세미콜론으로 종료된 각 PKCS11 드라이버 경로, PKCS11 토큰 레이블 및 PKCS11 토큰 비밀번호 문자열을 지정하십시오.

PKCS #11 드라이버 경로는 PKCS #11 카드 지원을 위해 제공되는 공유 라이브러리의 절대 경로입니다. PKCS #11 driver file name은 공유 라이브러리의 이름입니다. PKCS #11 경로 및 파일 이름에 필요한 값의 예는 다음과 같습니다.

```
/usr/lib/pkcs11/PKCS11_API.so
```

PKCS #11 토큰 레이블은 모두 소문자여야 합니다. 대소문자 혼합 또는 대문자 토큰 레이블로 하드웨어를 구성하는 경우 이 소문자 레이블로 재구성하십시오.

암호화 하드웨어 구성이 필요하지 않은 경우 필드를 공백 또는 널로 설정하십시오.

값이 필드의 길이보다 짧으면 널 문자로 값을 종료하거나 필드 길이에 맞게 공백으로 채웁니다. 이 값이 유효하지 않거나 암호용 하드웨어를 구성하는 데 사용될 때 호출은 이유 코드 RC2382로 실패합니다.

입력 필드입니다. 이 필드의 길이는 LNSSCH로 제공됩니다. 이 필드의 초기값은 공백 문자입니다.

### SCEPSUITEB(10자리의 부호 있는 정수)

이 필드는 스위트 B 준수 암호화가 사용되는지 여부와 이용되는 강도의 레벨을 지정합니다. 값은 다음 중 하나 이상입니다.

- SCEPSUITEB0

스위트 B 준수 암호화가 사용되지 않습니다.

- SCEPSUITEB1

스위트 B 128비트 강도 보안이 사용됩니다.

- SCEPSUITEB2

스위트 B 192비트 강도 보안이 사용됩니다.

**참고:** 이 필드에서 SCEPSUITEB0를 다른 값과 함께 사용하는 것은 유효하지 않습니다.

### SCFR(10자리의 부호 있는 정수)

사용되는 암호화 모듈이 하드웨어 제품에서 제공한 모듈이 되도록 IBM MQ를 암호화 하드웨어로 구성할 수 있습니다. 사용 중인 암호화 하드웨어 제품에 따라 특정 레벨로 FIPS 인증될 수 있습니다.

이 필드를 사용하여 암호화가 IBM MQ 제공 소프트웨어에 제공되는 경우 FIPS 인증 알고리즘만 사용되도록 지정합니다.

IBM MQ가 설치되는 경우 일부 FIPS 인증 모듈을 제공하는 TLS 암호화의 구현도 설치됩니다.

가능한 값은 다음과 같습니다.

#### MQSSL\_FIPS\_NO

이는 기본값입니다. 이 값으로 설정하는 경우:

- 특정 플랫폼에서 지원되는 CipherSpec을 사용할 수 있습니다.
- 암호화 하드웨어를 사용하지 않고 실행하는 경우 FIPS 140-2 인증 암호화를 사용하여 IBM MQ 플랫폼에서 다음 CipherSpecs을 실행합니다.
  - TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA
  - TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA
  - TLS\_RSA\_WITH\_AES\_256\_CBC\_SHA

#### MQSSL\_FIPS\_YES

이 값으로 설정하는 경우, 암호화를 수행하기 위한 암호화 하드웨어를 사용하지 않으면 다음 사항을 확실히 준수해야 합니다.

- 이 클라이언트 연결에 적용되는 CipherSpec에서는 FIPS 인증 암호화 알고리즘만 사용할 수 있습니다.
- 다음 Cipher Specs 중 하나가 사용되는 경우에만 인바운드 및 아웃바운드 TLS 채널 연결이 성공합니다.
  - TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA
  - TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA
  - TLS\_RSA\_WITH\_AES\_256\_CBC\_SHA

**참고:**

1. **Deprecated** CipherSpec TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA는 더 이상 사용되지 않습니다.
2. 가능한 경우 FIPS 전용 CipherSpec이 구성되면 RC2393으로 MQI 클라이언트가 비FIPS CipherSpec을 지정하는 연결을 거부합니다. IBM MQ는 이러한 모든 연결을 거부한다고 보장하지 않으며 IBM MQ 구성이 FIPS 준수인지 판별하는 것은 사용자의 책임입니다.

### SCKEYPWL (10자리의 부호 있는 정수)

TLS키 저장소 비밀번호 문구의 길이입니다.

키 저장소 비밀번호 문구의 최대 길이는 128자입니다. 키 저장소 비밀번호 문구가 허용되는 최대 길이보다 긴 경우 RC2381로 연결에 실패합니다.

입력 필드입니다. 이 필드의 초기값은 0입니다.

### SCKEYPWO (10자리의 부호 있는 정수)

이는 TLS키 저장소 비밀번호 문구의 오프셋 (바이트) 입니다. 오프셋은 양수 또는 음수일 수 있습니다.

SCKEYPWO 또는 SCKEYPWP 를 사용하여 키 저장소 비밀번호 문구를 지정할 수 있지만 둘 다 지정할 수는 없습니다. 자세한 정보는 SCKEYPWP 필드의 설명을 참조하십시오.



입력 필드입니다. 이 필드의 초기값은 0입니다.

### **SCKEYPWP (포인터)**

TLS키 저장소 비밀번호 문구의 주소입니다.

입력 필드입니다. 이 필드의 초기값은 널 포인터입니다.

키 저장소 비밀번호 문구는 일반 텍스트 문자열 또는 **runmqicred** 유틸리티를 사용하여 암호화된 비밀번호 문구로 지정할 수 있습니다.

이 필드를 사용하여 지정된 키 저장소 비밀번호 문구는 **MQKEYRPWD** 환경 변수 또는 클라이언트 구성 파일의 SSL 스탠자에 있는 **SSLKeyRepositoryPassword** 특성을 사용하여 지정된 키 저장소 비밀번호 문구를 대체합니다.

SCKEYPWO 또는 SCKEYPWP 를 사용하여 키 저장소 비밀번호 문구를 지정할 수 있지만 둘 다 지정할 수는 없습니다.

### **SCKR(10자리의 부호 있는 정수)**

이 필드는 키 및 인증서가 저장되는 키 데이터베이스 파일의 위치를 지정합니다. 파일 접미부를 지정하지 않으면 .kdb 접미부가 자동으로 추가됩니다.

각 키 데이터베이스 파일에는 연관된 암호 숨김 파일이 있을 수 있습니다. 이는 키 데이터베이스에 대한 프로그램 방식 액세스를 허용하는 데 사용되는 암호화 비밀번호를 보유합니다. 비밀번호 스테쉬 파일은 동일한 디렉토리에 상주해야 하고, 키 데이터베이스와 동일한 파일 스템이 있어야 하며 접미부 .sth로 끝나야 합니다.

예를 들어, 키 데이터베이스 파일이 /xxx/yyy/key.kdb인 경우 비밀번호 스테쉬 파일은 /xxx/yyy/key.sth여야 합니다. 여기서 xxx 및 yyy 는 디렉토리 이름을 나타냅니다.

키 데이터베이스 비밀번호는 SCKEYPWP 또는 SCKEYPWO 필드를 사용하여 지정할 수도 있습니다.

값이 필드의 길이보다 짧으면 널 문자로 값을 종료하거나 필드 길이에 맞게 공백으로 채웁니다. 이 값은 확인되지 않습니다. 키 저장소에 대한 액세스에 오류가 있는 경우 호출은 이유 코드 RC2381로 실패합니다.

IBM MQ MQI client에서 TLS 연결을 실행하려면 유효 키 데이터베이스 파일 이름에 **SCKR**을 설정하십시오.

입력 필드입니다. 이 필드의 길이는 LNSSKR로 제공됩니다. 이 필드의 초기값은 공백 문자입니다.

### **SCSID(10자리의 부호 있는 정수)**

구조 ID이며 값은 다음과 같아야 합니다.

#### **SCSIDV**

TLS 구성 옵션 구조의 ID입니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 SCSIDV입니다.

### **SCVER(10자리의 부호 있는 정수)**

구조 버전 번호이며 값은 다음과 같아야 합니다.

#### **SCVER1**

버전-1 TLS 구성 옵션 구조 ID.

#### **SCVER2**

버전-2 TLS 구성 옵션 구조 ID.

#### **SCVER3**

버전-3 TLS 구성 옵션 구조입니다.

#### **SCVER4**

버전-4 TLS 구성 옵션 구조입니다.

#### **SCVER5**

버전-5 TLS 구성 옵션 구조입니다.

#### **SCVER6**

Version-6 TLS 구성 옵션 구조입니다.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

**SCVERC**

TLS 구성 옵션 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 SCVER1입니다.

**초기값**

표 726. MQSCO의 필드		
필드 이름	상수의 이름	상수의 값
SCSID	SCSIDV	'SCO↵'
SCVER	SCVER1	1
SCKR	없음	널 문자열 또는 공백
SCCH	없음	널 문자열 또는 공백
SCAIC	없음	0
SCAIO	없음	0
SCAIP	없음	널 포인터 또는 널 바이트
SCKRC	없음	널 포인터 또는 널 바이트
SCFR	없음	널 포인터 또는 널 바이트
SCEPSUITEB	없음	널 포인터 또는 널 바이트
SCCERTVPOL	없음	널 포인터 또는 널 바이트
SCCERLBL	없음	널 포인터 또는 널 바이트
SCKEYPWP	없음	널 포인터 또는 널 바이트
SCKEYPWO	없음	0
SCKEYPWL	없음	0

**참고:**

- ↵ 기호는 단일 공백 문자를 나타냅니다.
- SCEPSUITEB 옵션은 [1158 페이지의 『RPG 선언』](#)의 내용을 참조하십시오.

**RPG 선언**

```

D*.1.....2.....3.....4.....5.....6.....7..
D* MQSCO Structure
D*
D* Structure identifier
D SCSID          1          4    INZ('SCO ')
D* Structure version number
D SCVER          5          8I 0 INZ(1)
D* Location of TLS key repository
D SCKR           9         264   INZ
D* Cryptographic hardware configuration string
D SCCH           265       520   INZ
D* Number of MQAIR records present
D SCAIC          521       524I 0 INZ(0)
D* Offset of first MQAIR record from start of MQSCO structure
D SCAIO          525       528I 0 INZ(0)
D* Address of first MQAIR record
D SCAIP          529       544*   INZ(*NULL)
D* Ver:1 **
D* Number of unencrypted bytes sent/received before secret key is
D* reset
    
```

```

D SCKRC 545 548I 0 INZ(0)
D* Using FIPS-certified algorithms
D SCFR 549 552I 0 INZ(0)
D* Ver:2 **
* Use only Suite B cryptographic algorithms
D SCEPSUITEB0
D SCEPSUITEB1 553 556I 0 INZ(1)
D SCEPSUITEB2 557 560I 0 INZ(0)
D SCEPSUITEB3 561 564I 0 INZ(0)
D SCEPSUITEB4 565 568I 0 INZ(0)
D SCEPSUITEB 10I 0 DIM(4) OVERLAY(SCEPSUITEB0)
D* Ver:3 **
D* Certificate validation policy
D SCCERTVPOL 569 572I 0 INZ(0)
D* Ver:4 **

```

## IBM i IBM i의 MQSD(구독 디스크립터)

MQSD 구조는 작성된 구독에 대한 세부사항을 지정하는 데 사용됩니다.

### 개요

#### 목적

구조는 MQSUB 호출의 입/출력 매개변수입니다.

#### 관리되는 구독

애플리케이션이 해당 구독과 일치하는 발행에 대한 목적지로 특별한 큐를 사용할 필요가 없고 관리되는 구독 기능을 사용할 수 있습니다. 애플리케이션이 관리되는 구독을 사용하도록 선택하는 경우, 큐 관리자는 MQSUB 호출의 출력으로 오브젝트 핸들을 제공하여, 발행된 메시지가 송신되는 목적지를 구독자에게 알려 줍니다. 자세한 정보는 [HOBJ\(10자리 부호 있는 정수\) - 입출력\(I/O\)의 내용을 참조하십시오.](#)

구독이 제거되면 다음 상황의 경우 큐 관리자 또한 관리되는 목적지에서 검색되지 않은 메시지 정리를 착수 합니다.

- 구독이 제거될 때 - CORMSB가 있는 MQCLOSE의 사용에 의해 - 관리된 Hobj가 처리완료됩니다.
- 연결이 지속 불가능 구독(SONDUR)를 사용하는 애플리케이션에 대한 연결이 손실됨을 암시적으로 의미
- 구독이 만료되었기 때문에 만기에 의해 제거되고 관리되는 Hobj가 처리완료됩니다.

위정리가 발생하고 닫힌 지속 불가능 구독의 메시지가 큐 관리자에서 장소를 차지하지 않도록 지속 불가능 구독으로 관리되는 구독을 사용해야 합니다. 지속 가능 구독은 또한 관리되는 목적지를 사용할 수 있습니다.

#### 문자 세트 및 인코딩

MQSD의 데이터는 **CodedCharSetId** 큐 관리자 속성에 의해 지정된 문자 세트와 ENNAT에 의해 지정된 로컬 큐 관리자의 인코딩에 있어야 합니다. 그러나 애플리케이션이 IBM MQ 클라이언트로 실행 중인 경우 구조는 클라이언트의 문자 세트와 인코딩으로 작성되어야 합니다.

- [1159 페이지의 『필드』](#)
- [1170 페이지의 『초기값』](#)
- [1171 페이지의 『RPG 선언』](#)

### 필드

MQSD 구조에는 다음과 같은 필드가 포함됩니다. 필드에 대해서 알파벳순으로 설명합니다.

#### SDAID(32바이트 문자 문자열)

이 값은 이 구독과 일치하는 모든 발행 메시지의 메시지 디스크립터(MQMD)의 **MDAID** 필드에 있습니다. **SDAID**는 메시지의 ID 컨텍스트의 일부입니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트](#)를 참조하십시오.

**MDAID**에 대한 자세한 정보는 [MDAID](#)를 참조하십시오.

SOSETI 옵션이 지정되지 않은 경우 이 구독을 위해 발행된 각 메시지에 설정되는 **MDAID**는 기본 컨텍스트 정보로서 공백입니다.

SOSETI 옵션을 지정하면 사용자가 *MDAID*를 생성하게 되며 이 필드는 이 구독의 각 발행에 설정할 *SDAID*를 포함하는 입력 필드입니다.

이 필드의 길이는 *LNAIDD*로 제공됩니다. 이 필드의 초기값은 32개의 공백 문자입니다.

SOALT 옵션을 사용하여 기존 구독을 변경하는 경우 추후 발행 메시지의 *SDAID*는 변경될 수 있습니다.

SORES를 사용하는 MQSUB 호출의 리턴에서 이 필드는 구독에 사용되는 현재 *MDAID*로 설정됩니다.

### **SDACC(32바이트 문자 문자열)**

이 값은 이 구독과 일치하는 모든 발행 메시지의 메시지 디스크립터(MQMD)의 *MDACC* 필드에 있습니다. *MDACC*는 메시지의 ID 컨텍스트의 일부입니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트](#)를 참조하십시오.

*MDACC*에 대한 자세한 정보는 [MDACC](#)를 참조하십시오.

*SDACC* 필드에 다음의 특수 값을 사용할 수 있습니다.

#### **ACNONE**

계정 토큰이 지정되지 않습니다.

이 값은 필드 길이 만큼의 2진 0입니다.

SOSETI 옵션이 지정되지 않은 경우 계정 토큰은 기본 컨텍스트 정보로 큐 관리자에 의해 생성되고 이 필드가 이 구독을 위해 발행된 각 메시지에 설정된 *MDACC*를 포함하는 출력 필드입니다.

SOSETI 옵션을 지정하는 경우 사용자가 계정 토큰을 생성하게 되며 이 필드는 이 구독의 각 발행에 설정할 *MDACC*를 포함하는 입력 필드가 됩니다.

이 필드의 길이는 *LNACCT*로 제공됩니다. 이 필드의 초기값은 *ACNONE*입니다.

SOALT 옵션을 사용하여 기존 구독을 변경하는 경우 추후 발행 메시지의 *MDACC* 값은 변경될 수 있습니다.

SORES를 사용하는 MQSUB 호출의 리턴에서 이 필드는 구독에 사용되는 현재 *MDACC*로 설정됩니다.

### **SDASI(40바이트 비트 문자열)**

이는 *SDAU*와 함께 권한 서비스로 전달되는 보안 ID이며 이를 통해 적절한 권한 검사를 수행할 수 있습니다.

SOALTU를 지정한 경우에만 *SDASI*가 사용되며 *SDAU* 필드가 첫 번째 널 문자까지 또는 필드의 끝까지 전부 공백이 되지 않습니다.

SORES를 사용한 MQSUB 호출로부터 리턴될 때 이 필드는 변경되지 않습니다.

자세한 정보는 MQOD 데이터 유형에서 [ODASI](#)에 대한 설명을 참조하십시오.

### **SDAU(12바이트 문자 문자열)**

SOALTU를 지정하는 경우 이 필드는 애플리케이션이 현재 실행되고 있는 사용자 ID 대신 구독 권한 및 목적지 큐(MQSUB 호출의 **Hobj** 매개변수에 지정됨)에 대한 출력을 확인하는 데 사용되는 대체 사용자 ID를 포함합니다.

성공적이면 이 필드에 지정된 사용자 ID는 애플리케이션이 현재 실행되고 있는 사용자 ID 대신 사용자 ID를 소유하는 구독으로 기록됩니다.

SOALTU가 지정되고 이 필드가 첫 번째 널 문자 또는 필드의 끝까지 완전히 비어 있는 경우 지정된 옵션 또는 출력을 위한 목적지 큐를 사용하여 이 토픽에 구독하는 데 필요한 사용자 권한이 없는 경우에만 성공할 수 있습니다.

SOALTU가 지정되지 않은 경우 이 필드는 무시됩니다.

SORES를 사용한 MQSUB 호출로부터 리턴될 때 이 필드는 변경되지 않습니다.

입력 필드입니다. 이 필드의 길이는 *LNUID*에서 제공합니다. 이 필드의 초기값은 12개의 공백 문자입니다.

## SDCID(24바이트 비트 문자열)

이 구독과 일치하기 위해 송신된 모든 발행물은 메시지 디스크립터에서 이 상관 ID를 포함합니다. 다중 구독이 동일한 큐를 사용하여 발행을 가져오는 경우 상관 ID로 MQGET를 사용하면 특정 구독에 대한 발행만 가져올 수 있습니다. 이 상관 ID는 큐 관리자나 사용자가 생성할 수 있습니다.

SOSCID 옵션을 지정되지 않은 경우 큐 관리자가 상관 ID를 생성하게 되며 이 필드는 이 구독에 발행된 각 메시지에 설정할 상관 ID를 포함하는 출력 필드입니다.

SOSCID 옵션을 지정하는 경우 사용자가 상관 ID를 생성하게 되며 이 필드는 이 구독의 각 발행에 설정할 상관 ID를 포함하는 입력 필드입니다. 이 경우 필드가 CINONE를 포함하면 이 구독을 위해 발행된 각 메시지에 설정되는 상관 ID는 메시지의 원래 넣기에서 작성된 상관 ID입니다.

SOGRP 옵션이 지정되고 지정된 상관 ID가 동일한 큐 및 중첩 토픽 문자열을 사용하는 기존 그룹화된 구독과 동일하면 그룹에서 가장 중요한 구독만 발행의 사본으로 제공됩니다.

이 필드의 길이는 LNCID로 제공됩니다. 이 필드의 초기값은 CINONE입니다.

구독이 SOGRP 옵션을 사용하여 작성되지 않는 한 SOALT 옵션을 사용하여 기존 구독을 변경하고 이 필드가 입력 필드이면 구독 상관 ID가 변경될 수 있습니다.

SORES를 사용하는 MQSUB 호출의 리턴에서 이 필드는 구독에 사용되는 현재 상관 ID로 설정됩니다.

## SDEXP(10자리의 부호 있는 정수)

10분의 1초로 표시되는 시간이며 이 시간 이후에 구독이 만료됩니다. 이 간격이 지나면 더 이상의 발행물은 이 구독과 일치하지 않습니다. 이 구독자에게 전송된 발행물의 MQMD에 있는 MDEXP 필드의 값으로도 사용됩니다.

다음 특수 값이 인식됩니다.

### EIULIM

구독에는 만기 시간이 없습니다.

SOALT 옵션을 사용하여 기존 구독을 변경하는 경우 구독의 만기가 변경될 수 있습니다.

SORES 옵션을 사용하는 MQSUB 호출의 리턴에서 이 필드는 남아있는 만기 시간이 아닌 구독의 원래 만기로 설정됩니다.

## SDON(48바이트 문자 문자열)

로컬 큐 관리자에서 정의된 토픽 오브젝트의 이름입니다.

이름에는 다음 문자가 포함될 수 있습니다.

- 대문자 알파벳(A - Z)
- 소문자 알파벳(a - z)
- 숫자(0 - 9)
- 마침표(.), 슬래시(/), 밑줄(\_), 퍼센트(%)

이름에 선두 문자 또는 임베드된 공백을 사용할 수 없으나 후미 문자 공백은 포함할 수 있습니다. 이름에서 중요한 데이터의 끝을 표시하기 위해 널 문자를 사용하십시오. 널 및 그 뒤에 오는 모든 문자는 공백으로 처리됩니다. 다음 제한사항이 적용됩니다.

- EBCDIC 가타카나를 사용하는 시스템에서는 소문자를 사용할 수 없습니다.
- 소문자, 슬래시 또는 퍼센트를 포함하는 이름은 명령에 지정할 때 따옴표로 묶어야 합니다. 이 인용부호는 구조의 필드로서 또는 호출의 매개변수로서 발생하는 이름에는 지정될 수 없습니다.

SDON은 전체 토픽 이름을 형성하는 데 사용됩니다.

전체 토픽 이름은 두 개의 서로 다른 필드 SDON 및 SDOS에서 빌드될 수 있습니다. 이 두 필드를 사용하는 방법에 대한 자세한 내용은 [토픽 문자열 결합](#)을 참조하십시오.

SORES를 사용한 MQSUB 호출로부터 리턴될 때 해당 옵션이 변경되지 않습니다.

이 필드의 길이는 LNTOPN로 제공됩니다. 이 필드의 초기값은 48개의 빈 문자입니다.

SOALT 옵션을 사용하여 기존 구독을 대체하는 경우 구독한 토픽 오브젝트의 이름을 변경할 수 없습니다. 이 필드 및 *SDOS*는 생략할 수 없습니다. 제공되는 경우 동일한 전체 토픽 이름으로 해결하지 않거나 RC2510로 호출에 실패하지 않아야 합니다.

### SDOPT(10자리의 부호 있는 정수)

다음 옵션 중 최소 하나를 지정해야 합니다.

- SOALT
- SORES
- SOCRT

값을 추가할 수 있습니다. 동일한 상수를 두 번 이상 추가하지 마십시오. 이 표에서는 해당 옵션을 결합할 수 있는 방법을 표시합니다. 올바르지 않은 결합이 기록됩니다. 다른 모든 결합은 올바릅니다.

#### 액세스 또는 작성 옵션

액세스 및 작성 옵션은 구독이 작성되었는지 또는 기존 구독이 리턴되거나 대체되는지 여부를 제어합니다. 해당 옵션 중 하나 이상 지정해야 합니다. 표에 액세스 또는 작성 옵션의 유효한 결합이 표시되어 있습니다.

표 727. 액세스 및 작성 옵션의 올바른 조합	
옵션 조합	참고
SOCRT	구독이 없는 경우 구독을 작성합니다. 구독이 존재하는 경우 실패합니다.
SORES	기존 구독을 재개합니다. 구독이 존재하지 않는 경우 실패합니다.
SOCRT + SORES	구독이 없는 경우 구독을 작성하고 구독이 있는 경우 일치하는 구독을 재개합니다. 여러 차례 실행되는 애플리케이션에 사용될 때 유용한 결합입니다.
SORES + SOALT(참고 참조)	기존 구독을 계속하여 MQSD에 지정된 것과 일치하도록 필드를 대체하십시오. 구독이 없는 경우 실패합니다.
SOCRT + SOALT(참고 참조)	구독이 없는 경우 구독을 작성하고 구독이 있는 경우 일치하는 구독을 계속하여 MQSD에 지정된 것과 일치하도록 필드를 대체합니다. 진행하기 전에 구독이 어떤 특정한 상태에 있는지 확인하려는 애플리케이션에서 사용될 때 유용한 결합입니다.

#### 참고:

SOALT를 지정하는 옵션이 SORES도 지정할 수 있으나 이 결합에는 SOALT만 지정하는 것 외에 추가 효과는 없습니다. 구독을 대체하기 위해 MQSUB를 호출하는 것은 구독 또한 재개됨을 암시하기 때문에 SOALT는 SORES를 암시합니다. 그러나 그 반대는 해당되지 않습니다. 구독을 재개한다고 해서 반드시 대체되는 것을 의미하지는 않습니다.

#### SOCRT

지정된 토픽에 대해 구독을 작성합니다. 동일한 *SDSN*을 사용하는 구독이 존재하는 경우 RC2432로 호출에 실패합니다. SORES 옵션과 SOCRT 옵션을 결합하여 이 실패를 방지할 수 있습니다. *SDSN*이 항상 필수인 것은 아닙니다. 자세한 정보는 해당 필드의 설명을 참조하십시오.

SOCRT를 SORES와 결합하면 먼저 지정된 *SDSN*에 대한 기존 구독이 있는지 여부를 확인하고 해당 기존 구독에 대한 핸들을 리턴하는 경우 이를 확인합니다. 그러나 기존 구독이 없는 경우 MQSD에 제공된 모든 필드를 사용하여 새 구독이 작성됩니다.

또한 SOCRT는 SOALT와 결합하여 비슷한 효과를 가질 수 있습니다(이 토픽 후자 부분의 SOALT에 대한 세부사항 참조).



## SORES

SDSN에 의해 지정된 일치하는 기존 구독에 핸들을 리턴합니다. 일치하는 구독 속성에 대한 변경이 없고 MQSD 구조의 출력이 리턴됩니다. MQSD의 대부분의 콘텐츠는 사용되지 않습니다. 사용된 필드는 SDSID, SDVER, SDOPT, SDAID 및 SDASI와 SDSN입니다.

전체 구독 이름이 일치하는 구독이 없으면 이유 코드 RC2428로 호출에 실패합니다. SORES 옵션과 SOCRT 옵션을 결합하여 이 실패를 방지할 수 있습니다. SOCRT에 대한 세부사항은 SOCRT를 참조하십시오.

구독의 사용자 ID는 구독을 작성한 사용자 ID이거나 나중에 다른 사용자 ID로 대체된 경우에는 가장 최근에 성공한 대체의 사용자 ID입니다. SDAID가 사용되는 경우 해당 사용자에 대해 대체 사용자 ID의 사용이 허용되며 구독을 작성한 사용자 ID 대신 SDAID가 구독을 작성한 사용자 ID로 기록됩니다.

해당 필드가 사용되는 경우 구독을 작성한 사용자 ID는 SDAU로 기록되고 대체 사용자 ID의 사용이 해당 사용자에 대해 허용됩니다.

SOAUID 옵션을 사용하지 않고 작성한 일치하는 구독이 있으며 구독의 사용자 ID가 애플리케이션이 구독에 대한 핸들을 요청하는 ID와 다른 경우 이유 코드 RC2434로 호출에 실패합니다.

일치하는 구독이 존재하고 현재 다른 애플리케이션에서 사용 중인 경우 호출은 이유 코드 RC2429로 실패합니다. 현재 동일한 연결로 사용 중인 경우 호출은 실패하지 않고 구독에 대한 핸들이 리턴됩니다.

SubName에서 이름이 지정된 구독이 애플리케이션으로부터 재개 또는 대체할 유효한 구독이 아닌 경우 RC2523으로 호출에 실패합니다.

SORES는 SOALT에 내재되어 해당 옵션과 결합할 필요가 없지만 두 옵션이 결합된 경우 오류가 아닙니다.

## SOALT

SDSN에 지정된 이름과 일치하는 전체 구독 이름을 가진 기존 구독에 대한 핸들을 리턴합니다. 해당 속성에 대해 대체가 허용되지 않는 경우 외에는 MQSD에서 지정된 이름과 다른 구독의 모든 속성은 구독에서 대체됩니다. 세부사항은 각 속성의 설명에 기록되며 다음 표에 요약되어 있습니다. 변경할 수 없는 속성을 대체하려고 시도하는 경우 다음 표에 표시된 이유 코드와 함께 호출에 실패합니다.

전체 구독 이름이 일치하는 구독이 없으면 이유 코드 RC2428로 호출에 실패합니다. SORES 옵션과 SOCRT 옵션을 결합하여 이 실패를 방지할 수 있습니다.

SOCRT를 SOALT와 결합하면 먼저 지정된 전체 구독 이름에 대한 기존 구독이 있는지 여부를 확인하고 이전에 자세하게 작성된 대체가 포함된 해당 기존 구독에 대한 핸들을 리턴하는 경우 이를 확인합니다. 그러나 기존 구독이 없는 경우 MQSD에 제공된 모든 필드를 사용하여 새 구독이 작성됩니다.

구독의 사용자 ID는 구독을 작성한 사용자 ID입니다. 또는 나중에 다른 사용자 ID로 대체된 경우에는 가장 최근에 성공적으로 대체된 사용자 ID입니다. SDAU가 사용된 경우 해당 사용자에 대해 대체 사용자 ID의 사용이 허용되며 구독을 작성한 사용자 ID 대신 대체 사용자 ID가 구독을 작성한 사용자 ID로 기록됩니다.

SOAUID 옵션을 사용하지 않고 작성한 일치하는 구독이 있으며 구독의 사용자 ID가 애플리케이션이 구독에 대한 핸들을 요청하는 ID와 다른 경우 이유 코드 RC2434로 호출에 실패합니다.

일치하는 구독이 존재하고 현재 다른 애플리케이션에서 사용 중인 경우 호출은 RC2429로 실패합니다. 현재 동일한 연결로 사용 중인 경우 호출은 실패하지 않고 구독에 대한 핸들이 리턴됩니다.

SubName에서 이름이 지정된 구독이 애플리케이션으로부터 재개 또는 대체할 유효한 구독이 아닌 경우 RC2523으로 호출에 실패합니다.

다음 표는 SOALT에 의해 변경될 수 있는 구독 속성을 표시합니다.

표 728. 대체할 수 있는 MQSD 및 MQSUB의 속성			
데이터 유형 디스크립터 또는 함수 호출	필드 이름	SOALT를 사용하여 이 속성을 대체할 수 있는지 여부	이유 코드
MQSD	지속 가능성 옵션	아니오	RC2509

표 728. 대체할 수 있는 MQSD 및 MQSUB의 속성 (계속)			
데이터 유형 디스크립터 또는 함수 호출	필드 이름	SOALT를 사용하여 이 속성을 대체할 수 있는지 여부	이유 코드
MQSD	목적지 옵션	예	없음
MQSD	등록 옵션	예(참고 1 참조)	SOGRP를 변경할 경우 RC2515
MQSD	발행물 옵션	예(참고 2 참조)	없음
MQSD	와일드카드 옵션	아니오	RC2510
MQSD	기타 옵션	아니오(참고 3 참조)	없음
MQSD	ObjectName	아니오	RC2510
MQSD	SDAU	아니오(참고 4 참조)	없음
MQSD	SDASI	아니오(참고 4 참조)	없음
MQSD	SDEXP	예	없음
MQSD	SDOS	아니오	RC2510
MQSD	SDSN	아니오(참고 5 참조)	없음
MQSD	SDSUD	예	없음
MQSD	SDCID	예(참고 6 참조)	RC2515 그룹화된 구독에 있는 경우
MQSD	SDPRI	예	없음
MQSD	SDACC	예	없음
MQSD	SDAID	예	없음
MQSD	SDSL	아니오	RC2512
MQSUB	Hobj	예(참고 6 참조)	RC2515 그룹화된 구독에 있는 경우

**참고:**

1. SOGRP는 대체될 수 없음
2. 구독에 속하지 않기 때문에 SONEWP를 대체할 수 없음
3. 해당 옵션은 구독의 일부가 아님
4. 이 속성은 구독의 일부가 아님
5. 이 속성은 대체되는 구독의 ID임
6. 그룹화된 하위 항목의 일부인 경우를 제외하고 변경 가능(SOGRP)

**지속성 옵션:** 다음 옵션은 구독의 지속성을 제어합니다. 이러한 옵션 중 하나만 지정할 수 있습니다. SOALT 옵션을 사용하여 기존 구독을 대체하는 경우 구독의 지속성을 변경할 수 없습니다. SORES를 사용하여 MQSUB 호출로부터 리턴 시 적절한 지속성 옵션이 설정됩니다.

**SODUR**

이 토픽에 대한 구독은 CORMSB 옵션이 있는 MQCLOSE를 사용하여 명시적으로 제거될 때까지 계속 요청하십시오. 이 구독이 명시적으로 제거되지 않는 경우 큐 관리자에 대한 이 애플리케이션 연결이 처리완료된 후에도 그대로 남습니다.

지속적 구독을 허용하지 않는 것으로 정의된 토픽에 대해 지속적 구독이 요청되는 경우 RC2436으로 호출에 실패합니다.



## SONDUR

큐 관리자에 대한 애플리케이션 연결이 처리완료될 때 구독이 아직 명시적으로 제거되지 않은 경우 이 토픽에 대한 구독이 제거되도록 요청합니다. SONDUR은 SONDUR 옵션과 반대이며 프로그램 문서화를 지원하기 위해 정의됩니다. 지정된 것이 없으면 이 값이 기본값입니다.

**목적지 옵션:** 다음 옵션은 구독된 토픽에 대한 발행을 보낼 목적지를 제어합니다. SOALT 옵션을 사용하여 기존 구독을 대체하는 경우 구독에 대한 발행물에 사용되는 목적지가 변경될 수 있습니다. SORES를 사용한 MQSUB 호출로부터 리턴 시 적절한 경우 이 옵션이 설정됩니다.

## SOMAN

발행물이 전송되는 목적지가 큐 관리자에 의해 관리되도록 요청합니다.

HOBJ에 리턴되는 오브젝트 핸들은 큐 관리자가 관리하는 큐를 나타내고 후속적인 MQGET, MQCB, MQINQ 또는 MQCLOSE 호출에서 사용됩니다.

SOMAN이 지정되어 있지 않은 경우 이전 MQSUB 호출로부터 리턴되는 오브젝트 핸들이 **Hobj** 매개변수에서 제공될 수 없습니다.

**등록 옵션:** 다음 옵션은 이 구독을 위한 큐 관리자 등록의 세부사항을 제어합니다. SOALT 옵션을 사용하여 기존 구독을 대체하는 경우 이런 등록 옵션은 변경될 수 있습니다. SORES를 사용한 MQSUB 호출로부터 리턴 시 적절한 지속성 옵션이 설정됩니다.

## SOGRP

이 구독은 같은 큐를 사용하고 같은 상관 ID를 지정하는 같은 SDSL의 다른 구독과 함께 그룹화되므로 겹치는 토픽 문자열 세트로 인해 둘 이상의 발행물 메시지를 구독 그룹으로 제공하는 원인이 되는 토픽에 대해 발행하면 한 메시지만 큐로 전달됩니다. 이 옵션이 사용되지 않는 경우에는 일치하는 각각의 고유 구독(SDSN으로 식별됨)이 발행물 사본과 함께 제공되며 이는 다수의 구독에서 공유하는 큐에 발행물 사본이 둘 이상 배치될 수도 있음을 의미하는 것일 수 있습니다.

그룹에서 가장 중요한 구독만 발행물 사본과 함께 제공됩니다. 가장 중요한 구독은 와일드카드가 있는 지점까지의 전체 토픽 이름을 바탕으로 합니다. 그룹 내에 와일드카드 설계 혼합이 사용된 경우 와일드카드 지점까지만 중요합니다. 동일한 큐를 공유하는 구독 그룹 내에 다른 와일드카드 설계를 결합하지 않도록 권장합니다.

그룹화된 구독을 새로 작성할 때 여전히 고유 SDSN이 있어야 하지만 그룹에 있는 기존 구독의 전체 토픽 이름과 일치하면 RC2514로 호출에 실패합니다.

그룹에서 가장 중요한 구독이 SONOLC도 지정하고 이것이 같은 애플리케이션의 발행물일 경우에는 큐로 발행물이 전달되지 않습니다.

이 옵션으로 작성한 구독을 대체하면 그룹화, MQSUB 호출의 Hobj (큐 및 큐 관리자 이름을 나타냄) 및 SDCID를 의미하는 필드는 변경될 수 없습니다. 필드를 대체하려고 하면 RC2515로 호출에 실패하게 됩니다.

이 옵션은 CINONE로 설정되지 않은 SDCID를 SOSCID에 결합해야 하고 SOMAN에 결합할 수 없습니다.

## SOAUID

SOAUID가 지정된 경우 구독자의 ID는 단일 사용자 ID로 제한되지 않습니다. 이로 인해 적절한 권한이 있을 때 사용자가 구독을 대체하거나 재개할 수 있습니다. 단일 사용자만 언제든지 구독 가능합니다. 다른 애플리케이션에서 현재 사용 중인 구독 사용을 계속하려고 시도하면 RC2429로 호출이 실패합니다.

SOALT를 사용하여 이 옵션을 기존 구독인 MQSUB 호출에 추가하려면 원래 구독 자체와 동일한 사용자 ID에서 시작되어야 합니다.

MQSUB 호출이 SOAUID가 설정된 기존 구독을 참조하고 사용자 ID가 원래 구독과 다른 경우 새 사용자 ID에 토픽을 구독할 수 있는 권한이 있을 때에만 호출에 성공합니다. 성공적으로 완료되면 이 구독자에 대한 추가 발행물을 발행물 메시지에 새 사용자 ID가 설정된 구독자 큐에 넣습니다.

SOAUID 및 SOFUID 모두 지정하지 마십시오. 둘 다 지정하지 않은 경우 기본값은 SOFUID입니다.

## SOFUID

SOFUID가 지정된 경우 구독을 대체하는 마지막 사용자 ID로만 구독을 대체하거나 계속할 수 있습니다. 구독이 변경되지 않은 경우 구독을 작성한 사용자 ID입니다.

MQSUB 동사가 SOAUID가 설정된 기존 구독을 참조하고 SOAUID를 사용하여 구독을 대체하여 SOFUID를 사용하는 경우 이제 구독의 사용자 ID는 이 새 사용자 ID와 혼합됩니다. 새 사용자 ID에 토픽을 구독할 수 있는 권한이 있는 경우에만 호출에 성공합니다.

구독을 소유할 때 기록된 사용자 ID와 다른 사용자 ID가 SOFUID 구독을 재개하거나 대체하려는 경우 RC2434로 호출에 실패합니다. **DISPLAY SBSTATUS** 명령을 사용하여 구독에 대한 소유 사용자 ID를 볼 수 있습니다.

SOAUID 및 SOFUID 모두 지정하지 마십시오. 둘 다 지정하지 않은 경우 기본값은 SOFUID입니다.

**발행 옵션:** 다음 옵션은 이 구독자에게 발행물이 발송되는 방식을 제어합니다. SOALT 옵션을 사용하여 기존 구독을 대체하는 경우 이런 발행 옵션은 변경될 수 있습니다.

## SONOLC

애플리케이션에서 자체 발행물을 확인하지 않음을 브로커에게 알립니다. 연결 핸들이 동일하면 발행물이 동일한 애플리케이션에서 비롯된 것으로 간주됩니다. SORES를 사용한 MQSUB 호출로부터 리턴 시 적절한 경우 이 옵션이 설정됩니다.

## SONEWP

이 구독이 작성될 때 현재 보유된 발행물이 전송되지 않으며 새 발행물만 전송됩니다. 이 옵션은 SOCRE가 지정되어 있을 때만 적용됩니다. 구독에 대한 어떠한 후속 변경도 발행물의 플로우를 변경하지 않으므로 토픽에 보유된 어떠한 발행물도 새 발행물로 구독자에게 전송되지 않습니다.

이 옵션이 SOCRE 없이 지정된 경우 RC2046으로 호출에 실패하게 됩니다. SORES를 사용한 MQSUB 호출로부터 리턴 시 이 옵션을 사용하여 구독이 작성되었더라도 이 옵션이 설정되지는 않습니다.

이 옵션이 사용되지 않는 경우 이전에 보유된 메시지는 제공된 목적지 큐로 전송됩니다. RC2525 또는 RC2526 오류로 인해 이 조치가 실패하는 경우 구독 작성에 실패합니다.

이 옵션은 SOPUBR과의 결합에 유효하지 않습니다.

## SOPUBR

이 옵션을 설정한다는 것은 필요할 때 구독자가 구체적으로 정보를 요청할 것이라는 의미입니다. 큐 관리자는 구독자에게 요청하지 않은 메시지는 보내지 않습니다. 이전 MQSUB 호출의 Hsub 핸들을 사용하여 MQSUBRQ 호출이 작성될 때마다 보유된 발행물(또는 토픽에서 와일드카드가 지정된 경우 가능한 다중 발행물)이 구독자에게 전송됩니다. 발행물이 이 옵션을 사용하여 MQSUB 호출의 결과로 송신됩니다. SORES를 사용한 MQSUB 호출로부터 리턴 시 적절한 경우 이 옵션이 설정됩니다.

이 옵션은 SONEWP와의 결합에 유효하지 않습니다.

**와일드카드 옵션:** 다음 옵션은 MQSD의 SDOS 필드에서 제공되는 문자열에서 와일드카드가 해석되는 방법을 제어합니다. 이러한 옵션 중 하나만 지정할 수 있습니다. SOALT 옵션을 사용하여 기존 구독을 대체하는 경우 이런 와일드카드 옵션은 변경될 수 없습니다. SORES를 사용한 MQSUB 호출로부터 리턴 시 적절한 와일드카드 옵션이 설정됩니다.

## SOWCHR

와일드카드는 토픽 문자열 내의 문자에 대해서만 작동합니다. SOWCHR 필드는 특별한 중요성 없이 바로 다른 특징으로 슬래시(/)를 처리합니다.

SOWCHR로 정의되는 작동은 다음 표에 표시되어 있습니다.

표 729. 와일드카드를 해석하는 방법	
특수 문자	동작
*	와일드카드, 0 또는 그 이상의 문자
?	와일드카드, 하나의 문자

표 729. 와일드카드를 해석하는 방법 (계속)	
특수 문자	동작
%	'*', '?' 또는 '%' 문자를 문자열에서 사용하고 특수 문자로 해석되지 않게 하는 이스케이프 문자(예: '%*', '%?' 또는 '%%').

예를 들어, 다음 토픽에 대한 발행입니다.

```
/level0/level1/level2/level3/level4
```

다음 토픽을 사용하여 구독자와 일치합니다.

```
*
/*
/ level0/level1/level2/level3/*
/ level0/level1/*/level3/level4
/ level0/level1/le?e12/level3/level4
```

**참고:** 와일드 카드의 이 사용은 발행/구독에 대한 MQRFH1 형식 메시지를 사용할 때 IBM MQ V6 및 WebSphere MB V6에서 제공된 의미를 정확하게 공급합니다. 새로 작성한 애플리케이션에는 사용하지 않고 이전에 그 버전에 대해 실행 중이고 SOWTOP에 설명된 것처럼 기본 와일드카드 작동을 사용하도록 변경되지 않은 애플리케이션에만 사용하는 것이 좋습니다.

#### SOWTOP

와일드카드는 토픽 문자열 내의 토픽 요소에 대해서만 작동합니다. 없음이 선택된 경우 이는 기본 작동입니다.

SOWTOP로 요청되는 작동은 다음 표에 표시되어 있습니다.

표 730. 와일드카드를 해석하는 방법	
특수 문자	동작
/	토픽 레벨 구분 기호
#	와일드카드: 다중 토픽 레벨
+	와일드카드: 단일 토픽 레벨

#### 참고:

'+' 및 '#'는 토픽 레벨에서 다른 문자(자신 포함)와 함께 사용되는 경우 와일드카드로 처리되지 않습니다. 다음 문자열에서 '#' 및 '+' 문자는 일반 문자로 처리됩니다.

```
level0/level1/#+/level3/level#
```

예를 들어, 다음 토픽에 대한 발행입니다.

```
/level0/level1/level2/level3/level4
```

다음 토픽을 사용하여 구독자와 일치합니다.

```
#
/#
/ level0/level1/level2/level3/#
/ level0/level1/+/level3/level4
```

**참고:** 와일드 카드의 이 사용은 발행/구독에 대한 MQRFH2 형식 메시지를 사용할 때 WebSphere Message Broker 6에서 제공된 의미를 공급합니다.

**기타 옵션:** 다음 옵션은 구독보다 API 호출이 실행되는 방법을 제어합니다. SORES를 사용하여 MQSUB 호출로부터 리턴 시 이런 옵션은 변경되지 않습니다.

**SOALTU**

SDAU 필드에는 이 MQSUB 호출을 유효성 검증하는 데 사용할 사용자 ID가 있습니다. 애플리케이션을 실행 중인 사용자 ID가 토픽을 구독하는 권한이 있는지에 관계없이 이 SDAU에 지정된 액세스 옵션을 사용하여 오브젝트를 열 수 있는 권한이 있는 경우에만 호출이 성공합니다.

**SOSCID**

구독이 SDCID 필드에서 제공되는 상관 ID를 사용합니다. 이 옵션이 지정되어 있지 않은 경우 큐 관리자가 구독 시간에 상관 ID를 자동으로 작성하고 이 ID는 SDCID 필드의 애플리케이션으로 리턴됩니다. 자세한 정보는 SDCID(24바이트 비트 문자열) SDCID를 참조하십시오.

**SOSETI**

구독은 SDACC 및 SDAID 필드에 제공된 계정 토큰 및 애플리케이션 ID 데이터를 사용하는 것입니다.

이 옵션이 지정되면 OOSSETI와 함께 MQOPEN 호출을 사용하여 대상 큐에 액세스할 때와 동일한 권한 부여 검사가 수행됩니다. 단, SOMAN 옵션도 사용되는 경우에는 제외되며, 이 경우 대상 큐에서 권한 부여 검사가 수행되지 않습니다.

이 옵션을 지정하지 않는 경우 다음과 같이 이 구독자에게 송신된 발행물에 연관된 기본 컨텍스트 정보가 있습니다.

표 731. 이 구독자에게 송신된 발행물에 대한 기본 컨텍스트 정보	
MQMD의 필드	사용된 값
MDUID	구독이 작성될 때 구독에 연관되는 사용자 ID입니다.
MDACC	가능한 경우 환경에서 결정됩니다. 그렇지 않으면 ACNONE로 설정됩니다.
MDAID	공백으로 설정

이 옵션은 SOCRE 및 SOALT로만 유효합니다. SORES와 함께 사용하는 경우 SDACC 및 SDAID 필드가 무시되고 따라서 이 옵션은 적용되지 않습니다.

이전에 구독이 ID 컨텍스트 정보를 제공한 경우 해당 옵션을 사용하지 않고 구독을 대체하면 대체된 구독에 대한 기본 컨텍스트 정보가 생성됩니다.

여러 사용자 ID가 SOAUID 옵션과 함께 이 옵션을 사용할 수 있게 하는 구독이 다른 사용자 ID에 의해 재개되는 경우 구독을 소유한 새 사용자 ID에 대한 기본 ID 컨텍스트가 생성되고 새 ID 컨텍스트가 포함된 후속 발행이 전달됩니다.

**SOFIQ**

큐 관리자가 정지 중인 경우 MQSUB 호출에 실패합니다. z/OS의 CICS 또는 IMS 애플리케이션의 경우 이 옵션은 연결이 정지 중 상태일 때 MQSUB 호출이 실패하게 합니다.

**SDAU(12바이트 문자 문자열)**

SOALTU를 지정하는 경우 이 필드는 애플리케이션이 현재 실행되고 있는 사용자 ID 대신 구독 권한 및 목적지 큐(MQSUB 호출의 **Hobj** 매개변수에 지정됨)에 대한 출력을 확인하는 데 사용되는 대체 사용자 ID를 포함합니다.

성공적이면 이 필드에 지정된 사용자 ID는 애플리케이션이 현재 실행되고 있는 사용자 ID 대신 사용자 ID를 소유하는 구독으로 기록됩니다.

SOALTU가 지정되고 이 필드가 첫 번째 널 문자 또는 필드의 끝까지 완전히 비어 있는 경우 지정된 옵션 또는 출력을 위한 목적지 큐를 사용하여 이 토픽에 구독하는 데 필요한 사용자 권한이 없는 경우에만 성공할 수 있습니다.

SOALTU가 지정되지 않은 경우 이 필드는 무시됩니다.

SORES를 사용한 MQSUB 호출로부터 리턴될 때 이 필드는 변경되지 않습니다.

입력 필드입니다. 이 필드의 길이는 LNUID에서 제공합니다. 이 필드의 초기값은 12개의 공백 문자입니다.

### SDPRI(10자리의 부호 있는 정수)

이 값은 이 구독과 일치하는 모든 발행 메시지의 메시지 디스크립터(MQMD)의 *MQPRI* 필드에 있습니다. MQMD의 *MQPRI* 필드에 대한 자세한 정보는 [MDPRI](#)를 참조하십시오.

값은 0보다 크거나 같아야 하며, 0은 가장 낮은 우선순위입니다. 다음 특수 값을 사용할 수도 있습니다.

#### PRQDEF

구독 큐가 관리되는 핸들이 아닌 MQSUB 호출의 *Hobj* 필드에서 제공되면 메시지의 우선순위를 이 큐의 **DefPriority** 속성에서 가져옵니다. 식별된 큐가 클러스터 큐이거나 큐 이름 해석 경로에 둘 이상의 정의가 있는 경우 발행 메시지가 [MDPRI](#)에 설명된 것처럼 큐에 넣을 때 우선순위가 판별됩니다.

MQSUB 호출에서 관리되는 핸들을 사용하는 경우 메시지에 대한 우선순위는 구독하는 주제와 연관된 모델 큐의 **DefPriority** 속성에서 가져옵니다.

#### PRPUB

메시지에 대한 우선순위는 원래 발행물의 우선순위입니다. 이는 필드의 초기값입니다.

SOALT 옵션을 사용하여 기존 구독을 변경하는 경우 추후 발행 메시지의 *MQPRI*는 변경될 수 있습니다.

SORES를 사용하는 MQSUB 호출에서 리턴 시 이 필드는 구독에 사용되는 현재 우선순위로 설정됩니다.

### SDRO(MQCHARV)

SDRO는 큐 관리자가 *SDON*에서 제공된 이름을 해석한 후의 긴 오브젝트 이름입니다.

긴 오브젝트 이름을 *SDOS*에 제공하고 *SDON*에는 아무 것도 입력하지 않은 경우 이 필드에 리턴되는 값은 *SDOS*에 제공한 값과 동일합니다.

이 필드가 생략되는 경우(즉, *SDRO.VSBufSize*가 0임) *SDRO*는 리턴되지 않지만 길이가 *SDRO.VSLength*에서 리턴됩니다. 길이가 전체 *SDRO*보다 짧으면 잘리고 제공된 길이에 맞을 수 있는 가장 오른쪽 문자를 리턴합니다.

[MQCHARV](#) 구조의 사용 방법 설명에 의하면 *SDRO*가 올바르게 지정되었거나 최대 길이를 초과하는 경우 호출은 RC2520으로 호출에 실패합니다.

### SDSID(4바이트 문자 문자열)

구조 ID이며 값은 다음과 같아야 합니다.

#### SDSIDV

구독 디스크립터 구조의 ID입니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 *SDSIDV*입니다.

### SDSL(10자리의 부호 있는 정수)

이는 구독과 연관된 레벨입니다. 발행물은 발행 시간에 사용된 *PubLevel*보다 작거나 같은 가장 높은 *SDSL* 값이 있는 구독 세트에 있는 경우에만 이 구독으로 전달됩니다.

값의 범위는 0 - 9여야 합니다. 0이 가장 낮은 레벨입니다.

이 필드의 초기값은 1입니다.

SOALT 옵션을 사용하여 기존 구독을 변경하면 *SDSL*을 변경할 수 없습니다.

### SDSN(MQCHARV)

SDSN은 등록 이름을 지정합니다.

이 필드는 *SDOPT*가 *SODUR* 옵션을 지정하는 경우에만 필요하지만 제공된다면 *SONDUR*의 큐 관리자에 의해 사용됩니다. 지정한 경우 *SDSN*은 구독을 식별하는 데 사용된 필드이기 때문에 큐 관리자 내에서 고유해야 합니다.

*SDSN*의 최대 길이는 10240입니다.

이 필드는 두 가지 목적을 서비스합니다. *SODUR* 구독의 경우 (*COKPSB* 옵션을 사용하여) 구독에 대한 핸들을 처리완료했거나 큐 관리자에서 연결 해제된 경우 작성된 후 재개하기 위해 구독을 식별하는 수단입니다.

작성 된 후 제거하기 위한 구독 식별은 SORES 옵션으로 MQSUB 호출을 사용하여 수행됩니다. SDSN 필드는 또한 DISPLAY SBSTATUS에서 SDSN 필드의 구독에 대한 관리 보기에 표시됩니다.

SDSN이 MQCHARV 구조를 사용하는 방법에 대한 설명에 따라 잘못 지정되거나 최대 길이를 초과하거나 또는 필요한 경우 생략되거나(즉, SDSN.VCHRL이 0임) 최대 길이를 초과하면 이유 코드 RC2440으로 호출에 실패합니다..

입력 필드입니다. 이 구조에 있는 필드의 초기값은 MQCHARV 구조의 값과 동일합니다.

SOALT 옵션을 사용하여 기존 구독을 대체하는 경우 이는 구독을 식별하는 데 사용되는 필드이기 때문에 구독 이름을 변경할 수 없습니다. SORES 옵션의 MQSUB 호출에서 출력은 변경되지 않습니다.

### SDSS(MQCHARV)

SDSS는 토픽에서 메시지를 구독할 때 사용되는 선택 기준을 제공하는 문자열입니다.

이 변수 길이 필드는 버퍼가 제공되고 VSBufSize 양수 버퍼 길이도 있는 경우 SORES 옵션을 사용하여 MQSUB 호출로부터 출력에 리턴됩니다. 버퍼가 호출에 제공되지 않는 경우 선택 문자열의 길이만 MQCHARV의 VSLength 필드에서 리턴됩니다. 제공된 버퍼가 필드를 리턴하는 데 필요한 공간 보다 작은 경우, 제공된 버퍼에는 VSBufSize 바이트만 리턴됩니다.

MQCHARV 구조의 사용 방법 설명에 의하면 SDSS가 올바르게 지정되었거나 최대 길이를 초과하는 경우 호출은 RC2519으로 호출에 실패합니다.

### SDSUD(MQCHARV)

이 필드의 구독에 대해 제공된 데이터는 이 구독에 전송된 모든 발행의 mq.SubUserData 메시지 특성으로 포함됩니다.

SDSUD의 최대 길이는 10240입니다.

MQCHARV 구조의 사용 방법 설명에 의하면 SDSUD가 올바르게 지정되었거나 최대 길이를 초과하는 경우 호출은 RC2431로 호출에 실패합니다.

입력 필드입니다. 이 구조에 있는 필드의 초기값은 MQCHARV 구조의 값과 동일합니다.

SOALT 옵션을 사용하여 기존 구독을 대체하는 경우 구독 사용자 데이터를 변경할 수 있습니다.

이 변수 길이 필드는 버퍼가 제공되고 VSBufLen에 양수 버퍼 길이가 있는 경우 SORES 옵션을 사용하여 MQSUB 호출로부터 출력에 리턴됩니다. 버퍼가 호출에 제공되지 않는 경우 구독 사용자 데이터의 길이만 MQCHARV의 VCHRL 필드에서 리턴됩니다. 제공된 버퍼가 필드를 리턴하는 데 필요한 공간보다 작은 경우 VSBufLen바이트만 제공된 버퍼에 리턴됩니다.

### SDVER(10자리의 부호 있는 정수)

구조 버전 번호이며 값은 다음과 같아야 합니다.

#### SDVER1

버전-1 구독 디스크립터 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### SDVERC

구독 디스크립터 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 SDVER1입니다.

### 초기값

표 732. MQSD의 필드		
필드 이름	상수의 이름	상수의 값
SDSID	SDSIDV	'SD--'
SDVER	SDVER1	1

표 732. MQSD의 필드 (계속)

필드 이름	상수의 이름	상수의 값
SDOPT	SONDUR	0
SDON	없음	공백
SDAU	없음	공백
SDASI	SINONE	널
SDEXP	EIULIM	-1
SDOS	MQCHARV에 정의된 이름과 값	
SDSN	MQCHARV에 정의된 이름과 값	
SDSUD	MQCHARV에 정의된 이름과 값	
SDCID	CINONE	널
SDPRI	PRQDEF	-3
SDACC	ACNONE	널
SDAID	없음	공백
SDSL	없음	1
SDRO	MQCHARV에 정의된 이름 및 값	

**참고:**

1. - 기호는 단일 공백 문자를 나타냅니다.

**RPG 선언**

```

D* .1.....2.....3.....4.....5.....6.....7..
D* MQSD Structure
D*
D* Structure identifier
D SDSID 1 4
D* Structure version number
D SDVER 5 8I 0
D* Options associated with subscribing
D SDOPT 9 12I 0
D* Object name
D SDON 13 60
D* Alternate user identifier
D SDAU 61 72
D* Alternate security identifier
D SDASI 73 112
D* Expiry of Subscription
D SDEXP 113 116I 0
D* Object Long name
D SDOSP 117 132*
D SDOSO 133 136I 0
D SDOSS 137 140I 0
D SDSL 141 144I 0
D SDOSC 145 148I 0
D* Subscription name
D SDSNP 149 164*
D SDSNO 165 168I 0
D SDSNS 169 172I 0
D SDSNL 173 176I 0
D SDSNC 177 180I 0
D* Subscription User data
D SDSUDP 181 196*
D SDSUDO 197 200I 0
D SDSUDS 201 204I 0
D SDSUDL 205 208I 0
D SDSUDC 209 212I 0
    
```

```

D* Correlation Id related to this subscription
D SDCID          213    236
D* Priority set in publications
D SDPRI          237    240I 0
D* Accounting Token set in publications
D SDACC          241    272
D* Appl Identity Data set in publications
D SDAID          273    304
D* Message Selector
D SDSSP          305    320*
D SDSSO          321    324I 0
D SDSSS          325    328I 0
D SDSSL          329    332I 0
D SDSSC          333    336
D* Subscription level
D SDSL           337    340  0
D* Resolved Long object name
D SDROP          341    356*
D SDR00          357    360I 0
D SDR0S          361    364I 0
D SDR0L          365    368I 0
D SDR0C          369    372I 0

```

## IBM i IBM i의 MQSMPO(메시지 특성 설정 옵션)

MQSMPO 구조를 사용하여 애플리케이션이 메시지 특성이 설정되는 방법을 제어하는 옵션을 지정할 수 있습니다.

### 개요

목적: 이 구조는 **MQSETMP** 호출의 입력 매개변수입니다.

문자 세트 및 인코딩: **MQSMPO**의 데이터는 애플리케이션의 문자 세트 및 애플리케이션의 인코딩(ENNAT)에 있어야 합니다.

- [1172 페이지의 『필드』](#)
- [1173 페이지의 『초기값』](#)
- [1173 페이지의 『RPG 선언』](#)

### 필드

MQSMPO 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

#### SPOPT(10자리의 부호 있는 정수)

**위치 옵션:** 다음 옵션은 특성 커서와 비교하여 특성의 상대 위치와 관련이 있습니다.

##### SPSETF

지정된 이름과 일치하는 첫 번째 특성의 값을 설정하거나 존재하지 않는 경우 일치 계층이 있는 다른 모든 특성 이후에 새 특성을 추가합니다.

##### SPSETC

특성 커서에 의해 지정된 특성의 값을 설정합니다. 특성 커서로 지정된 특성은 IPINQF 또는 IPINQN 옵션을 사용하여 마지막으로 조회된 특성입니다.

메시지 핸들이 재사용되거나 메시지 핸들이 MQGET 호출의 MQGMO 구조 또는 MQPUT 호출의 MQPMO 구조의 **HMSG** 필드에 지정될 때 재설정됩니다.

이 옵션은 특성 커서가 아직 설정되지 않았을 때 사용되거나 특성 커서로 지정된 특성이 삭제된 경우 완료 코드 CCFAIL 및 이유 코드 RC2471로 실패합니다.

##### SPSETA

특성 커서에 의해 지정된 특성 이후에 새 특성을 설정합니다. 특성 커서로 지정된 특성은 IPINQF 또는 IPINQO 옵션을 사용하여 마지막으로 조회된 특성입니다.

메시지 핸들이 재사용되거나 메시지 핸들이 MQGET 호출의 MQGMO 구조 또는 MQPUT 호출의 MQPMO 구조의 **HMSG** 필드에 지정될 때 재설정됩니다.



이 옵션은 특성 커서가 아직 설정되지 않았을 때 사용되거나 특성 커서로 지정된 특성이 삭제된 경우 완료 코드 CCFAIL 및 이유 코드 RC2471로 실패합니다.

설명된 옵션 중 필요한 옵션이 없는 경우 다음 옵션을 사용할 수 있습니다.

**SPNONE**

옵션이 지정되지 않았습니다.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 SPSETF입니다.

**SPSID(10자리의 부호 있는 정수)**

구조 ID이며 값은 다음과 같아야 합니다.

**SPSIDV**

세트 메시지 특성 옵션 구조의 ID.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 **SPSIDV**입니다.

**SPVAKCSI(10자리의 부호 있는 정수)**

값이 문자열인 경우 설정될 특성 값의 문자 세트.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 **CSAPL**입니다.

**SPVALENC(10자리의 부호 있는 정수)**

값이 숫자인 경우 설정될 특성 값의 인코딩.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 **ENNAT**입니다.

**SPVER(10자리의 부호 있는 정수)**

구조 버전 번호이며 값은 다음과 같아야 합니다.

**SPVER1**

버전-1 세트 메시지 특성 옵션 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

**SPVERC**

세트 메시지 특성 옵션 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 **SPVER1**입니다.

**초기값**

표 733. MQSMP0의 필드		
필드 이름	상수의 이름	상수의 값
SPSID	SPSIDV	'SMP0'
SPVER	SPVER1	1
SPOPT	SPNONE	0
SPVALENC	ENNAT	환경에 따라 다름
SPVALCSI	CSAPL	-3

**RPG 선언**

```
D* MQSMP0 Structure
D*
D*
D* Structure identifier
```

```

D SPSID                1      4    INZ('SMPO')
D*
D* Structure version number
D SPVER                5      8I 0 INZ(1)
D*
** Options that control the action of
D* MQSETMP
D SPOPT                9      12I 0 INZ(0)
D*
D* Encoding of Value
D SPVALENC            13     16I 0 INZ(273)
D*
D* Character set identifier of Value
D SPVALCSI            17     20I 0 INZ(-3)

```

## IBM i IBM i의 MQSRO(구독 요청 옵션)

MQSRO 구조로 애플리케이션이 구독 요청이 작성되는 방법을 제어하는 옵션을 지정할 수 있습니다.

### 개요

**목적:** 이 구조는 MQSUBRQ 호출의 입출력(I/O) 매개변수입니다.

**버전:** MQSRO의 현재 버전은 SRVER1입니다.

- [1174 페이지의 『필드』](#)
- [1175 페이지의 『초기값』](#)
- [1175 페이지의 『RPG 선언』](#)

### 필드

MQSRO 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

#### SRNMP(10자리의 부호 있는 정수)

이 호출의 결과로 구독 큐에 송신된 발행물의 수를 표시하기 위해 애플리케이션에 리턴된 출력 필드입니다. 이 수의 발행물이 이 호출의 결과로 송신되었다더라도 이 많은 메시지를 특히, 비지속 메시지인 경우 애플리케이션에서 가져올 수 있다고 보장하지 않습니다.

와일드카드가 포함된 토픽이 구독되는 경우 둘 이상의 발행물이 있을 수 있습니다. *HSUB*로 표시된 구독이 작성될 때 와일드카드가 토픽 문자열에 존재하지 않는 경우 하나 이하의 발행물이 이 호출의 결과로 전송됩니다.

#### SROPT(10자리의 부호 있는 정수)

다음 옵션 중 하나를 지정해야 합니다. 하나의 옵션만 지정할 수 있습니다.

**다른 옵션:** 다음 옵션은 큐 관리자가 정지 중일 때 발생하는 사항을 제어합니다.

##### SRFIQ

큐 관리자가 정지 중인 경우 MQSUBRQ 호출이 실패합니다.

**기본 옵션:** 이전에 설명된 옵션이 필요하지 않은 경우 다음 옵션을 사용해야 합니다.

##### SRNONE

기타 옵션이 지정되지 않았음을 표시하려면 이 값을 사용하십시오. 모든 옵션은 기본 값을 가정합니다.

SRNONE는 프로그램 문서화를 돕습니다. 이 옵션이 다른 옵션과 함께 사용하도록 작성된 것은 아니지만 해당 값이 0이기 때문에 이 사용을 감지할 수 없습니다.

#### SRSID(4바이트 문자열)

구조 ID이며 값은 다음과 같아야 합니다.

##### SRSIDV

구독 요청 SROPT 구조의 ID.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 SRSIDV입니다.

## SRVER(10자리의 부호 있는 정수)

구조 버전 번호이며 값은 다음과 같아야 합니다.

### SRVER1

버전-1 구독 요청 옵션 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

### SRVERC

구독 요청 옵션 구조의 현재 버전.

이 필드는 항상 입력 필드입니다. 이 필드의 초기값은 SRVER1입니다.

## 초기값

표 734. MQSRO의 필드		
필드 이름	상수의 이름	상수의 값
SRSID	SRSIDV	'SRO~'
SRVER	SRVER1	1
SROPT	SRNONE	0
SRNMP	없음	0

**참고:**

- ~ 기호는 단일 공백 문자를 나타냅니다.
- 널 문자열 값 또는 공백은 C의 널 문자열과 기타 프로그래밍 언어의 공백 문자를 나타냅니다.

## RPG 선언

```
D*..1.....2.....3.....4.....5.....6.....7..
D* MQSRO Structure
D*
D* Structure identifier
D SRSID          1          4
D* Structure version number
D SRVER          5          8I 0
D* Options that control the action of MQSUBRQ
D SROPT          9          12I 0
D* Number of publications sent
D SRNMP         13          16I 0
```

## IBM i IBM i의 MQSTS(상태 보고 구조)

MQSTS 구조는 MQSTAT 명령으로 리턴된 상태 구조의 데이터를 설명합니다.

### 개요

**문자 세트 및 인코딩:** MQSTS의 문자 데이터는 로컬 큐 관리자의 문자 세트입니다. 이는 *CodedCharSetId* 큐 관리자 속성으로 지정됩니다. MQSTS의 숫자 데이터는 원시 머신 인코딩으로 되어 있습니다. 이는 *ENNAT*를 통해 제공합니다.

**사용법:** MQSTAT 명령은 상태 정보를 검색하는 데 사용됩니다. 이 정보는 MQSTS 구조로 리턴됩니다. MQSTAT에 대한 정보는 1296 페이지의 『IBM i의 MQSTAT(상태 정보 검색)』의 내용을 참조하십시오.

- [1176 페이지의 『필드』](#)
- [1179 페이지의 『초기값』](#)
- [1179 페이지의 『RPG 선언』](#)

## 필드

MQSTS 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

### **STSCC(10자리의 부호 있는 정수)**

MQSTS 구조에서 보고된 첫 번째 오류의 결과인 완료 코드입니다.

이 필드는 항상 출력 필드입니다. 이 필드의 초기값은 CCOK입니다.

### **STSFCC(10자리의 부호 있는 정수)**

이는 실패한 비동기적 넣기 호출의 수입니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 0입니다.

### **STSOBJN(48바이트 문자열)**

이는 첫 번째 실패에 관련된 오브젝트의 로컬 이름입니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 48개의 빈 문자입니다.

### **STSOQMGR(48바이트 문자열)**

다음은 *STSOBJN* 오브젝트가 정의되어 있는 큐 관리자의 이름입니다. 첫 번째 널 문자 또는 필드의 마지막까지 완전히 비어 있는 이름은 애플리케이션이 연결된 큐 관리자를 나타냅니다(로컬 큐 관리자).

이 필드는 출력 필드입니다. 이 필드의 초기값은 48개의 빈 문자입니다.

### **STS00(10자리의 부호 있는 정수)**

STS00는 보고되는 오브젝트를 여는 데 사용됩니다. MQSTS의 버전 2 이상에만 제공됩니다.

STS00의 값은 MQSTAT **STYPE** 매개변수의 값에 따라 다릅니다.

#### **STATAPT**

0입니다.

#### **STATREC**

0입니다.

#### **STATRER**

STS00는 실패가 발생했을 때 사용됩니다. 실패에 대한 이유는 MQSTS 구조의 *STSCC* 및 *STSRC* 필드에 보고됩니다.

STS00는 출력 필드입니다. 초기값은 0입니다.

### **STSOS(MQCHARV)**

보고되고 있는 실패 오브젝트의 긴 오브젝트 이름. MQSTS의 버전 2 이상에만 제공됩니다.

STSOS는 최대 10240 길이의 MQCHARV 필드입니다. MQCHARV 구조를 사용하는 방법에 대한 설명은 [MQCHARV](#)를 참조하십시오.

STSOS의 해석은 MQSTAT **STYPE** 매개변수의 값에 따라 다릅니다.

#### **STATAPT**

이는 실패한 MQPUT 조작에 사용된 큐 또는 주제의 긴 오브젝트 이름입니다.

#### **STATREC**

0 길이 문자열

#### **STATRER**

재연결을 실패하게 만든 오브젝트의 긴 오브젝트 이름입니다.

STSOS는 출력 필드입니다. 해당 초기값은 길이가 0인 문자열입니다.

### STSOT(10자리의 부호 있는 정수)

*ObjectName*에 이름 지정된 오브젝트의 유형입니다. 가능한 값은 다음과 같습니다.

#### OTALSQ

알리어스 큐입니다.

#### OTLOCQ

로컬 큐.

#### OTMODQ

모델 큐입니다.

#### OTQ

큐.

#### OTREMQ

리모트 큐.

#### OTTOP

있습니다.

이 필드는 항상 출력 필드입니다. 이 필드의 초기값은 OTQ입니다.

### STSRC(10자리의 부호 있는 정수)

MQSTS 구조에서 보고된 첫 번째 오류의 결과인 이유 코드입니다.

이 필드는 항상 출력 필드입니다. 이 필드의 초기값은 RCNONE입니다.

### STSRBJN(48바이트 문자열)

로컬 큐 관리자가 이름을 해석한 후 이는 *STSRBJN*에 이름 지정된 목적지 큐의 이름입니다. 리턴된 이름은 *STSRQMGR*에 의해 식별된 큐 관리자에 있는 큐의 이름입니다.

오브젝트가 찾아보기, 입력 또는 출력(또는 결합)에 열린 단일 큐인 경우 비어 있는 값이 리턴됩니다. 열려 있는 오브젝트가 다음 중 하나인 경우 *STSRBJN*은(는) 공백으로 설정됩니다.

- 토픽
- 큐(단, 찾아보기, 입력 또는 출력을 위해 열려 있지 않음)

이 필드는 출력 필드입니다. 이 필드의 초기값은 48개의 빈 문자입니다.

### STSRQMGR(48바이트 문자열)

로컬 큐 관리자가 이름을 해석한 후 목적지 큐 관리자의 이름입니다. 리턴되는 이름은 *STSRBJN*(으)로 식별되는 큐를 소유하는 큐 관리자의 이름입니다. *STSRQMGR*은(는) 로컬 큐 관리자의 이름일 수 있습니다.

*STSRBJN*이(가) 로컬 큐 관리자가 속한 큐 공유 그룹이 소유하는 공유 큐인 경우, *STSRQMGR*은(는) 큐 공유 그룹의 이름입니다. 다른 큐 공유 그룹이 큐를 소유하는 경우, *STSRBJN*은(는) 큐 공유 그룹의 이름이거나 큐 공유 그룹의 구성원인 큐 관리자의 이름이 될 수 있습니다(리턴되는 값의 특성은 로컬 큐 관리자에 존재하는 큐 정의에 의해 결정됩니다).

오브젝트가 찾아보기, 입력 또는 출력(또는 결합)에 열린 단일 큐인 경우 비어 있는 값이 리턴됩니다. 열려 있는 오브젝트가 다음 중 하나인 경우 *STSRQMGR*은(는) 공백으로 설정됩니다.

- 토픽
- 큐(단, 찾아보기, 입력 또는 출력을 위해 열려 있지 않음)
- 지정된 OOBNDN이 있는 클러스터 큐(또는 **DefBind** 큐 속성에 값 OOBNDN이 있을 때 OOBNDQ가 적용됨)

이 필드는 출력 필드입니다. 이 필드의 초기값은 48개의 빈 문자입니다.

### STSSC(10자리의 부호 있는 정수)

이는 성공한 비동기적 넣기 호출의 수입니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 0입니다.

### **STSSID(4바이트 문자열)**

구조 ID입니다. 값은 다음과 같아야 합니다.

#### **STSSID**

상태 보고 구조의 ID입니다.

이 필드의 초기값은 STSSID입니다.

### **STSSO(10자리의 부호 있는 정수)**

STSSO는 실패한 구독을 여는 데 사용됩니다. MQSTS의 버전 2 이상에만 제공됩니다.

STSSO의 해석은 MQSTAT **STYPE** 매개변수의 값에 따라 다릅니다.

#### **STATAPT**

0입니다.

#### **STATREC**

0입니다.

#### **STATRER**

STSSO는 실패가 발생했을 때 사용됩니다. 실패에 대한 이유는 MQSTS 구조의 *STSCC* 및 *STSRC* 필드에 보고됩니다. 실패가 주제 구독과 관련이 없는 경우 리턴되는 값은 0입니다.

STSSO는 출력 필드입니다. 초기값은 0입니다.

### **STSSUN(MQCHARV)**

실패한 구독의 이름. MQSTS의 버전 2 이상에만 제공됩니다.

STSSUN은 최대 10240 길이의 MQCHARV 필드입니다. MQCHARV 구조를 사용하는 방법에 대한 설명은 [MQCHARV](#)를 참조하십시오.

STSSUN의 해석은 MQSTAT **STYPE** 매개변수의 값에 따라 다릅니다.

#### **STATAPT**

0 길이 문자열.

#### **STATREC**

0 길이 문자열.

#### **STATRER**

재연결을 실패하게 만든 구독의 이름입니다. 사용 가능한 구독 이름이 없거나 실패가 구독과 관련이 없는 경우에는 길이가 0인 문자열 때문입니다.

STSSUN는 출력 필드입니다. 해당 초기값은 길이가 0인 문자열입니다.

### **STSVR(10자리의 부호 있는 정수)**

구조 버전 번호입니다. 값은 다음과 같아야 합니다.

#### **STSVR1**

상태 보고 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### **STSVRC**

상태 보고 구조의 현재 버전입니다.

이 필드의 초기값은 STSVR1입니다.

### **STSWC(10자리의 부호 있는 정수)**

이는 경고로 완료된 비동기적 넣기 호출의 수입니다.

이 필드는 출력 필드입니다. 이 필드의 초기값은 0입니다.

## 초기값

표 735. MQSTS의 필드		
필드 이름	상수의 이름	상수의 값
STSSID	STSID	
STSVR	STSVRC	STSVR1
STSCC	CCOK	0
STSRC	RCNONE	0
STSSC	없음	0
STSWC	없음	0
STSFC	없음	0
STSOT	없음	0
STSOBJN	없음	공백
STSOQMGR	없음	공백
STSR OBJN	없음	공백
STSRQMGR	없음	공백
STSOS	MQCHARV에 정의된 이름과 값	
STSSUN	MQCHARV에 정의된 이름과 값	
STS00	없음	0
STSS0	없음	0

## RPG 선언

```

D*..1.....2.....3.....4.....5.....6.....7..
D* MQSTS Structure
D*
D* Structure identifier
D STSSID          1      4
D* Structure version number
D STSVR           5      8I 0
D* Completion code
D STSCC           9      12I 0
D* Reason code
D STSRC           13     16I 0
D* Success count
D STSSC           17     20I 0
D* Warning count
D STSWC           21     24I 0
D* Failure count
D STSFC           25     28I 0
D* Object type
D STSOT           29     32I 0
D* Object name
D STSOBJN         33     80
D* Object queue manager
D STSOQMGR        81     128
D* Resolved object name
D STSR OBJN      129     176
D* Resolved object queue manager name
D STSRQMGR       177     224
D* Ver:1 **
D* Failing object long name
D* Address of variable length string
D STSOSCHRP      225     240*
D* Offset of variable length string
D STSOSCHRO      241     244I 0

```

```

D* Size of buffer
D STSOSVSBS          245    248I 0
D* Length of variable length string
D STSOSCHRL         249    252I 0
D* CCSID of variable length string
D STSOSCHRC         253    256I 0
D* Failing subscription name
D* Address of variable length string
D STSSUNCHRP        257    272*
D* Offset of variable length string
D STSSUNCHRO        273    276I 0
D* Size of buffer
D STSSUNVSBS        277    280I 0
D* Length of variable length string
D STSSUNCHRL        281    284I 0
D* CCSID of variable length string
D STSSUNCHRC        285    288I 0
D* Failing open options
D STS00             289    292I 0
D* Failing subscription options
D STSS0             293    296I 0
D* Ver:2 **

```

## MQTM - 트리거 메시지

MQTM 구조는 트리거 이벤트가 큐에 발생할 때 트리거 모니터 애플리케이션에 큐 관리자가 전송한 트리거 메시지의 데이터를 설명합니다.

### 개요

**목적:** 이 구조는 TMI(IBM MQ Trigger Monitor Interface)의 일부이며 IBM MQ 프레임워크 인터페이스 중 하나입니다.

**형식 이름:** FMTM.

**문자 세트 및 인코딩:** MQTM의 문자 데이터는 MQTM을 생성하는 큐 관리자의 문자 세트 데이터입니다. MQTM의 숫자 데이터는 MQTM을 생성하는 큐 관리자의 시스템 인코딩 데이터입니다.

MQTM의 문자 세트 및 인코딩은 *MDCSI* 및 *MDENC* 필드에서 지정됩니다.

- MQMD(MQTM 구조가 메시지 데이터의 시작에 있는 경우) 또는
- MQTM 구조에 선행하는 헤더 구조(다른 모든 경우).

**사용법:** 애플리케이션은 트리거 모니터 애플리케이션에서 시작된 애플리케이션에 트리거 메시지의 일부 또는 모든 정보를 전달해야 할 수 있습니다. 시작된 애플리케이션에서 필요할 수 있는 정보는 *TMQN*, *TMTD* 및 *TMUD*를 포함합니다. 환경에서 허용되고 시작된 애플리케이션에 편리한 항목에 따라, 트리거 모니터 애플리케이션은 MQTM 구조를 시작된 애플리케이션에 직접 전달하거나 MQTMC2 구조를 대신 전달할 수 있습니다. MQTMC2에 대한 정보는 1184 페이지의 『IBM i의 MQTMC2(트리거 메시지 2 - 문자 형식)』의 내용을 참조하십시오.

- IBM i에서 IBM MQ가 있는 트리거 모니터 애플리케이션은 MQTMC2 구조를 시작된 애플리케이션에 전달합니다.

트리거에 대한 정보는 [트리거링의 전제조건](#)을 참조하십시오.

- [1180 페이지의 『트리거 메시지에 대한 MQMD』](#)
- [1181 페이지의 『필드』](#)
- [1183 페이지의 『초기값』](#)
- [1184 페이지의 『RPG 선언』](#)

### 트리거 메시지에 대한 MQMD

표 736. 큐 관리자가 생성한 트리거 메시지의 MQMD에 있는 필드에 대한 설정

MQMD의 필드	사용된 값
MDSID	MDSIDV



표 736. 큐 관리자가 생성한 트리거 메시지의 MQMD에 있는 필드에 대한 설정 (계속)

MQMD의 필드	사용된 값
MDVER	MDVER1
MDREP	RONONE
MDMT	MTDGRM
MDEXP	EIULIM
MDFB	FBNONE
MDENC	ENNAT
MDCSI	큐 관리자의 <b>CodedCharSetId</b> 속성
MDFMT	FMTM
MDPRI	이니시에이션 큐의 <b>DefPriority</b> 속성
MDPER	PENPER
MDMID	고유 값
MDCID	CINONE
MDBOC	0
MDRQ	공백
MDRM	큐 관리자의 이름
MDUID	공백
MDACC	ACNONE
MDAID	공백
MDPAT	ATQM 또는 메시지 채널 에이전트에 대해 적절하게 사용됨
MDPAN	큐 관리자 이름의 첫 번째 28바이트
MDPD	트리거 메시지를 송신한 날짜
MDPT	트리거 메시지를 송신한 시간
MDAOD	공백

트리거 메시지를 생성하는 애플리케이션은 다음을 제외하고 유사한 값을 설정하는 것이 좋습니다.

- *MDPRI* 필드를 *PRQDEF*로 설정할 수 있습니다(메시지를 넣을 때 큐 관리자가 이를 이니시에이션 큐의 기본 우선순위로 변경).
- *MDRM* 필드를 공백으로 설정할 수 있습니다(메시지를 넣을 때 큐 관리자가 이를 로컬 큐 관리자의 이름으로 변경합니다).
- 컨텍스트 필드는 애플리케이션에 적합하도록 설정되어야 합니다.

## 필드

MQTM 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

### TMAI(256바이트 문자열)

애플리케이션 ID입니다.

이는 시작될 애플리케이션을 식별하는 문자열이며, 트리거 메시지를 수신하는 트리거 모니터 애플리케이션에서 사용합니다. 큐 관리자는 *TMPN* 필드로 식별되는 프로세스 오브젝트의 **AppId** 속성 값을 사용하여 이 필드를 초기화합니다. 이 속성에 대한 세부사항은 1332 페이지의 『IBM i의 프로세스 정의에 대한 속성』의 내용을 참조하십시오. 이 데이터의 콘텐츠는 큐 관리자에 중요하지 않습니다.

*TMAI*의 의미는 트리거 모니터 애플리케이션에 의해 판별됩니다. IBM MQ에서 제공한 트리거 모니터에는 *TMAI*가 실행 가능 프로그램의 이름이 되어야 합니다.

이 필드의 길이는 LNPROA로 제공됩니다. 이 필드의 초기값은 256개의 공백 문자입니다.

#### **TMAT(10자리의 부호 있는 정수)**

애플리케이션 유형.

이는 시작될 프로그램의 네이처를 식별하고, 트리거 메시지를 수신하는 트리거 모니터 애플리케이션에서 사용됩니다. 큐 관리자는 *TMPN* 필드로 식별되는 프로세스 오브젝트의 **App1Type** 속성 값을 사용하여 이 필드를 초기화합니다. 이 속성에 대한 세부사항은 1332 페이지의 『IBM i의 프로세스 정의에 대한 속성』의 내용을 참조하십시오. 이 데이터의 콘텐츠는 큐 관리자에 중요하지 않습니다.

*TMAT*에는 다음 표준 값 중 하나가 있을 수 있습니다. 사용자 정의 유형 또한 사용될 수 있지만 *ATULST* - *ATUFST* 범위의 값으로 제한되어야 합니다.

#### **:CICS**

CICS 트랜잭션.

#### **ATVSE**

CICS/VSE 트랜잭션.

#### **AT400**

IBM i 애플리케이션입니다.

#### **ATUFST**

사용자 정의된 애플리케이션 유형의 최저값입니다.

#### **ATULST**

사용자 정의된 애플리케이션 유형의 최고값입니다.

이 필드의 초기값은 0입니다.

#### **TMED(128바이트 문자열)**

환경 데이터.

이는 시작될 애플리케이션과 관계가 있는 환경 관련 정보를 포함하는 문자열이며, 트리거 메시지를 수신하는 트리거 모니터 애플리케이션에서 사용됩니다. 큐 관리자는 *TMPN* 필드로 식별되는 프로세스 오브젝트의 **EnvData** 속성 값을 사용하여 이 필드를 초기화합니다. 이 속성에 대한 세부사항은 1332 페이지의 『IBM i의 프로세스 정의에 대한 속성』의 내용을 참조하십시오. 이 데이터의 콘텐츠는 큐 관리자에 중요하지 않습니다.

이 필드의 길이는 LNPROE로 제공됩니다. 이 필드의 초기값은 128개의 공백 문자입니다.

#### **TMPN(48바이트 문자열)**

프로세스 오브젝트의 이름.

이는 트리거된 큐에 지정된 큐 관리자 프로세스 오브젝트의 이름이며 트리거 메시지를 수신하는 트리거 모니터 애플리케이션에 의해 사용될 수 있습니다. 큐 관리자는 *TMQN* 필드로 식별되는 큐의 **ProcessName** 속성 값을 사용하여 이 필드를 초기화합니다. 이 속성에 대한 세부사항은 1305 페이지의 『큐의 속성』의 내용을 참조하십시오.

정의된 필드 길이보다 짧은 이름은 항상 오른쪽에 공백으로 채워집니다. 이름은 널 문자로 중간에 끝나지 않습니다.

이 필드의 길이는 LNPRON으로 제공됩니다. 이 필드의 초기값은 48개의 빈 문자입니다.

#### **TMQN(48바이트 문자열)**

트리거 큐의 이름.

이는 트리거 이벤트가 발생한 큐의 이름이며, 트리거 모니터 애플리케이션에 의해 시작된 애플리케이션에서 사용됩니다. 큐 관리자는 트리거된 큐의 **QName** 속성 값이 있는 이 필드를 초기화합니다. 이 속성의 세부사항은 1305 페이지의 『큐의 속성』의 내용을 참조하십시오.

정의된 필드 길이보다 짧은 이름은 오른쪽에 공백으로 채워집니다. 이름은 널 문자로 중간에 끝나지 않습니다.

이 필드의 길이는 LNQN으로 제공됩니다. 이 필드의 초기값은 48개의 빈 문자입니다.

#### TMSID(4바이트 문자열)

구조 ID입니다.

값은 다음과 같아야 합니다.

##### TMSIDV

트리거 메시지 구조의 ID입니다.

이 필드의 초기값은 TMSIDV입니다.

#### TMTD(64바이트 문자열)

트리거 데이터.

트리거 메시지를 수신하는 트리거 모니터 애플리케이션에서 사용할 자유 형식 데이터입니다. 큐 관리자는 *TMQN* 필드로 식별되는 큐의 **TriggerData** 속성 값을 사용하여 이 필드를 초기화합니다. 이 속성에 대한 세부사항은 1305 페이지의 『[큐의 속성](#)』의 내용을 참조하십시오. 이 데이터의 콘텐츠는 큐 관리자에 중요하지 않습니다.

이 필드의 길이는 LNTRGD로 제공됩니다. 이 필드의 초기값은 64개의 공백 문자입니다.

#### TMUD(128바이트 문자열)

사용자 데이터.

이는 시작될 애플리케이션과 관계가 있는 사용자 정보를 포함하는 문자열이며, 트리거 메시지를 수신하는 트리거 모니터 애플리케이션에서 사용됩니다. 큐 관리자는 *TMPN* 필드로 식별되는 프로세스 오브젝트의 **UserData** 속성 값을 사용하여 이 필드를 초기화합니다. 이 속성에 대한 세부사항은 1332 페이지의 『[IBM i의 프로세스 정의에 대한 속성](#)』의 내용을 참조하십시오. 이 데이터의 콘텐츠는 큐 관리자에 중요하지 않습니다.

이 필드의 길이는 LNPROU로 제공됩니다. 이 필드의 초기값은 128개의 공백 문자입니다.

#### TMVER(10자리의 부호 있는 정수)

구조 버전 번호입니다.

값은 다음과 같아야 합니다.

##### TMVER1

트리거 메시지 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

##### TMVERC

트리거 메시지 구조의 현재 버전.

이 필드의 초기값은 TMVER1입니다.

### 초기값

표 737. MQTM의 필드		
필드 이름	상수의 이름	상수의 값
TMSID	TMSIDV	'TM--'
TMVER	TMVER1	1
TMQN	없음	공백
TMPN	없음	공백
TMTD	없음	공백

표 737. MQTM의 필드 (계속)		
필드 이름	상수의 이름	상수의 값
TMAT	없음	0
TMAI	없음	공백
TMED	없음	공백
TMUD	없음	공백

**참고:**

1. ~ 기호는 단일 공백 문자를 나타냅니다.

### RPG 선언

```

D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQTM Structure
D*
D* Structure identifier
D TMSID          1      4    INZ(' TM ')
D* Structure version number
D TMVER          5      8I 0 INZ(1)
D* Name of triggered queue
D TMQN          9      56    INZ
D* Name of process object
D TMPN         57     104    INZ
D* Trigger data
D TMTD        105     168    INZ
D* Application type
D TMAT        169     172I 0 INZ(0)
D* Application identifier
D TMAI        173     428    INZ
D* Environment data
D TMED        429     556    INZ
D* User data
D TMUD        557     684    INZ

```

### IBM i IBM i의 MQTMC2(트리거 메시지 2 - 문자 형식)

트리거 모니터 애플리케이션이 트리거 메시지(MQTM)를 초기화 큐에서 검색할 때 트리거 모니터는 트리거 모니터에 의해 시작된 애플리케이션에 트리거 메시지 정보의 일부 또는 모두를 전달해야 합니다.

#### 개요

**목적:** 시작된 애플리케이션에서 필요할 수 있는 정보가 *TC2QN*, *TC2TD* 및 *TC2UD*를 포함합니다. 환경에서 허용되고 시작된 애플리케이션에 편리한 항목에 따라, 트리거 모니터 애플리케이션은 MQTM 구조를 시작된 애플리케이션에 직접 전달하거나 MQTMC2 구조를 대신 전달할 수 있습니다.

이 구조는 TMI(IBM MQ Trigger Monitor Interface)의 일부이며 IBM MQ 프레임워크 인터페이스 중 하나입니다.

**문자 세트 및 인코딩:** MQTMC2의 문자 데이터는 로컬 큐 관리자의 문자 세트입니다. 이는 **CodedCharSetId** 큐 관리자 속성에서 지정합니다.

**사용법:** MQTMC2 구조는 MQTM 구조의 형식과 유사합니다. 차이점은 MQTM의 비문자 필드가 MQTMC2의 동일한 길이의 문자 필드로 변경되었으며 큐 관리자 이름이 구조 끝에 추가되었습니다.

- IBM i에서 IBM MQ가 있는 트리거 모니터 애플리케이션은 MQTMC2 구조를 시작된 애플리케이션에 전달합니다.
- [1185 페이지의 『필드』](#)
- [1186 페이지의 『초기값』](#)

- [1186 페이지의 『RPG 선언』](#)

## 필드

MQTMC2 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

### **TC2AI(256바이트 문자열)**

애플리케이션 ID입니다.

MQTM 구조의 *TMAI* 필드를 참조하십시오.

### **TC2AT(4바이트 문자열)**

애플리케이션 유형.

원래 트리거 메시지의 MQTM 구조에 있는 *TMAT* 필드의 값에 상관 없이 이 필드에는 항상 공백이 포함됩니다.

### **TC2ED(128바이트 문자열)**

환경 데이터.

MQTM 구조의 *TMED* 필드를 참조하십시오.

### **TC2PN(48바이트 문자열)**

프로세스 오브젝트의 이름.

MQTM 구조의 *TMPN* 필드를 참조하십시오.

### **TC2QMN(48바이트 문자열)**

큐 관리자 이름.

트리거 이벤트가 발생한 큐 관리자의 이름입니다.

### **TC2QN(48바이트 문자열)**

트리거 큐의 이름.

MQTM 구조의 *TMQN* 필드를 참조하십시오.

### **TC2SID(4바이트 문자열)**

구조 ID입니다.

값은 다음과 같아야 합니다.

#### **TCSIDV**

트리거 메시지(문자 형식) 구조의 ID입니다.

### **TC2TD(64바이트 문자열)**

트리거 데이터.

MQTM 구조의 *TMTD* 필드를 참조하십시오.

### **TC2UD(128바이트 문자열)**

사용자 데이터.

MQTM 구조의 *TMUD* 필드를 참조하십시오.

### **TC2VER(4바이트 문자열)**

구조 버전 번호입니다.

값은 다음과 같아야 합니다.

#### **TCVER2**

버전 2 트리거 메시지(문자 형식) 구조입니다.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

## TCVERC

트리거 메시지(문자 형식화) 구조의 현재 버전.

### 초기값

표 738. MQTMC2의 필드		
필드 이름	상수의 이름	상수의 값
TC2SID	TCSIDV	'TMC~'
TC2VER	TCVER2	'~~2'
TC2QN	없음	공백
TC2PN	없음	공백
TC2TD	없음	공백
TC2AT	없음	공백
TC2AI	없음	공백
TC2ED	없음	공백
TC2UD	없음	공백
TC2QMN	없음	공백

**참고:**  
1. ~ 기호는 단일 공백 문자를 나타냅니다.

### RPG 선언

```
D*..1.....2.....3.....4.....5.....6.....7..
D* MQTMC2 Structure
D*
D* Structure identifier
D TC2SID 1 4
D* Structure version number
D TC2VER 5 8
D* Name of triggered queue
D TC2QN 9 56
D* Name of process object
D TC2PN 57 104
D* Trigger data
D TC2TD 105 168
D* Application type
D TC2AT 169 172
D* Application identifier
D TC2AI 173 428
D* Environment data
D TC2ED 429 556
D* User data
D TC2UD 557 684
D* Queue manager name
D TC2QMN 685 732
```

### IBM i IBM i의 MQWIH(작업 정보 헤더)

MQWIH 구조는 z/OS 워크로드 관리자에서 처리된 메시지의 시작 부분에 존재해야 하는 정보에 대해 설명합니다.

### 개요

형식 이름: FMWIH.

**문자 세트 및 인코딩:** MQWIH 구조의 필드는 MQWIH에 선행하는 헤더 구조의 *MDCSI* 및 *MDENC* 필드에 의해 제공된 문자 세트 및 인코딩에 있거나 MQWIH가 애플리케이션 메시지 데이터의 시작 부분에 있는 경우 MQMD 구조의 해당 필드에 의해 제공됩니다.

문자 세트는 큐 이름에 유효한 문자에 사용되는 단일 바이트 문자가 있어야 합니다.

**사용법:** 메시지가 z/OS 워크로드 관리자에 의해 처리되는 경우 메시지는 MQWIH 구조로 시작해야 합니다.

- [1187 페이지의 『필드』](#)
- [1189 페이지의 『초기값』](#)
- [1189 페이지의 『RPG 선언』](#)

## 필드

MQWIH 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

### WICSI(10자리의 부호 있는 정수)

MQWIH 뒤에 오는 데이터의 문자 세트 ID.

MQWIH 구조 뒤에 오는 데이터의 문자 세트 ID를 지정합니다. MQWIH 구조 자체의 문자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 다음 특수 값을 사용할 수 있습니다.

#### CSINHT

이 구조의 문자 세트 ID를 상속합니다.

이 구조 다음에 오는 데이터의 문자 데이터는 이 구조와 동일한 문자 세트로 되어 있습니다.

큐 관리자가 구조의 실제 문자 세트 ID에 메시지로 송신한 구조에서 이 값을 변경합니다. 오류가 발생하지 않으면 MQGET 호출에서 CSINHT 값을 리턴하지 않습니다.

MQMD의 *MDPAT* 필드 값이 ATBRKR이면 CSINHT를 사용할 수 없습니다.

이 필드의 초기값은 CSUNDF입니다.

### WIENC(10자리의 부호 있는 정수)

MQWIH 뒤에 오는 데이터의 숫자 인코딩

MQWIH 구조 뒤에 오는 데이터의 숫자 인코딩을 지정합니다. MQWIH 구조 자체의 숫자 데이터에는 적용되지 않습니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다.

이 필드의 초기값은 0입니다.

### WIFLG(10자리의 부호 있는 정수)

플래그

값은 다음과 같아야 합니다.

#### WINONE

플래그가 없습니다.

이 필드의 초기값은 WINONE입니다.

### WIFMT(8바이트 문자열)

MQWIH 뒤에 오는 데이터의 형식 이름.

MQWIH 구조 뒤에 오는 데이터의 형식 이름을 지정합니다.

MQPUT 또는 MQPUT1 호출에서 애플리케이션이 이 필드를 데이터에 적절한 값으로 설정해야 합니다. 이 필드의 코딩 규칙은 MQMD의 *MDFMT* 필드에 대한 규칙과 동일합니다.

이 필드의 길이는 LNFMT로 제공됩니다. 이 필드의 초기값은 FMNONE입니다.

**WILEN(10자리의 부호 있는 정수)**

MQWIH 구조 길이.  
값은 다음과 같아야 합니다.

**WILEN1**

버전-1 작업 정보 헤더 구조의 길이.  
다음 상수는 현재 버전의 길이를 지정합니다.

**WILENC**

작업 정보 헤더 구조의 현재 버전 길이입니다.  
이 필드의 초기값은 WILEN1입니다.

**WIRSV(32바이트 문자열)**

예약됨  
이는 예약된 필드이며 비어 있어야 합니다.

**WISID(4바이트 문자열)**

구조 ID입니다.  
값은 다음과 같아야 합니다.

**WISIDV**

작업 정보 헤더 구조의 ID입니다.  
이 필드의 초기값은 WISIDV입니다.

**WISNM(32바이트 문자열)**

서비스 이름.  
메시지를 처리하는 서비스의 이름입니다.  
이 필드의 길이는 LNSVNM으로 제공됩니다. 이 필드의 초기값은 32개의 공백 문자입니다.

**WISST(8바이트 문자열)**

서비스 단계 이름.  
이것은 메시지와 관련된 WISNM 단계의 이름입니다.  
이 필드의 길이는 LNSVST로 제공됩니다. 이 필드의 초기값은 8개의 빈 문자입니다.

**WITOK(16바이트 비트 문자열)**

메시지 토큰.  
고유하게 메시지를 식별하는 메시지 토큰입니다.  
MQPUT 및 MQPUT1 호출의 경우 이 필드는 무시됩니다. 이 필드의 길이는 LNMTOK로 제공됩니다. 이 필드의 초기값은 MTKNON입니다.

**WIVER(10자리의 부호 있는 정수)**

구조 버전 번호입니다.  
값은 다음과 같아야 합니다.

**WIVER1**

버전-1 작업 정보 헤더 구조 ID.  
다음 상수는 현재 버전의 버전 번호를 지정합니다.

**WIVERC**

작업 정보 헤더 구조의 현재 버전.  
이 필드의 초기값은 WIVER1입니다.



## 초기값

표 739. MQWIH의 필드		
필드 이름	상수의 이름	상수의 값
WISID	WISIDV	'WIH~'
WIVER	WIVER1	1
WILEN	WILEN1	120
WIENC	없음	0
WICSI	CSUNDF	0
WIFMT	FMNONE	공백
WIFLG	WINONE	0
WISNM	없음	공백
WISST	없음	공백
WITOK	MTKNON	널
WIRSV	없음	공백
참고:		
1. ~ 기호는 단일 공백 문자를 나타냅니다.		

## RPG 선언

```

D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQWIH Structure
D*
D* Structure identifier
D WISID          1          4    INZ('WIH ')
D* Structure version number
D WIVER          5          8I 0 INZ(1)
D* Length of MQWIH structure
D WILEN          9          12I 0 INZ(120)
D* Numeric encoding of data that followsMQWIH
D WIENC          13         16I 0 INZ(0)
D* Character-set identifier of data thatfollows MQWIH
D WICSI          17         20I 0 INZ(0)
D* Format name of data that followsMQWIH
D WIFMT          21         28    INZ('      ')
D* Flags
D WIFLG          29         32I 0 INZ(0)
D* Service name
D WISNM          33         64    INZ
D* Service step name
D WISST          65         72    INZ
D* Message token
D WITOK          73         88    INZ(X'00000000000000-
D                                     0000000000000000')
D* Reserved
D WIRSV          89         120   INZ
  
```

## IBM i IBM i의 MQXQH(전송 큐 헤더)

MQXQH 구조는 전송 큐에 있을 때 메시지의 애플리케이션 메시지 데이터의 앞에 붙여지는 정보를 설명합니다.

## 개요

**목적:** 전송 큐가 리모트 큐로 예정된 메시지를 임시로 보유하는 로컬 큐의 특수한 유형입니다(즉, 로컬 큐 관리자에 속하지 않는 큐로 예정됨). 전송 큐는 값 **USTRAN**이 있는 **Usage** 큐 속성으로 표시됩니다.

**형식 이름:** FMXQH.

**문자 세트 및 인코딩:** MQXQH의 데이터는 **CodedCharSetId** 큐 관리자 속성에 의해 지정된 문자 세트 및 C 프로그래밍 언어에 대한 ENNAT에 의해 지정된 로컬 큐 관리자의 인코딩에 있어야 합니다.

MQXQH의 문자 세트 및 인코딩은 **MDCSI** 및 **MDENC** 필드에 설정되어야 합니다.

- 별도의 MQMD(MQXQH 구조가 메시지 데이터의 시작에 있는 경우) 또는
- MQXQH 구조에 선행하는 헤더 구조(다른 모든 경우).

**사용법:** 전송 큐에 있는 메시지에는 두 개의 메시지 디스크립터가 있습니다.

- 하나의 메시지 디스크립터는 메시지 데이터와 별도로 저장됩니다. 개별 메시지 디스크립터라고 하며 메시지가 전송 큐에 우치되는 경우 큐 관리자에 의해 생성됩니다. 개별 메시지 디스크립터의 일부 필드는 MQPUT 또는 MQPUT1 호출에서 애플리케이션에 의해 제공된 메시지 디스크립터에서 복사됩니다.

별도의 메시지 디스크립터는 메시지가 전송 큐에서 제거될 때 MQGET 호출의 **MSGDSC** 매개변수의 애플리케이션으로 리턴됩니다.

- 두 번째 메시지 디스크립터는 메시지 데이터의 일부로 MQXQH 구조 내에 저장됩니다. 임베드된 메시지 디스크립터라고 하며 MQPUT 또는 MQPUT1 호출에서 (최소 변형으로) 애플리케이션에 의해 제공되는 메시지 디스크립터의 사본입니다.

임베드된 메시지 디스크립터는 항상 버전 1 MQMD입니다. 애플리케이션에서 넣은 메시지가 MQMD에 있는 하나 이상의 버전 2 필드에 대해 기본값이 아닌 값을 가지고 있는 경우 MQMDE 구조는 MQXQH 뒤에 오고, 차례대로 애플리케이션 메시지 데이터가 뒤에 옵니다(있는 경우). MQMDE는 다음 중 하나입니다.

- 큐 관리자에 의해 생성됨(애플리케이션에서 메시지를 넣는 데 버전 2 MQMD를 사용하는 경우) 또는
- 이미 애플리케이션 메시지 데이터의 시작에 존재함(애플리케이션에서 메시지를 넣는 데 버전 1 MQMD를 사용하는 경우).

임베드된 메시지 디스크립터는 메시지가 최종 목적지 큐에서 제거될 때 MQGET 호출의 **MSGDSC** 매개변수의 애플리케이션으로 리턴됩니다.

- [1190 페이지의 『별도의 메시지 디스크립터의 필드』](#)
- [1191 페이지의 『임베드된 메시지 디스크립터의 필드』](#)
- [1192 페이지의 『리모트 큐에 메시지 넣기』](#)
- [1192 페이지의 『전송 큐에 직접 메시지 넣기』](#)
- [1192 페이지의 『전송 큐에서 메시지 가져오기』](#)
- [1192 페이지의 『필드』](#)
- [1193 페이지의 『초기값』](#)
- [1194 페이지의 『RPG 선언』](#)

## 별도의 메시지 디스크립터의 필드

별도의 메시지 디스크립터의 필드는 다음 목록에 표시된 것처럼 큐 관리자로 설정됩니다. 큐 관리자가 버전 2 MQMD를 지원하지 않는 경우 버전 1 MQMD가 기능의 손실 없이 사용됩니다.

표 740. 별도의 메시지 디스크립터의 필드 및 사용되는 값

별도의 MQMD의 필드	사용된 값
MDSID	MDSIDV
MDVER	MDVER2

표 740. 별도의 메시지 디스크립터의 필드 및 사용되는 값 (계속)

별도의 MQMD의 필드	사용된 값
MDREP	임베드된 메시지 디스크립터에서 복사되지만 ROAUXM으로 식별된 비트가 0으로 설정됩니다 (메시지를 전송 큐에 넣거나 전송 큐에서 메시지가 제거될 때 COA 또는 COD 보고서 메시지가 생성되는 것을 방지합니다.)
MDMT	임베드된 메시지 디스크립터에서 복사합니다.
MDEXP	임베드된 메시지 디스크립터에서 복사합니다.
MDFB	임베드된 메시지 디스크립터에서 복사합니다.
MDENC	ENNAT
MDCSI	큐 관리자의 <b>CodedCharSetId</b> 속성.
MDFMT	FMXQH
MDPRI	임베드된 메시지 디스크립터에서 복사합니다.
MDPER	임베드된 메시지 디스크립터에서 복사합니다.
MDMID	새 값이 큐 관리자에 의해 생성됩니다. 이 메시지 ID는 큐 관리자가 임베드된 메시지 디스크립터에 대해 생성되었을 수 있는 <i>MDMID</i> 와 다릅니다(이전 설명 참조).
MDCID	임베드된 메시지 디스크립터의 <i>MDMID</i> 입니다.
MDBOC	0
MDRQ	임베드된 메시지 디스크립터에서 복사합니다.
MDRM	임베드된 메시지 디스크립터에서 복사합니다.
MDUID	임베드된 메시지 디스크립터에서 복사합니다.
MDACC	임베드된 메시지 디스크립터에서 복사합니다.
MDAID	임베드된 메시지 디스크립터에서 복사합니다.
MDPAT	ATQM
MDPAN	큐 관리자 이름의 첫 번째 28바이트.
MDPD	메시지를 전송 큐에 넣은 날짜.
MDPT	메시지를 전송 큐에 넣은 시간.
MDAOD	공백
MDGID	GINONE
MDSEQ	1
MDOFF	0
MDMFL	MFNONE
MDOLN	OLUNDF

### 임베드된 메시지 디스크립터의 필드

임베드된 메시지 디스크립터의 필드에는 다음을 제외하고 MQPUT 또는 MQPUT1 호출의 **MSGDSC** 매개변수의 필드와 동일한 값이 있습니다.

- *MDVER* 필드에는 항상 값 *MDVER1*이 있습니다.
- *MDPRI* 필드에 값 *PRQDEF*가 있는 경우 큐의 **DefPriority** 속성의 값으로 대체됩니다.
- *MDPER* 필드에 값 *PEQDEF*가 있는 경우 큐의 **DefPersistence** 속성의 값으로 대체됩니다.

- *MDMID* 필드에 값 *MINONE*가 있거나 *PMNMID* 옵션이 지정되었거나 메시지가 분배 목록 메시지인 경우 *MDMID*는 큐 관리자에서 생성한 새 메시지 ID로 대체됩니다.

분배 목록 메시지가 다른 전송 큐에 배치된 작은 분배 목록 메시지로 분할될 때 각각의 새 임베드된 메시지 디스크립터의 *MDMID* 필드는 원래 분배 목록 메시지의 필드와 동일합니다.

- *PMNCID* 옵션이 지정되는 경우 *MDCID*는 큐 관리자에서 생성한 새 상관 ID로 대체됩니다.
- 컨텍스트 필드는 **PMO** 매개변수에 지정된 *PM\** 옵션으로 표시되는 것처럼 설정됩니다. 컨텍스트 필드는 다음과 같습니다.
  - *MDACC*
  - *MDAID*
  - *MDAOD*
  - *MDPAN*
  - *MDPAT*
  - *MDPD*
  - *MDPT*
  - *MDUID*
- 하나 이상의 버전 2 필드에 기본값이 아닌 값이 있는 경우 MQMD에서 버전 2 필드(존재하는 경우)가 제거되고 MQMDE 구조로 이동됩니다.

## 리모트 큐에 메시지 넣기

: 애플리케이션이 리모트 큐에 메시지를 넣을 때(리모트 큐의 이름을 직접 지정하거나 리모트 큐의 로컬 정의를 사용하여) 로컬 큐 관리자는 다음과 같습니다.

- 임베드된 메시지 디스크립터를 포함하는 MQXQH 구조 작성
- 하나가 필요하고 이미 존재하지 않는 경우 MQMDE 추가
- 애플리케이션 메시지 데이터 추가
- 메시지를 적절한 전송 큐에 넣기

## 전송 큐에 직접 메시지 넣기

또한 애플리케이션이 전송 큐에 직접 메시지를 넣는 것도 가능합니다. 이 경우 애플리케이션은 애플리케이션 메시지 데이터에 접두부로 MQXQH 구조를 지정해야 하며 적절한 값으로 필드를 초기화해야 합니다. 또한 MQPUT 또는 MQPUT1 호출의 **MSGDSC** 매개변수에 있는 *MDFMT* 필드에는 *FMXQH*값이 있어야 합니다.

애플리케이션에서 작성한 MQXQH 구조의 문자 데이터는 로컬 큐 관리자의 문자 세트에 있어야 하고 (**CodedCharSetId** 큐 관리자 속성에 의해 정의됨) 정수 데이터는 고유 시스템 인코딩에 있어야 합니다. 뿐만 아니라, MQXQH 구조의 문자 데이터는 정의된 필드 길이까지 공백으로 채워져야 합니다. 큐 관리자가 널 및 후속 문자를 MQXQH 구조의 공백으로 변환하지 않기 때문에, 데이터는 널 문자로 중간에 끝나면 안됩니다.

그러나 큐 관리자는 MQXQH 구조가 존재하는지 또는 올바른 값이 필드에 지정되었는지 확인하지 않습니다.

## 전송 큐에서 메시지 가져오기

전송 큐에서 메시지를 가져오는 애플리케이션은 고유 방식으로 MQXQH 구조에서 정보를 처리해야 합니다. 애플리케이션 메시지 데이터의 시작 부분에 MQXQH 구조가 있으면 MQGET 호출의 **MSGDSC** 매개변수에서 *MDFMT* 필드에 리턴되는 *FMXQH*값으로 표시됩니다. **MSGDSC** 매개변수의 *MDCSI* 및 *MDENC* 필드에 리턴된 값은 MQXQH 구조에서 문자 및 정수 데이터의 문자 세트 및 인코딩을 표시합니다. 애플리케이션 메시지 데이터의 문자 세트 및 인코딩은 임베드된 메시지 디스크립터의 *MDCSI* 및 *MDENC* 필드에서 정의됩니다.

## 필드

MQXQH 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

## XQMD(MQMD1)

원래 메시지 디스크립터.

이는 임베드된 메시지 디스크립터이며, 메시지를 원래 리모트 큐에 넣을 때 MQPUT 또는 MQPUT1 호출에서 **MSGDSC** 매개변수로 지정되는 메시지 디스크립터 MQMD의 닫기 사본입니다.

**참고:** 이는 버전-1 MQMD입니다.

이 구조에서 필드의 초기값은 MQMD 구조의 초기값과 동일합니다.

## XQRQ(48바이트 문자열)

목적지 큐의 이름.

이는 메시지의 명백한 최종 목적지인 메시지 큐의 이름입니다(예를 들어, 이 큐가 XQRQM에서 다른 리모트 큐의 로컬 정의로 정의된 경우 실제 최종 목적지가 될 수 없음).

메시지가 분배 목록 메시지(즉, 임베드된 메시지 디스크립터의 *MDFMT* 필드가 FMDH임)인 경우 XQRQ는 비어 있습니다.

이 필드의 길이는 LNQN으로 제공됩니다. 이 필드의 초기값은 48개의 빈 문자입니다.

## XQRQM(48바이트 문자열)

목적지 큐 관리자의 이름.

메시지의 명확한 최종 목적지인 큐를 소유하는 큐 공유 그룹 또는 큐 관리자의 이름입니다.

메시지가 분배 목록 메시지인 경우, XQRQM은(는) 공백입니다.

이 필드의 길이는 LNQMN에서 제공됩니다. 이 필드의 초기값은 48개의 빈 문자입니다.

## XQSID(4바이트 문자열)

구조 ID입니다.

값은 다음과 같아야 합니다.

### XQSIDV

전송 큐 헤더 구조의 ID.

이 필드의 초기값은 XQSIDV입니다.

## XQVER(10자리의 부호 있는 정수)

구조 버전 번호입니다.

값은 다음과 같아야 합니다.

### XQVER1

전송 큐 헤더 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

### XQVERC

전송 큐 헤더 구조의 현재 버전.

이 필드의 초기값은 XQVER1입니다.

## 초기값

표 741. MQXQH의 필드		
필드 이름	상수의 이름	상수의 값
XQSID	XQSIDV	'XQH-'
XQVER	XQVER1	1
XQRQ	없음	공백

표 741. MQXQH의 필드 (계속)		
필드 이름	상수의 이름	상수의 값
XQRQM	없음	공백
XQMD	MQMD와 동일한 이름 및 값, <u>1101 페이지의 표 709 참조</u>	-
<b>참고:</b> 1. ~ 기호는 단일 공백 문자를 나타냅니다.		

## RPG 선언

```

D*..1.....2.....3.....4.....5.....6.....7..
D*
D* MQXQH Structure
D*
D* Structure identifier
D XQSID          1      4      INZ('XQH ')
D* Structure version number
D XQVER          5      8I 0  INZ(1)
D* Name of destination queue
D XQRQ           9      56     INZ
D* Name of destination queue manager
D XQRQM          57     104     INZ
D* Original message descriptor
D XQ1SID         105    108     INZ('MD ')
D XQ1VER         109    112I 0  INZ(1)
D XQ1REP         113    116I 0  INZ(0)
D XQ1MT          117    120I 0  INZ(8)
D XQ1EXP         121    124I 0  INZ(-1)
D XQ1FB          125    128I 0  INZ(0)
D XQ1ENC         129    132I 0  INZ(273)
D XQ1CSI         133    136I 0  INZ(0)
D XQ1FMT         137    144     INZ(' ')
D XQ1PRI         145    148I 0  INZ(-1)
D XQ1PER         149    152I 0  INZ(2)
D XQ1MID         153    176     INZ(X'00000000000000-
D                                     00000000000000000000-
D                                     000000000000')
D XQ1CID         177    200     INZ(X'00000000000000-
D                                     00000000000000000000-
D                                     000000000000')
D XQ1BOC         201    204I 0  INZ(0)
D XQ1RQ          205    252     INZ
D XQ1RM          253    300     INZ
D XQ1UID         301    312     INZ
D XQ1ACC         313    344     INZ(X'00000000000000-
D                                     00000000000000000000-
D                                     00000000000000000000-
D                                     000000')
D XQ1AID         345    376     INZ
D XQ1PAT         377    380I 0  INZ(0)
D XQ1PAN         381    408     INZ
D XQ1PD          409    416     INZ
D XQ1PT          417    424     INZ
D XQ1AOD         425    428     INZ

```

## IBM i IBM i의 함수 호출

이 정보를 사용하여 IBM i 프로그래밍에서 사용 가능한 함수 호출에 대해 학습합니다.

### IBM i의 호출 설명에 사용된 규칙

각 호출의 경우 토픽의 이 컬렉션은 호출의 매개변수와 사용법의 설명을 제공합니다. RPG 프로그래밍 언어로 매개변수의 일반 선언 및 일반적인 호출의 호출이 다음에 나타납니다.

**중요사항:** IBM MQ API 호출을 코딩할 때 다음 절에서 설명된 대로 모든 관련 매개변수가 제공되는지 확인해야 합니다. 이를 수행하지 못하면 예측 불가능한 결과가 생성될 수 있습니다.

각 호출의 설명에는 다음 섹션이 포함되어 있습니다.

#### 호출 이름

호출 이름, 뒤에 호출 목적에 대한 간단한 설명이 나옵니다.

#### 매개변수

각 매개변수의 경우 이름 뒤에는 괄호() 및 해당 방향으로 해당 데이터 유형이 옵니다(예:

`CMPCOD` (9자리 10진수 정수) - 출력)

957 페이지의 『기본 데이터 유형』에 구조 데이터 유형에 관한 자세한 정보가 있습니다.

매개변수 방향은 다음과 같을 수 있습니다.

#### 입력

프로그래머는 이 매개변수를 제공해야 합니다.

#### Output

호출은 이 매개변수를 리턴합니다.

#### 입출력(I/O)

이 매개변수를 제공해야 하지만 이 매개변수는 호출에서 수정됩니다.

매개변수가 가져올 수 있는 값 목록과 함께 매개변수의 용도에 대한 간단한 설명도 있습니다.

각 호출의 마지막 두 개 매개변수는 완료 코드와 이유 코드입니다. 완료 코드는 호출이 성공, 일부 성공 또는 실패되었는지를 표시합니다. 호출의 일부 성공 또는 실패에 대한 추가 정보는 이유 코드에서 제공됩니다.

#### 사용법 참고

호출에 대한 추가 정보이며, 사용 방법 및 사용에 대한 제한사항을 설명합니다.

#### RPG 호출

RPG로 일반적인 호출의 호출 및 해당 매개변수 선언.

기타 표기 규칙은 다음과 같습니다.

#### Constants

상수 이름은 대문자로 표시됩니다(예: `OOOUT`).

#### 배열

일부 호출의 경우 매개변수는 고정되지 않은 크기의 문자열의 배열입니다. 이 매개변수 설명에서 소문자 *n*은 숫자 상수를 표시합니다. 해당 매개변수에 대한 선언을 코드하는 경우 *n*을 필요한 숫자 값으로 바꾸십시오.

## IBM i IBM i의 MQBACK(변경사항 백아웃)

MQBACK 호출은 마지막 동기점 이후에 발생한 모든 메시지 가져오기 및 넣기가 큐 관리자에게 백아웃되도록 지시합니다. 작업 단위의 일부로 넣어진 메시지는 삭제되지만, 작업 단위의 일부로 검색된 메시지는 큐에 복구됩니다.

• 이 호출은 다음 환경에서 지원됩니다.

-  AIX
-  IBM i
-  Windows

- [1195 페이지의 『구문』](#)
- [1196 페이지의 『사용법 참고』](#)
- [1197 페이지의 『매개변수』](#)
- [1197 페이지의 『RPG 선언』](#)

## 구문

MQBACK (*Hconn*, *CompCode*, *Reason*)

## 사용법 참고

MQBACK을 사용할 때 이러한 사용법 참고사항을 고려하십시오.

1. 큐 관리자 자체가 작업 단위를 조정할 때만 이 호출은 사용될 수 있습니다. 이는 변경사항이 IBM MQ 자원에만 영향을 주는 작업의 로컬 단위입니다.
2. 큐 관리자가 작업 단위를 조정하지 않는 환경에서 적절한 백아웃 호출은 MQBACK 대신에 사용되어야 합니다. 또한 환경은 비정상적으로 종료되는 애플리케이션으로 인한 암시적 백아웃을 지원할 수도 있습니다.
  - IBM i에서 이 호출은 큐 관리자가 조정한 로컬 작업 단위에 사용될 수 있습니다. 이는 커밋 정의가 작업 레벨에 존재하지 않아야 함을 의미합니다. 즉, **CMTSCOPE(\*JOB)** 매개변수가 있는 **STRCMTCTL** 명령이 작업에 대해 실행되지 않았어야 합니다.
3. 애플리케이션이 작업 단위에서 커밋되지 않은 변경사항으로 종료되는 경우, 해당 변경사항의 배치는 애플리케이션이 정상으로 또는 비정상적으로 종료되는지 여부에 달려 있습니다. 추가적인 세부사항은 [1231 페이지의 『IBM i의 MQDISC\(큐 관리자 연결 끊기\)』](#)의 사용 시 참고사항을 참조하십시오.
4. 애플리케이션이 논리 메시지의 세그먼트 또는 그룹에서 메시지를 넣거나 가져올 때 큐 관리자는 마지막 성공한 MQPUT 및 MQGET 호출에 대한 메시지 그룹 및 논리 메시지와 관련된 정보를 보유하고 있습니다. 이 정보는 큐 핸들과 연관되어 있으며 다음과 같은 내용을 포함합니다.
  - MQMD에 있는 *MDGID*, *MDSEQ*, *MDOFF* 및 *MDMFL* 필드의 값입니다.
  - 메시지가 작업 단위의 일부인지 여부.
  - MQPUT 호출의 경우: 메시지가 지속적인지 또는 비지속적인지 여부.

큐 관리자는 다음 각각에 대해 한 세트씩, 그룹 및 세그먼트 정보의 세 개 세트를 유지합니다.

- 마지막 성공적인 MQPUT 호출(이는 작업 단위의 일부일 수 있음).
- 큐에서 메시지를 제거한 마지막 성공적인 MQGET 호출(이는 작업 단위의 일부일 수 있음).
- 큐에서 메시지를 찾아보는 마지막으로 성공한 MQGET 호출(작업 단위의 일부가 될 수 없음).

애플리케이션이 작업 단위의 일부로 메시지를 넣거나 가져온 다음 작업 단위를 백아웃하기로 결정한 경우, 그룹 및 세그먼트 정보가 이전 값으로 복원됩니다.

- MQPUT 호출과 연관된 정보가 첫 번째 성공적인 MQPUT 호출 이전 현재 작업 단위의 큐 핸들에 대한 값으로 복원됩니다.
- MQGET 호출과 연관된 정보가 현재 작업 단위에서 해당 큐 핸들에 대해 처음으로 성공한 MQGET 호출 전에 가지고 있던 값으로 복원됩니다.

작업 단위가 시작된 후 애플리케이션에 의해 작업 단위의 범위 외부에 업데이트된 큐가 해당 그룹 및 세그먼트 정보를 작업 단위가 외부에 돌아오면 복원되게 하지 않습니다.

작업의 단위가 백아웃될 때 그룹과 세그먼트 정보를 해당 이전 값으로 복구하면 애플리케이션이 여러 작업 단위 전체에 걸쳐 많은 세그먼트로 구성되어 대량 메시지 그룹 또는 대량 논리 메시지를 펼칠 수 있습니다. 그리고 작업 단위 중 하나가 실패하면 메시지 그룹 또는 논리 메시지의 정확한 위치에서 재시작하도록 허용합니다. 로컬 큐 관리자에 제한된 큐 스토리지만 있는 경우에는 여러 작업 단위를 사용하면 유용할 수 있습니다. 시스템 장애가 발생한 경우 그러나 애플리케이션은 정확한 위치에 메시지를 넣거나 가져와서 재시작할 수 있기 위한 충분한 정보를 유지보수해야 합니다. 시스템 장애 뒤에, 정확한 포인트에 재시작하는 방법에 대한 자세한 내용에 대해서 [1122 페이지의 『IBM i의 MQPMO\(메시지 넣기 옵션\)』](#)에서 설명된 **PMLOGO** 옵션 및 [1032 페이지의 『IBM i의 MQGMO\(메시지 가져오기 옵션\)』](#)에서 설명된 **GMLOGO** 옵션을 참조하십시오.

나머지 사용 시 참고사항은 큐 관리자가 작업 단위를 통합할 때만 적용됩니다.

1. 작업 단위는 연결 핸들과 동일한 범위를 갖습니다. 이는 특별한 작업 단위에 영향을 주는 모든 IBM MQ 호출이 동일한 연결 핸들을 사용하여 수행되어야 함을 의미합니다. 다른 연결 핸들을 사용하여 실행된 호출(예: 다른 애플리케이션이 발행한 호출)은 다음 작업 단위에 영향을 줍니다. 연결 핸들의 범위에 대한 정보는 [1219 페이지의 『IBM i의 MQCONN\(큐 관리자 연결\)』](#)에서 설명하는 **HCONN** 매개변수를 참조하십시오.
2. 현재 작업 단위의 일부로서 넣거나 검색된 메시지만 이 호출의 영향을 받습니다.
3. 작업 단위 내에서 MQGET, MQPUT 또는 MQPUT1 호출을 발행하지만 커밋 또는 백아웃 호출을 발행하지 않는 장기 실행 애플리케이션은 다른 애플리케이션으로 사용 불가능한 메시지로 큐를 채울 수 있습니다. 이 가능성을 예방하려면 **MaxUncommittedMsgs** 관리자는 속성을 애플리케이션이 큐를 채우지는 않지만 예상



된 메시징 애플리케이션이 올바르게 작동할 수 있을 만큼 충분히 높은 이탈을 방지하기에 충분히 낮은 값으로 설정해야 합니다.

## 매개변수

MQBACK 호출에는 다음 매개변수가 존재합니다.

### HCONN(10자리 부호 있는 정수) - 입력

연결 핸들입니다.

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. HCONN의 값은 이전 MQCONN 또는 MQCONNx 호출에 의해 리턴되었습니다.

### CMPCOD(10자리 부호 있는 정수) - 출력

완료 코드.

다음 중 하나입니다.

#### CCOK

정상적으로 완료되었습니다.

#### CCFAIL

호출에 실패했습니다.

### REASON(10자리 부호 있는 정수) - 출력

COMCOD을(를) 규정하는 이유 코드.

COMCOD가 CCOK일 경우:

#### RCNONE

(0, X'000') 보고할 이유가 없습니다.

COMCOD가 CCFail일 경우:

#### RC2219

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 다시 입력되었습니다.

#### RC2009

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

#### RC2018

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

#### RC2101

(2101, X'835') 오브젝트가 손상되었습니다.

#### RC2123

(2123, X'84B') 커밋 또는 백아웃 조작의 결과가 혼합되어 있습니다.

#### RC2162

(2162, X'872') 큐 관리자가 종료되었습니다.

#### RC2102

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

#### RC2071

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

#### RC2195

(2195, X'893') 예상치 못한 오류가 발생했습니다.

## RPG 선언

```
C*..1.....2.....3.....4.....5.....6.....7..  
C          CALLP      MQBACK(HCONN : COMCOD : REASON)
```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```
D*..1.....2.....3.....4.....5.....6.....7..
DMQBACK          PR          EXTPROC('MQBACK')
D* Connection handle
D HCONN          10I 0 VALUE
D* Completion code
D COMCOD          10I 0
D* Reason code qualifying COMCOD
D REASON          10I 0
```

## IBM i IBM i의 MQBEGIN(작업 단위 시작)

MQBEGIN 호출은 큐 관리자에 의해 조정되고 외부 자원 관리자를 포함할 수 있는 작업 단위를 시작합니다.

• 이 호출은 다음 환경에서 지원됩니다.

-  AIX
-  IBM i
-  Windows

- [1198 페이지의 『구문』](#)
- [1198 페이지의 『사용법 참고』](#)
- [1199 페이지의 『매개변수』](#)
- [1200 페이지의 『RPG 선언』](#)

### 구문

MQBEGIN (*HCONN*, *BEGOP*, *CMPCOD*, *REASON*)

### 사용법 참고

1. MQBEGIN 호출은 큐 관리자에 의해 조정되고 다른 자원 관리자에 의해 소유된 자원의 변경사항을 포함할 수 있는 작업 단위를 시작할 때 사용할 수 있습니다. 큐 관리자는 작업 단위의 세 가지 유형을 지원합니다.

#### 큐 관리자 통합 로컬 작업 단위

이는 큐 관리자가 참여하는 유일한 자원 관리자인 작업 단위이며 큐 관리자는 작업 단위 코디네이터의 역할을 합니다.

- 작업 단위의 이런 유형을 시작하려면 PMSYP 또는 GMSYP 옵션은 작업 단위에서 첫 번째 MQPUT 또는 MQPUT1 또는 MQGET 호출에 지정되어야 합니다.

작업 단위를 시작하려면 MQBEGIN 호출 발행이 애플리케이션에 필요하지 않지만 MQBEGIN이 사용되는 경우 호출이 CCWARN와 이유 코드 RC2121로 완료됩니다.

- 작업 단위의 이런 유형으로 커밋하거나 백아웃하려면 MQCMIT 또는 MQBACK 호출이 사용되어야 합니다.

#### 큐 관리자 통합 글로벌 작업 단위

이는 큐 관리자가 IBM MQ 자원 및 다른 자원 관리자에 속하는 자원 모두에 대해 작업 단위 코디네이터 역할을 하는 작업 단위입니다. 해당 자원 관리자는 작업 단위의 자원의 모든 변경사항이 커밋되거나 함께 백아웃되는지 확인하기 위해 큐 관리자와 협력합니다.

- 작업 단위의 이런 유형을 시작하려면 MQBEGIN 호출이 사용되어야 합니다.
- 작업 단위의 이런 유형으로 커밋하거나 백아웃하려면 MQCMIT 및 MQBACK 호출이 사용되어야 합니다.

#### 외부적으로 통합된 글로벌 작업 단위

이는 큐 관리자가 참가자이지만 큐 관리자가 작업 단위 코디네이터의 역할을 하지 않는 작업 단위입니다. 대신 큐 관리자가 협력하는 외부 작업 단위 코디네이터가 있습니다.

- 작업 단위의 이런 유형을 시작하려면 외부 작업 단위 코디네이터에 의해 제공된 관련 호출이 사용되어야 합니다.

MQBEGIN 호출이 작업 단위를 시작하려고 하는 데 사용되면, 호출은 이유 코드 RC2012로 실패합니다.

- 작업 단위의 이런 유형으로 커밋하거나 백아웃하려면 외부 작업 단위 코디네이터가 제공한 커밋 및 백아웃 호출이 사용되어야 합니다.

MQCMIT 또는 MQBACK 호출이 작업 단위를 커밋하거나 백아웃하는 데 사용되는 경우 호출은 이유 코드 RC2012로 실패합니다.

2. 애플리케이션이 작업 단위에서 커밋되지 않은 변경사항으로 종료되는 경우 해당 변경사항의 배치는 애플리케이션이 정상으로 또는 비정상적으로 종료되는지 여부에 달려 있습니다. 추가적인 세부사항은 [1231 페이지의 『IBM i의 MQDISC\(큐 관리자 연결 끊기\)』](#)의 사용 시 참고사항을 참조하십시오.
3. 애플리케이션은 동시에 단지 하나의 작업 단위에만 참여할 수 있습니다. 작업 단위의 유형과 관계없이 애플리케이션에 작업 단위가 이미 존재하는 경우 MQBEGIN 호출은 이유 코드 RC2128로 실패합니다.
4. MQBEGIN 호출은 IBM MQ 클라이언트 환경에서 유효하지 않습니다. 호출을 사용하기 위한 시도는 이유 코드 RC2012로 실패합니다.
5. 큐 관리자가 글로벌 작업 단위의 작업 단위 코디네이터 역할을 하고 있을 때 작업 단위에 참여할 수 있는 자원 관리자는 큐 관리자의 구성 파일에서 정의됩니다.
6. IBM i에서 작업 단위의 세 가지 유형은 다음과 같이 지원됩니다.
  - 큐 관리자 조정 로컬 작업 단위는 커밋 정의가 작업 레벨에 없는 경우에만 사용할 수 있습니다. 즉, **CMTSCOPE(\*JOB)** 매개변수가 있는 STRCMTCTL 명령이 작업에 대해 실행되지 않아야 합니다.
  - 큐 관리자 통합 글로벌 작업 단위는 지원되지 않습니다.
  - 외부적으로 조정된 글로벌 작업 단위는 커밋 정의가 작업 레벨에 있는 경우에만 사용할 수 있습니다. 즉, **CMTSCOPE(\*JOB)** 매개변수와 함께 STRCMTCTL 명령을 작업에 대해 실행해야 합니다. 이를 완료한 경우 IBM i COMMIT 및 ROLLBACK 조작은 다른 참여하는 자원 관리자에 속하는 자원뿐 아니라 IBM MQ 자원에도 적용됩니다.

## 매개변수

MQBEGIN 호출에는 다음 매개변수가 존재합니다.

### HCONN(10자리 부호 있는 정수) - 입력

연결 핸들입니다.

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. HCONN의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

### BEGOP(MQBO) - 입출력(I/O)

MQBEGIN 조치를 제어하는 옵션입니다.

자세한 내용은 977 페이지의 『IBM i의 MQBO(시작 옵션)』의 내용을 참조하십시오.

옵션이 필요하지 않은 경우 C 또는 S/390 어셈블러로 작성된 프로그램은 MQBO 구조의 주소를 지정하는 대신에 널 매개변수 주소를 지정할 수 있습니다.

### CMPCOD(10자리 부호 있는 정수) - 출력

완료 코드.

다음 중 하나입니다.

#### CCOK

정상적으로 완료되었습니다.

#### CCWARN

경고(일부 완료).

#### CCFAIL

호출에 실패했습니다.

## REASON(10자리 부호 있는 정수) - 출력

CMPCOD을(를) 규정하는 이유 코드.

CMPCOD가 CCOK일 경우:

### RCNONE

(0, X'000') 보고할 이유가 없습니다.

If CMPCOD is CCWARN:

### RC2121

(2121, X'849') 참여하는 자원 관리자가 등록되어 있지 않습니다.

### RC2122

(2122, X'84A') 참여하는 자원 관리자를 사용할 수 없습니다.

CMPCOD가 CCFAIL일 경우:

### RC2134

(2134, X'856') 시작 옵션 구조가 올바르지 않습니다.

### RC2219

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 다시 입력되었습니다.

### RC2009

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

### RC2012

(2012, X'7DC') 호출이 환경 내에서 유효하지 않습니다.

### RC2018

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

### RC2046

(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.

### RC2162

(2162, X'872') 큐 관리자가 종료되었습니다.

### RC2102

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

### RC2071

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

### RC2195

(2195, X'893') 예상치 못한 오류가 발생했습니다.

### RC2128

(2128, X'850') 작업 단위가 이미 시작되었습니다.

## RPG 선언

```
C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQBEGIN(HCONN : BEGOP : CMPCOD :
C                                REASON)
```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```
D*..1.....2.....3.....4.....5.....6.....7..
DMQBEGIN      PR          EXTPROC('MQBEGIN')
D* Connection handle
D HCONN              10I 0 VALUE
D* Options that control the action of MQBEGIN
D BEGOP              12A
D* Completion code
D CMPCOD              10I 0
D* Reason code qualifying CMPCOD
D REASON              10I 0
```

MQBUFMH 함수 호출은 버퍼를 메시지 핸들로 변환시키며 MQMHBUF 호출의 역수입니다.

이 호출은 버퍼의 메시지 디스크립터 및 MQRFH2 특성을 가져오고 메시지 핸들을 통해 사용 가능하게 합니다. 메시지 데이터의 MQRFH2 특성이 선택적으로 제거되었습니다. 특성이 제거된 후 메시지 디스크립터의 *Encoding*, *CodedCharSetId* 및 *Format* 필드가 업데이트되고 필요한 경우 버퍼의 콘텐츠를 올바르게 설명합니다.

- [1201 페이지의 『구문』](#)
- [1201 페이지의 『사용법 참고』](#)
- [1201 페이지의 『매개변수』](#)
- [1203 페이지의 『RPG 선언』](#)

## 구문

MQBUFMH (*Hconn*, *Hmsg*, *BufMsgHOpts*, *MsgDesc*, *Buffer*, *BufferLength*, *DataLength*, *CompCode*, *Reason*)

## 사용법 참고

MQBUFMH 호출은 API 엑시트로 인터셉트될 수 없으며 버퍼가 애플리케이션 공간에서 메시지 핸들로 변환됩니다. 호출은 큐 관리자에 도달하지 않습니다.

## 매개변수

MQBUFMH 호출에 다음 매개변수가 있습니다.

### HCONN(10자리 부호 있는 정수) - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *HCONN*의 값은 *Hmsg* 매개변수에 지정된 메시지 핸들을 작성하는 데 사용된 연결 핸들과 일치해야 합니다.

메시지 핸들이 HCUNAS를 사용하여 작성된 경우 올바른 연결은 버퍼를 메시지 핸들로 변환시키는 스레드에 설정되어야 합니다. 올바른 연결이 설정되지 않으면 RC2009가 표시되면서 호출이 실패합니다.

### HMSG(10자리 부호 있는 정수) - 입력

이 핸들은 버퍼가 요구되는 메시지 핸들입니다. 값은 이전 MQCRTMH 호출에서 리턴합니다.

### BMHOPT(MQBMHO) - 입력

MQBMHO 구조를 사용하여 애플리케이션이 메시지 핸들이 버퍼에서 생성되는 방법을 제어하는 옵션을 지정할 수 있습니다.

자세한 내용은 976 페이지의 『IBM i의 MQBMHO(메시지 핸들 옵션에 대한 버퍼)』의 내용을 참조하십시오.

### MSGDSC(MQMD) - 입출력(I/O)

*MSGDSC* 구조는 메시지 디스크립터 특성을 포함하고 버퍼 영역의 콘텐츠를 설명합니다.

호출의 출력에서 특성은 선택적으로 버퍼 영역에서 제거되고 이 경우에 메시지 디스크립터가 올바르게 버퍼 영역을 설명하기 위해 업데이트됩니다.

이 구조의 데이터는 애플리케이션의 문자 세트 및 인코딩에 있어야 합니다.

### BUFLEN(10자리 부호 있는 정수) - 입력

*BUFLEN*은 버퍼 영역의 길이(바이트)입니다.

0바이트의 *BUFLEN*은 올바르며 데이터를 포함하지 않은 버퍼 영역을 표시합니다.

## **BUFFER(1바이트 비트 문자열 x BUFLLEN) - 입출력(I/O)**

*BUFFER*는 메시지 버퍼가 있는 영역을 정의합니다. 대부분의 데이터에서는 4바이트 경계에 버퍼를 맞춰야 합니다.

*BUFFER*에 문자 또는 숫자 데이터가 포함된 경우 **MSGDSC** 매개변수의 *CodedCharSetId* 및 *Encoding* 필드를 데이터에 적합한 값으로 설정하십시오. 그러면 필요한 경우 데이터를 변환할 수 있습니다.

특성이 메시지 버퍼에 있으면 선택적으로 제거됩니다. 나중에 호출에서 리턴 시 메시지 핸들에서 사용할 수 있게 됩니다.

C 프로그래밍 언어에서 매개변수는 pointer-to-void로 선언됩니다. 이는 모든 유형의 데이터 주소를 매개변수로 지정할 수 있다는 의미입니다.

**BUFLLEN** 매개변수가 0인 경우 *BUFFER*은(는) 참조되지 않습니다. 이 경우에 C 또는 System/390 어셈블러로 작성된 프로그램이 전달하는 매개변수 주소는 널(null)일 수 있습니다.

## **DATLEN(10자리 부호 있는 정수) - 출력**

*DATLEN*은 특성을 제거할 수 있는 버퍼의 길이(바이트)입니다.

## **CMPCOD(10자리 부호 있는 정수) - 출력**

### **CCOK**

정상적으로 완료되었습니다.

### **CCFAIL**

호출에 실패했습니다.

## **REASON(10자리 부호 있는 정수) - 출력**

*CMPCOD*를 규정하는 이유 코드입니다.

*CMPCOD*가 CCOK일 경우:

### **RCNONE**

(0, X'000') 보고할 이유가 없습니다.

*CMPCOD*가 CCFAIL일 경우:

### **RC2204**

(2204, X'089C') 어댑터를 사용할 수 없습니다.

### **RC2130**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

### **RC2157**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

### **RC2489**

(2489, X'09B9') 메시지 핸들 옵션 구조에 대한 버퍼가 올바르지 않습니다.

### **RC2004**

(2004, X'07D4') 버퍼 매개변수가 올바르지 않습니다.

### **RC2005**

(2005, X'07D5') 버퍼 길이 매개변수가 올바르지 않습니다.

### **RC2219**

(2219, X'08AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

### **RC2009**

(2009, X'07D9') 큐 관리자에 대한 연결이 유실되었습니다.

### **RC2460**

(2460, X'099C') 메시지 핸들이 올바르지 않습니다.

### **RC2026**

(2026, X'07EA') 메시지 디스크립터가 올바르지 않습니다.

### **RC2499**

(2499, X'09C3') 메시지 핸들이 이미 사용 중입니다.

## RC2046

(2046, X'07FE') 옵션이 올바르지 않거나 일치하지 않습니다.

## RC2334

(2334, X'091E') MQRFH2 구조가 올바르지 않습니다.

## RC2421

(2421, X'0975') 특성을 포함하는 MQRFH2 폴더를 구문 분석할 수 없습니다.

## RC2195

(2195, X'893') 예상치 못한 오류가 발생했습니다.

## RPG 선언

```

C*..1.....2.....3.....4.....5.....6.....7..
C                                CALLP      MQBUFMH(HCONN : HMSG : BMHOPT :
                                MSGDSC : BUFLN : BUFFER :
                                DATLEN : CMPCOD : REASON)

```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```

DMQBUFMH          PR                EXTPROC('MQBUFMH')
D* Connection handle
D HCONN           10I 0
D* Message handle
D HMSG           10I 0
D* Options that control the action of MQBUFMH
D BMHOPT         12A VALUE
D* Message descriptor
D MSGDSC         364A
D* Length in bytes of the Buffer area
D BUFLN          10I 0
D* Area to contain the message buffer
D BUFFER         * VALUE
D* Length of the output buffer
D DATLEN         10I 0
D* Completion code
D CMPCOD         10I 0
D* Reason code qualifying CompCode
D REASON         10I 0

```

## IBM i IBM i의 MQCB(콜백 관리)

MQCB 호출은 지정된 오브젝트 핸들에 대한 콜백을 재등록하고 활성화를 제어하며 호출로 변경됩니다.

콜백은 특정 이벤트가 발생할 때 IBM MQ에서 호출하는 함수 포인터로 또는 동적으로 링크할 수 있는 함수 이름으로 지정되는 코드의 조각입니다.

V7 클라이언트에 MQCB 및 MQCTL을 사용하려면 V7 서버에 연결되어야 하고 채널의 **SHARECNV** 매개변수에 0이 아닌 값이 있어야 합니다.

정의될 수 있는 호출의 유형은 다음과 같습니다.

### 메시지 사용자

지정된 선택 기준을 충족하는 메시지를 오브젝트 핸들에서 사용할 수 있을 때 메시지 사용자 콜백 함수가 호출됩니다.

하나의 콜백 함수만 각 오브젝트 핸들에 등록할 수 있습니다. 단일 큐가 다중 선택 기준을 읽으면 큐는 여러 번 열려야 하고 사용자 함수가 각 핸들에 등록됩니다.

### 이벤트 핸들러

이벤트 핸들러는 전체 콜백 환경에 영향을 주는 조건에 호출됩니다.

이벤트 조건이 발생하면 함수가 호출됩니다(예를 들어, 큐 관리자나 연결 중지 또는 정지).

이 기능은 단일 메시지 사용자에게 특정한 조건에 호출되지 않습니다(예: RC2016). 그러나 콜백 함수가 정상적으로 종료되지 않은 경우 호출됩니다.

- [1204 페이지의 『구문』](#)
- [1204 페이지의 『MQCB의 사용법 참고』](#)
- [1205 페이지의 『MQCB의 매개변수』](#)
- [1211 페이지의 『RPG 선언』](#)

## 구문

MQCB (HCONN, OPERATN, HOBJ, CBDSC, MSGDSC, GMO, CMPCOD, REASON)

## MQCB의 사용법 참고

1. MQCB는 큐에서 사용 가능하고 지정된 기준과 일치하는, 각 메시지에 호출할 조치를 정의하는 데 사용됩니다. 조치가 처리되면 메시지가 큐에서 제거되고 정의된 메시지 이용자에게 전달되거나 메시지 토큰이 제공되며 이는 메시지를 검색하는 데 사용됩니다.
2. MQCB는 MQCTL로 이용을 시작하기 전에 콜백 루틴을 정의할 때 사용할 수 있거나 콜백 루틴으로부터 사용될 수 있습니다.
3. 콜백 루틴의 외부에서 MQCB를 사용하려면 먼저 MQCTL을 사용하여 메시지 이용을 일시중단한 후 이용을 재개해야 합니다.

## 메시지 이용자 콜백 순서

이용자의 라이프사이클 동안 키 포인트에서 콜백을 호출하도록 이용자를 구성할 수 있습니다. 예를 들면, 다음과 같습니다.

- 이용자가 먼저 등록하는 경우
- 연결이 시작된 경우
- 연결이 중지된 경우
- 이용자가 MQCLOSE에 의해 명시적 또는 내재적으로 등록 취소되는 경우

표 742. MQCTL 동사 정의	
동사	의미
MQCTL(START)	CTLSR 조작을 사용하여 MQCTL 호출
MQCTL(STOP)	CTLSP 조작을 사용하여 MQCTL 호출
MQCTL(WAIT)	CTLSW 조작을 사용하여 MQCTL 호출

이용자가 이용자와 연관된 상태를 유지할 수 있게 허용합니다. 호출이 애플리케이션에서 요청될 때 이용자 호출을 위한 규칙은 다음과 같습니다.

### REGISTER

항상 콜백의 첫 번째 호출 유형입니다.

항상 MQCB(CBREG) 호출과 동일한 스레드에 호출됩니다.

### 시작

항상 MQCTL(START) verb와 함께 동기적으로 호출됩니다.

- 모두 START 콜백은 MQCTL(START) verb 리턴 전에 완료됩니다.

CTLTHR이 요청되는 경우 항상 메시지 전달과 동일한 스레드에 있습니다.

예를 들어, 이전 콜백이 MQCTL(START) 동안 MQCTL(STOP)을 발행하면 start를 사용한 호출은 보증되지 않습니다.

### STOP

연결이 재시작될 때까지 이 호출 이후에 전달된 추가 메시지 또는 이벤트가 없습니다.

애플리케이션이 START, 메시지 또는 이벤트에 대해 이전에 호출된 경우 STOP은 보증됩니다.

### DEREGISTER

항상 콜백의 마지막 호출 유형입니다.



애플리케이션이 스레드 기반 초기화 및 정리를 START 및 STOP 호출에서 수행하는지 확인하십시오. REGISTER 및 DEREGISTER 콜백으로 비스레드 기반 초기화 및 정리를 완료할 수 있습니다.

명시된 사항 이외에 스레드의 수명 및 가용성에 대한 가정을 작성하지 마십시오. 예를 들어, DEREGISTER에 대한 마지막 호출을 넘어 남아 있는 스레드에 의존하지 마십시오. 마찬가지로 CTLTHR를 사용하지 않도록 선택했을 때 연결이 시작될 때마다 스레드가 존재한다고 가정하지 마십시오.

애플리케이션에 스레드 특성에 대한 특별한 요구사항이 있는 경우 항상 스레드를 작성한 후 MQCTL(WAIT)을 사용할 수 있습니다. 이 단계에서는 비동기 메시지 전달을 위해 스레드를 IBM MQ에 제공 합니다.

### 메시지 이용자 연결 사용법

일반적으로 하나가 미해결 상태인 동안 애플리케이션이 다른 MQI 호출을 발행할 때 RC2219로 호출에 실패 합니다.

그러나 이전 호출을 완료하기 전에 애플리케이션이 추가 MQI 호출을 발행해야 하는 특수한 경우가 있습니다. 예를 들어, 이용자는 CBRE가 있는 MQCB 호출 중 호출될 수 있습니다.

이러한 인스턴스에서 MQCB 또는 MQCTL verb를 발행하는 애플리케이션의 결과로 애플리케이션이 다시 호출되며 애플리케이션은 추가 MQI 호출을 발행하도록 허용됩니다. 이 인스턴스는 CBCTRC의 CBCCALLT 유형으로 호출될 때 이용자 함수에서 발행할 수 있음을 의미합니다(예: MQOPEN 호출). MQDISC를 제외한 MQI 호출이 허용됩니다.

### MQCB의 매개변수

MQCB 호출에 다음 매개변수가 있습니다.

#### HCONN(10자리 부호 있는 정수) - 입력

콜백 함수 관리 - HCONN 매개변수.

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. HCONN의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

#### OPERATN(10자리의 부호 있는 정수) - 입력

콜백 함수 관리 - OPERATN 매개변수.

지정된 오브젝트 핸들에 대해 정의된 콜백에서 처리 중인 조작입니다. 다음 옵션 중 하나를 지정해야 합니다. 둘 이상의 옵션이 필요한 경우 이 값을 추가하거나(두 번 이상 동일한 상수를 추가하지 않음) 비트 단위의 OR 연산을 사용하여 결합할 수 있습니다(프로그래밍 언어가 비트 연산을 지원하는 경우).

올바르지 않은 조합에 대해 설명되어 있습니다. 그 밖의 모든 조합은 올바릅니다.

#### CBREG

정의된 오브젝트 핸들에 대한 콜백 함수를 정의하십시오. 이 조작은 호출될 함수 및 사용될 선택 기준을 정의합니다.

콜백 함수가 오브젝트 핸들에 이미 정의된 경우 정의는 대체됩니다. 콜백을 대체하는 동안 오류가 감지되는 경우 함수가 등록 취소됩니다.

호출이 이전에 등록 취소된 동일한 콜백 함수에 등록되는 경우 이는 대체 작업으로 처리됩니다. 초기 또는 마지막 호출은 호출되지 않습니다.

CTLSU 또는 CTLRE가 있는 CBREG를 사용할 수 있습니다.

#### CBUNR

오브젝트 핸들의 메시지 이용을 중지하고 콜백에 적합한 핸들을 제거합니다.

연관된 핸들이 처리완료되는 경우 콜백이 자동으로 등록 취소됩니다.

CBUNR이 이용자 내에서 호출되고 콜백에 정의된 중지 호출이 있는 경우 이용자로부터 리턴에 호출됩니다.

이 조작이 등록된 이용자 없이 Hobj에 발행되는 경우 RC2448로 호출을 리턴합니다.

#### CTLSU

오브젝트 핸들에 대한 메시지의 이용 일시중단.

이 조작이 이벤트 핸들러에 적용되는 경우 일시중단되는 동안 이벤트 핸들러는 이벤트를 가져오지 않고 일시중단 상태인 동안에는 재개될 때 조작에 제공하지 않습니다.

일시중단되는 동안 사용자 함수는 제어 유형 콜백을 계속 가져옵니다.

#### CTLRE

오브젝트 핸들에 대한 메시지의 이용 재개.

이 조작이 이벤트 핸들러에 적용되는 경우 일시중단되는 동안 이벤트 핸들러는 이벤트를 가져오지 않고 일시중단 상태인 동안에는 재개될 때 조작에 제공하지 않습니다.

#### CBDSC(MQCB) - 입력

콜백 함수 관리 - CBDSC 매개변수.

이는 등록할 때 사용된 애플리케이션 및 옵션에 의해 등록되고 있는 콜백 함수를 식별하는 구조입니다.

구조에 대한 세부사항은 286 페이지의 『MQCB - 콜백 디스크립터』의 내용을 참조하십시오.

호출 디스크립터는 CBREG 옵션에만 필요합니다. 디스크립터가 요구되지 않는 경우 전달된 매개변수 주소는 널일 수 있습니다.

#### HOBJ(10자리 부호 있는 정수) - 입력

콜백 함수 관리 - HOBJ 매개변수.

이 핸들은 이용될 메시지에서 오브젝트에 설정된 액세스를 나타냅니다. 이는 이전 MQOPEN 또는 MQSUB 호출 (**HOBJ** 매개변수에서) 에서 리턴된 핸들입니다.

**HOBJ**는 이벤트 핸들러 루틴(CBTEH)을 정의할 때 필요하지 않고 HONONE로 지정되어야 합니다.

이 **Hobj**가 MQOPEN 호출에서 리턴되는 경우 큐는 다음 옵션 중 하나 이상으로 열려야 합니다.

- OOINPS
- OOINPX
- OOINPQ
- OOBWR

#### MSGDSC(MQMD) - 입력

콜백 함수 관리 - MSGDSC 매개변수.

이 구조는 필요한 메시지의 속성 및 검색된 메시지의 속성을 나타냅니다.

**MsgDesc** 매개변수는 이용자에게 필요한 메시지의 속성 및 메시지 이용자에게 전달될 MQMD의 버전을 정의합니다.

MQMD의 *MsgId*, *CorrelId*, *GroupId*, *MsgSeqNumber* 및 *Offset*은 **GetMsgOpts** 매개변수에 지정된 옵션에 따라 메시지 선택에 사용됩니다.

**GMCONV** 옵션을 지정하는 경우 *Encoding* 및 *CodedCharSetId*는 메시지 변환에 사용됩니다.

세부사항은 MQMD를 참조하십시오.

필드의 기본값 이외의 값을 요구하는 경우 *MsgDesc*는 **CBREG**에만 사용됩니다. *MsgDesc*는 이벤트 핸들러에 사용되지 않습니다.

디스크립터가 필요하지 않은 경우 전달된 매개변수 주소는 널일 수 있습니다.

다수 이용자가 중첩된 선택자가 있는 동일 큐에 대해 등록되는 경우 각 메시지에 대해 선택된 이용자가 정의되지 않습니다.

#### GMO(MQGM) - 입력

콜백 함수 관리 - GMO 매개변수.

메시지 이용자가 메시지를 가져오는 방법을 제어하는 옵션입니다.

모든 옵션은 MQGET 호출에 사용될 때 다음을 제외하고 1032 페이지의 『IBM i의 MQGM(메시지 가져오기 옵션)』에서 설명된 대로 의미를 가지고 있습니다.

## **GMSSIG**

이 옵션은 허용되지 않습니다.

## **GMBRWF, GMBRWN, GMMBH, GMMBC**

찾아보기 이용자에 전달된 메시지 순서는 해당 옵션의 결합으로 결정됩니다. 중요한 결합은 다음과 같습니다.

### **GMBRWF**

큐의 첫 번째 메시지는 이용자에게 반복해서 전달됩니다. 이는 이용자가 콜백에서 메시지를 파괴적으로 이용할 때 유용합니다. 이 옵션은 주의해서 사용하십시오.

### **GMBRWN**

큐의 종료에 도달할 때까지 현재 커서 위치에서 이용자에게 큐의 각 메시지가 제공됩니다.

### **GMBRWF + GMBRWN**

커서는 큐의 시작으로 재설정됩니다. 커서가 큐의 끝에 도달할 때까지 이용자에게 각 메시지가 제공됩니다.

### **GMBRWF + GMMBH 또는 GMMBC**

큐의 시작부터 이용자에게 큐에서 표시되지 않은 첫 번째 메시지가 제공된 후 이 이용자에 대해 표시됩니다. 이 결합은 이용자가 현재 커서 위치 뒤에 추가된 새 메시지를 수신할 수 있는지 확인합니다.

### **GMBRWN + GMMBH 또는 GMMBC**

커서 위치에서 시작하여 이용자는 큐의 다음 표시되지 않은 메시지를 수신한 후 이 이용자에 대해 표시됩니다. 현재 커서 위치 뒤의 큐에 메시지를 추가할 수 있으므로 이 결합을 주의해서 사용하십시오.

### **GMBRWF + GMBRWN + GMMBH 또는 GMMBC**

이 결합은 허용되지 않으며 사용되는 경우 호출이 RC2046을 리턴합니다.

## **GMNWT, GMWT 및 GMWI**

이러한 옵션은 이용자가 호출되는 방법을 제어합니다.

### **GMNWT**

이용자는 RC2033으로 호출되지 않습니다. 이용자는 메시지 및 이벤트에만 호출됩니다.

### **0 GMWI가 있는 GMWT**

메시지가 없는 경우 RC2033 코드만 이용자에게 전달됨

- 이용자가 시작됨
- 이용자는 마지막 메시지 이유 코드가 없기 때문에 하나 이상이 전달됨

이는 0 대기 간격이 지정될 때 이용자가 사용 중인 루프에서 폴링하는 것을 방지합니다.

### **GMWT 및 양수 GMWI**

사용자는 이유 코드 RC2033이 있는 지정된 대기 간격 뒤에 호출됩니다. 이 호출은 메시지가 이용자에게 전달되었는지 여부와 관계없이 작성됩니다. 이로 인해 사용자가 하트비트 또는 배치 유형 처리를 수행할 수 있습니다.

### **WIULIM의 GMWT 및 GMWI**

이는 RC2033을 리턴하기 전에 무한한 대기를 지정합니다. 이용자는 RC2033으로 호출되지 않습니다.

필드의 기본값 이외의 값을 요구하는 경우 *GMO*는 CBREG에만 사용됩니다. *GMO*는 이벤트 핸들러에 사용되지 않습니다.

옵션이 필요하지 않은 경우 전달된 매개변수 주소는 널일 수 있습니다.

메시지 특성 핸들이 MQGMO 구조에서 제공되는 경우 사본이 사용자 콜백에 전달된 MQGMO 구조에서 제공됩니다. MQCB 호출에서 리턴 시 애플리케이션은 메시지 특성 핸들을 삭제할 수 있습니다.

## **CMPCOD(10자리 부호 있는 정수) - 출력**

콜백 함수 관리 - CMPCOD 매개변수.

완료 코드는 다음 중 하나입니다.

### **CCOK**

정상적으로 완료되었습니다.

**CCWARN**

경고(일부 완료).

**CCFAIL**

호출에 실패했습니다.

**REASON(10자리 부호 있는 정수) - 출력**

콜백 함수 관리 - REASON 매개변수.

다음 이유 코드는 큐 관리자가 **REASON** 매개변수에 대해 리턴할 수 있는 코드입니다.

*CMPCOD*가 CCOK일 경우:

**RCNONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*가 CCFAIL인 경우:

**RC2204**

(2204, X'89C') 어댑터를 사용할 수 없습니다.

**RC2133**

(2133, X'855') 데이터 변환 서비스 모듈을 로드할 수 없습니다.

**RC2130**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

**RC2374**

(2374, X'946') API 엑시트가 실패했습니다.

**RC2183**

(2183, X'887') API 엑시트를 로드할 수 없습니다.

**RC2157**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

**RC2005**

(2005, X'7D5') 버퍼 길이 매개변수가 올바르지 않습니다.

**RC2219**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**RC2487**

(2487, X'9B7') 올바르지 않은 콜백 유형 필드.

**RC2448**

(2448, X'990') 등록된 콜백이 없기 때문에 등록 취소, 일시중단 또는 재개할 수 없습니다.

**RC2486**

(2486, X'9B6') *CallbackFunction* 또는 *CallbackName*은 지정해야 하지만 둘 다 지정해서는 안됩니다.

**RC2483**

(2483, X'9B3') 올바르지 않은 콜백 유형 필드.

**RC2484**

(2484, X'9B4') 올바르지 않은 MQCBD 옵션 필드입니다.

**RC2140**

(2140, X'85C') CICS에서 대기 요청이 거부되었습니다.

**RC2009**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**RC2217**

(2217, X'8A9') 연결할 권한이 부여되지 않았습니다.

**RC2202**

(2202, X'89A') 연결이 정지됩니다.

**RC2203**

(2203, X'89B') 연결이 종료됩니다.

- RC2207**  
(2207, X'89F') 상관 ID 오류입니다.
- RC2010**  
(2010, X'7DA') 데이터 길이 매개변수가 올바르지 않습니다.
- RC2016**  
(2016, X'7E0') 큐에 대해 가져오기가 금지되었습니다.
- RC2351**  
(2351, X'92F') 글로벌 작업 단위 충돌입니다.
- RC2186**  
(2186, X'88A') 메시지 가져오기 옵션 구조가 올바르지 않습니다.
- RC2353**  
(2353, X'931') 글로벌 작업 단위에 대해 사용 중인 핸들입니다.
- RC2018**  
(2018, X'7E2') 연결 핸들이 유효하지 않습니다.
- RC2019**  
(2019, X'7E3') 오브젝트 핸들이 올바르지 않습니다.
- RC2259**  
(2259, X'8D3') 찾아보기 지정에 일관성이 없습니다.
- RC2245**  
(2245, X'8C5') 작업 단위 지정에 일관성이 없습니다.
- RC2246**  
(2246, X'8C6') 커서에 있는 메시지가 검색에 대해 유효하지 않습니다.
- RC2352**  
(2352, X'930') 글로벌 작업 단위가 로컬 작업 단위와 충돌합니다.
- RC2247**  
(2247, X'8C7') 일치 옵션이 올바르지 않습니다.
- RC2485**  
(2485, X'9B4') Incorrect *MaxMsgLength* 필드.
- RC2026**  
(2026, X'7EA') 메시지 디스크립터가 올바르지 않습니다.
- RC2497**  
(2497, X'9C1') 지정된 함수 시작점을 모듈에서 찾을 수 없습니다.
- RC2496**  
(2496, X'9C0') 모듈을 발견했지만 틀린 유형이 있습니다. 32비트 또는 64비트 또는 유효한 동적 링크 라이브러리가 아닙니다.
- RC2495**  
(2495, X'9BF') 모듈을 검색 경로에서 찾을 수 없거나 로드 권한이 부여되지 않았습니다.
- RC2250**  
(2250, X'8CA') 메시지 순서 번호가 올바르지 않습니다.
- RC2331**  
(2331, X'91B') 메시지 토큰 사용이 올바르지 않습니다.
- RC2033**  
(2033, X'7F1') 사용 가능한 메시지가 없습니다.
- RC2034**  
(2034, X'7F2') 찾아보기 커서가 메시지에 위치해 있지 않습니다.
- RC2036**  
(2036, X'7F4') 큐가 읽기 전용으로 열려있지 않습니다.
- RC2037**  
(2037, X'7F5') 큐가 입력을 위해 열려있지 않습니다.

- RC2041**  
(2041, X'7F9') 오브젝트가 열린 후에 정의가 변경되었습니다.
- RC2101**  
(2101, X'835') 오브젝트가 손상되었습니다.
- RC2206**  
(2206, X'89E') API 호출에 대해 올바르지 않은 조작 코드입니다.
- RC2046**  
(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.
- RC2193**  
(2193, X'891') 페이지 세트 데이터 세트에 액세스하는 중에 오류가 발생했습니다.
- RC2052**  
(2052, X'804') 큐가 삭제되었습니다.
- RC2394**  
(2394, X'95A') 큐의 색인 유형이 올바르지 않습니다.
- RC2058**  
(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.
- RC2059**  
(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.
- RC2161**  
(2161, X'871') 큐 관리자가 정지됩니다.
- RC2162**  
(2162, X'872') 큐 관리자가 종료되었습니다.
- RC2102**  
(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.
- RC2069**  
(2069, X'815') 이 핸들의 미해결 신호.
- RC2071**  
(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.
- RC2109**  
(2109, X'83D') 엑시트 프로그램에서 호출이 억제되었습니다.
- RC2024**  
(2024, X'7E8') 현재 작업 단위 내에서 더 이상 메시지를 핸들링할 수 없습니다.
- RC2072**  
(2072, X'818') 동기점 지원을 사용할 수 없습니다.
- RC2195**  
(2195, X'893') 예상치 못한 오류가 발생했습니다.
- RC2354**  
(2354, X'932') 글로벌 작업 단위에서의 등록이 실패했습니다.
- RC2355**  
(2355, X'933') 작업 단위 호출의 혼합은 지원되지 않습니다.
- RC2255**  
(2255, X'8CF') 사용할 큐 관리자에 대해 작업 단위를 사용할 수 없습니다.
- RC2090**  
(2090, X'82A') MQGMO의 대기 간격이 올바르지 않습니다.
- RC2256**  
(2256, X'8D0') 올바르지 않은 버전의 MQGMO가 제공되었습니다.
- RC2257**  
(2257, X'8D1') 올바르지 않은 버전의 MQMD가 제공되었습니다.
- RC2298**  
(2298, X'8FA') 요청된 함수는 현재 환경에서 사용할 수 없습니다.

## RPG 선언

```
C*.1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQCB(HCONN : OPERATN : CBDSC :
                   HOBJ : MSGDSC : GMO :
                   DATLEN : CMPCOD : REASON)
```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```
DMQCB          PR          EXTPROC('MQCB')
D* Connection handle
D HCONN          10I 0 VALUE
D* Operation
D OPERATN        10I 0 VALUE
D* Callback descriptor
D CBDSC          180A
D* Object handle
D HOBJ           10I 0 VALUE
D* Message Descriptor
D MSGDSC         364A
D* Get options
D GMO           112A
D* Completion code
D CMPCOD         10I 0
* Reason code qualifying CompCode
D REASON        10I 0
```

## IBM i IBM i의 MQCLOSE(오브젝트 닫기)

MQCLOSE 호출은 오브젝트에 대한 액세스를 철회하며 MQOPEN 호출의 변환입니다.

- [1211 페이지의 『구문』](#)
- [1211 페이지의 『사용법 참고』](#)
- [1212 페이지의 『매개변수』](#)
- [1216 페이지의 『RPG 선언』](#)

## 구문

MQCLOSE (*HCONN*, *HOBJ*, *OPTS*, *CMPCOD*, *REASON*)

## 사용법 참고

1. 애플리케이션이 MQDISC 호출을 발행하거나 정상적으로 또는 비정상적으로 종료할 때 오브젝트는 애플리케이션에 의해 열리며 여전히 열린 오브젝트는 CONONE 옵션으로 자동으로 닫힙니다.
2. 처리완료되는 오브젝트가 큐인 경우 다음 사항이 적용됩니다.
  - 큐의 조작이 작업 단위의 일부로 수행되는 경우 큐는 동기점이 동기점의 결과에 영향을 주지 않고 발생할 때 처리완료될 수 있습니다.
  - 큐가 OOBROW 옵션으로 열린 경우 찾아보기 커서는 영구 삭제됩니다. 큐가 나중에 OBRW 옵션으로 다시 열리는 경우 새 찾아보기 커서가 작성됩니다(MQOPEN에 설명된 OOBROW 옵션 참조).
  - 메시지가 현재 MQCLOSE 호출 시간에 이 핸들에 대해 잠긴 경우 잠금은 릴리스됩니다([1032 페이지의 『IBM i의 MQGMO\(메시지 가져오기 옵션\)』](#)에 설명된 GMLK 옵션 참조).
3. 처리완료되는 오브젝트가 동적 큐인 경우 다음 사항이 적용됩니다(영구적 또는 임시).
  - 동적 큐의 경우 옵션 CODEL 또는 COPURG는 해당 MQOPEN 호출에 지정된 옵션에 관계없이 지정될 수 있습니다.
  - 동적 큐가 삭제될 때 큐에 대하여 미해결 상태인 GMWT 옵션이 있는 모든 MQGET 호출은 취소되고 이유 코드 RC2052가 리턴됩니다. [1032 페이지의 『IBM i의 MQGMO\(메시지 가져오기 옵션\)』](#)에서 설명된 GMWT 옵션을 참조하십시오.

동적 큐가 삭제된 후 이전에 확보한 *HOBJ* 핸들을 사용하여 큐를 참조하기 위해 시도하는 호출(MQCLOSE 이외)이 이유 코드 RC2052로 실패합니다.

삭제된 큐가 애플리케이션에 의해 액세스될 수 없고 큐가 시스템에서 제거되지 않으며 큐를 참조하는 모든 핸들이 처리완료될 때까지 연관된 자원이 무료로 아니고 큐에 영향을 주는 모든 작업 단위가 커밋되거나 백아웃됩니다.

- 영구적 동적 큐가 삭제될 때 MQCLOSE 호출에서 지정된 *HOBJ* 핸들이 큐를 작성한 MQOPEN 호출로 리턴된 핸들이 아닌 경우, MQOPEN 호출을 유효성 검증하는 데 사용된 사용자 ID에 큐를 삭제할 권한이 부여되었는지의 검사가 수행됩니다. OOALTU 옵션이 MQOPEN 호출에 지정되는 경우 확인된 사용자 ID는 ODAU입니다.

다음의 경우 이 검사는 수행되지 않습니다.

- 지정된 핸들은 큐를 작성한 MQOPEN 호출로 리턴된 핸들입니다.
- 삭제되고 있는 큐는 임시 동적 큐입니다.

- 임시 동적 큐가 처리완료될 때 MQCLOSE 호출에 지정된 *HOBJ* 핸들은 큐를 작성한 MQOPEN 호출로 리턴된 핸들이며 이 큐는 삭제됩니다. 이는 MQCLOSE 호출에 지정된 가까운 옵션과 관계없이 발생합니다. 큐에 메시지가 있는 경우 제거되며 보고 메시지가 생성되지 않습니다.

큐에 영향을 주는 커밋되지 않는 작업 단위가 있는 경우 큐 및 해당 메시지는 여전히 삭제되지만 이로 인해 작업 단위가 실패하지는 않습니다. 그러나 이전에 설명된 것처럼 각 작업 단위가 작업이 커밋되거나 백아웃될 때까지 작업 단위와 연관된 자원은 무료로 아닙니다.

#### 4. 처리완료되는 오브젝트가 분배 목록인 경우 다음 사항이 적용됩니다.

- 분배 목록에 대한 올바른 닫기 옵션만 CONONE입니다. 기타 옵션이 지정되는 경우 이유 코드 RC2046 또는 RC2045로 호출에 실패합니다.
- 분배 목록이 처리완료될 때 개별 완료 코드 및 이유 코드는 목록의 큐에 대해 리턴되지 않으며 호출의 **CMPCOD** 및 **REASON** 매개변수만 진단 목적에 사용 가능합니다.

큐 중 하나를 처리완료하는 중에 실패가 발생하는 경우 큐 관리자는 처리를 계속하고 분배 목록에 남아있는 큐를 처리완료하기 위해 시도합니다. 그런 다음 호출의 **CMPCOD** 및 **REASON** 매개변수는 실패를 설명하는 정보를 리턴하도록 설정됩니다. 따라서 대부분의 큐가 처리완료되더라도 완료 코드는 CCFAIL일 수 있습니다. 오류가 발생한 큐는 식별되지 않습니다.

둘 이상의 큐에 실패가 있는 경우 **CMPCOD** 및 **REASON** 매개변수에 보고된 실패는 정의되지 않습니다.

## 매개변수

MQCLOSE 호출에는 다음 매개변수가 존재합니다.

### HCONN(10자리 부호 있는 정수) - 입력

연결 핸들입니다.

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. HCONN의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

### HOBJ(10자리 부호 있는 정수) - 입출력(I/O)

오브젝트 핸들.

이 핸들은 처리완료되고 있는 오브젝트를 나타냅니다. 오브젝트는 다음 유형 중 하나일 수 있습니다. HOBJ의 값은 이전 MQOPEN 호출로 리턴됩니다.

호출의 정상 완료 시 큐 관리자는 환경에 대한 올바른 핸들이 아닌 값에 이 매개변수를 설정합니다. 이 값은 다음과 같습니다.

### HOUNUH

사용 불가능한 오브젝트 핸들.

### OPTS(10자리 부호 있는 정수) - 입력

MQCLOSE의 조치를 제어하는 옵션.



**OPTS** 매개변수는 오브젝트가 처리완료되는 방법을 제어합니다. 영구적 동적 큐 및 구독만 둘 이상의 방법으로 처리완료될 수 있습니다. 영구적 동적 큐는 보유되거나 삭제될 수 있습니다. 이는 값 **QDPERM**이 있는 **DefinitionType** 속성을 포함하는 큐입니다(1305 페이지의 『큐의 속성』에서 설명된 **DefinitionType** 속성 참조). 닫기 옵션은 이 주제에서 후자 부분의 표에 요약됩니다.

지속 가능 구독은 유지되거나 제거될 수 있으며 **SODUR** 옵션이 있는 **MQSUB** 호출을 사용하여 작성됩니다.

관리 목적지에 대한 핸들을 닫는 중(즉, **SOMAN** 옵션이 사용된 **MQSUB** 호출에서 리턴된 **Hobj** 매개변수) 연관된 구독 또한 제거되었을 때 큐 관리자는 검색되지 않은 발행물도 정리합니다. 이는 **MQSUB** 호출에서 리턴된 **Hsub** 매개변수에 **CORMSB** 옵션을 사용하여 수행됩니다. **CORMSB**가 지속 불가능 구독에 대한 **MQCLOSE**의 기본 작동이라는 점을 유의하십시오.

비관리 목적지에 대한 핸들을 닫을 때 발행물이 전송된 큐를 정리할 책임이 있습니다. 먼저 **CORMSB**를 사용하여 등록을 닫은 후 메시지가 남아 있을 때까지 큐에서 메시지를 처리하는 것이 좋습니다.

다음 중 하나(하나만)는 지정되어야 합니다.

### 동적 큐 닫기 옵션

이러한 옵션은 영구적 동적 큐가 닫히는 방법을 제어합니다.

#### CODEL

큐를 삭제하십시오.

다음 중 하나가 참이면 큐는 삭제됩니다.

- 이는 이전 **MQOPEN** 호출로 작성된 영구적 동적 큐이며 큐에 메시지가 없고 큐에 대해 미결해 상태인 커밋하지 않은 가져오기 또는 넣기 요청이 없습니다(현재 태스크 또는 기타 태스크에 대해).
- **HOBJ**를 리턴한 **MQOPEN** 호출에서 작성한 임시 동적 큐입니다. 이 경우 큐의 모든 메시지는 제거됩니다.

다른 모든 경우에 **Hobj**가 **MQSUB** 호출에 리턴된 경우를 포함하여 이유 코드 **RC2045**로 호출에 실패하고 오브젝트는 삭제되지 않습니다.

#### COPURG

큐의 메시지를 영구 제거하여 큐를 삭제하십시오.

다음 중 하나가 참이면 큐는 삭제됩니다.

- 이는 이전 **MQOPEN** 호출로 작성된 영구적 동적 큐이며 큐에 대해 미결해 상태인 커밋하지 않은 가져오기 또는 넣기 요청이 없습니다(현재 태스크 또는 기타 태스크에 대해).
- **HOBJ**를 리턴한 **MQOPEN** 호출에서 작성한 임시 동적 큐입니다.

다른 모든 경우에 **Hobj**가 **MQSUB** 호출에 리턴된 경우를 포함하여 이유 코드 **RC2045**로 호출에 실패하고 오브젝트는 삭제되지 않습니다.

다음 표는 올바른 닫기 옵션 및 오브젝트가 보유되거나 삭제되는지 여부를 보여줍니다.

표 743. 보유되거나 삭제된 오브젝트로 사용할 올바른 닫기 옵션			
큐의 오브젝트 유형	CONONE	CODEL	COPURG
큐 이외의 오브젝트	보유됨	올바르지 않음	올바르지 않음
사전 정의된 큐	보유됨	올바르지 않음	올바르지 않음
영구적 동적 큐	보유됨	비어 있고 보류 중인 업데이트가 없는 경우 삭제	메시지가 삭제됨, 보류 중인 업데이트가 없는 경우 큐가 삭제됨
임시 동적 큐(큐의 작성자가 발행한 호출)	삭제됨	삭제됨	삭제됨
임시 동적 큐(큐의 작성자가 발행하지 않은 호출)	보유됨	올바르지 않음	올바르지 않음
분배 목록	보유됨	올바르지 않음	올바르지 않음

표 743. 보유되거나 삭제된 오브젝트로 사용할 올바른 닫기 옵션 (계속)			
큐의 오브젝트 유형	CONONE	CODEL	COPURG
관리되는 구독 목적지	보유됨	올바르지 않음	올바르지 않음
분배 목록(구독이 제거됨)	메시지가 삭제됨, 큐가 삭제됨	올바르지 않음	올바르지 않음

### 구독 닫기 옵션

이 옵션은 핸들이 닫힐 때 지속 가능 구독이 제거되었는지 여부 및 애플리케이션에서 읽도록 여전히 대기 중인 발행물이 정리되었는지 여부를 제어합니다. 이러한 옵션은 MQSUB 호출의 **HSUB** 매개변수에서 리턴된 오브젝트 핸들과 함께 사용할 경우에만 유효합니다.

### COKPSB

구독에 대한 핸들은 닫히지만 작성된 구독은 유지됩니다. 발행물은 구독에 지정된 목적지로 계속 전송됩니다. 이 옵션은 옵션 SODUR로 구독이 작성된 경우에만 유효합니다. 구독이 지속 가능한 경우 COKPSB가 기본입니다.

### CORMSB

구독이 제거되고 구독에 대한 핸들이 닫힙니다.

MQSUB 호출의 **Hobj** 매개변수는 **Hsub** 매개변수의 클로저로 무효화되지 않고 남아있는 발행물을 수신하기 위해 MQGET 또는 MQCB에 계속 사용할 수 있습니다. MQSUB 호출의 **Hobj** 매개변수 또한 닫히면 관리 목적지인 경우 검색되지 않은 발행물이 제거됩니다.

구독이 지속 가능하지 않은 경우 CORMSB가 기본값입니다.

이러한 구독 클로저 옵션은 다음 표에 요약됩니다.

지속 가능 구독 핸들을 닫지만 구독을 남겨두려면 다음 구독 클로저 옵션을 사용하십시오.

표 744. 지속 가능한 구독 핸들을 닫고 구독을 그대로 두기 위한 태스크 옵션	
태스크	구독 클로저 옵션
MQOPENed 핸들에서 발행물 유지	COKPSB
MQOPENed 핸들에서 발행물 제거	조치가 허용되지 않음
SOMAN이 있는 핸들에 발행물 보관	COKPSB
SOMAN이 있는 핸들에서 발행물 제거	조치가 허용되지 않음

지속 가능 구독 핸들을 닫아 구독을 해제하거나 지속 불가능한 구독 핸들을 닫아 구독을 해제하려면 다음 클로저 옵션을 사용하십시오.

표 745. 구독을 해제하기 위한 태스크 옵션	
태스크	구독 클로저 옵션
MQOPENed 핸들에서 발행물 유지	CORMSB
MQOPENed 핸들에서 발행물 제거	조치가 허용되지 않음
SOMAN이 있는 핸들에 발행물 보관	CORMSB
SOMAN이 있는 핸들에서 발행물 제거	COPGSB

### 미리 읽기 옵션

다음 옵션은 애플리케이션이 요청했고 아직 애플리케이션에서 사용되기 전에 클라이언트에 전송한 비지속 메시지에 발생하는 사항을 제어합니다. 이 메시지는 MQCLOSE가 완료되기 전에 애플리케이션에서 요청하도록 대기 중인 클라이언트 미리 읽기 버퍼에 저장되고 큐에서 제거되거나 이용됩니다.

**COIMM**

이 오브젝트는 즉시 닫히며 요청된 애플리케이션이 제거되기 전 클라이언트에 전송한 메시지는 애플리케이션에서 사용할 수 없습니다. 이는 기본값입니다.

**COQSC**

오브젝트를 닫기 위한 요청이 작성되었지만 애플리케이션이 요청하기 전에 클라이언트에게 발송된 메시지가 여전히 클라이언트 버퍼에 상주하는 경우 MQCLOSE 호출이 RC2458의 경고 코드로 리턴되고 오브젝트 핸들이 유효한 상태로 남아 있습니다.

그런 다음 애플리케이션은 메시지를 검색하기 위해 오브젝트 핸들을 더 이상 사용할 수 없을 때까지 계속 사용한 후 오브젝트를 다시 닫을 수 있습니다. 애플리케이션 요청에 앞서 더 이상의 메시지가 클라이언트에 전송되지 않으며 이제 미리 읽기가 꺼집니다.

메시지는 COIMM이 사용되는 경우 제거되는 마지막 MQGET 호출 및 다음 MQCLOSE 사이에 도착할 수 있기 때문에 애플리케이션은 클라이언트가 미리 읽기 버퍼에 더 이상 메시지가 없는 지점에 도달하려고 하지 않고 COQSC를 사용하는 것이 좋습니다.

COQSC가 있는 MQCLOSE가 비동기적 콜백 함수에서 발행되는 경우 미리 읽기 메시지의 동일한 작동이 적용됩니다. 경고 코드 RC2458이 리턴되면 콜백 함수가 한번 이상 호출됩니다. 미리 읽은 마지막 남아 있는 메시지가 콜백 함수에 전달되면 CBCFLG 필드가 CBCFBE로 설정됩니다.

**기본 옵션**

이전에 설명된 옵션 중 필요한 옵션이 없는 경우 다음 옵션을 사용할 수 있습니다.

**CONONE**

선택적 닫기 처리가 필요하지 않습니다.

이는 다음에 대해 지정되어야 합니다.

- 큐 이외의 오브젝트
- 사전 정의된 큐
- 임시 동적 큐(HOBJ가 큐를 작성한 MQOPEN 호출로 리턴된 핸들이 아닌 경우에만)
- 분배 목록

이전의 모든 경우에 오브젝트는 유지되고 삭제되지 않습니다.

이 옵션이 임시 동적 큐에 지정된 경우:

- HOBJ를 리턴한 MQOPEN 호출로 작성된 경우 큐는 삭제되며 큐의 메시지도 제거됩니다.
- 다른 모든 경우에 큐(및 큐의 메시지)가 유지됩니다.

이 옵션이 영구적 동적 큐에 지정되는 경우 큐는 유지되고 삭제되지 않습니다.

**CMPCOD(10자리 부호 있는 정수) - 출력**

완료 코드.

다음 중 하나입니다.

**CCOK**

정상적으로 완료되었습니다.

**CCWARN**

경고(일부 완료).

**CCFAIL**

호출에 실패했습니다.

**REASON(10자리 부호 있는 정수) - 출력**

CMPCOD을(를) 규정하는 이유 코드.

CMPCOD가 CCOK일 경우:

**RCNONE**

(0, X'000') 보고할 이유가 없습니다.

If CMPCOD is CCWARN:

**RC2241**

(2241, X'8C1') 메시지 그룹이 완료되지 않았습니다.

**RC2242**

(2242, X'8C2') 논리 메시지가 완료되지 않았습니다.

CMPCOD가 CCFAIL일 경우:

**RC2219**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 다시 입력되었습니다.

**RC2009**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**RC2018**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**RC2019**

(2019, X'7E3') 오브젝트 핸들이 올바르지 않습니다.

**RC2035**

(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

**RC2101**

(2101, X'835') 오브젝트가 손상되었습니다.

**RC2045**

(2045, X'7FD') 옵션이 오브젝트 유형에 대해 유효하지 않습니다.

**RC2046**

(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.

**RC2058**

(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.

**RC2059**

(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.

**RC2162**

(2162, X'872') 큐 관리자가 종료되었습니다.

**RC2055**

(2055, X'807') 큐에 하나 이상의 메시지 또는 커밋되지 않은 Put 또는 Get 요청이 있습니다.

**RC2102**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**RC2063**

(2063, X'80F') 보안 오류가 발생했습니다.

**RC2071**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**RC2195**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

**RPG 선언**

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQCLOSE(HCONN : HOBJ : OPTS :
C                               CMPCOD : REASON)

```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```

D*..1.....2.....3.....4.....5.....6.....7..
DMQCLOSE      PR          EXTPROC('MQCLOSE')
D* Connection handle
D HCONN              10I 0 VALUE
D* Object handle
D HOBJ              10I 0

```

```

D* Options that control the action of MQCLOSE
D OPTS          10I 0 VALUE
D* Completion code
D CMPCOD        10I 0
D* Reason code qualifying CMPCOD
D REASON        10I 0

```

## IBM i IBM i의 MQCMIT(변경사항 커밋)

MQCMIT 호출은 마지막 동기점 이후에 발생한 모든 메시지 가져오기 및 넣기가 큐 관리자에게 백아웃되도록 지시합니다. 작업 단위의 일부로 넣어진 메시지는 삭제되지만 작업 단위의 일부로 검색된 메시지는 큐에 복구됩니다.

- [1217 페이지의 『구문』](#)
- [1217 페이지의 『사용법 참고』](#)
- [1218 페이지의 『매개변수』](#)
- [1219 페이지의 『RPG 선언』](#)

### 구문

MQCMIT (HCONN, COMCOD, REASON)

### 사용법 참고

MQCMIT를 사용할 때 이러한 사용법 참고사항을 고려하십시오.

1. 큐 관리자 자체가 작업 단위를 조정할 때만 이 호출은 사용될 수 있습니다. 이는 변경사항이 IBM MQ 자원에만 영향을 주는 작업의 로컬 단위입니다.
2. 큐 관리자가 작업 단위를 조정하지 않는 환경에서 적절한 백아웃 호출은 MQCMIT 대신에 사용되어야 합니다. 또한 환경은 비정상적으로 종료되는 애플리케이션으로 인한 암시적 백아웃을 지원할 수도 있습니다.
  - IBM i에서 이 호출은 큐 관리자가 조정할 로컬 작업 단위에서 사용될 수 있습니다. 이는 커밋 정의가 작업 레벨에 존재하지 않아야 함을 의미합니다. 즉, **CMTSCOPE(\*JOB)** 매개변수가 있는 STRCMTCTL 명령이 작업에 대해 실행되지 않았어야 합니다.
3. 애플리케이션이 작업 단위에서 커밋되지 않은 변경사항으로 종료되는 경우, 해당 변경사항의 배치는 애플리케이션이 정상으로 또는 비정상적으로 종료되는지 여부에 달려 있습니다. 추가적인 세부사항은 [1231 페이지의 『IBM i의 MQDISC\(큐 관리자 연결 끊기\)』](#)의 사용 시 참고사항을 참조하십시오.
4. 애플리케이션이 논리 메시지의 세그먼트 또는 그룹에서 메시지를 넣거나 가져올 때 큐 관리자는 마지막 성공한 MQPUT 및 MQGET 호출에 대한 메시지 그룹 및 논리 메시지와 관련된 정보를 보유하고 있습니다. 이 정보는 큐 핸들과 연관되어 있으며 다음과 같은 내용을 포함합니다.
  - MQMD에 있는 MDGID, MDSEQ, MDOFF 및 MDMFL 필드의 값입니다.
  - 메시지가 작업 단위의 일부인지 여부.
  - MQPUT 호출의 경우: 메시지가 지속적인지 또는 비지속적인지 여부.

작업의 단위가 커밋될 때 큐 관리자는 그룹과 세그먼트 정보를 보유하고 애플리케이션이 현재 메시지 그룹 또는 논리 메시지에 메시지를 넣거나 가져오기를 계속할 수 있습니다.

작업의 단위가 커밋될 때 그룹과 세그먼트 정보를 보유하는 것은 애플리케이션이 여러 작업 단위 전체에 걸쳐 많은 세그먼트로 구성되어 대량 메시지 그룹 또는 대량 논리 메시지를 펼칠 수 있게 허용합니다. 로컬 큐 관리자에 제한된 큐 스토리지만 있는 경우에는 여러 작업 단위를 사용하면 유용할 수 있습니다. 시스템 장애가 발생한 경우 그러나 애플리케이션은 정확한 위치에 메시지를 넣거나 가져와서 재시작할 수 있기 위한 충분한 정보를 유지보수해야 합니다. 시스템 장애 뒤에, 정확한 포인트에 재시작하는 방법에 대한 자세한 내용에 대해서 [1122 페이지의 『IBM i의 MQPMO\(메시지 넣기 옵션\)』](#)에서 설명된 PMLOGO 옵션 및 [1032 페이지의 『IBM i의 MQGMO\(메시지 가져오기 옵션\)』](#)에서 설명된 GMLOGO 옵션을 참조하십시오.

나머지 사용 시 참고사항은 큐 관리자가 작업 단위를 통합할 때만 적용됩니다.

1. 작업 단위는 연결 핸들과 동일한 범위를 갖습니다. 이는 특별한 작업 단위에 영향을 주는 모든 IBM MQ 호출이 동일한 연결 핸들을 사용하여 수행되어야 함을 의미합니다. 다른 연결 핸들을 사용하여 실행된 호출(예: 다른 애플리케이션이 발행한 호출)은 다음 작업 단위에 영향을 줍니다. 연결 핸들의 범위에 대한 정보는 MQCONN에서 설명된 **HCONN** 매개변수를 참조하십시오.
2. 현재 작업 단위의 일부로서 놓거나 검색된 메시지만 이 호출의 영향을 받습니다.
3. 작업 단위 내에 MQGET, MQPUT 또는 MQPUT1 호출을 실행하지만 커미트 또는 백아웃 호출을 발행하지 않는 장기 실행 애플리케이션으로 인해 다른 애플리케이션에 사용 불가능한 메시지로 큐를 채울 수 있습니다. 이 가능성을 예방하려면 **MaxUncommittedMsgs** 관리자는 속성을 애플리케이션이 큐를 채우지는 않지만 예상된 메시징 애플리케이션이 올바르게 작동할 수 있을 만큼 충분히 높은 이탈을 방지하기에 충분히 낮은 값으로 설정해야 합니다.

## 매개변수

MQCMIT 호출에는 다음 매개변수가 존재합니다.

### **HCONN(10자리 부호 있는 정수) - 입력**

연결 핸들입니다.

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *HCONN*의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

### **COMCOD(10자리의 부호 있는 정수) - 출력**

완료 코드.

다음 중 하나입니다.

#### **CCOK**

정상적으로 완료되었습니다.

#### **CCWARN**

경고(일부 완료).

#### **CCFAIL**

호출에 실패했습니다.

### **REASON(10자리 부호 있는 정수) - 출력**

*COMCOD*을(를) 규정하는 이유 코드.

*COMCOD*가 CCOK일 경우:

#### **RCNONE**

(0, X'000') 보고할 이유가 없습니다.

*COMCOD*가 CCWARN일 경우:

#### **RC2003**

(2003, X'7D3') 작업 단위가 백아웃되었습니다.

#### **RC2124**

(2124, X'84C') 커미트 조작의 결과가 보류 중입니다.

*COMCOD*가 CCFAIL일 경우:

#### **RC2219**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 다시 입력되었습니다.

#### **RC2009**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

#### **RC2018**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

#### **RC2101**

(2101, X'835') 오브젝트가 손상되었습니다.

**RC2123**

(2123, X'84B') 커밋 또는 백아웃 조작의 결과가 혼합되어 있습니다.

**RC2162**

(2162, X'872') 큐 관리자가 종료되었습니다.

**RC2102**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**RC2071**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**RC2195**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

**RPG 선언**

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQCMIT(HCONN : COMCOD : REASON)

```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```

D*..1.....2.....3.....4.....5.....6.....7..
DMQCMIT      PR          EXTPROC('MQCMIT')
D* Connection handle
D HCONN          10I 0 VALUE
D* Completion code
D COMCOD          10I 0
D* Reason code qualifying COMCOD
D REASON          10I 0

```

**IBM i IBM i의 MQCONN(큐 관리자 연결)**

MQCONN 호출은 애플리케이션 프로그램을 큐 관리자에 연결합니다. 애플리케이션이 후속 메시지 큐잉 호출에서 사용된 큐 관리자 연결 핸들을 제공합니다.

- 애플리케이션은 MQCONN 또는 MQCONNX 호출을 사용하여 큐 관리자에 연결하고 MQDISC 호출을 사용하여 큐 관리자에서 연결을 해제해야 합니다.

IBM MQ for Multiplatforms에서 애플리케이션의 각 스레드는 서로 다른 큐 관리자에 연결할 수 있습니다. 다른 시스템에서 프로세스 내의 모든 동시 연결은 동일한 큐 관리자여야 합니다.

- [1219 페이지의 『구문』](#)
- [1219 페이지의 『사용법 참고』](#)
- [1220 페이지의 『매개변수』](#)
- [1222 페이지의 『RPG 선언』](#)

**구문**

MQCONN (QMNAME, HCONN, CMPCOD, REASON)

**사용법 참고**

1. MQCONN 호출을 사용하여 연결이 작성된 큐 관리자를 로컬 큐 관리자라고 합니다.
2. 로컬 큐 관리자가 소유한 큐는 애플리케이션에 로컬 큐로 표시됩니다. 이러한 큐에 대해 메시지를 넣고 메시지를 가져올 수 있습니다.

로컬 큐 관리자가 속한 큐 공유 그룹이 소유한 공유 큐는 애플리케이션에 로컬 큐로 표시됩니다. 이러한 큐에 대해 메시지를 넣고 메시지를 가져올 수 있습니다.

리모트 큐 관리자가 소유한 큐는 리모트 큐로 표시됩니다. 이러한 큐에 대해 메시지를 넣을 수는 있으나 큐로부터 메시지를 가져올 수는 없습니다.

3. 애플리케이션이 실행 중인 동안 큐 관리자가 실패하는 경우 애플리케이션은 후속 IBM MQ 호출에 사용할 새 연결 핸들을 확보하기 위해 다시 MQCONN 호출을 발행해야 합니다. 애플리케이션은 호출이 성공할 때까지 정기적으로 MQCONN 호출을 발행할 수 있습니다.

애플리케이션이 큐 관리자에 연결되었는지 여부를 확신하지 못하는 경우에는 안전하게 MQCONN 호출을 발행하여 연결 핸들을 확보할 수 있습니다. 애플리케이션이 이미 연결된 경우에는 리턴되는 핸들이 이전 MQCONN 호출에 의해 리턴되는 핸들과 동일하나 완료 코드는 CCWARN이며 이유 코드는 RC2002입니다.

4. 애플리케이션이 IBM MQ 호출을 사용하여 완료된 경우 애플리케이션은 큐 관리자에서 연결 해제하기 위해 MQDISC 호출을 사용해야 합니다.
5. IBM i에서 비정상적으로 종료되는 프로그램은 자동으로 큐 관리자에서 연결 해제되지 않습니다. 따라서 애플리케이션은 완료 코드 CCWARN 및 이유 코드 RC2002를 리턴하여 MQCONN 또는 MQCONNX 호출의 가능성을 허용하기 위해 작성되어야 합니다. 이 상황에서 리턴된 연결 핸들은 정상적으로 사용될 수 있습니다.

## 매개변수

MQCONN 호출에는 다음 매개변수가 존재합니다.

### QMNAME(48바이트 문자열) - 입력

큐 관리자 이름.

이는 애플리케이션이 연결할 큐 관리자의 이름입니다. 이름에는 다음 문자가 포함될 수 있습니다.

- 대문자 알파벳(A - Z)
- 소문자 알파벳(a - z)
- 숫자(0 - 9)
- 마침표(.), 슬래시(/), 밑줄(\_), 퍼센트(%)

이름에 선두 문자 또는 임베드된 공백을 사용할 수 없으나 후미 문자 공백은 사용할 수 있습니다. 널 문자는 이름에서 중요한 데이터의 끝을 표시하는 데 사용할 수 있습니다. 널 및 그 다음에 오는 문자는 공백으로 처리됩니다. 다음 제한사항이 표시된 환경에서 적용됩니다.

- IBM i에서 소문자, 슬래시 또는 퍼센트를 포함하는 이름은 명령에 지정할 때 따옴표로 묶어야 합니다. 이러한 따옴표는 QMNAME 매개변수에 지정되지 않아야 합니다.

이름이 완전히 공백으로 구성된 경우 기본 큐 관리자의 이름이 사용됩니다.

QMNAME에 대해 지정된 이름은 연결 가능한 큐 관리자의 이름이어야 합니다.

**큐 공유 그룹** 여러 큐 관리자가 존재하고 큐 공유 그룹을 형성하도록 구성된 시스템에서는 큐 관리자의 이름 대신 QMNAME에 대해 큐 공유 그룹의 이름을 지정할 수 있습니다. 이로 인해 애플리케이션이 큐 공유 그룹에서 사용할 수 있는 모든 큐 관리자에 연결될 수 있습니다. 기본 큐 관리자 대신 큐 공유 그룹에 공백 QMNAME을 연결하도록 시스템을 구성할 수도 있습니다.

QMNAME이 큐 공유 그룹의 이름을 지정하지만 시스템에 해당 이름의 큐 관리자가 있는 경우 전자보다 후자에 우선적으로 연결이 설정됩니다. 해당 연결이 실패하는 경우에만 큐 공유 그룹의 큐 관리자 중 하나에 대한 연결을 시도합니다.

연결이 성공하면 MQCONN 또는 MQCONNX 호출에서 리턴되는 핸들은 연결이 설정된 특정 큐 관리자에 속한 모든 자원(공유 및 비공유 둘 다 포함)에 액세스에 사용할 수 있습니다. 이러한 자원에 대한 액세스에는 일반적인 권한 제어가 적용됩니다.

애플리케이션이 현재 연결을 설정하기 위해 두 개의 MQCONN 또는 MQCONNX 호출을 발행하고 하나 또는 두 호출이 큐 공유 그룹의 이름을 지정하는 경우에 두 번째 호출이 완료 코드 CCWARN 및 이유 코드 RC2002를 리턴합니다. 이는 두 번째 호출이 첫 번째 호출로 동일한 큐 관리자에 연결될 때 발생합니다.

큐 공유 그룹은 z/OS에서만 지원됩니다. 큐 공유 그룹에 대한 연결은 배치, RRS 배치 및 TSO 환경에서만 지원됩니다.

**IBM MQ 클라이언트 애플리케이션:** IBM MQ MQI client 애플리케이션의 경우 하나가 성공할 때까지 지정된 큐 관리자 이름의 각 클라이언트 연결 채널 정의에 대한 연결이 시도됩니다. 그러나 큐 관리자에는 지정된 이



름과 동일한 이름이 있어야 합니다. 모두 공백인 이름이 지정된 경우 연결이 성공할 때까지 모두 공백인 큐 관리자 이름을 가진 각 클라이언트 연결이 시도됩니다. 이 경우 큐 관리자의 실제 이름에 대응하는 검사는 수행되지 않습니다.

**IBM MQ 클라이언트 큐 관리자 그룹:** 지정된 이름이 별표(\*)로 시작하는 경우 연결이 작성된 실제 큐 관리자에 애플리케이션에서 지정된 큐 관리자와 다른 이름이 있을 수 있습니다. 별표 없이 지정된 이름은 연결에 대해 적합한 큐 관리자 그룹을 정의합니다. 구현은 연결을 설정할 수 있는 큐 관리자를 찾을 때까지 알파벳 순으로 차례로 하나씩 시도하여 그룹 중 하나를 선택합니다. 그룹에서 큐 관리자 중 연결에 사용 가능한 큐 관리자가 없는 경우 호출이 실패합니다. 각 큐 관리자는 한 번씩만 시도됩니다. 이름에 대해 별표만 지정된 경우에는 구현에서 정의된 기본 큐 관리자 그룹이 사용됩니다.

큐 관리자 그룹은 MQ 클라이언트 환경에서 실행 중인 애플리케이션에 대해서만 지원됩니다. 클라이언트 외의 애플리케이션이 별표로 시작되는 큐 관리자 이름을 지정하면 호출에 실패합니다. 그룹 내의 각 큐 관리자와 통신하기 위해 동일한 큐 관리자 이름(별표 없이 지정된 이름)을 사용한 여러 개의 클라이언트 연결 채널 정의를 제공하여 그룹이 정의됩니다. 기본 그룹은 하나 이상의 클라이언트 연결 채널 정의를 제공하여 정의되며 각각 공백 큐 관리자 이름을 갖습니다(모두 공백인 이름을 지정하여 클라이언트 애플리케이션의 이름에 대해 하나의 별표를 지정하는 것과 동일한 효과를 갖습니다).

그룹에서 하나의 큐 관리자에 연결된 후 애플리케이션은 애플리케이션이 실제로 연결된 큐 관리자(로컬 큐 관리자)의 이름을 의미하는 메시지 및 오브젝트 디스크립터의 큐 관리자 이름 필드에 일반적인 방법으로 공백을 지정할 수 있습니다. 애플리케이션이 이 이름을 알아야 하는 경우 MQINQ 호출은 **QMgrName** 큐 관리자 속성을 조회하기 위해 발행될 수 있습니다.

연결 이름에 별표를 접두부로 사용하면 애플리케이션이 그룹 내의 특정 큐 관리자에 대한 연결에 의존하지 않음을 나타냅니다. 적절한 애플리케이션은 다음일 수 있습니다.

- 메시지를 넣지만 메시지를 가져오지 않는 애플리케이션입니다.
- 요청 메시지를 넣은 후 임시 동적 큐에서 응답 메시지를 가져오는 애플리케이션입니다.

적합하지 않은 애플리케이션은 특정 큐 관리자의 특정 큐에서 메시지를 가져와야 하는 애플리케이션입니다. 이러한 애플리케이션은 별표를 이름에 접두부로 사용하지 마십시오.

별표를 지정한 경우 이름의 나머지 최대 길이는 47자입니다.

이 필드의 길이는 LNQMNM로 제공됩니다.

### **HCONN(10자리 부호 있는 정수) - 출력**

연결 핸들입니다.

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. 애플리케이션에서 발행하는 모든 후속 메시지 큐잉 호출에 지정되어야 합니다. 이는 MQDISC 호출이 발행되거나 핸들의 범위를 정의하는 처리 단위가 종료될 때 유효하도록 사용이 중단됩니다.

핸들의 범위가 애플리케이션이 애플리케이션이 실행 중인 플랫폼에서 지원되는 병렬 처리입니다. 핸들은 MQCONN 호출이 실행된 단위의 병렬 처리 외부에 유효하지 않습니다.

- IBM i에서 핸들의 범위는 호출을 발행하는 작업입니다.

### **CMPCOD(10자리 부호 있는 정수) - 출력**

완료 코드.

다음 중 하나입니다.

#### **CCOK**

정상적으로 완료되었습니다.

#### **CCWARN**

경고(일부 완료).

#### **CCFAIL**

호출에 실패했습니다.

### **REASON(10자리 부호 있는 정수) - 출력**

CMPCOD을(를) 규정하는 이유 코드.

CMPCOD가 CCOK일 경우:

**RCNONE**

(0, X'000') 보고할 이유가 없습니다.

If CMPCOD is CCWARN:

**RC2002**

(2002, X'7D2') 애플리케이션이 이미 연결되어 있습니다.

CMPCOD가 CCFAIL일 경우:

**RC2219**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 다시 입력되었습니다.

**RC2267**

(2267, X'8DB') 클러스터 워크로드 엑시트를 로드하지 못했습니다.

**RC2009**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**RC2018**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**RC2035**

(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

**RC2137**

(2137, X'859') 오브젝트가 성공적으로 열리지 않았습니다.

**RC2058**

(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.

**RC2059**

(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.

**RC2161**

(2161, X'871') 큐 관리자가 정지됩니다.

**RC2162**

(2162, X'872') 큐 관리자가 종료되었습니다.

**RC2102**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**RC2063**

(2063, X'80F') 보안 오류가 발생했습니다.

**RC2071**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**RC2195**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

## RPG 선언

```
C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQCONN(QMNAME : HCONN : CMPCOD :
C                                REASON)
```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```
D*..1.....2.....3.....4.....5.....6.....7..
DMQCONN      PR          EXTPROC('MQCONN')
D* Name of queue manager
D QMNAME          48A
D* Connection handle
D HCONN          10I 0
D* Completion code
D CMPCOD          10I 0
```

## IBM i IBM i의 MQCONNX(큐 관리자 연결(확장))

MQCONNX 호출은 애플리케이션 프로그램을 큐 관리자에 연결합니다. 애플리케이션이 후속 IBM MQ 호출에서 사용된 큐 관리자 연결 핸들을 제공합니다.

MQCONNX 호출은 MQCONN 호출과 유사합니다. 단, MQCONNX는 호출이 작동하는 방법을 제어하기 위한 옵션을 지정하도록 허용합니다.

IBM MQ for Multiplatforms에서 애플리케이션의 각 스레드는 서로 다른 큐 관리자에 연결할 수 있습니다. 다른 시스템에서 프로세스 내의 모든 동시 연결은 동일한 큐 관리자여야 합니다.

- [1223 페이지의 『구문』](#)
- [1223 페이지의 『매개변수』](#)
- [1224 페이지의 『RPG 선언』](#)

### 구문

MQCONNX (QMNAME, CNOPT, HCONN, CMPCOD, REASON)

### 매개변수

MQCONNX 호출에 다음 매개변수가 있습니다.

#### QMNAME(48바이트 문자열) - 입력

큐 관리자 이름.

자세한 내용은 [1219 페이지의 『IBM i의 MQCONN\(큐 관리자 연결\)』](#)에서 설명한 **QMNAME** 매개변수를 참조하십시오.

#### CNOPT(MQCNO) - 입출력(I/O)

MQCONNX의 조치를 제어하는 옵션입니다.

자세한 내용은 [1005 페이지의 『IBM i의 MQCNO\(연결 옵션\)』](#)의 내용을 참조하십시오.

#### HCONN(10자리 부호 있는 정수) - 출력

연결 핸들입니다.

자세한 내용은 [1219 페이지의 『IBM i의 MQCONN\(큐 관리자 연결\)』](#)에서 설명한 **HCONN** 매개변수를 참조하십시오.

#### CMPCOD(10자리 부호 있는 정수) - 출력

완료 코드.

자세한 내용은 [1219 페이지의 『IBM i의 MQCONN\(큐 관리자 연결\)』](#)에서 설명한 **CMPCOD** 매개변수를 참조하십시오.

#### REASON(10자리 부호 있는 정수) - 출력

CMPCOD을(를) 규정하는 이유 코드.

가능한 이유 코드의 세부사항은 [1219 페이지의 『IBM i의 MQCONN\(큐 관리자 연결\)』](#)에서 설명한 **REASON** 매개변수를 참조하십시오.

다음 추가 이유 코드는 MQCONNX 호출에 의해 리턴될 수 있습니다.

CMPCOD가 CCFAIL일 경우:

#### RC2278

(2278, X'8E6') 클라이언트 연결 필드가 유효하지 않습니다.

## RC2139

(2139, X'85B') 연결 옵션 구조가 올바르지 않습니다.

## RC2046

(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.

## RPG 선언

```

C*..1.....2.....3.....4.....5.....6.....7..
C                                CALLP      MQCONN(QMNAME : HCONN : CMPCOD :
C                                REASON)

```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```

D*..1.....2.....3.....4.....5.....6.....7..
DMQCONN      PR                EXTPROC('MQCONN')
D* Name of queue manager
D QMNAME                48A
D* Options that control the action of MQCONNX
D HCONN                224A
D* Connection handle
D HCONN                10I 0
D* Completion code
D CMPCOD                10I 0
D* Reason code qualifying CMPCOD
D REASON                10I 0

```

## IBM i IBM i의 MQCRTMH(메시지 핸들 작성)

MQCRTMH 호출은 메시지 핸들을 리턴합니다.

애플리케이션은 후속 큐잉 호출에서 사용할 수 있습니다.

- [MQSETMP](#) 호출을 사용하여 메시지 핸들의 특성을 설정하십시오.
- [MQINQMP](#) 호출을 사용하여 메시지 핸들의 특성 값을 조회하십시오.
- [MQDLTMP](#) 호출을 사용하여 메시지 핸들의 특성을 삭제하십시오.

MQPUT 및 MQPUT1 호출에서 메시지 핸들을 사용하여 메시지 핸들의 특성을 넣어 메시지의 특성과 연관시킬 수 있습니다. 이와 비슷하게 MQGET 호출에서 메시지 핸들을 지정하여 MQGET 호출이 완료될 때 메시지를 사용하여 검색할 메시지의 특성을 액세스할 수 있습니다.

[MQDLTMH](#)를 사용하여 메시지 핸들을 삭제하십시오.

- [1224 페이지의 『구문』](#)
- [1224 페이지의 『매개변수』](#)
- [1226 페이지의 『RPG 선언』](#)

## 구문

MQCRTMH (*Hconn, CrtMsgHOpts, Hmsg, CompCode, Reason*)

## 매개변수

MQCRTMH 호출에는 다음 매개변수가 존재합니다.

### HCONN(10자리 부호 있는 정수) - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *HCONN*의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다. 큐 관리자에 대한 연결이 유효하지 않고 IBM MQ 호출도 메시지 핸들에 작동하지 않는 경우 [MQDLTMH](#)가 내재적으로 호출되어 메시지를 삭제합니다.

또는 다음 값을 지정할 수 있습니다.

## HCUNAS

연결 핸들이 특정 큐 관리자에 대한 연결을 표시하지 않습니다.

이 값이 사용될 때 메시지 핸들은 할당된 스토리지를 릴리스하기 위해 MQDLTMH에 명시적 호출로 삭제되어야 합니다. IBM MQ는 내재적으로 메시지 핸들을 삭제하지 않습니다.

메시지 핸들을 작성하는 스레드에서 큐 관리자에 대해 최소 하나 이상의 유효한 연결이 구축되어 있어야 합니다. 그렇지 않으면 호출이 RC2018 오류와 함께 실패합니다.

## CRTOPT(MQCMHO) - 입력

MQCRTMH의 조치를 제어하는 옵션. 세부사항은 MQCMHO를 참조하십시오.

## HMSG(10자리 부호 있는 정수) - 출력

출력에서 메시지 핸들의 특성을 설정, 조회 및 삭제하는 데 사용할 수 있는 메시지 핸들이 리턴됩니다. 초기에는 메시지에 특성이 없습니다.

메시지 핸들에는 연관된 메시지 디스크립터가 있습니다. 처음에는 이 메시지 디스크립터에 기본값이 포함됩니다. MQSETMP 및 MQINQMP 호출을 사용하여 연관된 메시지 디스크립터 필드의 값을 설정하고 조회할 수 있습니다. MQDLTMP 호출은 메시지 디스크립터의 필드를 기본값으로 다시 재설정합니다.

*HCONN* 매개변수가 값 HCUNAS로 지정되는 경우 리턴된 메시지 핸들은 단위의 처리 내에 연결로 MQGET, MQPUT 또는 MQPUT1 호출에 사용될 수 있지만 동시에 하나의 IBM MQ 호출에 의해서만 사용될 수 있습니다. 두 번째 IBM MQ 호출이 동일한 메시지 핸들을 사용하려고 시도할 때 핸들이 사용 중인 경우 두 번째 IBM MQ 호출은 이유 코드 RC2499로 실패합니다.

*HCONN* 매개변수가 HCUNAS가 아닌 경우 리턴된 메시지 핸들은 특정 연결에서만 사용될 수 있습니다.

동일한 *HCONN* 매개변수 값은 이 메시지 핸들이 사용되는 이후의 MQI 호출에서 사용되어야 합니다.

- MQDLTMH
- MQSETMP
- MQINQMP
- MQDLTMP
- MQMHBUF
- MQBUFMH

리턴된 메시지 핸들은 메시지 핸들에 대해 MQDLTMH 호출이 발행되거나 핸들의 범위를 정의하는 처리 단위가 종료될 때 유효하도록 사용이 중단됩니다. 메시지 핸들이 작성되고 큐 관리자에 대한 연결의 유효성이 중지될 때(예: MQDBC가 호출될 때) MQDLTMH가 내재적으로 호출됩니다.

## CMPCOD(10자리 부호 있는 정수) - 출력

완료 코드는 다음 중 하나입니다.

### CCOK

정상적으로 완료되었습니다.

### CCFAIL

호출에 실패했습니다.

## REASON(10자리 부호 있는 정수) - 출력

*CMPCOD*를 규정하는 이유 코드입니다.

*CMPCOD*가 CCOK일 경우:

### RCNONE

(0, X'000') 보고할 이유가 없습니다.

*CMPCOD*가 CCFAIL일 경우:

### RC2204

(2204, X'089C') 어댑터를 사용할 수 없습니다.

### RC2130

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

**RC2157**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

**RC2219**

(2219, X'08AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**RC2461**

(2461, X'099D') 올바르지 않은 메시지 핸들 옵션 구조가 작성되었습니다.

**RC2273**

(2273, X'7D9') 큐 관리자에 대한 연결이 손실되었습니다.

**RC2017**

(2017, X'07E1') 사용 가능한 핸들이 없습니다.

**RC2018**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**RC2460**

(2460, X'099C') 메시지 핸들 포인터가 올바르지 않습니다.

**RC2046**

(2046, X'07FE') 옵션이 올바르지 않거나 일치하지 않습니다.

**RC2071**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**RC2195**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

자세한 내용은 [1359 페이지](#)의 『IBM i(ILE RPG)의 리턴 코드』의 내용을 참조하십시오.

**RPG 선언**

```

C*.1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQCRTMH(HCONN : CRTOPT : HMSG :
                          CMPCOD : REASON)

```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```

DMQCRTMH      PR                EXTPROC('MQCRTMH')
D* Connection handle
D HCONN              10I 0 VALUE
D* Options that control the action of MQCRTMH
D CRTOPT              12A
D* Message handle
D HMSG                20I 0
D* Completion code
D CMPCOD              10I 0
D* Reason code qualifying CompCode
D REASON              10I 0

```

**IBM i IBM i의 MQCTL(콜백 제어)**

MQCTL 호출은 연결을 위해 열린 오브젝트 핸들의 제어 조치를 수행합니다.

- [1226 페이지](#)의 『구문』
- [1227 페이지](#)의 『사용법 참고』
- [1227 페이지](#)의 『매개변수』
- [1231 페이지](#)의 『RPG 선언』

**구문**

MQCTL (*Hconn, Operation, ControlOpts, CompCode, Reason*)

## 사용법 참고

1. 콜백 루틴은 호출하는 모든 서비스의 응답을 확인하고 루틴이 해결할 수 없는 조건을 감지하는 경우 콜백 루틴에 반복된 호출을 방지하기 위해 MQCB(CBREG) 명령을 발행해야만 합니다.

## 매개변수

MQCTL 호출에 다음 매개변수가 있습니다.

### HCONN(10자리 부호 있는 정수) - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. HCONN의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

### OPERATN(10자리의 부호 있는 정수) - 입력

지정된 오브젝트 핸들에 대해 정의된 콜백에서 처리 중인 조작입니다. 다음 옵션 중 하나 또는 하나만 지정해야 합니다.

#### CTLSR

지정된 연결 핸들의 정의된 모든 메시지 사용자 함수에 대해 메시지 이용을 시작합니다.

콜백은 시스템에서 시작된 스레드에서 실행하며 모든 애플리케이션 스레드와 다릅니다.

이 작동은 시스템에 연결 핸들을 제공한 제어를 제공합니다. 사용자 스레드 이외의 스레드에서 실행할 수 있는 유일한 MQI 호출은 다음과 같습니다.

- 조작 CTLSR를 사용하는 QCTL
- 조작 CTLSU를 사용하는 MQCTL
- MQDISC - 이는 HConn 연결 해제 전에 조작 CTLSR로 MQCTL을 수행합니다.

연결 핸들이 시작되는 동안 IBM MQ API 호출이 발행되는 경우 RC2500이 리턴되고 호출은 메시지 사용자 함수에서 발생하지 않습니다.

연결에 실패하는 경우 이는 가능한 한 빨리 대화를 중지합니다. 따라서 잠시 리턴 코드 RC2500을 수신하기 위해 메인 스레드에 실행되고 연결이 중지된 상태로 되돌아갈 때 리턴 코드 RC2009이 뒤에 오는 IBM MQ API 호출에 대해 가능합니다.

이는 사용자 함수에 발행될 수 있습니다. 콜백 루틴과 같은 연결의 경우 유일한 목적은 이전에 발행된 CTLSR 작동을 취소하는 것입니다.

애플리케이션이 스레드되지 않은 IBM MQ 라이브러리에 바인딩되면 이 옵션은 지원되지 않습니다.

#### CTLSW

지정된 연결 핸들의 정의된 모든 메시지 사용자 함수에 대해 메시지 이용을 시작합니다.

동일한 스레드 및 제어에 실행하는 메시지 이용자는 MQCTL의 호출자로 리턴되지 않습니다.

- MQCTL CTLSR 또는 CTLSU 조작을 사용하여 릴리스됩니다. 또는
- 모든 사용자 루틴은 등록 취소되거나 일시중단되었습니다.

모든 사용자가 등록 취소되거나 일시중단되는 경우 내재적 CTLSR 조작이 발행됩니다.

이 옵션은 또한 현재 연결 핸들 또는 기타 연결 핸들에 대해 콜백 루틴에서 사용할 수 없습니다. 호출이 시도되면 RC2012로 리턴합니다.

CTLSW 조작 중 등록되지 않은 경우 일시중단되지 않은 이용자는 RC2446의 이유 코드로 호출에 실패합니다.

CTLSW 조작 중 연결이 일시중단되는 경우 MQCTL 호출은 RC2521의 경고 이유 코드를 리턴합니다. 연결은 '시작됨' 상태로 남아 있습니다.

애플리케이션은 CTLSR 또는 CTLRE를 발행하도록 선택할 수 있습니다. 이 인스턴스에서 CTLRE 조작은 차단됩니다.

이 옵션은 단일 스레드 클라이언트에서 지원되지 않습니다.

## **CTLSP**

메시지의 사용을 중지하고 모든 이용자가 이 옵션이 완료하기 전에 해당 작동을 완료하도록 대기하십시오. 이 조작은 연결 핸들을 릴리스합니다.

콜백 루틴 내에서 발행되는 경우 이 옵션은 루틴 종료까지 적용되지 않습니다. 메시지에 대한 콜백 루틴이 이미 완료된 후와 콜백 루틴이 작성된 호출을 중지한 후(요청된 경우) 호출된 메시지 이용자 루틴이 없습니다.

콜백 루틴이 외부에서 발행되는 경우 콜백이 작성된 호출을 중지한 후(요청된 경우) 이미 읽기가 완료된 메시지의 이용자 루틴까지 발행자에게 리턴하지 않습니다. 그러나 콜백 자체는 등록된 상태로 남아 있습니다.

이 함수는 메시지 미리 읽기에 영향을 미치지 않습니다. 이용자가 전달할 수 있는 추가 메시지가 있는지 여부를 판별하기 위해 콜백 함수 내에서 이용자가 MQCLOSE(COQSC)를 실행하는지 확인하십시오.

## **CTLSU**

메시지 이용 일시정지. 이 조작은 연결 핸들을 릴리스합니다.

이는 애플리케이션의 메시지 미리 읽기에 영향을 주지 않습니다. 장기간 메시지 이용을 중지하는 경우 이용을 계속할 때 큐 닫기 및 다시 열기를 고려하십시오.

콜백 루틴 내에서 발행되는 경우 이는 루틴 종료까지 적용되지 않습니다. 더 이상 메시지 이용자 루틴은 현재 루틴 엑시트 이후에 호출되지 않습니다.

호출이 외부에 발행되면 현재 이용자 루틴이 완료되어 더 이상 호출되지 않을 때까지 제어는 호출자로 돌아가지 않습니다.

## **CTLRE**

메시지 이용 일시정지.

이 옵션은 일반적으로 주요 애플리케이션 스레드에서 발행되지만 동일한 루틴에서 발행된 이전 일시중단 요청을 취소시키기 위해서도 콜백루틴으로부터 사용될 수 있습니다.

CTLRE가 CTLSW를 재개하는 데 사용되면 조작이 차단됩니다.

## **PCTLOP(MQCTLO) - 입력**

MQCTL의 조치를 제어하는 옵션

구조에 대한 세부사항은 MQCTLO의 내용을 참조하십시오.

## **CMPCOD(10자리 부호 있는 정수) - 출력**

완료 코드는 다음 중 하나입니다.

### **CCOK**

정상적으로 완료되었습니다.

### **CCWARN**

경고(일부 완료).

### **CCFAIL**

호출에 실패했습니다.

## **REASON(10자리 부호 있는 정수) - 출력**

다음 이유 코드는 큐 관리자가 **Reason** 매개변수에 대해 리턴할 수 있는 코드입니다.

CMPCOD가 CCOK일 경우:

### **RCNONE**

(0, X'000') 보고할 이유가 없습니다.

CMPCOD가 CCFAIL일 경우:

### **RC2133**

(2133, X'855') 데이터 변환 서비스 모듈을 로드할 수 없습니다.

### **RC2204**

(2204, X'89C') 어댑터를 사용할 수 없습니다.



- RC2130**  
(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.
- RC2374**  
(2374, X'946') API 엑시트가 실패했습니다.
- RC2183**  
(2183, X'887') API 엑시트를 로드할 수 없습니다.
- RC2157**  
(2157, X'86D') 1차 및 홈 ASID가 다릅니다.
- RC2005**  
(2005, X'7D5') 버퍼 길이 매개변수가 올바르지 않습니다.
- RC2487**  
(2487, X'9B7') 콜백루틴을 호출할 수 없음
- RC2448**  
(2448, X'990') 등록된 콜백이 없기 때문에 등록 취소, 일시중단 또는 재개할 수 없습니다.
- RC2486**  
(2486, X'9B6') Either, CallbackFunction 및 CallbackName은 CBREG 호출에 지정되었거나 CallbackFunction 또는 CallbackName 중 하나가 지정되었지만 현재 등록된 콜백 함수와 일치하지 않습니다.
- RC2483**  
(2483, X'9B3') 올바르지 않은 콜백 유형 필드.
- RC2219**  
(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.
- RC2444**  
(2444, X'98C') 옵션 차단이 올바르지 않습니다.
- RC2484**  
(2484, X'9B4') 올바르지 않은 MQCBD 옵션 필드입니다.
- RC2140**  
(2140, X'85C') CICS에서 대기 요청이 거부되었습니다.
- RC2009**  
(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.
- RC2217**  
(2217, X'8A9') 연결할 권한이 부여되지 않았습니다.
- RC2202**  
(2202, X'89A') 연결이 정지됩니다.
- RC2203**  
(2203, X'89B') 연결이 종료됩니다.
- RC2207**  
(2207, X'89F') 상관 ID 오류입니다.
- RC2016**  
(2016, X'7E0') 큐에 대해 가져오기가 금지되었습니다.
- RC2351**  
(2351, X'92F') 글로벌 작업 단위 충돌입니다.
- RC2186**  
(2186, X'88A') 메시지 가져오기 옵션 구조가 올바르지 않습니다.
- RC2353**  
(2353, X'931') 글로벌 작업 단위에 대해 사용 중인 핸들입니다.
- RC2018**  
(2018, X'7E2') 연결 핸들이 유효하지 않습니다.
- RC2019**  
(2019, X'7E3') 오브젝트 핸들이 올바르지 않습니다.

- RC2259**  
(2259, X'8D3') 찾아보기 지정에 일관성이 없습니다.
- RC2245**  
(2245, X'8C5') 작업 단위 지정에 일관성이 없습니다.
- RC2246**  
(2246, X'8C6') 커서에 있는 메시지가 검색에 대해 유효하지 않습니다.
- RC2352**  
(2352, X'930') 글로벌 작업 단위가 로컬 작업 단위와 충돌합니다.
- RC2247**  
(2247, X'8C7') 일치 옵션이 올바르지 않습니다.
- RC2485**  
(2485, X'9B5') 올바르지 않은 MaxMsgLength 필드
- RC2026**  
(2026, X'7EA') 메시지 디스크립터가 올바르지 않습니다.
- RC2497**  
(2497, X'9C1') 지정된 함수 시작점은 모듈에서 발견할 수 없습니다.
- RC2496**  
(2496, X'9C0') 모듈을 찾았지만 틀린 유형(32비트 또는 64비트)이거나 유효한 dll이 아닙니다.
- RC2495**  
(2495, X'9BF') 모듈을 검색 경로에서 찾을 수 없거나 로드 권한이 부여되지 않았습니다.
- RC2206**  
(2206, X'89E') 메시지 ID 오류입니다.
- RC2250**  
(2250, X'8CA') 메시지 순서 번호가 올바르지 않습니다.
- RC2331**  
(2331, X'91B') 메시지 토큰 사용이 올바르지 않습니다.
- RC2036**  
(2036, X'7F4') 큐가 읽기 전용으로 열려있지 않습니다.
- RC2037**  
(2037, X'7F5') 큐가 입력을 위해 열려있지 않습니다.
- RC2041**  
(2041, X'7F9') 오브젝트가 열린 후에 정의가 변경되었습니다.
- RC2101**  
(2101, X'835') 오브젝트가 손상되었습니다.
- RC2488**  
(2488, X'9B8') API 호출에 대해 올바르지 않은 조작 코드입니다.
- RC2046**  
(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.
- RC2193**  
(2193, X'891') 페이지 세트 데이터 세트에 액세스하는 중에 오류가 발생했습니다.
- RC2052**  
(2052, X'804') 큐가 삭제되었습니다.
- RC2394**  
(2394, X'95A') 큐의 색인 유형이 올바르지 않습니다.
- RC2058**  
(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.
- RC2059**  
(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.
- RC2161**  
(2161, X'871') 큐 관리자가 정지됩니다.

**RC2162**

(2162, X'872') 큐 관리자가 종료되었습니다.

**RC2102**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**RC2069**

(2069, X'815') 이 핸들의 미해결 신호.

**RC2071**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**RC2109**

(2109, X'83D') 엑시트 프로그램에서 호출이 억제되었습니다.

**RC2072**

(2072, X'818') 동기점 지원을 사용할 수 없습니다.

**RC2195**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

**RC2354**

(2354, X'932') 글로벌 작업 단위에서의 등록이 실패했습니다.

**RC2355**

(2355, X'933') 작업 단위 호출의 혼합은 지원되지 않습니다.

**RC2255**

(2255, X'8CF') 사용할 큐 관리자에 대해 작업 단위를 사용할 수 없습니다.

**RC2090**

(2090, X'82A') MQGMO의 대기 간격이 올바르지 않습니다.

**RC2256**

(2256, X'8D0') 올바르지 않은 버전의 MQGMO가 제공되었습니다.

**RC2257**

(2257, X'8D1') 올바르지 않은 버전의 MQMD가 제공되었습니다.

**RC2298**

(2298, X'8FA') 요청된 함수는 현재 환경에서 사용할 수 없습니다.

**RPG 선언**

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQCTL(HCONN : OPERATN : PCTLOP :
                          CMPCOD : REASON)

```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```

DMQCTL          PR          EXTPROC('MQCTL')
D* Connection handle
D HCONN          10I 0 VALUE
D* Operation
D OPERATN        10I 0 VALUE
D* Control options
D PCTLOP          32A
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CompCode
D REASON          10I 0

```

**IBM i IBM i의 MQDISC(큐 관리자 연결 끊기)**

MQDISC 호출은 큐 관리자 및 애플리케이션 프로그램 사이의 연결을 중단하고 MQCONN 또는 MQCONNX 호출의 역수입니다.

- [1232 페이지의 『구문』](#)
- [1232 페이지의 『사용법 참고』](#)
- [1232 페이지의 『매개변수』](#)
- [1233 페이지의 『RPG 선언』](#)

## 구문

MQDISC (*HCONN*, *CMPCOD*, *REASON*)

## 사용법 참고

1. 애플리케이션에 아직 오브젝트가 열려 있을 때 MQDISC 호출이 발행되는 경우 해당 오브젝트는 닫기 옵션이 CONONE으로 설정된 채 큐 관리자에 의해 닫힙니다.
2. 애플리케이션이 작업 단위에서 커밋되지 않은 변경사항으로 종료되는 경우 해당 변경사항의 배치는 애플리케이션이 종료되는 방법에 따라 다릅니다.
  - a. 애플리케이션이 종료되기 전에 MQDISC 호출을 발행하는 경우:
    - 큐 관리자 통합 작업 단위의 경우 큐 관리자는 애플리케이션 대신 MQCMIT 호출을 발행합니다. 가능한 경우 작업 단위가 커밋되고 가능하지 않은 경우 백아웃됩니다.
    - 외부적으로 통합된 작업 단위의 경우 작업 단위의 상태에 변경사항이 없지만 작업 단위는 작업 단위 코디네이터가 응답할 때 커밋되어야 함을 표시합니다.
  - b. 애플리케이션이 MQDISC 호출을 발행하지 않고 정상적으로 종료되는 경우 작업 단위는 백아웃됩니다.
  - c. 애플리케이션이 MQDISC 호출을 발행하지 않고 비정상적으로 종료하는 경우 작업 단위가 백아웃됩니다.

## 매개변수

MQDISC 호출에 다음 매개변수가 있습니다.

### **HCONN(10자리 부호 있는 정수) - 입출력(I/O)**

연결 핸들입니다.

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *HCONN*의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

호출의 정상 완료 시 큐 관리자는 환경에 대한 올바른 핸들이 아닌 값에 *HCONN*을 설정합니다. 이 값은 다음과 같습니다.

#### **HCUNUH**

사용 불가능한 연결 핸들입니다.

### **CMPCOD(10자리 부호 있는 정수) - 출력**

완료 코드.

다음 중 하나입니다.

#### **CCOK**

정상적으로 완료되었습니다.

#### **CCWARN**

경고(일부 완료).

#### **CCFAIL**

호출에 실패했습니다.

### **REASON(10자리 부호 있는 정수) - 출력**

*CMPCOD*을(를) 규정하는 이유 코드.

*CMPCOD*가 CCOK일 경우:

**RCNONE**

(0, X'000') 보고할 이유가 없습니다.

CMPCOD가 CCFAIL일 경우:

**RC2219**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 다시 입력되었습니다.

**RC2009**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**RC2018**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**RC2058**

(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.

**RC2059**

(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.

**RC2162**

(2162, X'872') 큐 관리자가 종료되었습니다.

**RC2102**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**RC2071**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**RC2195**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

**RPG 선언**

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQDISC(HCONN : CMPCOD : REASON)

```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```

D*..1.....2.....3.....4.....5.....6.....7..
DMQDISC          PR          EXTPROC('MQDISC')
D* Connection handle
D HCONN          10I 0
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CMPCOD
D REASON          10I 0

```

**IBM i IBM i의 MQDLTMH(메시지 핸들 삭제)**

MQDLTMH 호출은 메시지 핸들을 삭제하며 MQCRTMH 호출의 역수입니다.

- [1233 페이지의 『구문』](#)
- [1233 페이지의 『사용법 참고』](#)
- [1235 페이지의 『매개변수』](#)
- [1236 페이지의 『RPG 선언』](#)

**구문**

MQDLTMH ((Hconn, Hmsg, DltMsgH0pts, CompCode, Reason)

**사용법 참고**

1. 큐 관리자가 작업 단위를 조정하는 경우에만 이 호출을 사용할 수 있습니다. 다음과 같습니다.

- 변경사항이 IBM MQ 자원에만 영향을 미치는 경우, 로컬 작업 단위.
  - 변경사항이 다른 자원 관리자에 속한 자원과 IBM MQ 자원에 영향을 미칠 수 있는 경우, 글로벌 작업 단위.
- 로컬 및 글로벌 작업 단위에 대한 세부사항은 1198 페이지의 『IBM i의 MQBEGIN(작업 단위 시작)』의 내용을 참조하십시오.

2. 큐 관리자가 작업 단위를 조정하지 않는 환경에서는 MQBACK 대신 적절한 백아웃을 사용하십시오. 이 환경에서는 애플리케이션의 비정상적인 종료로 인해 발생한 암시적 백아웃을 지원할 수도 있습니다.
  - z/OS에서는 다음 호출을 사용하십시오.
    - 배치 프로그램 (IMS 배치 DL/I 프로그램 포함) 은 작업 단위가 IBM MQ 자원에만 영향을 주는 경우 MQBACK 호출을 사용할 수 있습니다. 그러나, 작업 단위가 IBM MQ 자원 및 다른 자원 관리자에 속한 자원(예: Db2) 모두에 영향을 미치는 경우에는 z/OS Recoverable Resource Service(RRS)에서 제공하는 SRRBACK 호출을 사용하십시오. SRRBACK 호출은 RRS 조정을 위해 설정한 자원 관리자에 속한 자원의 변경사항을 백아웃합니다.
    - CICS 애플리케이션은 EXEC CICS SYNCPOINT ROLLBACK 명령을 사용하여 작업 단위를 백아웃해야 합니다. CICS 애플리케이션에 대해 MQBACK 호출을 사용하지 마십시오.
    - IMS 애플리케이션(일괄 처리 DL/I 프로그램 제외)은 작업 단위를 백아웃하기 위해 ROLB과(와) 같은 IMS 호출을 사용해야 합니다. (배치 DL/I 프로그램 이외의) IMS 애플리케이션에 대해 MQBACK 호출을 사용하지 마십시오.
  - IBM i에서는 큐 관리자가 조정하는 로컬 작업 단위에 이 호출을 사용하십시오. 이는 커밋 정의가 작업 레벨에 존재하지 않아야 함을 의미합니다. 즉, **CMTSCOPE(\*JOB)** 매개변수가 있는 STRCMTCTL 명령이 작업에 대해 실행되지 않았어야 합니다.
3. 애플리케이션이 작업 단위에서 커밋되지 않은 변경사항으로 종료되는 경우, 해당 변경사항의 배치는 애플리케이션이 정상으로 또는 비정상적으로 종료되는지 여부에 달려 있습니다. 추가적인 세부사항은 1231 페이지의 『IBM i의 MQDISC(큐 관리자 연결 끊기)』의 사용 시 참고사항을 참조하십시오.
4. 애플리케이션이 논리 메시지의 세그먼트 또는 그룹에서 메시지를 넣거나 가져올 때 큐 관리자는 마지막 성공한 MQPUT 및 MQGET 호출에 대한 메시지 그룹 및 논리 메시지와 관련된 정보를 보유하고 있습니다. 이 정보는 큐 핸들과 연관되어 있으며 다음과 같은 내용을 포함합니다.
  - MQMD에 있는 *GroupId*, *MsgSeqNumber*, *Offset* 및 *MsgFlags* 필드의 값입니다.
  - 메시지가 작업 단위의 일부인지 여부.
  - MQPUT 호출의 경우: 메시지가 지속적인지 또는 비지속적인지 여부.

큐 관리자는 다음 각각에 대해 한 세트씩, 그룹 및 세그먼트 정보의 세 개 세트를 유지합니다.

- 마지막 성공적인 MQPUT 호출(이는 작업 단위의 일부일 수 있음).
- 큐에서 메시지를 제거한 마지막 성공적인 MQGET 호출(이는 작업 단위의 일부일 수 있음).
- 큐에서 메시지를 찾아보는 마지막으로 성공한 MQGET 호출(작업 단위의 일부가 될 수 없음).

애플리케이션이 작업 단위의 일부로 메시지를 넣거나 가져오는 경우 애플리케이션은 작업 단위를 백아웃하고 그룹 및 세그먼트 정보는 이전에 있었던 값으로 복원됩니다.

- MQPUT 호출과 연관된 정보가 첫 번째 성공적인 MQPUT 호출 이전 현재 작업 단위의 큐 핸들에 대한 값으로 복원됩니다.
- MQGET 호출과 연관된 정보가 현재 작업 단위에서 해당 큐 핸들에 대해 처음으로 성공한 MQGET 호출 전에 가지고 있던 값으로 복원됩니다.

작업 단위가 시작된 이후 애플리케이션을 통해 업데이트되었지만 작업 단위 범위의 외부에 있는 큐에서는 작업 단위가 백업된 경우 그룹 및 세그먼트 정보가 복원되지 않습니다.

작업의 단위가 백아웃될 때 그룹과 세그먼트 정보를 해당 이전 값으로 복구하면 애플리케이션이 여러 작업 단위 전체에 걸쳐 많은 세그먼트로 구성되어 대량 메시지 그룹 또는 대량 논리 메시지를 펼칠 수 있습니다. 그리고 작업 단위 중 하나가 실패하면 메시지 그룹 또는 논리 메시지의 정확한 위치에서 재시작하도록 허용합니다. 로컬 큐 관리자에 제한된 큐 스토리지만 있는 경우에는 여러 작업 단위를 사용하면 유용할 수 있습니다. 그러나 애플리케이션은 시스템 장애가 발생하는 경우 정확한 위치에 메시지를 넣거나 가져올 수 있도록 위한 충분한 정보를 유지해야 합니다.

시스템 장애 후 현재 지점에서 재시작하는 방법에 대한 자세한 정보는 PMOPT (10자리 부호 있는 정수)에 설명된 PMLOGO 옵션 및 GMOPT (10자리 부호 있는 정수)에 설명된 GMLOGO 옵션을 참조하십시오.

나머지 사용 시 참고사항은 큐 관리자가 작업 단위를 통합할 때만 적용됩니다.

5. 작업 단위는 연결 핸들과 동일한 범위를 갖습니다. 특정 작업 단위에 영향을 미치는 모든 IBM MQ 호출은 동일한 연결 핸들을 사용하여 수행해야 합니다. 다른 연결 핸들을 사용하여 실행된 호출(예: 다른 애플리케이션이 발행한 호출)은 다음 작업 단위에 영향을 줍니다. 연결 핸들의 범위에 대한 정보는 HCONN(10자리 부호 있는 정수) - 출력의 내용을 참조하십시오.
6. 현재 작업 단위의 일부로서 넣거나 검색된 메시지만 이 호출의 영향을 받습니다.
7. 작업 단위 내에서 MQGET, MQPUT 또는 MQPUT1 호출을 발행했지만 커밋 또는 백아웃 호출을 발행하지 않고 장시간 실행 중인 애플리케이션이 다른 애플리케이션에서 사용할 수 없는 메시지로 큐를 채울 수 있습니다. 이러한 가능성에 대비하기 위해 관리자는 **MaxUncommittedMsgs** 큐 관리자 속성을 런어웨이 애플리케이션이 큐를 채우는 것을 방지할 만큼 충분히 낮지만 예상 메시지 애플리케이션이 올바르게 작동할 수 있을 만큼 충분히 높은 값으로 설정해야 합니다.

## 매개변수

MQDLTMH 호출에 다음 매개변수가 있습니다.

### HCONN(10자리 부호 있는 정수) - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다.

값이 **HMSG** 매개변수에 지정된 메시지 핸들 작성에 사용된 연결 핸들과 일치해야 합니다.

메시지 핸들이 HCUNAS를 사용하여 작성되면 유효한 연결은 메시지 핸들을 삭제하는 스레드에 설정되어야 합니다. 그렇지 않으면 호출은 RC2009로 실패합니다.

### HMSG(10자리 부호 있는 정수) - 입출력(I/O)

이는 삭제될 메시지 핸들입니다. 값은 이전 MQCRTMH 호출에서 리턴합니다.

호출의 정상 완료 시 핸들은 환경에 올바르지 않은 값으로 설정됩니다. 이 값은 다음과 같습니다.

#### HMUNUH

사용 불가능한 메시지 핸들.

동일한 메시지 핸들을 전달한 다른 IBM MQ 호출이 진행 중에 있는 경우 메시지 핸들은 삭제될 수 없습니다.

### DLTOPT(MQDMHO) - 입력

세부사항은 MQDMHO를 참조하십시오.

### CMPCOD(10자리 부호 있는 정수) - 출력

완료 코드는 다음 중 하나입니다.

#### CCOK

정상적으로 완료되었습니다.

#### CCFAIL

호출에 실패했습니다.

### REASON(10자리 부호 있는 정수) - 출력

CMPCOD를 규정하는 이유 코드입니다.

CMPCOD가 CCOK일 경우:

#### RCNONE

(0, X'000') 보고할 이유가 없습니다.

CMPCOD가 CCFAIL일 경우:

#### RC2204

(2204, X'089C') 어댑터를 사용할 수 없습니다.

**RC2130**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

**RC2157**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

**RC2219**

(2219, X'08AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**RC2009**

(2009, X'07D9') 큐 관리자에 대한 연결이 유실되었습니다.

**RC2462**

(2462, X'099E') 올바르지 않은 메시지 핸들 옵션 구조가 삭제되었습니다.

**RC2460**

(2460, X'099C') 메시지 핸들 포인터가 올바르지 않습니다.

**RC2499**

(2499, X'09C3') 메시지 핸들이 이미 사용 중입니다.

**RC2046**

(2046, X'07FE') 옵션이 올바르지 않거나 일치하지 않습니다.

**RC2071**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**RC2195**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

자세한 내용은 [1359 페이지](#)의 『IBM i(ILE RPG)의 리턴 코드』의 내용을 참조하십시오.

**RPG 선언**

```

C*.1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQDLTMH(HCONN : HMSG : DLTOPT :
                      CMPCOD : REASON)

```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```

DMQDLTMH      PR          EXTPROC('MQDLTMH')
D* Connection handle
D HCONN              10I 0 VALUE
D* Message handle
D HMSG              20I 0
D* Options that control the action of MQDLTMH
D DLTOPT            12A
D* Completion code
D CMPCOD            10I 0
D* Reason code qualifying CompCode
D REASON            10I 0

```

**MQDLTMP - 메시지 특성 삭제**

MQDLTMP 호출은 메시지 핸들의 특성을 삭제하며 MQSETMP 호출의 역수입니다.

- [1236 페이지](#)의 『구문』
- [1237 페이지](#)의 『매개변수』
- [1238 페이지](#)의 『RPG 선언』

**구문**

MQDLTMP (*Hconn*, *Hmsg*, *DltPropOpts*, *Name*, *CompCode*, *Reason*)



## 매개변수

MQDLTMP 호출에 다음 매개변수가 있습니다.

### HCONN(10자리 부호 있는 정수) - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. 값이 **HMSG** 매개변수에 지정된 메시지 핸들 작성에 사용된 연결 핸들과 일치해야 합니다.

메시지 핸들이 HCUNAS를 사용하여 작성되면 유효한 연결은 메시지 핸들을 삭제하는 스레드에 설정되어야 합니다. 그렇지 않으면 호출은 RC2009로 실패합니다.

### HMSG(10자리 부호 있는 정수) - 입력

이는 삭제될 특성을 포함하는 메시지 핸들입니다. 값은 이전 MQCRTMH 호출에서 리턴합니다.

### DLTOPT(MQDMPO) - 입력

세부사항은 [MQDMPO](#) 데이터 유형을 참조하십시오.

### PRNAME (MQCHARV) - 입력

삭제할 특성 이름입니다. 특성 이름에 대한 추가 정보는 [특성 이름](#)을 참조하십시오.

와일드 카드는 특성 이름으로 허용되지 않습니다.

### CMPCOD(10자리 부호 있는 정수) - 출력

완료 코드는 다음 중 하나입니다.

#### CCOK

정상적으로 완료되었습니다.

#### CCWARN

경고(일부 완료).

#### CCFAIL

호출에 실패했습니다.

### REASON(10자리 부호 있는 정수) - 출력

CMPCOD를 규정하는 이유 코드입니다.

CMPCOD가 CCOK일 경우:

#### RCNONE

(0, X'000') 보고할 이유가 없습니다.

If CMPCOD is CCWARN:

#### RC2471

(2471, X'09A7') 특성을 사용할 수 없습니다.

#### RC2421

(2421, X'0975') 특성을 포함하는 MQRFH2 폴더를 구문 분석할 수 없습니다.

CMPCOD가 CCFAIL일 경우:

#### RC2204

(2204, X'089C') 어댑터를 사용할 수 없습니다.

#### RC2130

(2130, X'0852') 어댑터 서비스 모듈을 로드할 수 없습니다.

#### RC2157

(2157, X'086D') 1차 및 홈 ASID가 다릅니다.

#### RC2219

(2219, X'08AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

#### RC2009

(2009, X'07D9') 큐 관리자에 대한 연결이 유실되었습니다.

**RC2481**

(2481, X'09B1') 삭제 메시지 특성 옵션 구조는 올바르지 않습니다.

**RC2460**

(2460, X'099C') 메시지 핸들이 올바르지 않습니다.

**RC2499**

(2499, X'09C3') 메시지 핸들이 이미 사용 중입니다.

**RC2046**

(2046, X'07FE') 옵션이 올바르지 않거나 일치하지 않습니다.

**RC2442**

(2442, X'098A') 올바르지 않은 특성 이름입니다.

**RC2111**

(2111, X'083F') 특성 이름 코드화 문자 세트 ID가 올바르지 않습니다.

**RC2195**

(2195, X'0893') 예상치 못한 오류가 발생했습니다.

이러한 코드에 대한 자세한 정보는 [API 완료 및 이유 코드](#)를 참조하십시오.

**RPG 선언**

```

C*.1.....2.....3.....4.....5.....6.....7..
C          CALLP          MQDLTMP(HCONN : HMSG : DLTOPT :
                          PRNAME : CMPCOD : REASON)

```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```

DMQDLTMP          PR          EXTPROC('MQDLTMP')
D* Connection handle
D HCONN          10I 0 VALUE
D* Message handle
D HMSG          20I 0 VALUE
D* Options that control the action of MQDLTMP
D DLTOPT          12A
D* Property name
D PRNAME          32A
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CompCode
D REASON          10I 0

```

**IBM i IBM i의 MQGET(메시지 가져오기)**

MQGET 호출은 MQOPEN 호출을 사용하여 열린 로컬 큐에서 메시지를 검색합니다.

- [1238 페이지의 『구문』](#)
- [1238 페이지의 『사용법 참고』](#)
- [1241 페이지의 『매개변수』](#)
- [1245 페이지의 『RPG 선언』](#)

**구문**

MQGET (HCONN, HOBJ, MSGDSC, GMO, BUFLN, BUFFER, DATLEN, CMPCOD, REASON)

**사용법 참고**

1. 검색된 메시지는 일반적으로 큐에서 삭제됩니다. 이 삭제는 MQGET 호출 자체의 일부로 또는 동기점의 일부로 발행할 수 있습니다. GMBRWF 또는 GMBRWN 옵션이 GMO 매개변수에 지정된 경우 메시지 삭제가 발생하지 않습니다([1032 페이지의 『IBM i의 MQGMO\(메시지 가져오기 옵션\)』](#)에 설명된 GMOPT 필드 참조).

2. GMLK 옵션이 찾아보기 옵션 중 하나와 함께 지정된 경우, 찾아본 메시지는 이 핸들에서만 볼 수 있도록 잠깁니다.
- GMUNLK 옵션이 지정되면 이전에 잠긴 메시지의 잠금이 해제됩니다. 이 경우 메시지를 검색하지 않고, **MSGDSC, BUFLN, BUFFER** 및 **DATLEN** 매개변수를 검사하거나 대체하지 않습니다.
3. MQGET 호출을 발행하는 애플리케이션이 IBM MQ MQI client로 실행 중인 경우, MQGET 호출을 처리하는 동안 IBM MQ MQI client가 비정상적으로 종료하거나 클라이언트 연결이 끊어지면 검색되는 메시지가 손실될 수 있습니다. 이 문제는 큐 관리자의 플랫폼에서 실행되고 클라이언트 대신 MQGET 호출을 발행한 대리가 메시지를 클라이언트로 메시지를 리턴하려고 할 때까지 클라이언트의 손실을 감지할 수 없기 때문에 발생합니다. 메시지가 큐에서 제거된 후 일어나는 일입니다. 지속 메시지 및 비지속 메시지 모두에 대해 발생할 수 있습니다.
- 항상 작업 단위 내에서 메시지를 검색하면(즉, MQGET 호출에 GMSYP 옵션을 지정하고, 메시지 처리가 완료될 때 MQCMIT 또는 MQBACK 호출을 사용하여 작업 단위를 커밋 또는 백아웃함) 이런 식의 메시지 손실 위험을 해결할 수 있습니다. GMSYP가 지정되고 클라이언트가 비정상적으로 종료되거나 연결이 끊긴 경우, 대리가 큐 관리자에서 작업 단위를 백아웃하고 메시지는 큐에 복원됩니다.
- 원칙적으로 큐 관리자의 플랫폼에서 실행되는 애플리케이션에서도 동일한 상황이 발생할 수 있지만, 이 경우 메시지가 손실될 수 있는 시간대가 짧습니다. 그러나, IBM MQ MQI clients와 마찬가지로 작업 단위 내에서 메시지를 검색하면 이러한 위험을 해결할 수 있습니다.
4. 애플리케이션이 단일 작업 단위 내의 특정큐에 메시지 시퀀스를 배치하고 작업 단위를 성공적으로 커밋하는 경우 다음과 같이 메시지를 검색에 사용할 수 있게 됩니다.
- 큐가 비공유 큐(즉, 로컬 큐)이면 작업 단위 내의 모든 메시지를 동시에 사용할 수 있게 됩니다.
  - 큐가 공유 큐이면 작업 단위 내의 메시지가 동시에 모두가 아닌 놓여진 순서대로 사용할 수 있게 됩니다. 시스템의 로드가 큰 경우 작업 단위에 있는 첫 번째 메시지 검색에는 성공하지만 작업 단위의 두 번째 또는 후속 메시지에 대한 MQGET 호출은 RC2033이 표시되면서 실패할 수 있습니다. 실패하면 애플리케이션은 잠시 대기한 다음 조작을 다시 시도해야 합니다.
5. 애플리케이션이 메시지 그룹을 사용하지 않고 동일 큐에 메시지 순서를 넣는 경우, 특정 조건을 충족하는 경우 이러한 메시지의 순서가 보존됩니다. 자세한 정보는 MQPUT 호출 설명에 나온 사용 시 참고사항을 참조하십시오. 다음 조건을 충족하면 메시지가 송신된 순서대로 수신 애플리케이션에 제공됩니다.
- 한 수신자만 큐에서 메시지를 가져옵니다.  
 큐에서 메시지를 가져오는 애플리케이션이 둘 이상 있는 경우 순서에 속하는 메시지를 식별하는 데 사용할 메커니즘을 송신자와 동의해야 합니다. 예를 들어, 송신자가 순서의 메시지에 있는 MDCID 필드를 메시지의 해당 순서에 고유한 값으로 설정할 수 있습니다.
  - 수신자가 의도적으로 검색 순서를 변경하지 않습니다(예: 특정 MDMID 또는 MDCID를 지정하는 방법으로).
- 송신 애플리케이션이 메시지를 메시지 그룹으로 넣을 경우 수신 애플리케이션이 MQGET 호출에 GMLOGO 옵션을 지정하면 메시지는 수신 애플리케이션에 올바른 순서로 제공됩니다. 메시지 그룹에 대한 자세한 정보는 다음을 참조하십시오.
- MQMD의 MDMFL 필드
  - MQPMO의 PMLOGO 옵션
  - MQGMO의 GMLOGO 옵션
6. 애플리케이션은 **MSGDSC** 매개변수의 MDFB 필드에서 피드백 코드 FBQUIT를 테스트합니다. 이 값이 있으면 애플리케이션이 종료됩니다. 자세한 정보는 [1063 페이지의 『IBM i의 MQMD\(메시지 디스크립터\)』](#)에 설명된 MDFB 필드를 참조하십시오.
7. HOBj로 식별된 큐가 OOSAVA 옵션으로 열렸고 MQGET 호출의 완료 코드가 CCOK 또는 CCWARN인 경우, 큐 핸들 HOBj와 연관된 컨텍스트가 검색된 메시지의 컨텍스트로 설정됩니다(GMBRWF 또는 GMBRWn 옵션이 설정되지 않은 경우 컨텍스트가 사용 불가능으로 표시됨). PMPASI 또는 PMPASA 옵션을 지정하면 후속 MQPUT 또는 MQPUT1 호출에서 이 컨텍스트를 사용할 수 있습니다. 이렇게 하면 수신된 메시지의 컨텍스트를 전체 또는 일부로 다른 메시지에 전송할 수 있습니다(예: 메시지를 다른 큐에 전달하는 경우). 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어](#)를 참조하십시오.

8. GMCONV 옵션이 **GMO** 매개변수에 포함된 경우, 데이터가 **BUFFER** 매개변수에 놓이기 전에 애플리케이션 메시지 데이터가 수신 애플리케이션에서 요청한 표현으로 변환됩니다.
- 메시지의 제어 정보에 있는 MDFMT 필드는 애플리케이션 데이터의 구조를 식별하고, 메시지의 제어 정보에 있는 MDCSI 및 MDENC 필드는 해당 문자 세트 ID와 인코딩을 지정합니다.
  - MQGET 호출을 발행하는 애플리케이션은 **MSGDSC** 매개변수의 MDCSI 및 MDENC 필드에 애플리케이션 메시지 데이터를 변환해야 하는 문자 세트 ID 및 인코딩을 지정합니다.

메시지 데이터 변환이 필요한 경우, 메시지의 제어 정보에 있는 MDFMT 필드의 값에 따라 큐 관리자 자체에서 또는 사용자 작성 엑시트에서 변환이 수행됩니다.

- 다음 형식 이름은 큐 관리자에 의해 자동으로 변환되며, 이 형식을 "내장" 형식이라고 합니다.

FMADMN	FMMDE
FMCICS	FMPCF
FMCMD1	FMRMH
FMCMD2	FMRFH
FMDLH	FMRFH2
FMDH	FMSTR
FMEVNT	FMTM
FMIMS	FMXQH
FMIMVS	

- 형식 이름 FMNONE은 메시지에 데이터의 네이처가 정의되어 있지 않음을 표시하는 특수 값입니다. 따라서, 메시지를 큐에서 검색하면 큐 관리자가 변환을 시도하지 않습니다.

**참고:** 형식 이름이 FMNONE인 메시지에 대해 MQGET 호출에 GMCONV를 지정하고 메시지의 문자 세트 또는 인코딩이 **MSGDSC** 매개변수에 지정된 것과 다른 경우 메시지가 **BUFFER** 매개변수에서 리턴되지만 (다른 오류가 없다고 가정), 완료 코드 CCWARN 및 이유 코드 RC2110이 표시되면서 호출이 완료됩니다.

FMNONE은 메시지 데이터의 네이처가 변환이 필요하지 않음을 나타내거나 송신 및 수신 애플리케이션이 두 애플리케이션 간에 메시지 데이터 송신 형식에 동의한 경우, FMNONE을 사용할 수 있습니다.

- 그 외 모든 형식 이름의 경우 변환을 위해 메시지가 사용자 작성 엑시트에 전달됩니다. 엑시트의 이름은 환경 특정 추가 항목 없이 형식과 동일합니다. 앞으로 지원될 형식 이름과 충돌할 수 있으므로 사용자 지정 형식 이름은 "MQ" 문자로 시작하지 않아야 합니다.

메시지의 사용자 데이터는 지원되는 문자 세트와 인코딩 간에 변환될 수 있습니다. 그러나, 메시지에 하나 이상의 IBM MQ 헤더 구조가 있으면 해당 메시지는 큐 이름에서 유효한 문자에 대해 1바이트 또는 2바이트 문자를 포함하는 문자 세트로(부터) 변환될 수 없습니다. 이러한 동작을 시도하면 이유 코드 RC2111 또는 RC2115가 표시되고 메시지가 변환 없이 리턴됩니다. 유니코드 문자 세트 UTF-16은 이러한 문자 세트의 예입니다.

MQGET에서의 리턴에서 다음 이유 코드는 메시지가 성공적으로 변환되었음을 표시합니다.

- RCNONE

다음 이유 코드는 메시지가 성공적으로 변환되었을 수 있음을 표시합니다. 애플리케이션은 **MSGDSC** 매개변수에서 MDCSI 및 MDENC 필드를 확인하여 찾아야 합니다.

- RC2079

기타 모든 이유 코드는 메시지가 변환되지 않았음을 표시합니다.

**참고:** 이 예에 설명된 이유 코드의 해석은 엑시트가 처리 지침을 준수하는 경우에만 사용자 작성 엑시트에 의해 수행된 변환에 적용됩니다.

9. 앞서 나온 내장 형식의 경우, GMCONV 옵션이 지정되면 메시지에 있는 문자열의 기본 변환을 큐 관리자가 수행할 수 있습니다. 기본 변환을 사용하면 큐 관리자가 문자열 데이터 변환 시 실제 문자 세트에 가까운 설치 지정 기본 문자 세트를 사용할 수 있습니다. 따라서 MQGET 호출은 CCWARN 및 이유 코드 RC2111 또는 RC2115가 표시되면서 완료되는 것이 아니라 완료 코드 CCOK와 함께 성공할 수 있습니다.

**참고:** 근사치의 문자 세트를 사용하여 문자열 데이터를 변환하면 일부 문자가 올바르게 않게 변환되는 결과가 발생할 수 있습니다. 이러한 상황을 방지하려면 문자열에 실제 문자 세트 및 기본 문자 세트 모두에 공통되는 문자만 사용하십시오.

기본 변환은 MQMD와 MQMDE 구조의 문자 필드 및 애플리케이션 메시지 데이터 모두에 적용됩니다.

- 애플리케이션 메시지 데이터의 기본 변환은 다음 명령문이 모두 true인 경우에만 발생합니다.
  - 애플리케이션이 GMCONV를 지정합니다.
  - 지원되지 않는 문자 세트로(부터) 변환되어야 하는 데이터가 메시지에 포함되어 있습니다.
  - 큐 관리자를 설치하거나 재시작할 때 기본 변환을 사용하도록 설정되었습니다.
- 큐 관리자에 기본 변환이 사용되는 경우, 필요에 따라 MQMD 및 MQMDE 구조의 문자 필드가 기본 변환됩니다. 이 변환은 애플리케이션에서 MQGET 호출에 GMCONV 옵션을 지정하지 않은 경우에도 수행됩니다.

10. RPG 프로그래밍 예에 표시된 **BUFFER** 매개변수는 문자열로 선언됩니다. 따라서 매개변수의 최대 길이가 256바이트로 제한됩니다. 더 큰 버퍼가 필요하면 매개변수를 물리적 파일에 있는 필드 또는 구조로 선언해야 합니다.

매개변수를 구조로 선언하면 가능한 최대 길이가 9999바이트로 커지는 반면, 매개변수를 물리적 파일의 필드로 선언하면 가능한 최대 길이가 약 32KB로 커집니다.

## 매개변수

MQGET 호출의 매개변수는 다음과 같습니다.

### HCONN(10자리 부호 있는 정수) - 입력

연결 핸들입니다.

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. HCONN의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

### HOBJ(10자리 부호 있는 정수) - 입력

오브젝트 핸들.

이 핸들은 메시지를 검색할 큐를 나타냅니다. HOBJ의 값은 이전 MQOPEN 호출로 리턴됩니다. 큐는 다음 옵션 중 하나 이상을 사용하여 열어야 합니다(세부사항은 [1262 페이지의 『IBM i의 MQOPEN\(오브젝트 열기\)』](#) 참조).

- OOINPS
- OOINPX
- OOINPQ
- OOBROW

### MSGDSC(MQMD) - 입출력(I/O)

메시지 디스크립터.

이 구조는 필요한 메시지의 속성 및 검색된 메시지의 속성을 나타냅니다. 자세한 내용은 [1063 페이지의 『IBM i의 MQMD\(메시지 디스크립터\)』](#)의 내용을 참조하십시오.

BUFLEN가 메시지 길이보다 작은 경우, GMATM이 **GMO** 매개변수에 지정되었는지 여부에 관계없이 큐 관리자가 여전히 MSGDSC를 입력합니다 ([1032 페이지의 『IBM i의 MQGMO\(메시지 가져오기 옵션\)』](#)에 설명된 GMOPT 필드 참조).

애플리케이션이 버전-1 MQMD를 제공할 경우, 리턴된 메시지에서 애플리케이션 메시지 데이터에 접두부 MQMDE가 붙어 있지만 이는 MQMDE에 있는 하나 이상의 필드에 기본값이 아닌 값이 있는 경우에만 해당됩니다. MQMDE에 있는 모든 필드에 기본값이 있으면 MQMDE가 생략됩니다. MQMD에 있는 MDFMT 필드의 FMMDE 형식 이름은 MQMDE가 있음을 표시합니다.

### GMO (MQGMO) - 입출력(I/O)

MQGET의 조치를 제어하는 옵션.

자세한 내용은 1032 페이지의 『IBM i의 MQGMO(메시지 가져오기 옵션)』의 내용을 참조하십시오.

#### **BUFLEN(10자리 부호 있는 정수) - 입력**

BUFFER 영역의 길이(바이트).

데이터가 없는 메시지에 대해, 또는 메시지가 큐에서 제거될 예정이고 데이터가 제거된 경우 0을 지정해야 합니다(이 경우 GMATM을 지정해야 함).

**참고:** 큐에서 읽을 수 있는 가장 긴 메시지의 길이는 **MaxMsgLength** 큐 속성에서 제공됩니다. 1305 페이지의 『큐의 속성』의 내용을 참조하십시오.

#### **BUFFER(1바이트 비트 문자열 x BUFLEN) - 출력**

메시지 데이터를 포함하는 영역.

버퍼는 메시지에 있는 데이터의 네이처에 적절한 경계에 맞춰야 합니다. 4바이트 맞추기는 대부분의 메시지(IBM MQ 헤더 구조가 있는 메시지 포함)에 적절해야 하지만, 일부 메시지는 보다 엄격한 맞추기를 요청할 수 있습니다. 예를 들어, 64비트 2진 정수를 포함하는 메시지에는 8바이트 맞추기가 필요할 수 있습니다.

BUFLEN 가 메시지 길이보다 작은 경우 가능한 많은 메시지가 BUFFER 로 이동됩니다. 이는 GMATM이 **GMO** 매개변수에 지정되는지 여부에 따라 발생합니다(자세한 정보는 1032 페이지의 『IBM i의 MQGMO(메시지 가져오기 옵션)』에 설명된 **GMOPT** 필드 참조).

**BUFFER** 에 있는 데이터의 문자 세트 및 인코딩은 **MSGDSC** 매개변수에서 리턴되는 **MDCSI** 및 **MDENC** 필드에 의해 제공됩니다. 이 값이 수신자에 필요한 값과 다른 경우 수신자는 애플리케이션 메시지 데이터를 필요한 문자 세트 및 인코딩으로 변환해야 합니다. **GMCONV** 옵션을 사용자 작성 엑시트와 함께 사용하여 메시지 데이터의 변환을 수행할 수 있습니다(이 옵션에 대한 자세한 정보는 1032 페이지의 『IBM i의 MQGMO(메시지 가져오기 옵션)』 참조).

**참고:** MQGET 호출의 다른 모든 매개변수는 로컬 큐 관리자의 문자 세트 및 인코딩에 있습니다(**CodedCharSetId** 큐 관리자 속성 및 **ENNAT**에서 제공).

호출이 실패해도 버퍼의 콘텐츠가 여전히 변경되었을 수 있습니다.

#### **DATLEN(10자리 부호 있는 정수) - 출력**

메시지의 길이

이는 메시지에 있는 애플리케이션 데이터의 길이(바이트)입니다. 이 메시지 길이가 **BUFLEN**보다 크면 **BUFLEN** 바이트만 **BUFFER** 매개변수에 리턴됩니다(즉, 메시지가 잘림). 값이 0이면 메시지에 애플리케이션 데이터가 없다는 의미입니다.

**BUFLEN** 이 메시지 길이보다 작은 경우, **GMATM**이 **GMO** 매개변수에 지정되었는지 여부에 관계없이 큐 관리자가 여전히 **DATLEN** 를 입력합니다(자세한 정보는 1032 페이지의 『IBM i의 MQGMO(메시지 가져오기 옵션)』에 설명된 **GMOPT** 필드 참조). 이렇게 하면 애플리케이션이 메시지 데이터를 수용하는 데 필요한 버퍼의 크기를 판별하고 적절한 크기의 버퍼를 사용하여 호출을 다시 발행할 수 있습니다.

그러나, **GMCONV** 옵션이 지정되고 변환된 메시지 데이터가 **BUFFER**에 비해 너무 길면 **DATLEN**에 리턴되는 값은 다음과 같습니다.

- 큐 관리자 정의 형식의 경우, 변환되지 않은 데이터의 길이.
  - 이 경우 데이터의 네이처로 인해 데이터가 변환 중에 확장되면 애플리케이션이 **DATLEN**에 대해 큐 관리자가 리턴한 값보다 큰 버퍼를 할당해야 합니다.
- 애플리케이션 정의 형식의 경우, 데이터 변환 엑시트에 의해 리턴된 값.

#### **CMPCOD(10자리 부호 있는 정수) - 출력**

완료 코드.

다음 중 하나입니다.

##### **CCOK**

정상적으로 완료되었습니다.

##### **CCWARN**

경고(일부 완료).

**CCFAIL**

호출에 실패했습니다.

**REASON(10자리 부호 있는 정수) - 출력**

CMPCOD을(를) 규정하는 이유 코드.

다음 이유 코드는 큐 관리자가 **REASON** 매개변수에 대해 리턴할 수 있는 코드입니다. 애플리케이션이 GMCONV 옵션을 지정하고 메시지 데이터의 일부 또는 전부를 변환하기 위해 사용자 작성 엑시트가 호출되는 경우, **REASON** 매개변수에 리턴되는 값은 엑시트가 결정합니다. 따라서 이 절의 후반부에 설명하는 값 이외의 값을 사용할 수 있습니다.

CMPCOD가 CCOK일 경우:

**RCNONE**

(0, X'000') 보고할 이유가 없습니다.

If CMPCOD is CCWARN:

**RC2120**

(2120, X'848') 변환된 데이터가 버퍼에 비해 너무 큽니다.

**RC2190**

(2190, X'88E') 변환된 문자열이 필드에 비해 너무 큽니다.

**RC2150**

(2150, X'866') DBCS 문자열이 올바르지 않습니다.

**RC2110**

(2110, X'83E') 메시지 형식이 올바르지 않습니다.

**RC2243**

(2243, X'8C3') 메시지 세그먼트에 서로 다른 CCSID가 있습니다.

**RC2244**

(2244, X'8C4') 메시지 세그먼트에 서로 다른 인코딩이 있습니다.

**RC2209**

(2209, X'8A1') 잠긴 메시지가 없습니다.

**RC2119**

(2119, X'847') 메시지 데이터가 변환되지 않았습니다.

**RC2272**

(2272, X'8E0') 메시지 데이터가 부분적으로 변환되었습니다.

**RC2145**

(2145, X'861') 소스 버퍼 매개변수가 올바르지 않습니다.

**RC2111**

(2111, X'83F') 소스 코드화 문자 세트 ID가 올바르지 않습니다.

**RC2113**

(2113, X'841') 메시지의 팩형 10진수 인코딩이 인식되지 않습니다.

**RC2114**

(2114, X'842') 메시지의 부동 소수점 인코딩이 인식되지 않습니다.

**RC2112**

(2112, X'840') 소스 정수 인코딩이 인식되지 않습니다.

**RC2143**

(2143, X'85F') 소스 길이 매개변수가 올바르지 않습니다.

**RC2146**

(2146, X'862') 대상 버퍼 매개변수가 올바르지 않습니다.

**RC2115**

(2115, X'843') 대상 코드화 문자 세트 ID가 올바르지 않습니다.

**RC2117**

(2117, X'845') 수신자가 지정한 팩형 10진수 인코딩을 인식할 수 없습니다.

- RC2118**  
(2118, X'846') 수신자에 의해 지정되는 부동 소수점 인코딩이 인식되지 않습니다.
- RC2116**  
(2116, X'844') 대상 정수 인코딩이 인식되지 않습니다.
- RC2079**  
(2079, X'81F') 잘린 메시지가 리턴되었습니다(처리 완료됨).
- RC2080**  
(2080, X'820') 잘린 메시지가 리턴되었습니다(처리 미완료).
- CMPCOD가 CCFAIL일 경우:
- RC2004**  
(2004, X'7D4') 버퍼 매개변수가 올바르지 않습니다.
- RC2005**  
(2005, X'7D5') 버퍼 길이 매개변수가 올바르지 않습니다.
- RC2219**  
(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 다시 입력되었습니다.
- RC2009**  
(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.
- RC2010**  
(2010, X'7DA') 데이터 길이 매개변수가 올바르지 않습니다.
- RC2016**  
(2016, X'7E0') 큐에 대해 가져오기가 금지되었습니다.
- RC2186**  
(2186, X'88A') 메시지 가져오기 옵션 구조가 올바르지 않습니다.
- RC2018**  
(2018, X'7E2') 연결 핸들이 유효하지 않습니다.
- RC2019**  
(2019, X'7E3') 오브젝트 핸들이 올바르지 않습니다.
- RC2241**  
(2241, X'8C1') 메시지 그룹이 완료되지 않았습니다.
- RC2242**  
(2242, X'8C2') 논리 메시지가 완료되지 않았습니다.
- RC2259**  
(2259, X'8D3') 찾아보기 지정에 일관성이 없습니다.
- RC2245**  
(2245, X'8C5') 작업 단위 지정에 일관성이 없습니다.
- RC2246**  
(2246, X'8C6') 커서에 있는 메시지가 검색에 대해 유효하지 않습니다.
- RC2247**  
(2247, X'8C7') 일치 옵션이 올바르지 않습니다.
- RC2026**  
(2026, X'7EA') 메시지 디스크립터가 올바르지 않습니다.
- RC2250**  
(2250, X'8CA') 메시지 순서 번호가 올바르지 않습니다.
- RC2033**  
(2033, X'7F1') 사용 가능한 메시지가 없습니다.
- RC2034**  
(2034, X'7F2') 찾아보기 커서가 메시지에 위치해 있지 않습니다.
- RC2036**  
(2036, X'7F4') 큐가 읽기 전용으로 열려있지 않습니다.



**RC2037**

(2037, X'7F5') 큐가 입력을 위해 열려있지 않습니다.

**RC2041**

(2041, X'7F9') 오브젝트가 열린 후에 정의가 변경되었습니다.

**RC2101**

(2101, X'835') 오브젝트가 손상되었습니다.

**RC2046**

(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.

**RC2052**

(2052, X'804') 큐가 삭제되었습니다.

**RC2058**

(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.

**RC2059**

(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.

**RC2161**

(2161, X'871') 큐 관리자가 정지됩니다.

**RC2162**

(2162, X'872') 큐 관리자가 종료되었습니다.

**RC2102**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**RC2071**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**RC2024**

(2024, X'7E8') 현재 작업 단위 내에서 더 이상 메시지를 핸들링할 수 없습니다.

**RC2072**

(2072, X'818') 동기점 지원을 사용할 수 없습니다.

**RC2195**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

**RC2255**

(2255, X'8CF') 사용할 큐 관리자에 대해 작업 단위를 사용할 수 없습니다.

**RC2090**

(2090, X'82A') MQGMO의 대기 간격이 올바르지 않습니다.

**RC2256**

(2256, X'8D0') 올바르지 않은 버전의 MQGMO가 제공되었습니다.

**RC2257**

(2257, X'8D1') 올바르지 않은 버전의 MQMD가 제공되었습니다.

**RPG 선언**

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQGET(HCONN : HOBJ : MSGDSC : GMO :
C                               BUFLN : BUFFER : DATLEN :
C                               CMPCOD : REASON)

```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```

D*..1.....2.....3.....4.....5.....6.....7..
DMQGET          PR          EXTPROC('MQGET')
D* Connection handle
D HCONN          10I 0 VALUE
D* Object handle
D HOBJ          10I 0 VALUE
D* Message descriptor
D MSGDSC          364A

```

```

D* Options that control the action of MQGET
D GMO 112A
D* Length in bytes of the Buffer area
D BUFLN 10I 0 VALUE
D* Area to contain the message data
D BUFFER * VALUE
D* Length of the message
D DATLEN 10I 0
D* Completion code
D CMPCOD 10I 0
D* Reason code qualifying CMPCOD
D REASON 10I 0

```

## IBM i IBM i의 MQINQ(오브젝트 속성 조회)

MQINQ 호출은 오브젝트의 속성을 포함한 문자열 세트 및 정수 배열을 리턴합니다.

올바른 오브젝트의 유형은 다음과 같습니다.

- 큐
- 이름 목록
- 프로세스 정의
- 큐 관리자
- [1246 페이지의 『구문』](#)
- [1246 페이지의 『사용법 참고』](#)
- [1247 페이지의 『매개변수』](#)
- [1254 페이지의 『RPG 선언』](#)

### 구문

MQINQ (*HCONN, HOBJ, SELCNT, SELS, IACNT, INTATR, CALEN, CHRATR, CMPCOD, REASON*)

### 사용법 참고

1. 리턴되는 값이 선택된 속성의 스냅샷입니다. 애플리케이션이 리턴되는 값에 따라 조치를 취하기 전에 속성이 변경되지 않도록 보장할 수 없습니다.
2. 모델 큐를 열면 동적 로컬 큐가 작성됩니다. 해당 속성을 조회하기 위해 모델 큐를 열 때에도 마찬가지입니다. 특정 예외가 있으나 동적 큐의 속성은 동적 큐가 작성되는 시점의 모델 큐의 속성과 동일합니다. 이 큐에서 MQINQ 호출을 사용하면 큐 관리자는 동적 큐의 속성을 리턴하지만 모델 큐의 속성은 리턴하지 않습니다. 동적 큐에서 상속되는 모델 큐의 속성에 대한 자세한 정보는 [표 1](#)의 내용을 참조하십시오.
3. 조회하는 오브젝트가 알리어스 큐이면 MQINQ 호출에 의해 리턴되는 속성 값이 알리어스 큐의 값이며 알리어스가 해석된 기본 큐의 값이 아닙니다.
4. 조회하는 오브젝트가 클러스터 큐이면 조회할 수 있는 속성은 큐를 여는 방법에 따라 다릅니다.
  - 조회는 물론 입력, 찾아보기 또는 설정을 한 번 이상 수행하기 위해 클러스터 큐를 연 경우, 열기가 성공하려면 클러스터 큐의 로컬 인스턴스가 있어야 합니다. 이 경우 조회 가능한 속성은 로컬 큐에 올바른 속성입니다.
  - 클러스터 큐가 조회용으로만 열린 경우 또는 조회 및 출력용으로 열린 경우에는 다음 속성만 조회할 수 있습니다. 이 경우, **QType** 속성의 값은 QTCLUS입니다.
    - CAQD
    - CAQN
    - IADBND
    - IADPER
    - IADPRI
    - IAIPUT

- IAQTYP

클러스터 큐가 고정된 바인딩 없이 열린 경우(즉, MQOPEN 호출에 OOBNDN이 지정되거나 **DefBind** 속성의 값이 BNDNOT일 때 OOBNDQ가 지정된 경우) 일반적으로 모든 인스턴스가 동일한 속성 값을 갖지만 큐에 대한 후속 MQINQ 호출이 클러스터 큐의 다른 인스턴스를 조회할 수 있습니다.

클러스터 큐에 대한 자세한 정보는 [큐 관리자 클러스터 구성의 내용](#)을 참조하십시오.

5. 많은 속성을 조회한 다음 그 일부를 MQSET 호출을 사용하여 설정하려면 수가 감소된 동일한 배열을 MQSET에 사용할 수 있도록 해당 설정할 속성을 선택자 배열의 시작 부분에 배치하는 것이 편리합니다.
6. 둘 이상의 경고 상황이 발생하면(**CMPCOD** 매개변수 참조), 해당되는 다음 목록 중 첫 번째 이유 코드가 리턴됩니다.
  - a. RC2068
  - b. RC2022
  - c. RC2008
7. 오브젝트 속성에 대한 자세한 정보는 다음을 참조하십시오.
  - [1305 페이지의 『큐의 속성』](#)
  - [1331 페이지의 『이름 목록에 대한 속성』](#)
  - [1332 페이지의 『IBM i의 프로세스 정의에 대한 속성』](#)
  - [1334 페이지의 『IBM i의 큐 관리자에 대한 속성』](#)
8. 명령을 실행할 때마다 생성되는 큐잉 메시지에 대해 새 로컬 큐 SYSTEM.ADMIN.COMMAND.EVENT가 사용됩니다. CMDEV 큐 관리자 속성의 설정 방법에 따라, 대부분의 명령에 대해 이 큐에 메시지를 넣습니다.
  - ENABLED - 모든 성공적인 명령에 대해 명령 이벤트 메시지가 생성되고 큐에 메시지를 넣습니다.
  - NODISPLAY - DISPLAY (MQSC) 명령과 Inquire (PCF) 명령을 제외한 모든 성공적인 명령에 대해 명령 이벤트 메시지가 생성되고 큐에 메시지를 넣습니다.
  - DISABLED - 명령 이벤트 메시지가 생성되지 않습니다(큐 관리자의 초기 기본값임).

## 매개변수

MQINQ 호출에는 다음 매개변수가 존재합니다.

### HCONN(10자리 부호 있는 정수) - 입력

연결 핸들입니다.

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. HCONN의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

### HOBJ(10자리 부호 있는 정수) - 입력

오브젝트 핸들.

이 핸들은 필요한 속성을 포함한 오브젝트(임의 유형)를 표시합니다. 핸들이 OOINQ 옵션을 지정한 이전 MQOPEN 호출을 통해 리턴되어야 합니다.

### SELCNT(10자리 부호 있는 정수) - 입력

선택자의 수.

이는 SELS 배열에서 제공되는 선택자의 수입니다. 리턴될 속성의 수입니다. 0은 올바른 값입니다. 허용된 최대 수는 256입니다.

### SELS(10자리 부호 있는 정수 x SELCNT) - 입력

속성 선택자의 배열.

이는 SELCNT 속성 선택자의 배열입니다. 각 선택자는 필요한 값이 있는 속성(정수 또는 문자)을 식별합니다.

각 선택자는 HOBJ가 나타내는 오브젝트 유형에 대해 올바른 것이어야 합니다. 그렇지 않으면 완료 코드 CCFAIL 및 이유 코드 RC2067이 표시되면서 호출이 실패합니다.

특수한 유형의 큐인 경우

- 선택자가 모든 유형의 큐에 올바르지 않으면 완료 코드 RC2067 및 이유 코드 RC2067이 표시되면서 호출이 실패합니다.
- 선택자가 오브젝트 유형 이외의 유형인 큐에 한정되어 적용되는 경우, 완료 코드 CCWARN 및 이유 코드 RC2068을 표시하며 호출이 성공합니다.
- 조회하는 큐가 클러스터 큐인 경우, 올바른 선택자는 큐가 해석되는 방법에 따라 결정됩니다. 세부사항은 사용 시 참고사항 4를 참조하십시오.

임의의 순서로 선택자를 지정할 수 있습니다. 정수 속성 선택자(IA\* 선택자)에 해당되는 속성 값은 SELS에서 해당 선택자가 발생한 순서와 동일한 순서대로 INTATR에서 리턴됩니다. 문자 속성 선택자(CA\* 선택자)에 해당되는 속성 값은 해당 선택자가 발생한 순서와 동일한 순서대로 CHRATR에서 리턴됩니다. IA\* 선택자는 CA\* 선택자로 인터리브될 수 있습니다. 각 유형에서 상대 순서만이 중요합니다.

참고:

1. 정수 및 문자 속성 선택자는 두 가지 다른 범위에서 할당됩니다. IA\* 선택자는 IAFRST에서 IALAST 사이의 범위에 상주하며 CA\* 선택자는 CAFRST에서 CALAST 사이의 범위에 상주합니다.  
각 범위에 대해 상수 IALSTU 및 CALSTU가 큐 관리자가 승인할 수 있는 가장 높은 값을 정의합니다.
2. 모든 IA\* 선택자가 처음 발생하면 동일한 요소 번호를 사용하여 SELS 및 INTATR 배열에서 해당되는 요소를 처리할 수 있습니다.

조회할 수 있는 속성이 다음 표에 나와 있습니다. CA\* 선택자의 경우, CHRATR에서 결과 문자열의 길이를 바이트 단위로 정의하는 상수가 괄호로 묶여 제공됩니다.

표 746. 큐에 대한 MQINQ 속성 선택자		
선택기	설명	참고
CAALTD	가장 최근 대체 날짜(LNDATE).	1
CAALTT	가장 최근 대체 시간(LNTIME).	1
CABRQN	초과 백아웃 리큐 이름(LNQN).	5
CABASQ	알리어스가 해석되는 큐의 이름(LNQN).	
CACFSN	커플링 기능 구조 이름(LNCFSN).	3
CACLN	클러스터 이름(LNCLUN).	1
CACLNL	클러스터 이름 목록(LNNLN).	1
CACRTD	큐 작성 날짜(LNCRTD).	
CACRTT	큐 작성 시간(LNCRTT).	
CAINIQ	이니시에이션 큐 이름(LNQN).	
CAPRON	프로세스 정의 이름(LNPRON).	
CAQD	큐 설명(LNQD).	
CAQN	큐 이름(LNQN).	
CARQMN	리모트 큐 관리자의 이름(LNQMN).	
CARQN	리모트 큐 관리자에 인식되는 리모트 큐의 이름(LNQN).	
CATRGD	트리거 데이터(LNTRGD).	5
CAXQN	전송 큐 이름(LNQN).	
IABTHR	백아웃 임계값.	5
IACDEP	큐의 메시지 수	

표 746. 큐에 대한 MQINQ 속성 선택자 (계속)		
선택기	설명	참고
IADBND	기본 바인딩입니다.	1
IADINP	기본 입력 열기 옵션.	5
IADPER	기본 메시지 지속성.	
IADPRI	기본 메시지 우선순위입니다.	5
IADEFT	큐 정의 유형입니다.	
IADIST	분배 목록 지원.	2
IAHGB	백아웃 수 (디스크에) 기록 여부.	5
IAIGET	가져오기 조작성 허용되는지 여부.	
IAIPUT	넣기 조작성 허용되는지 여부.	
IAMLEN	최대 메시지 길이.	
IAMDEP	큐에 허용되는 최대 메시지 수.	
IAMDS	메시지 우선순위가 적절한지 여부.	5
IAOIC	입력을 위해 큐를 연 MQOPEN 호출의 수.	
IAOOC	출력을 위해 큐를 연 MQOPEN 호출의 수.	
IAQDHE	큐 용량 상한 이벤트의 제어 속성.	4, 5
IAQDHL	큐 용량의 상한입니다.	4, 5
IAQDLE	큐 용량 하한 이벤트의 제어 속성.	4, 5
IAQDLL	큐 용량의 하한입니다.	4, 5
IAQDME	큐 용량 최대 이벤트의 제어 속성.	4, 5
IAQSI	큐 서비스 간격의 한계.	4, 5
IAQSIE	큐 서비스 간격 이벤트에 대한 제어 속성.	4, 5
IAQTYP	큐 유형.	
IAQSGD	큐 공유 그룹 속성 지정.	3
IARINT	큐 보유 간격.	5
IASCOP	큐 정의 범위.	4, 5
IASHAR	입력에 대한 큐 공유 여부입니다.	
IATRGC	트리거 제어.	
IATRGD	트리거 용량.	5
IATRGP	트리거에 대한 메시지 우선순위 임계값입니다.	5
IATRGT	트리거 유형.	
IAUSAG	사용법.	
CLWLUSEQ	리모트 큐를 사용합니다.	

**참고:**

1. 다음 플랫폼에서 지원됩니다.

-  AIX
-  IBM i
-  Windows
-  z/OS

및 해당 시스템에 연결된 IBM MQ MQI clients

2. 다음 플랫폼에서 지원됩니다.

-  AIX
-  IBM i
-  Windows

및 해당 시스템에 연결된 IBM MQ 클라이언트



3.  z/OS에서 지원됩니다.
4.  z/OS에서는 지원되지 않습니다.
5. VSE/ESA에서 지원되지 않습니다.

표 747. 이름 목록에 대한 MQINQ 속성 선택자		
선택기	설명	참고
CAALTD	가장 최근 대체 날짜(LNDATE)	1
CAALTT	가장 최근 대체 시간(LNTIME)	1
CALSTD	이름 목록 설명(LNNLD)	1
CALSTN	이름 목록 오브젝트의 이름(LNNLN)	1
CANAMS	이름 목록에 있는 이름(LNQX x 목록에 있는 이름 수)	1
IANAMC	이름 목록에 있는 이름 수	1
IAQSGD	큐 공유 그룹 속성 지정	3

표 748. 프로세스 정의에 대한 MQINQ 속성 선택자		
선택기	설명	참고
CAALTD	가장 최근 대체 날짜(LNDATE)	1
CAALTT	가장 최근 대체 시간(LNTIME)	1
CAAPPI	애플리케이션 ID(LNPROA)	5
CAENVV	환경 데이터(LNPROE)	5
CAPROD	프로세스 정의에 대한 설명(LNPROD)	5
CAPRON	프로세스 정의 이름(LNPRON)	5
CAUSRD	사용자 데이터(LNPROU)	5
IAAPPT	애플리케이션 유형	5
IAQSGD	큐 공유 그룹 속성 지정	3

표 749. 큐 관리자에 대한 MQINQ 속성 선택자		
선택기	설명	참고
CAALTD	가장 최근 대체 날짜(LNDATE)	1
CAALTT	가장 최근 대체 시간(LNTIME)	1
CACADX	자동 채널 정의 엑시트 이름(LNEXN)	1
CACLWD	클러스터 워크로드 엑시트로 전달된 데이터(LNEXDA)	1
CACLWX	클러스터 워크로드 엑시트 이름(LNEXN)	1
CACMDQ	시스템 명령 입력 큐 이름(LNQN)	5
CADLQ	데드-레터 큐 이름(LNQN)	5
CADXQN	기본 전송 큐 이름(LNQN)	5
CAQMD	큐 관리자 설명(LNQMD)	5
CAQMID	큐 관리자 ID(LNQMID)	1
CAQMN	로컬 큐 관리자 이름(LNQMN)	5
CAQSGN	큐 공유 그룹 이름(LNQSGN)	3
CARPN	큐 관리자가 저장소 서비스를 제공하는 클러스터의 이름(LNQMN)	1
CARPNL	큐 관리자가 저장소 서비스를 제공하는 클러스터의 이름을 포함하는 이름 목록 오브젝트의 이름(LNNLN)	1
CMDEV	명령이 실행될 때 생성된 메시지를 큐에 넣을 것인지 여부를 판별하는 제어 속성	8
IAAUTE	권한 이벤트에 대한 제어 속성	4, 5
IACAD	자동 채널 정의에 대한 제어 속성	2
IACADE	자동 채널 정의 이벤트에 대한 제어 속성	2
IACLXQ	기본 클러스터 전송 큐 유형	4
IACLWL	클러스터 워크로드 길이	1
IACCSI	코드화 문자 세트 ID	5
IACMDL	큐 관리자가 지원하는 명령 레벨	5
IACFGE	구성 이벤트에 대한 제어 속성	3
IADIST	분배 목록 지원	2
IAINHE	금지 이벤트에 대한 제어 속성	4, 5
IACLE	로컬 이벤트에 대한 제어 속성	4, 5
IAMHND	핸들의 최대 수입입니다.	5
IAMLEN	최대 메시지 길이	5
IAMPRI	최대 우선순위입니다.	5
IAMUNC	작업 단위 내에서 커밋되지 않은 최대 메시지 수입입니다.	5
IAPFME	성능 이벤트에 대한 제어 속성	4, 5
IAPLAT	큐 관리자가 상주하는 플랫폼	5
IARMTE	리모트 이벤트에 대한 제어 속성	4, 5
IASSE	시작 중지 이벤트에 대한 제어 속성	4, 5

표 749. 큐 관리자에 대한 MQINQ 속성 선택자 (계속)		
선택기	설명	참고
IASYNC	동기점 가용성	5
IATRLFT	사용하지 않는 비관리 토픽의 지속 시간	
IATRGI	트리거 간격	5

#### IACNT(10자리 부호 있는 정수) - 입력

정수 속성의 수.

이는 INTATR 배열에 있는 요소의 수입니다. 0은 올바른 값입니다.

이 값이 최소한 SELS 매개변수의 IA\* 선택자의 수인 경우, 요청된 모든 정수 속성이 리턴됩니다.

#### INTATR(10자리 부호 있는 정수 x IACNT) - 출력

정수 속성의 배열.

이는 IACNT 정수 속성 값의 배열입니다.

정수 속성 값은 SELS 매개변수의 IA\* 선택자와 동일한 순서대로 리턴됩니다. 배열에 IA\* 선택자의 수보다 많은 요소가 포함된 경우에는 초과 요소가 변경되지 않습니다.

HOBG가 큐를 나타내지만 속성 선택자가 해당 유형의 큐에 적용되지 않는 경우, INTATR 배열 내의 해당되는 요소에 대해 특정 값 IAVNA가 리턴됩니다.

#### CALEN(10자리 부호 있는 정수) - 입력

문자 속성 버퍼의 길이.

이는 CHRATR 매개변수의 길이(바이트)입니다.

이 값이 최소한 요청된 문자 속성 길이의 합계여야 합니다(SELS 참조). 0은 올바른 값입니다.

#### CHRATR(1바이트 문자열 x CALEN) - 출력

문자 속성.

함께 연결된 문자 속성이 리턴되는 버퍼입니다. 버퍼의 길이는 CALEN 매개변수로 제공됩니다.

문자 속성은 SELS 매개변수의 CA\* 선택자와 동일한 순서대로 리턴됩니다. 각 속성 문자열의 길이는 속성별로 고정되며(SELS 참조), 필요에 따라 해당 값의 오른쪽에 공백이 입력됩니다. 버퍼가 요청된 모든 문자 속성(채우기 포함)을 포함하는 데 필요한 것보다 큰 경우에는 리턴된 마지막 속성 값 이상의 바이트가 변경되지 않습니다.

HOBG가 큐를 나타내지만 속성 선택자가 해당 유형의 큐에 적용되지 않는 경우, CHRATR에서 해당 속성의 값으로 완전히 별표(\*)로 구성된 문자열이 리턴됩니다.

#### CMPCOD(10자리 부호 있는 정수) - 출력

완료 코드.

다음 중 하나입니다.

##### CCOK

정상적으로 완료되었습니다.

##### CCWARN

경고(일부 완료).

##### CCFAIL

호출에 실패했습니다.

#### REASON(10자리 부호 있는 정수) - 출력

CMPCOD을(를) 규정하는 이유 코드.



CMPCOD가 CCOK일 경우:

**RCNONE**

(0, X'000') 보고할 이유가 없습니다.

If *CMPCOD* is CCWARN:

**RC2008**

(2008, X'7D8') 문자 속성에 허용된 공간이 충분하지 않습니다.

**RC2022**

(2022, X'7E6') 정수 속성에 허용되는 공간이 충분하지 않습니다.

**RC2068**

(2068, X'814') 큐 유형에 적용할 수 없는 선택자입니다.

*CMPCOD*가 CCFAIL일 경우:

**RC2219**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 다시 입력되었습니다.

**RC2006**

(2006, X'7D6') 문자 속성의 길이가 올바르지 않습니다.

**RC2007**

(2007, X'7D7') 문자 속성 문자열이 올바르지 않습니다.

**RC2009**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**RC2018**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**RC2019**

(2019, X'7E3') 오브젝트 핸들이 올바르지 않습니다.

**RC2021**

(2021, X'7E5') 정수 속성의 수가 유효하지 않습니다.

**RC2023**

(2023, X'7E7') 정수 속성 배열이 유효하지 않습니다.

**RC2038**

(2038, X'7F6') 조회를 위해 큐를 열지 못했습니다.

**RC2041**

(2041, X'7F9') 오브젝트가 열린 후에 정의가 변경되었습니다.

**RC2101**

(2101, X'835') 오브젝트가 손상되었습니다.

**RC2052**

(2052, X'804') 큐가 삭제되었습니다.

**RC2058**

(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.

**RC2059**

(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.

**RC2162**

(2162, X'872') 큐 관리자가 종료되었습니다.

**RC2102**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**RC2065**

(2065, X'811') 선택자의 수가 유효하지 않습니다.

**RC2067**

(2067, X'813') 속성 선택자가 올바르지 않습니다.

**RC2066**

(2066, X'812') 선택자 수가 너무 큼니다.

**RC2071**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**RC2195**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

**RPG 선언**

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQINQ(HCONN : HOBJ : SELCNT :
C                               SELS(1) : IACNT : INTATR(1) :
C                               CALEN : CHRATR : CMPCOD :
C                               REASON)

```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```

D*..1.....2.....3.....4.....5.....6.....7..
DMQINQ          PR          EXTPROC('MQINQ')
D* Connection handle
D HCONN          10I 0 VALUE
D* Object handle
D HOBJ          10I 0 VALUE
D* Count of selectors
D SELCNT        10I 0 VALUE
D* Array of attribute selectors
D SELS          10I 0
D* Count of integer attributes
D IACNT         10I 0 VALUE
D* Array of integer attributes
D INTATR        10I 0
D* Length of character attributes buffer
D CALEN         10I 0 VALUE
D* Character attributes
D CHRATR        *   VALUE
D* Completion code
D CMPCOD        10I 0
D* Reason code qualifying CMPCOD
D REASON        10I 0

```

**IBM i IBM i의 MQINQMP(메시지 특성 조회)**

MQINQMP 호출은 메시지의 특성 값을 리턴합니다.

- [1254 페이지의 『구문』](#)
- [1254 페이지의 『매개변수』](#)
- [1258 페이지의 『RPG 선언』](#)

**구문**

MQINQMP (*Hconn*, *Hmsg*, *InqPropOpts*, *Name*, *PropDesc*, *Type*, *ValueLength*, *Value*, *DataLength*, *CompCode*, *Reason*)

**매개변수**

MQINQMP 호출의 매개변수는 다음과 같습니다.

**HCONN(10자리 부호 있는 정수) - 입력**

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *Hconn* 값이 **Hmsg** 매개변수에 지정된 메시지 핸들 작성에 사용된 연결 핸들과 일치해야 합니다.

HCUNAS를 사용하여 메시지 핸들을 작성한 경우, 메시지 핸들의 특성을 조회하는 스레드에 올바른 연결이 설정되어야 합니다. 그렇지 않으면 RC2009가 표시되면서 호출이 실패합니다.

## HMSG(10자리 부호 있는 정수) - 입력

조회할 메시지 핸들입니다. 이 값은 이전 MQCRTMH 호출을 통해 리턴됩니다.

## INQOPT (MQIMPO) - 입력

자세한 정보는 [MQIMPO](#) 데이터 유형을 참조하십시오.

## PRNAME (MQCHARV) - 입력

이 항목은 조회할 특성의 이름을 설명합니다.

이 이름의 특성이 없으면 이유 RC2471이 표시되면서 호출이 실패합니다.

특성 이름 끝에 퍼센트 부호(%) 문자를 사용할 수 있습니다. 와일드카드는 마침표(.) 문자를 포함하여 0개 이상의 문자와 일치합니다. 와일드카드를 사용하면 애플리케이션에서 많은 특성의 값을 조회할 수 있습니다. MQINQMP 호출에 IPINQF 옵션을 사용하면 첫 번째 일치하는 특성을 가져오고, 다시 IPINQN 옵션을 사용하면 그 다음으로 일치하는 특성을 가져옵니다. 더 이상 일치하는 특성이 없으면 RC2471이 표시되면서 호출이 실패합니다. InqPropOpts 구조의 ReturnedName 필드가 리턴된 특성 이름의 오프셋 또는 주소로 초기화되는 경우, 일치한 특성의 이름과 함께 MQINQMP의 리턴에서 완료됩니다. InqPropOpts 구조의 ReturnedName의 VSBufSize 필드가 리턴된 특성 이름의 길이보다 짧으면 이유는 RC2465로, 완료 코드는 CCFAIL로 설정됩니다.

알려진 동의어가 있는 특성은 다음과 같이 리턴됩니다.

1. 접두부 "mqps."가 있는 특성이 IBM MQ 특성 이름과 함께 리턴됩니다. 예를 들어, "MQTopicString"은 "mqps.Top"이 아닌 리턴된 이름입니다.
2. 접두부 "jms." 또는 "mcd."가 있는 특성이 JMS 헤더 필드 이름으로 리턴됩니다. 예를 들어, "JMSExpiration"은 "jms.Exp"가 아닌 리턴된 이름입니다.
3. 접두부 "usr."가 있는 특성이 해당 접두부 없이 리턴됩니다. 예를 들어, "usr.Color"가 아니라 "Color"가 리턴됩니다.

동의어가 있는 특성은 한 번만 리턴됩니다.

RPG 프로그래밍 언어에서 다음 매크로 변수는 모든 특성을 조회한 다음 "usr."로 시작하는 모든 특성을 조회하기 위해 정의됩니다.

## INQALL

메시지의 모든 특성을 조회합니다.

## INQUSR

"usr"로 시작하는 메시지의 모든 특성을 조회하십시오. 리턴되는 이름은 "usr." 접두부 없이 리턴됩니다.

IPINQN이 지정되었지만 이전 호출 이후 이름이 변경되거나 첫 번째 호출인 경우, IPINQF를 암시합니다.

특성 이름 사용에 대한 자세한 정보는 [특성 이름 및 특성 이름 제한사항](#)을 참조하십시오.

## PRPDSC (MQPD) - 출력

이 구조는 특성이 지원되지 않을 때 일어나는 일, 특성이 속한 메시지 컨텍스트, 특성이 복사되어야 하는 메시지 등 특성의 속성을 정의하는 데 사용됩니다. 이 구조에 대한 자세한 정보는 [MQPD](#)를 참조하십시오.

## TYPE(10자리 부호 있는 정수) - 입출력(I/O)

MQINQMP 호출의 리턴에서 이 매개변수는 Value 데이터 유형으로 설정됩니다. 데이터 유형은 다음 중 하나일 수 있습니다.

### TYPBOL

부울.

### TYPBST

바이트 문자열.

### TYPI8

8비트 서명 정수입니다.

### TYPI16

16비트 서명 정수입니다.

**TYPI32**

32비트 서명 정수입니다.

**TYPI64**

64비트 서명 정수입니다.

**TYPF32**

32비트 부동 소수점 숫자입니다.

**TYPF64**

64비트 부동 소수점 숫자입니다.

**TYPSTR**

문자열.

**TYPNUL**

특성이 존재하지만 값이 널입니다.

특성 값의 데이터 유형이 인식되지 않으면 TYPSTR이 리턴되고 값의 문자열 표현이 *Value* 영역에 놓입니다. 데이터 유형의 문자열 표현은 *IPOPT* 매개변수의 *IPTYP* 필드에서 찾을 수 있습니다. 이유 RC2467과 함께 경고 완료 코드가 리턴됩니다.

또한, IPCTYP 옵션을 지정하면 특성 값의 변환이 요청됩니다. 리턴되는 특성에 대해 원하는 데이터 유형을 지정하려면 *Type*을 입력으로 사용하십시오. 데이터 유형 변환에 대한 자세한 정보는 1056 페이지의 『IBM i의 MQIMPO(메시지 특성 조회 옵션)』의 IPCTYP 옵션에 대한 설명을 참조하십시오.

유형 변환을 요청하지 않은 경우, 입력에 다음 값을 사용할 수 있습니다.

**TYPAST**

데이터 유형을 변환하지 않고 특성 값이 리턴됩니다.

**VALLEN(10자리 부호 있는 정수) - 입력**

값 영역의 길이(바이트)입니다.

값이 리턴될 필요가 없는 특성에 대해서는 0을 지정하십시오. 이들 특성은 널 값이나 빈 문자열을 갖도록 애플리케이션이 설계한 특성일 수 있습니다. IPQLEN 옵션이 지정된 경우에도 0을 지정하십시오. 이 경우, 값이 리턴되지 않습니다.

**VALUE(1바이트 비트 stringxVALLEN) - 출력**

조회한 특성 값을 포함할 영역입니다. 버퍼는 리턴되는 값에 적절한 경계에 맞춰야 합니다. 이렇게 하지 못하면 나중에 값에 액세스할 때 오류가 발생할 수 있습니다.

VALLEN이 특성 값의 길이보다 작으면 최대한 많은 특성 값이 VALUE로 이동하고 완료 코드 CCFAIL 및 이유 RC2469가 표시되면서 호출이 실패합니다.

VALUE의 데이터 문자 세트는 INQOPT 매개변수의 IPRETCI 필드를 통해 제공됩니다. VALUE의 데이터 인코딩은 INQOPT 매개변수의 IPRETEC 필드를 통해 제공됩니다.

VALLEN 매개변수가 0이면 VALUE는 참조되지 않습니다.

**DATLEN(10자리 부호 있는 정수) - 출력**

Value 영역에 리턴되는 실제 특성 값의 길이(바이트)입니다.

DataLength가 특성 값 길이보다 작아도 MQINQMP 호출의 리턴에서 DataLength는 여전히 입력됩니다. 따라서 애플리케이션이 특성 값을 수용하는 데 필요한 버퍼의 크기를 판별하고 적절한 크기의 버퍼를 사용하여 호출을 다시 실행할 수 있게 됩니다.

다음 값도 리턴될 수 있습니다.

Type 매개변수를 TYPSTR 또는 TYPBST로 설정한 경우:

**VLEMP**

특성이 존재하지만, 문자나 바이트를 포함하지 않습니다.

**CMPCOD(10자리 부호 있는 정수) - 출력**

완료 코드는 다음 중 하나입니다.

**CCOK**

정상적으로 완료되었습니다.

**CCWARN**

경고(일부 완료).

**CCFAIL**

호출에 실패했습니다.

**REASON(10자리 부호 있는 정수) - 출력**

*CompCode*를 규정하는 이유 코드입니다.

*CMPCOD*가 CCOK일 경우:

**RCNONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*가 CCWARN인 경우:

**RC2492**

(2492, X'09BC') 리턴된 특성 이름이 변환되지 않습니다.

**RC2466**

(2466, X'09A2') 특성 값이 변환되지 않습니다.

**RC2467**

(2467, X'09A3') 특성 데이터 유형이 지원되지 않습니다.

**RC2421**

(2421, X'0975') 특성을 포함하는 MQRFH2 폴더를 구문 분석할 수 없습니다.

*CMPCOD*가 CCFAIL일 경우:

**RC2204**

(2204, X'089C') 어댑터를 사용할 수 없습니다.

**RC2130**

(2130, X'0852') 어댑터 서비스 모듈을 로드할 수 없습니다.

**RC2157**

(2157, X'086D') 1차 및 홈 ASID가 다릅니다.

**RC2004**

(2004, X'07D4') 값 매개변수가 올바르지 않습니다.

**RC2005**

(2005, X'07D5') 값 길이 매개변수가 올바르지 않습니다.

**RC2219**

(2219, X'08AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**RC2009**

(2009, X'07D9') 큐 관리자에 대한 연결이 유실되었습니다.

**RC2010**

(2010, X'07DA') 데이터 길이 매개변수가 올바르지 않습니다.

**RC2464**

(2464, X'09A0') 메시지 특성 조회 옵션 구조가 올바르지 않습니다.

**RC2460**

(2460, X'099C') 메시지 핸들이 올바르지 않습니다.

**RC2499**

(2499, X'09C3') 메시지 핸들이 이미 사용 중입니다.

**RC2064**

(2064, X'07F8') 옵션이 올바르지 않거나 일치하지 않습니다.

**RC2482**

(2482, X'09B2') 특성 디스크립터 구조가 올바르지 않습니다.

**RC2470**

(2470, X'09A6') 실제 항목에서 요청된 데이터 유형으로 변환할 수 없습니다.

**RC2442**

(2442, X'098A') 올바르지 않은 특성 이름입니다.

**RC2465**

(2465, X'09A1') 리턴된 이름 버퍼에 비해 특성 이름이 너무 길니다.

**RC2471**

(2471, X'09A7) 특성을 사용할 수 없습니다.

**RC2469**

(2469, X'09A5') 값 영역에 비해 특성 값이 너무 큽니다.

**RC2472**

(2472, X'09A8') 숫자 형식 오류가 값 데이터에서 발견되었습니다.

**RC2473**

(2473, X'09A9') 요청된 특성 유형이 올바르지 않습니다.

**RC2111**

(2111, X'083F') 특성 이름 코드화 문자 세트 ID가 올바르지 않습니다.

**RC2071**

(2071, X'0871') 사용 가능한 스토리지가 충분하지 않습니다.

**RC2195**

(2195, X'0893') 예상치 못한 오류가 발생했습니다.

이러한 코드에 대한 자세한 정보는 다음을 참조하십시오.

- [IBM MQ for z/OS 메시지, 완료 및 이유 코드 for IBM MQ for z/OS](#)
- 기타 모든 IBM MQ 플랫폼의 경우 [메시지 및 이유 코드](#)

**RPG 선언**

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP          MQINQMP(HCONN : HMSG : INQOPT :
                          PRNAME : PRPDSC : TYPE :
                          VALLEN : VALUE : DATLEN :
                          CMPCOD : REASON)

```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```

DMQINQMP          PR          EXTPROC('MQINQMP')
D* Connection handle
D HCONN          10I 0 VALUE
D* Message handle
D HMSG          20I 0 VALUE
D* Options that control the action of MQINQMP
D INQOPT          72A
D* Property name
D PRNAME          32A
D* Property descriptor
D PRPDSC          24A
D* Property data type
D TYPE          10I 0
D* Length in bytes of the Value area
D VALLEN          10I 0 VALUE
D* Property value
D VALUE          * VALUE
D* Length of the property value
D DATLEN          10I 0
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CompCode
D REASON          10I 0

```

MQMHBUF는 메시지 핸들을 버퍼로 변환하고 MQBUFMH 호출의 역입니다.

- [1259 페이지의 『구문』](#)
- [1259 페이지의 『사용법 참고』](#)
- [1259 페이지의 『매개변수』](#)
- [1261 페이지의 『RPG 선언』](#)

## 구문

MQMHBUF (*Hconn*, *Hmsg*, *MsgHBufOpts*, *Name*, *MsgDesc*, *BufferLength*, *Buffer*, *DataLength*, *CompCode*, *Reason*)

## 사용법 참고

MQMHBUF는 메시지 핸들을 버퍼로 변환합니다.

MQGET API 엑시트와 함께 사용하면 메시지 특성 API를 사용하여 특정 특성에 액세스할 수 있고, 버퍼에 있는 이 특성을 메시지 핸들이 아닌 MQRFH2 헤더를 사용하도록 설계된 애플리케이션으로 다시 전달할 수 있습니다.

이 호출은 MQBUFMH 호출의 역으로, 버퍼의 메시지 특성을 메시지 핸들에 구문 분석하는 데 사용할 수 있습니다.

## 매개변수

MQMHBUF 호출의 매개변수는 다음과 같습니다.

### HCONN(10자리 부호 있는 정수) - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다.

HCONN 값이 HMSG 매개변수에 지정된 메시지 핸들 작성에 사용된 연결 핸들과 일치해야 합니다.

HCUNAS를 사용하여 메시지 핸들을 작성한 경우, 메시지 핸들을 삭제하는 스레드에 올바른 연결이 설정되어야 합니다. 올바른 연결이 설정되지 않으면 RC2009가 표시되면서 호출이 실패합니다.

### HMSG(10자리 부호 있는 정수) - 입력

이 핸들은 버퍼가 요구되는 메시지 핸들입니다.

값은 이전 MQCRTMH 호출에서 리턴합니다.

### MHBOPT (MQMHBO) - 입력

MQMHBO 구조에서는 애플리케이션이 메시지 핸들에서 버퍼가 생성되는 방법을 제어하는 옵션을 지정할 수 있습니다.

자세한 내용은 [976 페이지의 『IBM i의 MQBMHO\(메시지 핸들 옵션에 대한 버퍼\)』](#)의 내용을 참조하십시오.

### PRNAME (MQCHARV) - 입력

버퍼에 넣을 특성의 이름.

이 이름과 일치하는 특성을 찾을 수 없으면 RC2471이 표시되면서 호출이 실패합니다.

### 와일드카드

와일드카드를 사용하면 둘 이상의 특성을 버퍼에 넣을 수 있습니다. 그렇게 하려면 특성 이름 끝에 퍼센트 부호(%)를 사용하십시오. 이 와일드카드는 마침표(.) 문자를 포함하여 0개 이상의 문자와 일치합니다.

특성 이름 사용에 대한 자세한 정보는 [특성 이름 및 특성 이름 제한사항](#)을 참조하십시오.

### MSGDSC(MQMD) - 입출력(I/O)

MSGDSC 구조는 버퍼 영역의 콘텐츠를 설명합니다.

출력에서, 호출이 작성한 대로 버퍼 영역에 있는 데이터의 인코딩, 문자 세트 ID 및 형식을 올바르게 설명하도록 *Encoding*, *CodedCharSetId* 및 *Format* 필드가 설정됩니다.

이 구조의 데이터는 애플리케이션의 문자 세트와 인코딩에 있습니다.

#### **BUFLLEN(10자리 부호 있는 정수) - 입력**

*BUFLLEN*은 버퍼 영역의 길이(바이트)입니다.

#### **BUFFER(1바이트 비트 문자열 x BUFLLEN) - 입출력(I/O)**

*BUFFER*는 메시지 버퍼가 있는 영역을 정의합니다. 대부분의 데이터에서는 4바이트 경계에 버퍼를 맞춰야 합니다.

*BUFFER*에 문자 또는 숫자 데이터가 포함된 경우 *MSGDSC* 매개변수의 *CodedCharSetId* 및 *Encoding* 필드를 데이터에 적합한 값으로 설정하십시오. 그러면 필요한 경우 데이터를 변환할 수 있습니다.

특성이 메시지 버퍼에 있으면 선택적으로 제거됩니다. 나중에 호출에서 리턴 시 메시지 핸들에서 사용할 수 있게 됩니다.

C 프로그래밍 언어에서 매개변수는 pointer-to-void로 선언됩니다. 이는 모든 유형의 데이터 주소를 매개변수로 지정할 수 있다는 의미입니다.

*BUFLLEN* 매개변수가 0인 경우 *BUFFER*은(는) 참조되지 않습니다. 이 경우에 C 또는 System/390 어셈블러로 작성된 프로그램이 전달하는 매개변수 주소는 널(null)일 수 있습니다.

#### **DATLEN(10자리 부호 있는 정수) - 출력**

*DATLEN*은 버퍼에 있는 리턴된 특성의 길이(바이트)입니다. 값이 0이면 *PRNAME*에 제공된 값과 일치하는 특성이 없고 이유 코드 RC2471이 표시되면서 호출이 실패합니다.

*BUFLLEN*이 버퍼에 특성을 저장하는 데 필요한 길이보다 작으면 RC2469가 표시되면서 *MQMHBUF* 호출이 실패하지만, 값은 여전히 *DATLEN*에 입력됩니다. 이렇게 하면 애플리케이션이 특성을 수용하는 데 필요한 버퍼의 크기를 판별한 다음 필요한 *BUFLLEN*을 포함한 호출을 다시 실행할 수 있습니다.

#### **CMPCOD(10자리 부호 있는 정수) - 출력**

완료 코드는 다음 중 하나입니다.

##### **CCOK**

정상적으로 완료되었습니다.

##### **CCFAIL**

호출에 실패했습니다.

#### **REASON(10자리 부호 있는 정수) - 출력**

*CMPCOD*를 규정하는 이유 코드입니다.

*CMPCOD*가 CCOK일 경우:

##### **RCNONE**

(0, X'000') 보고할 이유가 없습니다.

*CMPCOD*가 CCFAIL일 경우:

##### **RC2204**

(2204, X'089C') 어댑터를 사용할 수 없습니다.

##### **RC2130**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

##### **RC2157**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

##### **RC2501**

(2501, X'095C') 메시지 핸들 버퍼 옵션 구조가 올바르지 않습니다.

##### **RC2004**

(2004, X'07D4') 버퍼 매개변수가 올바르지 않습니다.



**RC2005**

(2005, X'07D5') 버퍼 길이 매개변수가 올바르지 않습니다.

**RC2219**

(2219, X'08AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**RC2009**

(2009, X'07D9') 큐 관리자에 대한 연결이 유실되었습니다.

**RC2010**

(2010, X'07DA') 데이터 길이 매개변수가 올바르지 않습니다.

**RC2460**

(2460, X'099C') 메시지 핸들이 올바르지 않습니다.

**RC2026**

(2026, X'07EA') 메시지 디스크립터가 올바르지 않습니다.

**RC2499**

(2499, X'09C3') 메시지 핸들이 이미 사용 중입니다.

**RC2046**

(2046, X'07FE') 옵션이 올바르지 않거나 일치하지 않습니다.

**RC2442**

(2442, X'098A') 특성 이름이 올바르지 않습니다.

**RC2471**

(2471, X'09A7') 특성을 사용할 수 없습니다.

**RC2469**

(2469, X'09A5') 지정된 특성을 포함하기에 BufferLength 값이 너무 작습니다.

**RC2195**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

**RPG 선언**

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQMHBUF(HCONN : HMSG : MHBOPT :
                          PRNAME : MSGDSC : BUFLN :
                          BUFFER : DATLEN :
                          CMPCOD : REASON)

```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```

DMQMHBUF          PR          EXTPROC('MQMHBUF')
D* Connection handle
D HCONN           10I 0 VALUE
D* Message handle
D HMSG            20I 0 VALUE
D* Options that control the action of MQMHBUF
D MHBOPT          12A
D* Property name
D PRNAME          32A
D* Message descriptor
D MSGDSC          364A
D* Length in bytes of the Buffer area
D BUFLN           10I 0 VALUE
D* Area to contain the properties
D BUFFER          *   VALUE
D* Length of the properties
D DATLEN          10I 0
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CompCode
D REASON          10I 0

```

MQOPEN 호출은 오브젝트에 대한 액세스를 설정합니다.

올바른 오브젝트의 유형은 다음과 같습니다.

- 큐(분배 목록 포함)
- 이름 목록
- 프로세스 정의
- 큐 관리자
- 토픽

## 색인

- [1262 페이지의 『구문』](#)
- [1262 페이지의 『사용법 참고』](#)
- [1266 페이지의 『매개변수』](#)
- [1271 페이지의 『RPG 선언』](#)

## 구문

MQOPEN (*HCONN, OBJDSC, OPTS, HOBJ, CMPCOD, REASON*)

## 사용법 참고

1. 열린 오브젝트는 다음 중 하나입니다.

- 다음 목적으로 사용되는 큐:
  - 메시지 가져오기 또는 찾아보기(MQGET 호출 사용)
  - 메시지 넣기(MQPUT 호출 사용)
  - 큐의 속성 조회(MQINQ 호출 사용)
  - 큐의 속성 설정(MQSET 호출 사용)

이름 지정된 큐가 모델 큐이면 동적 로컬 큐가 작성됩니다.

분배 목록은 큐 목록을 포함하는 특수 유형의 큐 오브젝트입니다. 메시지를 넣기 위해 열 수 있으나 메시지를 가져오거나 찾아보는 목적 또는 속성을 조회하거나 설정하는 목적으로는 열 수 없습니다. 자세한 정보는 사용 시 참고사항 8을 참조하십시오.

QSGDISP(GROUP)가 있는 큐는 MQOPEN 또는 MQPUT1 호출에 사용할 수 없는 특수 유형의 큐 정의입니다.

- 다음 목적으로 사용되는 이름 목록:
  - 목록에 있는 큐의 이름 조회(MQINQ 호출 사용).
- 다음 목적으로 사용되는 프로세스 정의:
  - 프로세스 속성 조회(MQINQ 호출 사용).
- 다음 목적으로 사용되는 큐 관리자:
  - 로컬 큐 관리자의 속성 조회(MQINQ 호출 사용).

2. 애플리케이션이 동일한 오브젝트를 두 번 이상 열 수 있습니다. 각각 열 때마다 다른 오브젝트 핸들이 리턴됩니다. 리턴되는 각 핸들은 해당 열기가 수행된 함수에 사용될 수 있습니다.

3. 열리는 오브젝트가 클러스터 큐 이외의 큐인 경우, MQOPEN 호출 시 로컬 큐 관리자 내에 있는 모든 이름이 해석됩니다. 특정 MQOPEN 호출의 경우 여기에 다음 중 하나 이상이 포함될 수 있습니다.

- 기본 큐의 이름으로 알리어스 해석
- 리모트 큐 로컬 정의의 이름을 리모트 큐 관리자 이름 및 리모트 큐 관리자에 큐가 인식되는 이름으로 해석

- 리모트 큐 관리자 이름을 로컬 전송 큐의 이름으로 해석

그러나, 핸들에 대한 후속 MQINQ 또는 MQSET 호출은 열린 이름에만 관련이 있고 이름 해석 이후 발생한 오브젝트에는 관련이 없습니다. 예를 들어, 열린 오브젝트가 알리어스이면 MQINQ 호출이 리턴한 속성은 알리어스가 해석되는 기본 큐의 속성이 아니라 알리어스의 속성입니다. 하지만 해당 MQOPEN에서 **OPTS** 매개변수에 무엇이 지정되든 이름 해석 검사가 여전히 수행됩니다.

열리는 오브젝트가 클러스터 큐이면 이름 해석이 MQOPEN 호출 시 발생하거나 나중에 지연될 수 있습니다. 해석이 발생하는 시점은 MQOPEN 호출에서 지정된 OOBND\* 옵션에 의해 제어됩니다.

- OOBND0
- OOBNDN
- OOBNDQ

클러스터 큐의 이름 해석에 대한 자세한 정보는 [이름 해석](#)의 내용을 참조하십시오.

4. 오브젝트의 속성은 애플리케이션이 오브젝트를 여는 동안 변경될 수 있습니다. 대부분의 경우, 애플리케이션에서 이를 표시하지 않지만 특정 속성은 큐 관리자가 핸들을 더 이상 올바르게 읽은 것으로 표시합니다. 이러한 항목은 다음과 같습니다.

- 오브젝트의 이름 해석에 영향을 미치는 모든 속성. 이 속성은 사용되는 열기 옵션에 상관없이 적용되며 다음을 포함합니다.

- 열린 알리어스 큐의 **BaseQName** 속성 변경사항
- **RemoteQName** 또는 **RemoteQMgrName** 큐 속성의 변경사항. 이 큐 또는 이 정의를 통해 큐 관리자 알리어스로 해석되는 큐에 열린 모든 핸들이 대상입니다.
- 현재 리모트 큐에 대해 열린 핸들이 다른 전송 큐로 해석되거나 해석되지 못하도록 하는 변경사항. 예를 들어, 다음과 같습니다.
  - 정의가 큐 또는 큐 관리자 알리어스에 사용되는지 여부에 상관없이 리모트 큐 로컬 정의의 **XmitQName** 속성 변경사항.

한 가지 예외가 있는데, 바로 새 전송 큐를 작성하는 경우입니다. 핸들을 열 때 핸들이 있으면 이 큐로 해석되어야 하지만 기본 전송 큐로 해석된 핸들은 올바르게 읽은 것으로 간주되지 않습니다.

- **DefXmitQName** 큐 관리자 속성 변경사항. 이 경우, 이전에 명명된 큐로 해석된 모든 열기 핸들(기본 전송 큐란 이유만으로 해당 큐로 해석됨)이 올바르게 읽은 것으로 표시됩니다. 그 외 이유로 이 큐로 해석된 핸들은 영향을 받지 않습니다.
- 현재 이 큐 또는 이 큐로 해석되는 큐에 대해 OOINPS 액세스를 제공하는 핸들이 두 개 이상 있는 경우, **Shareability** 큐 속성. 해당되는 경우 이 큐 또는 이 큐에 대해 해석하는 큐에 대해 열려 있는 모든 핸들이 열기 옵션에 상관없이 올바르게 읽은 것으로 표시됩니다.
- 열기 옵션에 상관없이, 이 큐 또는 이 큐로 해석되는 큐에 열려 있는 모든 핸들에 대한 **Usage** 큐 속성.

핸들이 올바르게 읽은 것으로 표시되면 이 핸들을 사용하는 이후의 모든 호출(MQCLOSE 제외)은 이유 코드 RC2041이 표시되면서 실패합니다. 애플리케이션이 (원본 핸들을 사용하여) MQCLOSE 호출을 실행한 다음 큐를 다시 열어야 합니다. 이전의 성공적인 호출에서 이전 핸들에 대해 커밋되지 않은 업데이트는 애플리케이션 논리에 따라 여전히 커밋되거나 백아웃된 상태일 수 있습니다.

속성을 변경하면 이런 현상이 발생하는 경우, "강제 실행"되는 특별한 형태의 명령을 사용해야 합니다.

5. MQOPEN 호출 발행 시 액세스 허용 전에 애플리케이션이 실행 중인 사용자 ID에 적절한 레벨의 권한이 있는지 확인하기 위해 큐 관리자가 보안 검사를 수행합니다. 권한 검사는 이름이 해석된 후에 발생하는 이름이 아니라 열리는 오브젝트의 이름에서 수행됩니다.

열리는 오브젝트가 모델 큐인 경우, 큐 관리자가 모델 큐의 이름 및 작성되는 동적 큐의 이름 둘 다에 대해 전체 보안 검사를 수행합니다. 그럼 다음, 결과 동적 큐를 명시적으로 열면 동적 큐의 이름에 대해 추가적인 자원 보안 검사가 수행됩니다.

6. 리모트 큐는 이 호출의 **OBJDSC** 매개변수에서 두 가지 방법 중 하나를 사용하여 지정할 수 있습니다(1109 페이지의 『IBM i의 MQOD(오브젝트 디스크립터)』에 설명된 **ODON** 및 **ODMN** 필드 참조).

- **ODON**에 리모트 큐의 로컬 정의 이름을 지정합니다. 이 경우, **ODMN**은 로컬 큐 관리자를 참조하며 공백 또는 공백으로 지정할 수 있습니다.

로컬 큐 관리자에 의해 수행되는 보안 유효성 검증은 사용자가 리모트 큐의 로컬 정의를 열 수 있는 권한이 있는지 확인합니다.

- *ODON*에 리모트 큐 관리자에 인식되는 리모트 큐의 이름을 지정합니다. 이 경우, *ODMN*은 리모트 큐 관리자의 이름입니다.

로컬 큐 관리자에 의해 수행되는 보안 유효성 검증은 사용자가 이름 해석 프로세스에서 발생하는 전송 큐에 메시지를 송신할 수 있는 권한이 있는지 확인합니다.

두 경우 모두 다음 사항이 적용됩니다.

- 사용자가 큐에 메시지를 넣을 수 있는 권한이 있는지 확인하기 위해 로컬 큐 관리자에 의해 리모트 큐 관리자에 메시지가 송신되지 않습니다.
- 메시지가 리모트 큐 관리자에 도착할 때 메시지를 생성한 사용자에게 권한이 없으므로 리모트 큐 관리자가 이를 거부할 수 있습니다.

7. 오브젝트 핸들 및 찾아보기 옵션 중 하나를 지정하는 *MQGET* 호출과 함께 사용하기 위해 *OOBRW* 옵션이 있는 *MQOPEN* 호출이 찾아보기 커서를 설정합니다. 그러면 큐의 콘텐츠를 대체하지 않고 스캔할 수 있습니다. *GMMUC* 옵션을 사용하여 찾아보기를 통해 발견한 메시지를 큐에서 제거할 수 있습니다.

동일한 큐에 여러 개의 *MQOPEN* 요청을 발행하면 단일 애플리케이션을 상대로 여러 찾아보기 커서가 활성화될 수 있습니다.

8. 다음은 분배 목록의 사용에 적용되는 참고입니다.

- *MQOD* 구조 내의 필드는 분배 목록을 열 때 다음과 같이 설정되어야 합니다.

- *ODVER*는 *ODVER2* 이상이어야 합니다.
- *ODOT*는 *OTQ*이어야 합니다.
- *ODON*은 공백 또는 널 문자열이어야 합니다.
- *ODMN*은 공백 또는 널 문자열이어야 합니다.
- *ODREC*는 0보다 커야 합니다.
- *ODORO* 및 *ODORP* 중 하나는 0이고 다른 하나는 0이 아니어야 합니다.
- *ODRRO* 및 *ODRRP* 중 하나만 0이 아닐 수 있습니다.
- *ODORO* 또는 *ODORP*에 의해 처리되는 *ODREC* 오브젝트 레코드가 있어야 합니다. 오브젝트 레코드는 열리는 목적지 큐의 이름으로 설정되어야 합니다.
- *ODRRO* and *ODRRP* 중 하나가 0이 아니면 *ODREC* 응답 레코드가 있어야 합니다. 이유 코드 *RC2136*이 표시되면서 호출이 완료되면 응답 레코드가 큐 관리자에 의해 설정됩니다.

*ODREC*가 0인지 확인하여 분배 목록에 없는 단일 큐를 여는 데 버전 2 *MQOD*를 사용할 수도 있습니다.

- **OPTS** 매개변수에서는 다음 열기 옵션만 유효합니다.

- *OOOUT*
- *OOPAS\**
- *OOSSET\**
- *OOALTU*
- *OOFIQ*

- 분배 목록의 목적지 큐는 로컬, 알리어스 또는 리모트 큐가 될 수 있으나 모델 큐는 될 수 없습니다. 모델 큐가 지정된 경우 이유 코드 *RC2057*이 표시되면서 해당 큐를 열지 못합니다. 그러나 이것 때문에 목록의 다른 큐를 열지 못하는 것은 아닙니다.

- 완료 코드 및 이유 코드 매개변수는 다음과 같이 설정됩니다.

- 분배 목록에 있는 큐에 대한 열기 조작이 같은 방법으로 모두 성공하거나 실패하면 완료 코드 및 이유 코드 매개변수가 공용 결과를 설명하도록 설정됩니다. *MQRR* 응답 레코드(애플리케이션에서 제공하는 경우)는 이 경우에 설정되지 않습니다.

예를 들어, 모든 열기가 성공하면 완료 코드가 *CCOK*로 설정되고 이유 코드가 *RCNONE*입니다. 존재하는 큐가 없어 모든 열기가 실패하면 매개변수가 *CCFAIL* 및 *RC2085*로 설정됩니다.

- 분배 목록에 있는 큐에 대한 열기 조작이 모두 같은 방법으로 성공하거나 실패하지 않는 경우
    - 최소한 하나의 열기가 성공하면 완료 코드 매개변수가 CCWARN으로 설정되고 모두 실패하면 CCFAIL로 설정됩니다.
    - 이유 코드 매개변수는 RC2136으로 설정됩니다.
    - 응답 레코드(애플리케이션에 의해 제공되는 경우)가 분배 목록 내의 큐에 대해 개별 완료 코드 및 이유 코드로 설정됩니다.
  - 분배 목록이 성공적으로 열린 경우, 호출에 의해 리턴되는 HOBJ 핸들을 후속 MQPUT 호출에 사용하여 분배 목록에 있는 큐에 메시지를 넣을 수 있으며 MQCLOSE 호출에 사용하여 분배 목록에 대한 액세스를 철회할 수 있습니다. 분배 목록에 올바른 단 하나의 닫기 옵션은 CONONE입니다.
 

분배 목록에 메시지를 넣는 데 MQPUT1 호출을 사용할 수도 있습니다. 목록에서 큐를 정의하는 MQOD 구조가 해당 호출에서 매개변수로 지정됩니다.
  - 애플리케이션이 허용되는 최대 핸들 수를 초과했는지 검사할 때 성공적으로 열린 분배 목록 내의 각 목적지는 별도의 별도의 핸들로 계수됩니다(MaxHandles 큐 관리자 속성 참조). 분배 목록에 있는 목적지 중 둘 이상이 동일한 물리적 큐로 해석된 경우에도 마찬가지입니다. 분배 목록에 대한 MQOPEN 또는 MQPUT1 호출로 인해 애플리케이션에 사용 중인 핸들의 수가 MaxHandles를 초과하는 경우, 이유 코드 RC2017이 표시되면서 호출이 실패합니다.
  - 목적지가 성공적으로 열릴 때마다 OpenOutputCount 속성의 값이 1씩 증가합니다. 분배 목록에 있는 목적지 중 둘 이상이 동일한 물리적 큐로 해석되는 경우, 큐의 OpenOutputCount 속성이 해당 큐로 해석된 분배 목록에 있는 목적지의 수만큼 증가합니다.
  - 핸들을 올바르게 않게 만드는 큐 정의의 변경사항으로 인해 큐가 개별적으로 열린 경우(예: 해석 경로 변경), 분배 목록 핸들이 올바르게 않게 되지 않습니다. 그러나 분배 목록 핸들이 후속 MQPUT 호출에서 사용될 때 이는 특정 큐의 실패를 유발합니다.
  - 분배 목록은 단 하나의 목적지만 포함할 수 있습니다.
9. 다음은 클러스터 큐 사용에 적용되는 참고입니다.
- 클러스터 큐가 처음으로 열리고 로컬 큐 관리자가 전체 저장소 큐 관리자가 아닌 경우, 로컬 큐 관리자는 전체 저장소 큐 관리자에서 클러스터 큐에 대한 정보를 가져옵니다. 네트워크가 사용 중인 경우, 로컬 큐 관리자가 저장소 큐 관리자로부터 필요한 정보를 수신하는 데 몇 초 정도 걸릴 수 있습니다. 결과적으로 MQOPEN 호출을 발행하는 애플리케이션은 MQOPEN 호출로부터의 리턴을 제어하기 전에 최대 10초까지 대기해야 할 수도 있습니다. 로컬 큐 관리자가 이 시간 안에 클러스터 큐에 대해 필요한 정보를 수신하지 못하면 이유 코드 RC2189가 표시되면서 호출이 실패합니다.
  - 클러스터 큐가 열리고 클러스터 내에 큐 인스턴스가 여러 개 있는 경우, 실제 열리는 인스턴스는 MQOPEN 호출에서 지정된 옵션에 따라 결정됩니다.
    - 옵션을 지정하는 경우, 다음을 포함하십시오.
      - OOBWRW
      - OOINPQ
      - OOINPX
      - OOINPS
      - OOSSET

열린 클러스터 큐의 인스턴스는 로컬 인스턴스여야 합니다. 큐의 로컬 인스턴스가 없으면 MQOPEN 호출이 실패합니다.
    - 지정된 옵션에 위에 설명한 옵션이 하나도 포함되지 않은 경우에는 다음 중 하나 또는 둘 다를 포함하십시오.
      - OOINQ
      - OOOUT

해당 항목이 있으면 열린 인스턴스가 로컬 인스턴스이고, 그렇지 않으면 리모트 인스턴스입니다. 그러나 클러스터 워크로드 엑시트를 통해 큐 관리자가 선택한 인스턴스를 대체할 수 있습니다(해당 항목이 있는 경우).

클러스터 큐에 대한 자세한 정보는 [클러스터 큐](#)를 참조하십시오.

10. 트리거 모니터에 의해 시작된 애플리케이션은 애플리케이션이 시작될 때 애플리케이션과 연관된 큐의 이름이 전달됩니다. 이 큐 이름은 큐를 열기 위해 **OBJDSC** 매개변수에 지정할 수 있습니다. 자세한 정보는 MQTMC 구조에 대한 설명을 참조하십시오.
11. OORLOQ 옵션을 사용하는 경우, 로컬, 알리어스 또는 모델 큐를 열 때 로컬 큐가 이미 리턴되어 있지만, 리모트 큐 또는 로컬이 아닌 클러스터 큐를 열 때는 그렇지 않습니다. ResolvedQName 및 ResolvedQMgrName 이 리모트 큐 정의에 있는 RemoteQName 및 RemoteQMgrName과 함께 입력되거나, 선택한 리모트 클러스터 큐에서도 마찬가지로입니다. 예를 들어, 열 때 OORLOQ를 지정하면 리모트 큐 ResolvedQName이 메시지를 넣는 전송 큐가 됩니다. ResolvedQMgrName에는 전송 큐를 호스팅하는 로컬 큐 관리자의 이름이 입력됩니다. 사용자에게 큐에 대한 찾아보기, 입력 또는 출력 권한이 있으면 이 플래그를 MQOPEN 호출에 지정하기 위한 필수 권한도 있습니다. 특별한 권한이 필요하지는 않습니다.

## 매개변수

MQOPEN 호출의 매개변수는 다음과 같습니다.

### HCONN(10자리 부호 있는 정수) - 입력

연결 핸들입니다.

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. HCONN의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

### OBJDSC (MQOD) - 입출력(I/O)

오브젝트 디스크립터.

이는 열리는 오브젝트를 식별하는 구조입니다. 세부사항은 [1109 페이지의 『IBM i의 MQOD\(오브젝트 디스크립터\)』](#)의 내용을 참조하십시오.

**OBJDSC** 매개변수의 *ODON* 필드가 모델 큐의 이름인 경우 동적 로컬 큐는 모델 큐의 속성으로 작성됩니다. 이 동작은 **OPTS** 매개변수에 지정된 열기 옵션에 상관없이 발생합니다. MQOPEN 호출에 의해 리턴되는 *HOBJ*를 사용하는 후속 조작은 모델 큐가 아니라 새 동적 큐에서 수행됩니다. 이는 MQINQ 및 MQSET 호출에 대해서도 마찬가지입니다. **OBJDSC** 매개변수에서 모델 큐의 이름이 작성되는 동적 큐의 이름으로 대체됩니다. 동적 큐의 유형은 모델 큐의 **DefinitionType** 속성 값에 따라 판별됩니다([1305 페이지의 『큐의 속성』](#) 참조). 동적 큐에 적용할 수 있는 닫기 옵션에 대한 정보는 MQCLOSE 호출의 설명을 참조하십시오.

### OPTS(10자리 부호 있는 정수) - 입력

MQOPEN의 조치를 제어하는 옵션.

다음 옵션 중 하나 이상을 지정해야 합니다.

- OOBW
- OOINP\*(다음 중 하나만 해당됨)
- OOINQ
- OOUT
- OSET
- OORLQ

필요에 따라 다른 옵션을 지정할 수 있습니다. 둘 이상의 옵션이 필요하면 값을 추가할 수 있습니다(동일한 상수를 두 번 이상 추가하지 않음). 올바르지 않은 조합에 대해 설명되어 있습니다. 그 밖의 모든 조합은 올바릅니다. **OBJDSC**에 지정된 오브젝트 유형에 적용 가능한 옵션만 사용할 수 있습니다([각 큐 유형의 올바른 MQOPEN 옵션 참조](#)).

**액세스 옵션:** 다음 옵션은 오브젝트에서 수행할 수 있는 조작 유형을 제어합니다.

#### OOINPQ

큐가 정의한 디폴트를 사용하여 메시지를 가져오기 위해 큐를 엽니다.

해당 큐가 후속 MQSET 호출에 사용하도록 열립니다. 액세스 유형은 **DefInputOpenOption** 큐 속성 값에 따라 공유 또는 독점입니다. 자세한 정보는 [1305 페이지의 『큐의 속성』](#)의 내용을 참조하십시오.



이 옵션은 로컬, 알리어스 및 모델 큐에서만 올바르며, 리모트 큐, 분배 목록과 큐에 없는 오브젝트에는 올바르지 않습니다.

### OOINPS

공유 액세스를 메시지로 가져오기 위해 큐를 엽니다.

해당 큐가 후속 MQSET 호출에 사용하도록 열립니다. 현재 이 애플리케이션 또는 다른 애플리케이션에 의해 OOIINPS를 사용하여 큐가 열려 있으면 호출이 성공합니다. 그러나 큐가 현재 OOINPX를 사용하여 열려 있으면 이유 코드 RC2042가 표시되면서 호출이 실패합니다.

이 옵션은 로컬, 알리어스 및 모델 큐에서만 올바르며, 리모트 큐, 분배 목록과 큐에 없는 오브젝트에는 올바르지 않습니다.

### OOINPX

배타적 액세스를 메시지로 가져오기 위해 큐를 엽니다.

해당 큐가 후속 MQSET 호출에 사용하도록 열립니다. 임의 유형의 입력(OOINPS 또는 OOINPX)을 대상으로 현재 이 애플리케이션 또는 다른 애플리케이션에 의해 큐가 열려 있으면 이유 코드 RC2042가 표시되면서 호출이 실패합니다.

이 옵션은 로컬, 알리어스 및 모델 큐에서만 올바르며, 리모트 큐, 분배 목록과 큐에 없는 오브젝트에는 올바르지 않습니다.

다음은 이러한 옵션에 적용되는 참고입니다.

- 이러한 옵션 중 하나만 지정될 수 있습니다.
- **InhibitGet** 큐 속성이 QAGETI로 설정된 경우, 속성이 이 값으로 설정된 동안 후속 MQGET 호출이 실패하더라도 이러한 옵션 중 하나를 사용한 MQOPEN 호출은 성공할 수 있습니다.
- 큐가 공유할 수 없는 것으로 정의된 경우(즉, **Shareability** 큐 속성 값이 QANSHR임), 공유 액세스를 위해 큐를 열려는 시도가 독점 액세스를 큐를 열려는 시도로 간주됩니다.
- 이러한 옵션 중 하나를 사용하여 알리어스 큐가 열린 경우, 알리어스 큐가 해석되는 기본 큐에 대해 독점 사용(또는 다른 애플리케이션의 독점 사용 여부)에 대한 테스트가 수행됩니다.
- **ODMN**이 큐 관리자 알리어스의 이름인 경우 이 옵션이 올바르지 않습니다. 큐 관리자 알리어스에 사용되는 리모트 큐의 로컬 정의에서 **RemoteQMgrName** 속성 값이 로컬 큐 관리자의 이름인 경우에도 마찬가지입니다.

### OOBRW

메시지를 열람하기 위해 큐를 엽니다.

다음 옵션 중 하나를 사용하여 후속 MQGET 호출과 함께 사용하기 위해 큐가 열립니다.

- GMBRWF
- GMBRWN
- GMBRWC

큐가 현재 OOINPX에 대해 열려 있더라도 허용됩니다. OOBRW 옵션을 사용한 MQOPEN 호출은 찾아보기 커서를 설정하고 이를 큐의 첫 번째 메시지 앞에 논리적으로 배치합니다. 자세한 정보는 [1032 페이지의 『IBM의 MQGMO\(메시지 가져오기 옵션\)』](#)에 설명된 **GMOPT** 필드를 참조하십시오.

이 옵션은 로컬, 알리어스 및 모델 큐에서만 유효하고 리모트 큐, 분배 목록과 큐가 아닌 오브젝트에는 유효하지 않습니다. **ODMN**이 큐 관리자 알리어스의 이름인 경우에도 유효하지 않습니다. 큐 관리자 알리어스에 사용되는 리모트 큐의 로컬 정의에서 **RemoteQMgrName** 속성 값이 로컬 큐 관리자의 이름인 경우에도 마찬가지입니다.

### OOOUT

메시지를 발행하기 위해 메시지 넣기, 토픽 또는 토픽 문자열에 대한 큐를 엽니다.

후속 MQPUT 호출과 함께 사용하기 위해 큐가 열립니다.

**InhibitPut** 큐 속성이 QAPUTI로 설정된 경우, 속성이 이 값으로 설정된 동안 후속 MQPUT 호출이 실패하더라도 이 옵션을 사용한 MQOPEN 호출은 성공할 수 있습니다.

이 옵션은 분배 목록 및 토픽을 포함하여 모든 유형의 큐에 대해 유효합니다.

## OOINQ

속성을 조회할 오브젝트를 엽니다.

큐, 이름 목록, 프로세스 정의 또는 큐 관리자가 후속 MQINQ 호출과 함께 사용하기 위해 열립니다.

이 옵션은 분배 목록을 제외한 모든 유형의 오브젝트에 대해 유효합니다. ODMN이 큐 관리자 알리어스의 이름인 경우에는 유효하지 않습니다. 큐 관리자 알리어스에 사용되는 리모트 큐의 로컬 정의에서 **RemoteQMgrName** 속성 값이 로컬 큐 관리자의 이름인 경우에도 마찬가지입니다.

## OASET

속성을 설정하기 위해 큐를 엽니다.

후속 MQSET 호출과 함께 사용하기 위해 큐가 열립니다.

이 옵션은 분배 목록을 제외한 모든 유형의 큐에 대해 유효합니다. ODMN이 리모트 큐의 로컬 정의 이름인 경우에는 이 옵션이 유효하지 않습니다. 큐 관리자 알리어스에 사용되는 리모트 큐의 로컬 정의에서 **RemoteQMgrName** 속성 값이 로컬 큐 관리자의 이름인 경우에도 마찬가지입니다.

**바인딩 옵션:** 다음 옵션은 열리는 오브젝트가 클러스터 큐인 경우에 적용됩니다. 이러한 옵션은 클러스터 큐의 인스턴스에 대한 큐 핸들 바인딩을 제어합니다.

## OQBNDQ

큐를 열 때 핸들을 목적지에 바인딩합니다.

이로 인해 로컬 큐 관리자가 큐를 열 때 목적지 큐의 인스턴스에 큐 핸들을 바인딩하게 됩니다. 따라서 이 핸들을 사용하여 넣는 모든 메시지가 동일한 라우트로 목적지 큐의 동일한 인스턴스에 전송됩니다.

이 옵션은 큐에 대해서만 유효하며 클러스터 큐에만 영향을 미칩니다. 클러스터 큐가 아닌 큐에 대해 지정된 경우에는 옵션이 무시됩니다.

## OQBNDN

특정 목적지에 바인딩하지 않습니다.

로컬 큐 관리자가 큐 핸들을 목적지 큐의 인스턴스에 바인딩하는 것을 중지합니다. 따라서 이 핸들을 사용하는 후속 MQPUT 호출이 목적지 큐의 다른 인스턴스에 메시지를 송신하거나 다른 라우트로 동일한 인스턴스에 메시지를 송신합니다. 네트워크 조건에 따라 로컬 큐 관리자, 리모트 큐 관리자 또는 메시지 채널 에이전트(MCA)에 의해 인스턴스가 나중에 변경되도록 선택할 수도 있습니다.

**참고:** 트랜잭션을 완료하기 위해 메시지 시리즈를 교환해야 하는 클라이언트 및 서버 애플리케이션은 OQBNDN(또는 *DefBind*의 값이 BNDNOT인 경우에는 OQBNDQ)을 사용하지 않아야 합니다. 이후 메시지가 연속적으로 서버 애플리케이션의 다른 인스턴스로 송신될 수 있기 때문입니다.

클러스터 큐에 대해 OQBNDN 또는 OOINP\* 옵션 중 하나가 지정된 경우에는 큐 관리자가 강제로 클러스터 큐의 로컬 인스턴스를 선택하도록 설정됩니다. 결과적으로 OQBNDN이 지정되었더라도 큐 핸들의 바인딩이 고정됩니다.

OOINQ가 OQBNDN과 함께 지정된 경우, 일반적으로 모든 인스턴스의 속성 값이 동일하더라도 핸들을 사용하는 연속 MQINQ 호출이 클러스터 큐의 다른 인스턴스를 조회할 수 있습니다.

OQBNDN은 큐에 대해서만 유효하고 클러스터 큐에만 영향을 미칩니다. 클러스터 큐가 아닌 큐에 대해 지정된 경우에는 옵션이 무시됩니다.

## OQBNDQ

큐의 기본 바인딩을 사용합니다.

이로 인해 로컬 큐 관리자가 **DefBind** 큐 속성에 정의된 방식으로 큐 핸들을 바인딩합니다. 이 속성의 값은 BNDOPN 또는 BNDNOT입니다.

OQBNDQ 및 OQBNDN이 지정되지 않으면 OQBNDQ가 기본값입니다.

OQBNDQ는 프로그램 문서를 보조하기 위해 정의됩니다. 이 옵션은 기타 두 바인드 옵션 중 하나와 함께 사용하기 위한 용도가 아니며 값이 0이므로 이러한 사용을 감지할 수 없습니다.

**컨텍스트 옵션:** 다음 옵션은 메시지 컨텍스트의 처리를 제어합니다.

## OOSAVA

메시지가 검색되면 컨텍스트를 저장합니다.



컨텍스트 정보는 이 큐 핸들과 연관됩니다. 이 정보는 이 핸들을 사용하여 검색된 메시지의 컨텍스트에서 설정됩니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어를 참조하십시오](#).

이 컨텍스트 정보는 MQPUT 또는 MQPUT1 호출을 사용하여 나중에 큐에 넣을 메시지에 전달될 수 있습니다. 1122 페이지의 『IBM i의 MQPMO(메시지 넣기 옵션)』에 설명된 PMPASI 및 PMPASA 옵션을 참조하십시오.

메시지가 성공적으로 검색될 때까지 큐에 넣는 메시지에 컨텍스트를 전달할 수 없습니다.

GMBRW\* 찾아보기 옵션 중 하나를 사용하여 검색된 메시지는 **MSGDSC** 매개변수의 컨텍스트 필드가 찾아보기 후에 설정된 경우에도 컨텍스트 정보를 저장하지 않습니다.

이 옵션은 로컬, 알리어스 및 모델 큐에서만 유효하고 리모트 큐, 분배 목록과 큐가 아닌 오브젝트에는 유효하지 않습니다. OOINP\* 옵션 중 하나를 지정해야 합니다.

## OOPASI

ID 컨텍스트의 전달을 허용합니다.

메시지를 큐에 넣을 때 PMPASI 옵션을 **PMO** 매개변수에 지정할 수 있도록 합니다. 따라서 OOSAVA 옵션을 사용하여 열린 입력 큐의 ID 컨텍스트 정보를 메시지에 제공할 수 있습니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어를 참조하십시오](#).

OOOUT 옵션을 지정해야 합니다.

이 옵션은 분배 목록을 포함하여 모든 유형의 큐에 유효합니다.

## OOPASA

모든 컨텍스트의 전달을 허용합니다.

메시지를 큐에 넣을 때 PMPASA 옵션을 **PMO** 매개변수에 지정할 수 있도록 합니다. 따라서 OOSAVA 옵션을 사용하여 열린 입력 큐의 ID 및 원본 컨텍스트 정보를 메시지에 제공할 수 있습니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어를 참조하십시오](#).

이 옵션은 OOPASI를 의미하므로 지정할 필요가 없습니다. OOOUT 옵션을 지정해야 합니다.

이 옵션은 분배 목록을 포함하여 모든 유형의 큐에 유효합니다.

## OOSSETI

ID 컨텍스트의 설정을 허용합니다.

메시지를 큐에 넣을 때 PMSETI 옵션을 **PMO** 매개변수에 지정할 수 있도록 합니다. 따라서 MQPUT 또는 MQPUT1 호출에서 지정된 **MSGDSC** 매개변수에 들어 있는 ID 컨텍스트 정보를 메시지에 제공합니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어를 참조하십시오](#).

이 옵션은 OOPASI를 의미하므로 지정할 필요가 없습니다. OOOUT 옵션을 지정해야 합니다.

이 옵션은 분배 목록을 포함하여 모든 유형의 큐에 유효합니다.

## OOSSETA

모든 컨텍스트의 설정을 허용합니다.

메시지를 큐에 넣을 때 PMSETA 옵션을 **PMO** 매개변수에 지정할 수 있도록 합니다. 따라서 MQPUT 또는 MQPUT1 호출에서 지정된 **MSGDSC** 매개변수에 들어 있는 ID 및 원본 컨텍스트 정보를 메시지에 제공합니다. 메시지 컨텍스트에 대한 자세한 정보는 [메시지 컨텍스트 및 컨텍스트 정보 제어를 참조하십시오](#).

이 옵션은 다음 옵션을 의미하므로 지정할 필요가 없습니다.

- OOPASI
- OOPASA
- OOSSETI

OOOUT 옵션을 지정해야 합니다.

이 옵션은 분배 목록을 포함하여 모든 유형의 큐에 유효합니다.

**기타 옵션:** 다음 옵션은 권한 검사 및 큐 관리자가 정지 중일 때 발생하는 현상을 제어합니다.

**OOALTU**

지정된 사용자 ID로 유효화합니다.

이는 **OBJDSC** 매개변수의 **ODAU** 필드에 이 MQOPEN 호출을 유효성 검증하는 데 사용할 사용자 ID가 포함되어 있음을 표시합니다. 애플리케이션을 실행 중인 사용자 ID가 오브젝트를 열 수 있는 권한이 있는 지에 상관없이 이 **ODAU**가 지정된 액세스 옵션을 포함한 오브젝트를 열 수 있는 권한이 있는 경우에만 호출이 성공합니다. 이는 지정된 어떠한 컨텍스트 옵션에도 적용되지 않지만 애플리케이션이 실행 중인 사용자 ID에 대해서는 항상 검사가 수행됩니다.

이 옵션은 모든 유형의 오브젝트에 대해 유효합니다.

**OOFIQ**

큐 관리자가 정지 중인 경우 실패합니다.

이 옵션은 큐 관리자가 정지 상태인 경우 MQOPEN 호출이 실패하도록 강제합니다.

이 옵션은 모든 유형의 오브젝트에 대해 유효합니다.

**OORLQ**

열린 로컬 큐의 이름을 입력합니다.

이 옵션은 MQOD 구조에서 ResolvedQName(사용 가능한 경우)에 열린 로컬 큐의 이름이 입력되어야 하도록 지정합니다. 이와 유사하게 ResolvedQMgrName에는 로컬 큐를 호스팅하는 로컬 큐 관리자의 이름이 입력됩니다.

표 750. 큐 유형별 올바른 MQOPEN 옵션						
옵션	알리어스 ( <a href="#">1271 페이지</a> 의 『1』)	로컬 및 모델	원격	로컬이 아닌 클러스터	분배 목록	토픽
OOINPQ	✓	✓	-	-	-	-
OOINPS	✓	✓	-	-	-	-
OOINPX	✓	✓	-	-	-	-
OOBRW	✓	✓	-	-	-	-
OOOUT	✓	✓	✓	✓	✓	✓
OOINQ	✓	✓	<a href="#">1271 페이지</a> 의 『2』	✓	-	-
OOSSET	✓	✓	<a href="#">1271 페이지</a> 의 『2』	-	-	-
OOBNDQ ( <a href="#">1271 페이지</a> 의 『3』)	✓	✓	✓	✓	✓	-
OOBNDN ( <a href="#">1271 페이지</a> 의 『3』)	✓	✓	✓	✓	✓	-
OOBNDQ ( <a href="#">1271 페이지</a> 의 『3』)	✓	✓	✓	✓	✓	-
OOSAVA	✓	✓	-	-	-	-
OOPASI	✓	✓	✓	✓	✓	<a href="#">1271 페이지</a> 의 『5』
OOPASA	✓	✓	✓	✓	✓	<a href="#">1271 페이지</a> 의 『5』

표 750. 큐 유형별 올바른 MQOPEN 옵션 (계속)						
옵션	알리어스 (1271 페이지 의 『1』)	로컬 및 모델	원격	로컬이 아닌 클러스터	분배 목록	토픽
OOSSETI	✓	✓	✓	✓	✓	1271 페이지 의 『5』
OOSSETA	✓	✓	✓	✓	✓	1271 페이지 의 『5』
OOSALTU	✓	✓	✓	✓	✓	✓
OOSFIQ	✓	✓	✓	✓	✓	✓
OOSRLQ	✓	✓	✓	✓	-	-

**참고:**

1. 알리어스에 대한 옵션 검증은 알리어스가 해석되는 큐에 대한 옵션의 검증에 따라 결정됩니다.
2. 이 옵션은 리모트 큐의 로컬 정의에 대해서만 유효합니다.
3. 이 옵션은 모든 큐 유형에 대해 지정할 수 있지만 큐가 클러스터 큐가 아니면 무시됩니다.
4. 이 속성은 토픽에 대해 무시됩니다.
5. 이러한 속성은 토픽과 함께 사용될 수 있으나 모든 구독자에게 송신된 컨텍스트 필드가 아니라 보유한 메시지에 대한 컨텍스트 세트에만 영향을 미칩니다.

**HOBJS(10자리 부호 있는 정수) - 출력**

오브젝트 핸들.

이 핸들은 오브젝트에 설정된 액세스를 나타냅니다. 오브젝트에서 작동하는 후속 메시지 큐잉 호출에 지정되어야 합니다. 이는 MQCLOSE 호출이 발행되거나 핸들의 범위를 정의하는 처리 단위가 종료될 때 유효성이 중단됩니다.

핸들의 범위가 애플리케이션이 실행 중인 플랫폼에 지원되는 가장 작은 병렬 처리 단위로 제한됩니다. MQOPEN 호출을 발행한 병렬 처리 단위 외부에서는 핸들이 유효하지 않습니다.

- IBM i에서 핸들의 범위는 호출을 발행하는 작업입니다.

**CMPCOD(10자리 부호 있는 정수) - 출력**

완료 코드.

다음 중 하나입니다.

**CCOK**

정상적으로 완료되었습니다.

**CCWARN**

경고(일부 완료).

**CCFAIL**

호출에 실패했습니다.

**RPG 선언**

```
C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQOPEN(HCONN : OBJDSC : OPTS :
C                               HOBJS : CMPCOD : REASON)
```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```

D*..1.....2.....3.....4.....5.....6.....7..
DMQOPEN          PR          EXTPROC('MQOPEN')
D* Connection handle
D HCONN          10I 0 VALUE
D* Object descriptor
D OBJDSC          468A
D* Options that control the action of MQOPEN
D OPTS          10I 0 VALUE
D* Object handle
D HOBJ          10I 0
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CMPCOD
D REASON          10I 0

```

## IBM i IBM i의 MQPUT(메시지 넣기)

MQPUT 호출은 큐, 분배 목록 또는 토픽에 메시지를 넣습니다. 큐, 분배 목록 또는 토픽은 이미 열려 있는 상태여야 합니다.

- [1272 페이지의 『구문』](#)
- [1272 페이지의 『사용법 참고』](#)
  - [1272 페이지의 『토픽』](#)
  - [1273 페이지의 『MQPUT 및 MQPUT1』](#)
  - [1273 페이지의 『목적지 큐』](#)
  - [1274 페이지의 『분배 목록』](#)
  - [1275 페이지의 『헤더』](#)
  - [1275 페이지의 『버퍼』](#)
- [1276 페이지의 『매개변수』](#)
- [1280 페이지의 『RPG 선언』](#)

### 구문

MQPUT (*HCONN*, *HOBJ*, *MSGDSC*, *PMO*, *BUFLEN*, *BUFFER*, *CMPCOD*, *REASON*)

### 사용법 참고

#### 토픽

다음 참고사항이 토픽 사용에 적용됩니다.

1. MQPUT를 사용하여 메시지를 토픽에 발행할 때 구독자 큐의 문제(예: 가득 참) 때문에 해당 토픽에 대한 하나 이상의 구독자에게 발행물을 제공할 수 없으면 MQPUT 호출에 리턴되는 이유 코드와 전달 동작은 TOPIC의 PMSGDLV 또는 NPMSGDLV 속성 설정에 따라 다릅니다. RODLQ를 지정한 경우 데드-레터 큐에 발행물이 전달되는 점에 유의하십시오. 또는 RODISC를 지정하면 메시지가 제거되고 이는 성공적인 메시지 전달로 간주됩니다. 전달되는 발행물이 없으면 MQPUT이 RC2502와 함께 리턴됩니다. 이러한 상태는 다음의 경우에 발생할 수 있습니다.
  - (메시지 지속성에 따라) PMSGDLV 또는 NPMSGDLV를 ALL로 설정한 상태에서 메시지가 TOPIC에 발행되고, 구독(지속 또는 비지속)에 발행물을 수신할 수 없는 큐가 있습니다.
  - (메시지 지속성에 따라) PMSGDLV 또는 NPMSGDLV를 ALLDUR로 설정한 상태에서 메시지가 TOPIC에 발행되고, 지속 구독에 발행물을 수신할 수 없는 큐가 있습니다.

다음과 같은 경우 발행물을 일부 구독자에게 전달할 수 없더라도 MQPUT가 RCNONE과 함께 리턴될 수 있습니다.

- (메시지 지속성에 따라) PMSGDLV 또는 NPMSGDLV를 ALLAVAIL로 설정한 상태에서 메시지가 TOPIC에 발행되고, 구독(지속 또는 비지속)에 발행물을 수신할 수 없는 큐가 있습니다.
  - (메시지 지속성에 따라) PMSGDLV 또는 NPMSGDLV를 ALLDUR로 설정한 상태에서 메시지가 TOPIC에 발행되고, 비지속 구독에 발행물을 수신할 수 없는 큐가 있습니다.
2. 사용 중인 토픽에 대한 구독자가 없으면 발행한 메시지가 큐에 송신되지 않고 제거됩니다. 구독자가 없는 경우 이 메시지가 지속되는지, 만기 제한이 없는지 또는 만기 시간까지 얼마 안 남았는지 여부는 아무런 차이가 없습니다. 이에 대한 예외로, 메시지가 보유되는 경우에는 메시지가 구독자의 큐로 송신되지 않더라도 새 구독 또는 MQSUBRQ를 사용하여 보유된 발행물을 요청하는 구독자에게 전달할 토픽을 위해 저장됩니다.

## MQPUT 및 MQPUT1

MQPUT 및 MQPUT1 호출을 사용하여 메시지를 큐에 넣을 수 있습니다. 사용할 호출은 환경에 따라 다릅니다.

- 여러 메시지를 동일한 큐에 넣는 경우 MQPUT 호출을 사용해야 합니다.

OOOUT 옵션을 지정하는 MQOPEN 호출을 먼저 발행한 다음 메시지를 큐에 추가하라는 하나 이상의 MQPUT 요청이 이어지고, 마지막으로 MQCLOSE 호출과 함께 큐가 닫힙니다. 이렇게 하면 MQPUT1 호출을 반복 사용하는 것보다 성능이 높아집니다.

- 단 하나의 메시지를 큐에 넣는 경우에는 MQPUT1 호출을 사용해야 합니다.

이 호출은 MQOPEN, MQPUT 및 MQCLOSE 호출을 단일 호출로 캡슐화하며, 실행되어야 하는 호출의 수를 최소화합니다.

## 목적지 큐

애플리케이션이 메시지 그룹을 사용하지 않고 동일 큐에 메시지 순서를 넣는 경우, 다음 조건을 만족하는 경우 이 메시지의 순서가 유지됩니다. 일부 조건은 로컬 및 리모트 목적지 큐 모두에 적용됩니다. 기타 조건은 리모트 목적지 큐에만 적용됩니다.

### 로컬 및 리모트 목적지 큐의 조건

- 모든 MQPUT 호출이 동일한 작업 단위 내에 있거나, 작업 단위 내에 해당 호출이 없습니다.

단일 작업 단위 내의 특정 큐에 메시지를 넣는 경우, 다른 애플리케이션의 메시지가 큐의 메시지 순서에 맞게 사이사이에 배치될 수 있습니다.

- 모든 MQPUT 호출은 동일한 오브젝트 핸들 *HOBJ*를 사용하여 작성됩니다.

일부 환경에서는 동일한 애플리케이션에서 호출하는 경우 다른 오브젝트 핸들을 사용해도 메시지 순서가 보존됩니다. "동일한 애플리케이션"의 의미는 환경에 따라 결정됩니다.

- IBM i에서는 애플리케이션이 작업입니다.

- 모든 메시지는 우선순위가 같습니다.

### 리모트 목적지 큐의 추가 조건

- 송신 큐 관리자에서 목적지 큐 관리자로 이동하는 유일한 경로입니다.

순서 대기 중인 일부 메시지가 다른 경로로 이동할 가능성이 있는 경우(예: 재구성, 트래픽 균형 조정 또는 메시지 크기에 따른 경로 선택 때문에) 목적지 큐 관리자에서 메시지의 순서를 보장할 수 없습니다.

- 메시지를 송신, 중간 또는 목적지 큐 관리자의 데드-레터 큐에 일시적으로 넣지 않습니다.

하나 이상의 메시지를 데드-레터 큐에 일시적으로 넣는 경우(예: 전송 큐 또는 목적지 큐가 일시적으로 가득 참) 메시지가 순서대로 목적지 큐에 도착하지 않을 수 있습니다.

- 메시지는 모두 지속적이거나 모두 지속적이지 않습니다.

송신 및 목적지 큐 관리자의 라우트에 있는 채널의 **CDNPM** 속성을 **NPFAST**로 설정한 경우, 비지속 메시지가 지속 메시지보다 먼저 이동하므로 비지속 메시지 대비 지속 메시지의 순서가 보존되지 않습니다. 그러나, 서로 연관된 지속 메시지의 순서와 서로 연관된 비지속 메시지의 순서는 보존됩니다.

조건을 충족하지 않는 경우 메시지 그룹을 사용하여 메시지 순서를 보존할 수 있지만, 이렇게 하려면 송신 및 수신 애플리케이션이 메시지 그룹 지원을 사용해야 합니다. 메시지 그룹에 대한 자세한 정보는 다음을 참조하십시오.

- MQMD의 MDMFL 필드
- MQPMO의 PMLOGO 옵션
- MQGMO의 GMLOGO 옵션

## 분배 목록

다음은 분배 목록의 사용에 적용되는 참고입니다.

1. 버전-1 또는 버전-2 MQPMO를 사용하여 메시지를 분배 목록에 넣을 수 있습니다. 버전-1 MQPMO(또는 PMREC가 0인 버전-2 MQPMO)를 사용하는 경우 애플리케이션에서 메시지 넣기 레코드 또는 응답 레코드가 제공되지 않습니다. 이는 메시지가 분배 목록의 일부 큐에 성공적으로 송신되고 그 외 큐에는 송신되지 않는 경우 오류가 발생하는 큐를 식별할 수 없음을 의미합니다.

애플리케이션에서 메시지 넣기 레코드 또는 응답 레코드가 제공되는 경우, PMVER 필드를 PMVER2로 설정해야 합니다.

PMREC가 0인지 확인하는 것으로 버전-2 MQPMO를 사용하여 분배 목록에 없는 단일 큐에 메시지를 송신할 수도 있습니다.

2. 완료 코드 및 이유 코드 매개변수는 다음과 같이 설정됩니다.

- 분배 목록에 있는 큐에 넣기 작업이 같은 식으로 모두 성공하거나 실패하면, 공통 결과를 설명하는 완료 코드 및 이유 코드 매개변수가 설정됩니다. MQRR 응답 레코드(애플리케이션에서 제공하는 경우)는 이 경우에 설정되지 않습니다.

예를 들어, 모든 넣기 작업이 성공하면 완료 코드가 CCOK로 설정되고 이유 코드가 RCNONE입니다. 모든 큐가 넣기 금지되어 모든 넣기 작업이 실패하면 매개변수가 CCFAIL 및 RC2051로 설정됩니다.

- 분배 목록에 있는 큐에 넣기 작업이 모두 같은 식으로 성공하거나 실패하지 않는 경우:
  - 최소한 하나의 넣기가 성공하면 완료 코드 매개변수가 CCWARN으로 설정되고 모두 실패하면 CCFAIL로 설정됩니다.
  - 이유 코드 매개변수는 RC2136으로 설정됩니다.
  - 응답 레코드(애플리케이션에 의해 제공되는 경우)가 분배 목록 내의 큐에 대해 개별 완료 코드 및 이유 코드로 설정됩니다.

목적지를 위해 열기가 실패했기 때문에 목적지에 넣기 작업이 실패하면 응답 레코드의 필드가 CCFAIL 및 RC2137로 설정되고, 해당 목적지가 PMIDC에 포함됩니다.

3. 분배 목록에 있는 목적지가 로컬 큐로 해석되면 메시지를 해당 큐에 (분배 목록 메시지가 아닌) 일반 양식으로 넣습니다. 둘 이상의 목적지가 동일한 로컬 큐로 해석되면 하나의 메시지를 이러한 각 목적지의 큐에 넣습니다.

분배 목록의 목적지가 리모트 큐로 해석되는 경우, 메시지가 적절한 전송 큐에 놓입니다. 몇 가지 목적지가 동일한 전송 큐로 해석되면 목적지가 애플리케이션에서 제공한 목적지 목록에서 인접해 있지 않더라도 해당 목적지를 포함한 단일 분배 목록 메시지가 전송 큐에 놓일 수 있습니다. 그러나, 이 동작은 전송 큐가 분배 목록 메시지를 지원하는 경우에만 수행될 수 있습니다(1305 페이지의 『큐의 속성』에 설명된 **DistLists** 큐 속성 참조).

전송 큐가 분배 목록을 지원하지 않으면 일반 양식의 메시지 사본 한 개가 해당 전송 큐를 사용하는 각 목적지의 전송 큐에 놓입니다.

애플리케이션 메시지 데이터가 있는 분배 목록이 전송 큐에 비해 너무 크면 분배 목록 메시지가 더 작은 분배 목록 메시지로 분할되고 각각은 더 적은 수의 목적지를 포함합니다. 애플리케이션 메시지 데이터가 큐에만 적합한 경우, 분배 목록 메시지를 사용할 수 없고 큐 관리자가 전송 큐를 사용하는 각 목적지에 대해 일반 양식의 메시지 사본을 한 개 생성합니다.

목적지마다 각기 다른 메시지 우선순위 또는 메시지 지속성이 적용되는 경우(애플리케이션에서 PRQDEF 또는 PEQDEF를 지정하면 발생함), 메시지가 동일한 분배 목록 메시지에 보유되지 않습니다. 대신, 다른 우선순위 및 지속성 값을 수용하는 데 필요한 수만큼 분배 목록 메시지를 큐 관리자에서 생성합니다.

4. 분배 목록에 넣으면 다음 결과로 이어집니다.

- 하나의 분배 목록 메시지, 또는

- 다수의 작은 분배 목록 메시지, 또는
- 분배 목록 메시지와 일반 메시지의 혼합, 또는
- 일반 메시지만.

앞의 항목 중 어느 것이 발생하느냐는 다음 내용의 여부에 따라 결정됩니다.

- 목록에 있는 목적지가 로컬, 리모트 또는 혼합입니다.
- 목적지의 메시지 우선순위와 메시지 지속성이 같습니다.
- 전송 큐가 분배 목록 메시지를 보유할 수 있습니다.
- 전송 큐의 최대 메시지 길이가 분배 목록 양식의 메시지를 수용할 만큼 충분한 크기입니다.

그러나 위의 사항 중 어느 것이 발생하든 상관없이 결과로 발생하는 각 실제 메시지(즉 넣기로 인해 발생한 각 일반 메시지 또는 분배 목록 메시지)는 다음의 경우 한 개의 메시지로만 계수됩니다.

- 애플리케이션이 작업 단위에서 허용되는 최대 메시지 수를 초과했는지 여부를 확인합니다 (**MaxUncommittedMsgs** 큐 관리자 속성 참조).
  - 트리거 조건을 충족하는지 여부를 확인합니다.
  - 큐 용량을 늘리고 큐의 최대 큐 용량을 초과하는지 여부를 확인합니다.
5. 핸들을 올바르게 없게 만드는 큐 정의의 변경사항으로 인해 큐가 개별적으로 열린 경우(예: 해석 경로 변경), 분배 목록 핸들이 올바르게 되지 않습니다. 그러나 분배 목록 핸들이 후속 MQPUT 호출에서 사용될 때 이는 특정 큐의 실패를 유발합니다.

## 헤더

애플리케이션 메시지 데이터의 시작 부분에 하나 이상의 IBM MQ 헤더 구조가 있는 상태에서 메시지를 넣으면 큐 관리자가 헤더 구조에 특정 검사를 수행하여 해당 구조가 올바르게 확인합니다. 큐 관리자가 오류를 감지하면 적절한 이유 코드와 함께 호출이 실패합니다. 수행되는 검사는 특정 구조에 따라 다릅니다. 또한, 버전-2 이상 MQMD를 MQPUT 또는 MQPUT1 호출에 사용하는 경우에만 검사가 수행됩니다. 버전-1 MQMD를 사용하는 경우에는 MQMDE가 애플리케이션 메시지 데이터 시작 시 존재하더라도 검사가 수행되지 않습니다.

MQDH, MQMDE IBM MQ 헤더 구조는 큐 관리자를 통해 완전히 유효성 검증됩니다.

다른 IBM MQ 헤더 구조의 경우, 큐 관리자가 몇 가지 유효성 검증을 수행하지만 모든 필드를 검사하지는 않습니다. 로컬 큐 관리자에 지원되지 않는 구조 및 메시지의 첫 번째 MQDLH를 따르는 구조는 유효성 검증되지 않습니다.

IBM MQ 구조의 필드에 실행되는 일반 검사 외에, 다음 조건을 충족해야 합니다.

- IBM MQ 구조가 둘 이상의 세그먼트로 분할되지 않아야 합니다. 구조가 완전히 하나의 세그먼트 내에 포함되어 있어야 합니다.
- PCF 메시지의 구조 길이의 합계는 MQPUT 또는 MQPUT1 호출에서 **BUFLEN** 매개변수에 지정된 길이와 같아야 합니다. PCF 메시지는 형식 이름이 다음 중 하나인 메시지입니다.
  - FMADMN
  - FMEVNT
  - FMPCF
- IBM MQ 구조는 잘린 구조가 허용되는 다음 상황을 제외하고, 잘리지 않아야 합니다.
  - 보고 메시지인 메시지.
  - PCF 메시지.
  - MQDLH 구조를 포함하는 메시지. (첫 번째 MQDLH 뒤의 구조는 자를 수 있지만, MQDLH 앞의 구조는 자를 수 없습니다.)

## 버퍼

RPG 프로그래밍 예에 표시된 **BUFFER** 매개변수는 문자열로 선언됩니다. 따라서 매개변수의 최대 길이가 256바이트로 제한됩니다. 더 큰 버퍼가 필요하다면 매개변수를 물리적 파일에 있는 필드 또는 구조로 선언해야 합니다. 이렇게 하면 가능한 최대 길이가 약 32KB로 증가합니다.



## 매개변수

MQPUT 호출의 매개변수는 다음과 같습니다.

### HCONN(10자리 부호 있는 정수) - 입력

연결 핸들입니다.

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. HCONN의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

### HOBJ(10자리 부호 있는 정수) - 입력

오브젝트 핸들.

이 핸들은 메시지가 추가되는 큐 또는 메시지가 발행되는 토픽을 나타냅니다. OOOUT 옵션을 지정한 이전 MQOPEN 호출에서 HOBJ 값이 리턴되었습니다.

### MSGDSC(MQMD) - 입출력(I/O)

메시지 디스크립터.

이 구조는 송신하는 메시지의 속성을 설명하며, Put 요청이 완료된 후 메시지에 대한 정보를 수신합니다. 자세한 내용은 1063 페이지의 『IBM i의 MQMD(메시지 디스크립터)』의 내용을 참조하십시오.

애플리케이션이 버전-1 MQMD를 제공하는 경우, 버전-1이 아닌 버전-2 MQMD에 있는 필드에 대한 값을 지정하기 위해 메시지 데이터에 MQMDE 구조가 접두부로 붙을 수 있습니다. MQMDE가 있음을 표시하려면 MQMD의 MDFMT 필드가 FMMDE로 설정되어야 합니다. 자세한 내용은 1103 페이지의 『IBM i의 MQMDE(메시지 디스크립터 확장자)』의 내용을 참조하십시오.

### PMO (MQPMO) - 입출력(I/O)

MQPUT의 조치를 제어하는 옵션.

자세한 내용은 1122 페이지의 『IBM i의 MQPMO(메시지 넣기 옵션)』의 내용을 참조하십시오.

### BUFLEN(10자리 부호 있는 정수) - 입력

BUFFER의 메시지 길이.

올바른 값은 0이며, 메시지에 애플리케이션 데이터가 없음을 표시합니다. BUFLEN의 상한은 다양한 요인에 따라 다릅니다.

- 목적지 큐가 공유 큐이면 상한은 63KB(64 512바이트)입니다.
  - 목적지가 로컬 큐이거나 로컬 큐로 해석되면(공유 큐가 아님) 상한은 다음에 따라 다릅니다.
    - 로컬 큐 관리자가 세그먼트화를 지원합니다.
    - 전송 애플리케이션은 큐 관리자가 메시지를 세그먼트화하도록 하는 플래그를 지정합니다. 이 플래그는 MFSEGA이고, 버전-2 MQMD 또는 버전-1 MQMD가 사용된 MQMDE에 지정할 수 있습니다.
- 두 조건을 모두 충족하는 경우, BUFLEN은 999 999 999에서 MQMD의 MDOFF 필드 값을 차감한 값을 초과하지 않아야 합니다. 따라서 넣을 수 있는 가장 긴 논리 메시지는 999 999 999바이트입니다(MDOFF가 0인 경우). 그러나, 애플리케이션이 실행 중인 운영 체제 또는 환경에서 적용한 자원 제한조건 때문에 하한이 설정될 수 있습니다.
- 이전에 설명한 조건 중 하나 또는 모두를 충족하지 않으면 BUFLEN이 큐의 MaxMsgLength 속성과 큐 관리자의 MaxMsgLength 속성 중 더 작은 값을 초과하지 않아야 합니다.
- 대상이 리모트 큐이거나 리모트 큐로 해석되는 경우, 로컬 큐의 조건이 적용됩니다. 목적지 큐에 도달하기 위해 메시지가 통과해야 하는 각 큐 관리자와 다음 큐에서 특히 그렇습니다.
    1. 로컬 큐 관리자에 임시로 메시지를 저장하는 데 사용되는 로컬 송신 큐
    2. 로컬 및 목적지 큐 관리자 간 라우트에 있는 큐 관리자에 메시지를 저장하는 데 사용되는 중간 전송 큐(있는 경우)
    3. 목적지 큐 관리자에 있는 목적지 큐
- 따라서 넣을 수 있는 가장 긴 메시지는 이 큐 및 큐 관리자의 가장 제한적인 값에 따라 통제됩니다.



메시지가 전송 큐에 있을 때 추가 정보가 메시지 데이터와 함께 있으며, 이로 인해 이동할 수 있는 애플리케이션 데이터의 양이 줄어듭니다. 이 상황에서 *BUFLLEN*의 한계를 판별하는 경우 전송 큐의 *MaxMsgLength* 값에서 *LNMHD* 바이트를 차감하는 것이 좋습니다.

**참고:** 메시지를 넣을 때 조건 1의 준수 실패만 동기적으로 진단할 수 있습니다(이유 코드 RC2030 또는 RC2031 표시). 조건 2 또는 3을 충족하지 않는 경우, 메시지가 중간 큐 관리자 또는 목적지 큐 관리자에서 데드 레터(미배달 메시지) 큐로 경로 재지정됩니다. 이 경우, 송신자의 요청이 있으면 보고 메시지가 생성됩니다.

#### **BUFFER(1바이트 비트 문자열 x BUFLLEN) - 입력**

메시지 데이터.

송신할 애플리케이션 데이터가 있는 버퍼입니다. 버퍼는 메시지에 있는 데이터의 네이처에 적절한 경계에 맞춰야 합니다. 4바이트 맞추기는 대부분의 메시지(MQ 헤더 구조가 있는 메시지 포함)에 적절하지만, 일부 메시지는 보다 엄격한 맞추기를 요청할 수 있습니다. 예를 들어, 64비트 2진 정수를 포함하는 메시지에는 8바이트 맞추기가 필요할 수 있습니다.

*BUFFER*에 문자 데이터, 숫자 데이터 또는 둘 다 포함되는 경우, **MSGDSC** 매개변수의 *MDCSI* 및 *MDENC* 필드는 데이터에 적합한 값으로 설정되어야 합니다. 이를 통해 메시지 수신자는 필요한 경우 데이터를 수신자가 사용하는 문자 세트 및 인코딩으로 변환할 수 있습니다.

**참고:** MQPUT 호출의 다른 모든 매개변수가 **CodedCharSetId** 큐 관리자 속성이 제공한 문자 세트 및 ENNAT가 제공한 로컬 큐 관리자의 인코딩에 있어야 합니다.

#### **CMPCOD(10자리 부호 있는 정수) - 출력**

완료 코드.

다음 중 하나입니다.

##### **CCOK**

정상적으로 완료되었습니다.

##### **CCWARN**

경고(일부 완료).

##### **CCFAIL**

호출에 실패했습니다.

#### **REASON(10자리 부호 있는 정수) - 출력**

*CMPCOD*을(를) 규정하는 이유 코드.

*CMPCOD*가 CCOK일 경우:

##### **RCNONE**

(0, X'000') 보고할 이유가 없습니다.

If *CMPCOD* is CCWARN:

##### **RC2104**

(2104, X'838') 메시지 디스크립터의 보고 옵션이 인식되지 않습니다.

##### **RC2136**

(2136, X'858') 다중 이유 코드가 리턴되었습니다.

*CMPCOD*가 CCFAIL일 경우:

##### **RC2004**

(2004, X'7D4') 버퍼 매개변수가 올바르지 않습니다.

##### **RC2005**

(2005, X'7D5') 버퍼 길이 매개변수가 올바르지 않습니다.

##### **RC2009**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

##### **RC2013**

(2013, X'7DD') 만기 시간이 올바르지 않습니다.

- RC2014**  
(2014, X'7DE') 피드백 코드가 올바르지 않습니다.
- RC2018**  
(2018, X'7E2') 연결 핸들이 유효하지 않습니다.
- RC2019**  
(2019, X'7E3') 오브젝트 핸들이 올바르지 않습니다.
- RC2024**  
(2024, X'7E8') 현재 작업 단위 내에서 더 이상 메시지를 핸들링할 수 없습니다.
- RC2026**  
(2026, X'7EA') 메시지 디스크립터가 올바르지 않습니다.
- RC2027**  
(2027, X'7EB') 응답 대상 큐가 누락되었습니다.
- RC2029**  
(2029, X'7ED') 메시지 디스크립터의 메시지 유형이 올바르지 않습니다.
- RC2030**  
(2030, X'7EE') 메시지 길이가 큐의 최대 길이보다 깁니다.
- RC2031**  
(2031, X'7EF') 메시지 길이가 큐 관리자의 최대값보다 큽니다.
- RC2039**  
(2039, X'7F7') 출력을 큐를 열지 못했습니다.
- RC2041**  
(2041, X'7F9') 오브젝트가 열린 후에 정의가 변경되었습니다.
- RC2046**  
(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.
- RC2047**  
(2047, X'7FF') 지속이 올바르지 않습니다.
- RC2048**  
(2048, X'800') 큐가 지속 메시지를 지원하지 않습니다.
- RC2050**  
(2050, X'802') 메시지 우선순위가 올바르지 않습니다.
- RC2051**  
(2051, X'803') 큐에 대해 금지된 Put 호출입니다.
- RC2052**  
(2052, X'804') 큐가 삭제되었습니다.
- RC2053**  
(2053, X'805') 큐에 이미 최대 수의 메시지가 포함되어 있습니다.
- RC2056**  
(2056, X'808') 큐에 사용할 수 있는 공간이 디스크에 없습니다.
- RC2058**  
(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.
- RC2059**  
(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.
- RC2061**  
(2061, X'80D') 메시지 디스크립터의 보고 옵션이 올바르지 않습니다.
- RC2071**  
(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.
- RC2072**  
(2072, X'818') 동기점 지원을 사용할 수 없습니다.
- RC2093**  
(2093, X'82D') 모든 컨텍스트 전달을 위해 큐를 열지 못했습니다.

- RC2094**  
(2094, X'82E') ID 컨텍스트 전달을 위해 큐를 열지 못했습니다.
- RC2095**  
(2095, X'82F') 모든 컨텍스트 설정을 위해 큐를 열지 못했습니다.
- RC2096**  
(2096, X'830') ID 컨텍스트 설정을 위해 큐를 열지 못했습니다.
- RC2097**  
(2097, X'831') 참조되는 큐 핸들이 컨텍스트를 저장하지 않습니다.
- RC2098**  
(2098, X'832') 참조되는 큐 핸들에 대해 컨텍스트를 사용할 수 없습니다.
- RC2101**  
(2101, X'835') 오브젝트가 손상되었습니다.
- RC2102**  
(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.
- RC2135**  
(2135, X'857') 분배 헤더 구조가 올바르지 않습니다.
- RC2136**  
(2136, X'858') 다중 이유 코드가 리턴되었습니다.
- RC2137**  
(2137, X'859') 오브젝트가 성공적으로 열리지 않았습니다.
- RC2149**  
(2149, X'865') PCF 구조가 올바르지 않습니다.
- RC2154**  
(2154, X'86A') 현재 레코드 수가 유효하지 않습니다.
- RC2156**  
(2156, X'86C') 응답 레코드가 올바르지 않습니다.
- RC2158**  
(2158, X'86E') 메시지 넣기 레코드 플래그가 올바르지 않습니다.
- RC2159**  
(2159, X'86F') 메시지 넣기 레코드가 올바르지 않습니다.
- RC2161**  
(2161, X'871') 큐 관리자가 정지됩니다.
- RC2162**  
(2162, X'872') 큐 관리자가 종료되었습니다.
- RC2173**  
(2173, X'87D') 메시지 넣기 옵션 구조가 올바르지 않습니다.
- RC2185**  
(2185, X'889') 불일치하는 지속성 스펙.
- RC2188**  
(2188, X'88C') 클러스터 워크로드 엑시트에 의해 호출이 거부되었습니다.
- RC2189**  
(2189, X'88D') 클러스터 이름을 해석하지 못했습니다.
- RC2195**  
(2195, X'893') 예상치 못한 오류가 발생했습니다.
- RC2219**  
(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 다시 입력되었습니다.
- RC2241**  
(2241, X'8C1') 메시지 그룹이 완료되지 않았습니다.
- RC2242**  
(2242, X'8C2') 논리 메시지가 완료되지 않았습니다.

**RC2245**

(2245, X'8C5') 작업 단위 지정에 일관성이 없습니다.

**RC2248**

(2248, X'8C8') 메시지 디스크립터 확장이 올바르지 않습니다.

**RC2249**

(2249, X'8C9') 메시지 플래그가 올바르지 않습니다.

**RC2250**

(2250, X'8CA') 메시지 순서 번호가 올바르지 않습니다.

**RC2251**

(2251, X'8CB') 메시지 세그먼트 오프셋이 올바르지 않습니다.

**RC2252**

(2252, X'8CC') 원래 길이가 올바르지 않습니다.

**RC2253**

(2253, X'8CD') 메시지 세그먼트의 데이터 길이가 0입니다.

**RC2255**

(2255, X'8CF') 사용할 큐 관리자에 대해 작업 단위를 사용할 수 없습니다.

**RC2257**

(2257, X'8D1') 올바르지 않은 버전의 MQMD가 제공되었습니다.

**RC2258**

(2258, X'8D2') 그룹 ID가 올바르지 않습니다.

**RC2266**

(2266, X'8DA') 클러스터 워크로드 엑시트가 실패했습니다.

**RC2269**

(2269, X'8DD') 클러스터 자원에 오류가 있습니다.

**RC2270**

(2270, X'8DE') 목적지 큐가 사용 가능하지 않습니다.

**RC2420**

(2420) MQPUT 호출이 발행되었지만, 메시지 데이터에 올바르지 않은 MQEPH 구조가 있습니다.

**RC2479**

(2479, X'9AF') 발행물을 보유할 수 없습니다.

**RC2480**

(2480, X'9B0') 대상 유형이 변경되었습니다. 알리어스 큐가 큐를 참조했지만 이제는 토픽을 참조합니다.

**RC2502**

(2502, X'9C6') 발행에 실패했고, 발행물이 구독자에게 전달되지 않았습니다.

**RC2551**

(2551, X'9F7') 지정된 선택 문자열을 사용할 수 없습니다.

**RC2554**

(2554, X'9FA') 메시지 콘텐츠를 구문 분석하여 메시지가 확장된 메시지 선택자와 함께 구독자에게 전달되어야 하는지 여부를 판별할 수 없습니다.

**RPG 선언**

```

C*.1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQPUT(HCONN : HOBJ : MSGDSC : PMO :
C          BUFLLEN : BUFFER : CMPCOD :
C          REASON)

```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```

D*.1.....2.....3.....4.....5.....6.....7..
DMQPUT      PR          EXTPROC('MQPUT')

```

```

D* Connection handle
D HCONN                10I 0 VALUE
D* Object handle
D HOBJ                10I 0 VALUE
D* Message descriptor
D MSGDSC                364A
D* Options that control the action of MQPUT
D PMO                  200A
D* Length of the message in Buffer
D BUFLN                10I 0 VALUE
D* Message data
D BUFFER                *   VALUE
D* Completion code
D CMPCOD                10I 0
D* Reason code qualifying CMPCOD
D REASON                10I 0

```

## IBM i IBM i의 MQPUT1(하나의 메시지 넣기)

MQPUT1 호출은 큐나 분배 목록 또는 토픽에 메시지를 넣습니다. 큐, 분배 목록 또는 토픽을 열 필요가 없습니다.

- [1281 페이지의 『구문』](#)
- [1281 페이지의 『사용법 참고』](#)
- [1282 페이지의 『매개변수』](#)
- [1287 페이지의 『RPG 선언』](#)

### 구문

MQPUT1 (*HCONN*, *OBJDSC*, *MSGDSC*, *PMO*, *BUFLN*, *BUFFER*, *CMPCOD*, *REASON*)

### 사용법 참고

1. MQPUT 및 MQPUT1 호출을 사용하여 메시지를 큐에 넣을 수 있습니다. 사용할 호출은 환경에 따라 다릅니다.

- 여러 메시지를 동일한 큐에 넣는 경우 MQPUT 호출을 사용해야 합니다.

OOOUT 옵션을 지정하는 MQOPEN 호출을 먼저 발행한 다음 메시지를 큐에 추가하라는 하나 이상의 MQPUT 요청이 이어지고, 마지막으로 MQCLOSE 호출과 함께 큐가 닫힙니다. 이렇게 하면 MQPUT1 호출을 반복 사용하는 것보다 성능이 높아집니다.

- 단 하나의 메시지를 큐에 넣는 경우에는 MQPUT1 호출을 사용해야 합니다.

이 호출은 MQOPEN, MQPUT 및 MQCLOSE 호출을 단일 호출로 캡슐화하며, 실행되어야 하는 호출의 수를 최소화합니다.

2. 애플리케이션이 메시지 그룹을 사용하지 않고 동일 큐에 메시지 순서를 넣는 경우, 특정 조건을 충족하는 경우 이러한 메시지의 순서가 보존됩니다. 그러나, 대부분의 환경에서는 MQPUT1 호출이 이 조건을 충족하지 않으므로 메시지 순서가 보존되지 않습니다. 이 환경에서는 MQPUT 호출을 대신 사용해야 합니다. 자세한 정보는 MQPUT 호출 설명에 나온 사용 시 참고사항을 참조하십시오.

3. MQPUT1 호출을 사용하여 메시지를 분배 목록에 넣을 수 있습니다. 이에 관한 일반적인 정보는 MQOPEN 및 MQPUT 호출에 대한 사용 시 참고사항을 참조하십시오.

MQPUT1 호출을 사용하는 경우 다음 차이점이 적용됩니다.

- 애플리케이션에서 MQRR 응답 레코드를 제공한 경우, MQOD 구조를 사용하여 제공되어야 합니다. MQPMO 구조를 사용해서는 제공할 수 없습니다.

- MQPUT1은 응답 레코드에 이유 코드 RC2137을 리턴하지 않습니다. 큐를 열지 못하면 해당 큐의 응답 레코드에 열기 조작에 따른 실제 이유 코드가 들어 있습니다.

큐의 열기 조작이 완료 코드 CCWARN과 함께 성공하면, 해당 큐의 응답 레코드에 있는 완료 코드 및 이유 코드가 넣기 조작으로 발생한 완료 코드 및 이유 코드로 대체됩니다.

MQOPEN 및 MQPUT 호출과 마찬가지로, 큐 관리자는 호출 출력이 분배 목록의 모든 큐에 동일하지 않은 경우에만 응답 레코드를 설정합니다(제공되는 경우). 이는 이유 코드 RC2136과 함께 완료되는 호출을 통해 알 수 있습니다.

4. 메시지를 클러스터 큐에 넣는 데 MQPUT1 호출이 사용되면 해당 호출은 OOBNDN이 MQOPEN 호출에 지정된 것처럼 작동합니다.
5. 애플리케이션 메시지 데이터의 시작 부분에 하나 이상의 IBM MQ 헤더 구조가 있는 상태에서 메시지를 넣으면 큐 관리자가 헤더 구조에 특정 검사를 수행하여 해당 구조가 올바른지 확인합니다. 이에 대한 자세한 정보는 MQPUT 호출에 대한 사용 시 참고사항을 참조하십시오.
6. 둘 이상의 경고 상황이 발생하면(**CMPCOD** 매개변수 참조), 해당되는 다음 목록 중 첫 번째 이유 코드가 리턴됩니다.
  - a. RC2136
  - b. RC2242
  - c. RC2241
  - d. RC2049 또는 RC2104
7. RPG 프로그래밍 예에 표시된 **BUFFER** 매개변수는 문자열로 선언됩니다. 따라서 매개변수의 최대 길이가 256바이트로 제한됩니다. 더 큰 버퍼가 필요하면 매개변수를 물리적 파일에 있는 필드 또는 구조로 선언해야 합니다. 이렇게 하면 가능한 최대 길이가 약 32KB로 증가합니다.

## 매개변수

MQPUT1 호출의 매개변수는 다음과 같습니다.

### HCONN(10자리 부호 있는 정수) - 입력

연결 핸들입니다.

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *HCONN*의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

### OBJDSC(MQOD) - 입출력(I/O)

오브젝트 디스크립터.

메시지가 추가될 큐를 식별하는 구조입니다. 자세한 내용은 [1109 페이지의 『IBM i의 MQOD\(오브젝트 디스크립터\)』](#)의 내용을 참조하십시오.

사용자에게 출력을 위해 큐를 열 수 있는 권한이 있어야 합니다. 큐는 모델 큐가 **아니어야** 합니다.

### MSGDSC(MQMD) - 입출력(I/O)

메시지 디스크립터.

이 구조는 송신하는 메시지의 속성을 설명하고, 넣기 요청이 완료된 후 피드백 정보를 수신합니다. 자세한 내용은 [1063 페이지의 『IBM i의 MQMD\(메시지 디스크립터\)』](#)의 내용을 참조하십시오.

애플리케이션이 버전-1 MQMD를 제공하는 경우, 버전-1이 아닌 버전-2 MQMD에 있는 필드에 대한 값을 지정하기 위해 메시지 데이터에 MQMDE 구조가 접두부로 붙을 수 있습니다. MQMDE가 있음을 표시하려면 MQMD의 *MDFMT* 필드가 FMMDE로 설정되어야 합니다. 자세한 내용은 [1103 페이지의 『IBM i의 MQMDE\(메시지 디스크립터 확장자\)』](#)의 내용을 참조하십시오.

### PMO(MQPMO) - 입출력(I/O)

MQPUT1의 조치를 제어하는 옵션.

자세한 내용은 [1122 페이지의 『IBM i의 MQPMO\(메시지 넣기 옵션\)』](#)의 내용을 참조하십시오.

### BUFLEN(10자리 부호 있는 정수) - 입력

*BUFFER*의 메시지 길이.

올바른 값은 0이며, 메시지에 애플리케이션 데이터가 없음을 표시합니다. 상한은 다양한 요인에 따라 결정됩니다. 자세한 정보는 MQPUT 호출의 **BUFLEN** 매개변수 설명을 참조하십시오.

## **BUFFER(1바이트 비트 문자열 x BUFLLEN) - 입력**

메시지 데이터.

송신할 애플리케이션 메시지 데이터가 있는 버퍼입니다. 버퍼는 메시지에 있는 데이터의 네이처에 적절한 경계에 맞춰야 합니다. 4바이트 맞추기는 대부분의 메시지(IBM MQ 헤더 구조가 있는 메시지 포함)에 적절해야 하지만, 일부 메시지는 보다 엄격한 맞추기를 요청할 수 있습니다. 예를 들어, 64비트 2진 정수를 포함하는 메시지에는 8바이트 맞추기가 필요할 수 있습니다.

*BUFFER*에 문자 데이터, 숫자 데이터 또는 둘 다 포함되는 경우, **MSGDSC** 매개변수의 *MDCSI* 및 *MDENC* 필드는 데이터에 적합한 값으로 설정되어야 합니다. 이를 통해 메시지 수신자는 필요한 경우 데이터를 수신자가 사용하는 문자 세트 및 인코딩으로 변환할 수 있습니다.

**참고:** MQPUT1 호출의 다른 모든 매개변수가 **CodedCharSetId** 큐 관리자 속성이 제공한 문자 세트 및 ENNAT에 제공된 로컬 큐 관리자의 인코딩에 있어야 합니다.

## **CMPCOD(10자리 부호 있는 정수) - 출력**

완료 코드.

다음 중 하나입니다.

### **CCOK**

정상적으로 완료되었습니다.

### **CCWARN**

경고(일부 완료).

### **CCFAIL**

호출에 실패했습니다.

## **REASON(10자리 부호 있는 정수) - 출력**

*CMPCOD*을(를) 규정하는 이유 코드.

*CMPCOD*가 CCOK일 경우:

### **RCNONE**

(0, X'000') 보고할 이유가 없습니다.

If *CMPCOD* is CCWARN:

### **RC2104**

(2104, X'838') 메시지 디스크립터의 보고 옵션이 인식되지 않습니다.

### **RC2136**

(2136, X'858') 다중 이유 코드가 리턴되었습니다.

### **RC2049**

(2049, X'801') 메시지 우선순위가 지원되는 최대 값을 초과합니다.

### **RC2241**

(2241, X'8C1') 메시지 그룹이 완료되지 않았습니다.

### **RC2242**

(2242, X'8C2') 논리 메시지가 완료되지 않았습니다.

*CMPCOD*가 CCFAIL일 경우:

### **RC2001**

(2001, X'7D1') 알리어스 기본 큐가 올바른 유형이 아닙니다.

### **RC2004**

(2004, X'7D4') 버퍼 매개변수가 올바르지 않습니다.

### **RC2005**

(2005, X'7D5') 버퍼 길이 매개변수가 올바르지 않습니다.

### **RC2009**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

- RC2013**  
(2013, X'7DD') 만기 시간이 올바르지 않습니다.
- RC2014**  
(2014, X'7DE') 피드백 코드가 올바르지 않습니다.
- RC2017**  
(2017, X'7E1') 사용 가능한 핸들이 없습니다.
- RC2018**  
(2018, X'7E2') 연결 핸들이 유효하지 않습니다.
- RC2024**  
(2024, X'7E8') 현재 작업 단위 내에서 더 이상 메시지를 핸들링할 수 없습니다.
- RC2026**  
(2026, X'7EA') 메시지 디스크립터가 올바르지 않습니다.
- RC2027**  
(2027, X'7EB') 응답 대상 큐가 누락되었습니다.
- RC2029**  
(2029, X'7ED') 메시지 디스크립터의 메시지 유형이 올바르지 않습니다.
- RC2030**  
(2030, X'7EE') 메시지 길이가 큐의 최대 길이보다 깁니다.
- RC2031**  
(2031, X'7EF') 메시지 길이가 큐 관리자의 최대값보다 큼니다.
- RC2035**  
(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.
- RC2042**  
(2042, X'7FA') 오브젝트가 이미 충돌하는 옵션으로 열렸습니다.
- RC2043**  
(2043, X'7FB') 오브젝트 유형이 유효하지 않습니다.
- RC2044**  
(2044, X'7FC') 오브젝트 디스크립터 구조가 유효하지 않습니다.
- RC2046**  
(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.
- RC2047**  
(2047, X'7FF') 지속이 올바르지 않습니다.
- RC2048**  
(2048, X'800') 큐가 지속 메시지를 지원하지 않습니다.
- RC2050**  
(2050, X'802') 메시지 우선순위가 올바르지 않습니다.
- RC2051**  
(2051, X'803') 큐에 대해 금지된 Put 호출입니다.
- RC2052**  
(2052, X'804') 큐가 삭제되었습니다.
- RC2053**  
(2053, X'805') 큐에 이미 최대 수의 메시지가 포함되어 있습니다.
- RC2056**  
(2056, X'808') 큐에 사용할 수 있는 공간이 디스크에 없습니다.
- RC2057**  
(2057, X'809') 큐 유형이 유효하지 않습니다.
- RC2058**  
(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.
- RC2059**  
(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.



- RC2061**  
(2061, X'80D') 메시지 디스크립터의 보고 옵션이 올바르지 않습니다.
- RC2063**  
(2063, X'80F') 보안 오류가 발생했습니다.
- RC2071**  
(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.
- RC2072**  
(2072, X'818') 동기점 지원을 사용할 수 없습니다.
- RC2082**  
(2082, X'822') 알 수 없는 알리어스 기본 큐입니다.
- RC2085**  
(2085, X'825') 알 수 없는 오브젝트 이름입니다.
- RC2086**  
(2086, X'826') 알 수 없는 오브젝트 큐 관리자입니다.
- RC2087**  
(2087, X'827') 알 수 없는 리모트 큐 관리자입니다.
- RC2091**  
(2091, X'82B') 전송 큐가 로컬이 아닙니다.
- RC2092**  
(2092, X'82C') 사용법이 올바르지 않은 전송 큐입니다.
- RC2097**  
(2097, X'831') 참조되는 큐 핸들이 컨텍스트를 저장하지 않습니다.
- RC2098**  
(2098, X'832') 참조되는 큐 핸들에 대해 컨텍스트를 사용할 수 없습니다.
- RC2101**  
(2101, X'835') 오브젝트가 손상되었습니다.
- RC2102**  
(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.
- RC2135**  
(2135, X'857') 분배 헤더 구조가 올바르지 않습니다.
- RC2136**  
(2136, X'858') 다중 이유 코드가 리턴되었습니다.
- RC2149**  
(2149, X'865') PCF 구조가 올바르지 않습니다.
- RC2154**  
(2154, X'86A') 현재 레코드 수가 유효하지 않습니다.
- RC2155**  
(2155, X'86B') 오브젝트 레코드가 올바르지 않습니다.
- RC2156**  
(2156, X'86C') 응답 레코드가 올바르지 않습니다.
- RC2158**  
(2158, X'86E') 메시지 넣기 레코드 플래그가 올바르지 않습니다.
- RC2159**  
(2159, X'86F') 메시지 넣기 레코드가 올바르지 않습니다.
- RC2161**  
(2161, X'871') 큐 관리자가 정지됩니다.
- RC2162**  
(2162, X'872') 큐 관리자가 종료되었습니다.
- RC2173**  
(2173, X'87D') 메시지 넣기 옵션 구조가 올바르지 않습니다.

- RC2184**  
(2184, X'888') 리모트 큐 이름이 유효하지 않습니다.
- RC2188**  
(2188, X'88C') 클러스터 워크로드 엑시트에 의해 호출이 거부되었습니다.
- RC2189**  
(2189, X'88D') 클러스터 이름을 해석하지 못했습니다.
- RC2195**  
(2195, X'893') 예상치 못한 오류가 발생했습니다.
- RC2196**  
(2196, X'894') 알 수 없는 전송 큐입니다.
- RC2197**  
(2197, X'895') 알 수 없는 기본 전송 큐입니다.
- RC2198**  
(2198, X'896') 기본 전송 큐가 로컬이 아닙니다.
- RC2199**  
(2199, X'897') 기본 전송 큐 사용법 오류입니다.
- RC2258**  
(2258, X'8D2') 그룹 ID가 올바르지 않습니다.
- RC2248**  
(2248, X'8C8') 메시지 디스크립터 확장이 올바르지 않습니다.
- RC2219**  
(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 다시 입력되었습니다.
- RC2249**  
(2249, X'8C9') 메시지 플래그가 올바르지 않습니다.
- RC2250**  
(2250, X'8CA') 메시지 순서 번호가 올바르지 않습니다.
- RC2251**  
(2251, X'8CB') 메시지 세그먼트 오프셋이 올바르지 않습니다.
- RC2252**  
(2252, X'8CC') 원래 길이가 올바르지 않습니다.
- RC2253**  
(2253, X'8CD') 메시지 세그먼트의 데이터 길이가 0입니다.
- RC2255**  
(2255, X'8CF') 사용할 큐 관리자에 대해 작업 단위를 사용할 수 없습니다.
- RC2257**  
(2257, X'8D1') 올바르지 않은 버전의 MQMD가 제공되었습니다.
- RC2266**  
(2266, X'8DA') 클러스터 워크로드 엑시트가 실패했습니다.
- RC2269**  
(2269, X'8DD') 클러스터 자원에 오류가 있습니다.
- RC2270**  
(2270, X'8DE') 목적지 큐가 사용 가능하지 않습니다.
- RC2420**  
(2420) MQPUT1 호출이 발행되었지만, 메시지 데이터에 올바르지 않은 MQEPH 구조가 있습니다.
- RC2551**  
(2551, X'9F7') 지정된 선택 문자열을 사용할 수 없습니다.
- RC2554**  
(2554, X'9FA') 메시지 콘텐츠를 구문 분석하여 메시지가 확장된 메시지 선택자와 함께 구독자에게 전달되어야 하는지 여부를 판별할 수 없습니다.

## RPG 선언

```
C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQPUT1(HCONN : OBJDSC : MSGDSC :
C          PMO : BUFLN : BUFFER :
C          CMPCOD : REASON)
```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```
D*..1.....2.....3.....4.....5.....6.....7..
DMQPUT1      PR          EXTPROC('MQPUT1')
D* Connection handle
D HCONN          10I 0 VALUE
D* Object descriptor
D OBJDSC          468A
D* Message descriptor
D MSGDSC          364A
D* Options that control the action of MQPUT1
D PMO            200A
D* Length of the message in BUFFER
D BUFLN          10I 0 VALUE
D* Message data
D BUFFER          * VALUE
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CMPCOD
D REASON          10I 0
```

## IBM i IBM i의 MQSET(오브젝트 속성 설정)

MQSET 호출은 핸들이 나타내는 오브젝트의 속성을 변경하는 데 사용됩니다. 오브젝트는 큐여야 합니다.

- [1287 페이지의 『구문』](#)
- [1287 페이지의 『사용법 참고』](#)
- [1288 페이지의 『매개변수』](#)
- [1291 페이지의 『RPG 선언』](#)

## 구문

MQSET (HCONN, HOBJ, SELCNT, SELS, IACNT, INTATR, CALEN, CHRATR, CMPCOD, REASON)

## 사용법 참고

1. 이 호출을 사용하여 애플리케이션이 정수 속성 배열, 문자 속성 문자열 콜렉션 또는 둘 다를 지정할 수 있습니다. 오류가 발생하지 않으면 지정된 속성이 모두 동시에 설정됩니다. 오류가 발생하면(예: 선택자가 올바르지 않거나 속성을 올바르게 않은 값으로 설정하려고 함), 호출이 실패하고 속성이 설정되지 않습니다.
2. 속성의 값은 MQINQ 호출을 사용하여 판별할 수 있습니다. 세부사항은 [1246 페이지의 『IBM i의 MQINQ\(오브젝트 속성 조회\)』](#)의 내용을 참조하십시오.

**참고:** MQINQ 호출을 사용하여 조회할 수 있는 값을 가진 모든 속성이 MQSET 호출을 사용하여 값을 변경할 수 있는 것은 아닙니다. 예를 들어, process-object 또는 큐 관리자 속성은 이 호출을 사용하여 설정할 수 없습니다.

3. 속성 변경은 큐 관리자를 재시작해도 보존됩니다. (단, 임시 동적 큐에 대한 변경사항은 큐 관리자를 재시작하면 지속되지 않습니다.)
4. MQSET 호출을 사용하여 모델 큐의 속성을 변경할 수는 없습니다. 그러나, MQOO\_SET 옵션과 MQOPEN 호출을 사용하여 모델 큐를 여는 경우 MQSET 호출을 사용하여 MQOPEN 호출에 의해 작성된 동적 로컬 큐의 속성을 설정할 수 있습니다.
5. 설정되는 오브젝트가 클러스터 큐인 경우 열기가 성공하려면 클러스터 큐의 인스턴스가 있어야 합니다.

오브젝트 속성에 대한 자세한 정보는 다음을 참조하십시오.

- [1305 페이지의 『큐의 속성』](#)
- [1331 페이지의 『이름 목록에 대한 속성』](#)
- [1332 페이지의 『IBM i의 프로세스 정의에 대한 속성』](#)
- [1334 페이지의 『IBM i의 큐 관리자에 대한 속성』](#)

## 매개변수

MQSET 호출의 매개변수는 다음과 같습니다.

### HCONN(10자리 부호 있는 정수) - 입력

연결 핸들입니다.

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. HCONN의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

### HOBJ(10자리 부호 있는 정수) - 입력

오브젝트 핸들.

이 핸들은 설정될 속성을 가진 큐 오브젝트를 나타냅니다. 핸들이 OOSSET 옵션을 지정한 이전 MQOPEN 호출에 의해 리턴되었습니다.

### SELCNT(10자리 부호 있는 정수) - 입력

선택자의 수.

이는 SELS 배열에서 제공되는 선택자의 수입니다. 설정되는 속성의 수입니다. 0은 올바른 값입니다. 허용된 최대 수는 256입니다.

### SELS(10자리 부호 있는 정수 x SELCNT) - 입력

속성 선택자의 배열.

**SELCNT** 속성 선택자의 배열입니다. 각 선택자는 설정될 값을 사용하여 속성(정수 또는 문자)을 식별합니다.

각 선택자는 HOBJ가 나타내는 큐 유형에 대해 유효해야 합니다. 특정 IA\* 및 CA\* 값만 허용됩니다. 이러한 값은 이 절의 후반부에 나열됩니다.

임의의 순서로 선택자를 지정할 수 있습니다. 정수 속성 선택자(IA\* 선택자)에 해당되는 속성 값은 SELS에서 해당 선택자가 발생한 순서와 동일한 순서대로 INTATR에서 지정되어야 합니다. 문자 속성 선택자(CA\* 선택자)에 해당되는 속성 값은 해당 선택자가 발생한 순서와 동일한 순서대로 CHRATR에서 지정되어야 합니다. IA\* 선택자는 CA\* 선택자로 인터리브될 수 있습니다. 각 유형에서 상대 순서만이 중요합니다.

동일한 선택자를 두 번 이상 지정하는 것은 오류가 아닙니다. 그런 경우, 특정 선택자에 마지막으로 지정된 값이 적용됩니다.

#### 참고:

1. 정수 및 문자 속성 선택자는 두 가지 다른 범위에서 할당됩니다. IA\* 선택자는 IAFRST에서 IALAST 사이의 범위에 상주하며 CA\* 선택자는 CAFRST에서 CALAST 사이의 범위에 상주합니다.  
각 범위에 대해 상수 IALSTU 및 CALSTU가 큐 관리자가 승인할 수 있는 가장 높은 값을 정의합니다.
2. 모든 IA\* 선택자가 처음 발생하면 동일한 요소 번호를 사용하여 SELS 및 INTATR 배열에서 해당되는 요소를 처리할 수 있습니다.

설정할 수 있는 속성이 다음 표에 나와 있습니다. 그 외 속성은 이 호출을 사용하여 설정할 수 없습니다. CA\* 속성 선택자의 경우, CHRATR에 필요한 문자열의 길이를 바이트 단위로 정의하는 상수는 괄호로 묶여 제공됩니다.

표 751. 큐에 대한 MQSET 속성 선택자		
선택기	설명	참고
CATRGD	트리거 데이터 (LNTRGD).	<a href="#">1289 페이지의 『2』</a>

표 751. 큐에 대한 MQSET 속성 선택자 (계속)		
선택기	설명	참고
IADIST	분배 목록 지원.	1289 페이지의 『1』
IAIGET	가져오기 조작이 허용되는지 여부.	
IAIPUT	넣기 조작이 허용되는지 여부.	
IATRGC	트리거 제어.	1289 페이지의 『2』
IATRGD	트리거 용량.	1289 페이지의 『2』
IATRGP	트리거에 대한 메시지 우선순위 임계값입니다.	1289 페이지의 『2』
IATRGT	트리거 유형.	1289 페이지의 『2』

**참고:**

1. 다음 플랫폼에서만 지원됩니다.

-  AIX
-  IBM i
-  Windows

및 해당 시스템에 연결된 IBM MQ 클라이언트

2. VSE/ESA에서 지원되지 않습니다.

**IACNT(10자리 부호 있는 정수) - 입력**

정수 속성의 수.

INTATR 배열의 요소 수이며 최소한 SELS 매개변수의 IA\* 선택자 수여야 합니다. 요소가 없으면 0이 올바른 값입니다.

**INTATR(10자리 부호 있는 정수 x rxIACNT) - 입력**

정수 속성의 배열.

이는 IACNT 정수 속성 값의 배열입니다. 이러한 속성 값은 SELS 배열의 IA\* 선택자와 순서가 동일해야 합니다.

**CALEN(10자리 부호 있는 정수) - 입력**

문자 속성 버퍼의 길이.

CHRATR 매개변수의 길이(바이트)이며 최소한 SELS 배열에 지정된 문자 속성 길이의 합계여야 합니다. SELS에 CA\* 선택자가 없으면 0이 올바른 값입니다.

**CHRATR(1바이트 문자열 x CALEN) - 입력**

문자 속성.

함께 연결된 문자 속성 값을 포함하는 버퍼입니다. 버퍼의 길이는 CALEN 매개변수로 제공됩니다.

문자 속성은 SELS 배열의 CA\* 선택자와 동일한 순서대로 지정되어야 합니다. 각 문자 속성의 길이는 고정됩니다(SELS 참조). 속성에 대해 설정할 값이 속성의 정의된 길이보다 더 적은 수의 공백이 아닌 문자를 포함하는 경우, 속성 값이 속성의 정의된 길이와 일치하도록 CHRATR의 값을 공백으로 오른쪽을 채워야 합니다.

## **CMPCOD(10자리 부호 있는 정수) - 출력**

완료 코드.

다음 중 하나입니다.

### **CCOK**

정상적으로 완료되었습니다.

### **CCFAIL**

호출에 실패했습니다.

## **REASON(10자리 부호 있는 정수) - 출력**

CMPCOD을(를) 규정하는 이유 코드.

CMPCOD가 CCOK일 경우:

### **RCNONE**

(0, X'000') 보고할 이유가 없습니다.

CMPCOD가 CCFail일 경우:

### **RC2219**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 다시 입력되었습니다.

### **RC2006**

(2006, X'7D6') 문자 속성의 길이가 올바르지 않습니다.

### **RC2007**

(2007, X'7D7') 문자 속성 문자열이 올바르지 않습니다.

### **RC2009**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

### **RC2018**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

### **RC2019**

(2019, X'7E3') 오브젝트 핸들이 올바르지 않습니다.

### **RC2020**

(2020, X'7E4') Get 금지(inhibit-get) 또는 Put 금지(inhibit-put) 큐 속성의 값이 올바르지 않습니다.

### **RC2021**

(2021, X'7E5') 정수 속성의 수가 유효하지 않습니다.

### **RC2023**

(2023, X'7E7') 정수 속성 배열이 유효하지 않습니다.

### **RC2040**

(2040, X'7F8') 설정을 위해 큐를 열지 못했습니다.

### **RC2041**

(2041, X'7F9') 오브젝트가 열린 후에 정의가 변경되었습니다.

### **RC2101**

(2101, X'835') 오브젝트가 손상되었습니다.

### **RC2052**

(2052, X'804') 큐가 삭제되었습니다.

### **RC2058**

(2058, X'80A') 큐 관리자 이름이 올바르지 않거나 알 수 없습니다.

### **RC2059**

(2059, X'80B') 연결에 큐 관리자를 사용할 수 없습니다.

### **RC2162**

(2162, X'872') 큐 관리자가 종료되었습니다.

### **RC2102**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**RC2065**

(2065, X'811') 선택자의 수가 유효하지 않습니다.

**RC2067**

(2067, X'813') 속성 선택자가 올바르지 않습니다.

**RC2066**

(2066, X'812') 선택자 수가 너무 큼니다.

**RC2071**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**RC2075**

(2075, X'81B') trigger-control 속성의 값이 올바르지 않습니다.

**RC2076**

(2076, X'81C') trigger-depth 속성의 값이 올바르지 않습니다.

**RC2077**

(2077, X'81D') trigger-message-priority 속성의 값이 올바르지 않습니다.

**RC2078**

(2078, X'81E') trigger-type 속성의 값이 올바르지 않습니다.

**RC2195**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

**RPG 선언**

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQSET(HCONN : HOBJ : SELCNT :
C                               SELS(1) : IACNT : INTATR(1) :
C                               CALEN : CHRATR : CMPCOD :
C                               REASON)

```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```

D*..1.....2.....3.....4.....5.....6.....7..
DMQSET          PR          EXTPROC('MQSET')
D* Connection handle
D HCONN          10I 0 VALUE
D* Object handle
D HOBJ          10I 0 VALUE
D* Count of selectors
D SELCNT        10I 0 VALUE
D* Array of attribute selectors
D SELS          10I 0
D* Count of integer attributes
D IACNT         10I 0 VALUE
D* Array of integer attributes
D INTATR        10I 0
D* Length of character attributes buffer
D CALEN         10I 0 VALUE
D* Character attributes
D CHRATR          *   VALUE
D* Completion code
D CMPCOD        10I 0
D* Reason code qualifying CMPCOD
D REASON        10I 0

```

**IBM i IBM i의 MQSETMP(메시지 핸들 특성 설정)**

MQSETMP 호출은 메시지 핸들의 특성을 설정하거나 수정합니다.

- [1292 페이지의 『구문』](#)
- [1292 페이지의 『사용법 참고』](#)
- [1293 페이지의 『매개변수』](#)
- [1295 페이지의 『RPG 선언』](#)

## 구문

MQSETMP (*Hconn, Hmsg, SetPropOpts, Name, PropDesc, Type, ValueLength, Value, CompCode, Reason*)

## 사용법 참고

- 큐 관리자가 작업 단위를 조정하는 경우에만 이 호출을 사용할 수 있습니다. 다음과 같습니다.
  - 변경사항이 IBM MQ 자원에만 영향을 미치는 경우, 로컬 작업 단위.
  - 변경사항이 다른 자원 관리자에 속한 자원과 IBM MQ 자원에 영향을 미칠 수 있는 경우, 글로벌 작업 단위.로컬 및 글로벌 작업 단위에 대한 세부사항은 [1198 페이지의 『IBM i의 MQBEGIN\(작업 단위 시작\)』](#)의 내용을 참조하십시오.
- 큐 관리자가 작업 단위를 조정하지 않는 환경에서는 MQBACK 대신 적절한 백아웃을 사용하십시오. 이 환경에서는 애플리케이션의 비정상적인 종료로 인해 발생한 암시적 백아웃을 지원할 수도 있습니다.
  - z/OS에서는 다음 호출을 사용하십시오.
    - 배치 프로그램 (IMS 배치 DL/I 프로그램 포함) 은 작업 단위가 IBM MQ 자원에만 영향을 주는 경우 MQBACK 호출을 사용할 수 있습니다. 그러나, 작업 단위가 IBM MQ 자원 및 다른 자원 관리자에 속한 자원(예: Db2) 모두에 영향을 미치는 경우에는 z/OS Recoverable Resource Service(RRS)에서 제공하는 SRRBACK 호출을 사용하십시오. SRRBACK 호출은 RRS 조정을 위해 설정한 자원 관리자에 속한 자원의 변경사항을 백아웃합니다.
    - CICS 애플리케이션은 EXEC CICS SYNCPOINT ROLLBACK 명령을 사용하여 작업 단위를 백아웃해야 합니다. CICS 애플리케이션에 대해 MQBACK 호출을 사용하지 마십시오.
    - IMS 애플리케이션(일괄 처리 DL/I 프로그램 제외)은 작업 단위를 백아웃하기 위해 ROLB과(와) 같은 IMS 호출을 사용해야 합니다. (배치 DL/I 프로그램 이외의) IMS 애플리케이션에 대해 MQBACK 호출을 사용하지 마십시오.
    - IBM i에서는 큐 관리자가 조정하는 로컬 작업 단위에 이 호출을 사용하십시오. 이는 커밋 정의가 작업 레벨에 존재하지 않아야 함을 의미합니다. 즉, **CMTSCOPE(\*JOB)** 매개변수가 있는 STRCMTCTL 명령이 작업에 대해 실행되지 않았어야 합니다.
- 애플리케이션이 작업 단위에서 커밋되지 않은 변경사항으로 종료되는 경우, 해당 변경사항의 배치는 애플리케이션이 정상으로 또는 비정상적으로 종료되는지 여부에 달려 있습니다. 추가적인 세부사항은 [1231 페이지의 『IBM i의 MQDISC\(큐 관리자 연결 끊기\)』](#)의 사용 시 참고사항을 참조하십시오.
- 애플리케이션이 논리 메시지의 세그먼트 또는 그룹에서 메시지를 넣거나 가져올 때 큐 관리자는 마지막 성공한 MQPUT 및 MQGET 호출에 대한 메시지 그룹 및 논리 메시지와 관련된 정보를 보유하고 있습니다. 이 정보는 큐 핸들과 연관되어 있으며 다음과 같은 내용을 포함합니다.
  - MQMD에 있는 *GroupId, MsgSeqNumber, Offset* 및 *MsgFlags* 필드의 값입니다.
  - 메시지가 작업 단위의 일부인지 여부.
  - MQPUT 호출의 경우: 메시지가 지속적인지 또는 비지속적인지 여부.큐 관리자는 다음 각각에 대해 한 세트씩, 그룹 및 세그먼트 정보의 세 개 세트를 유지합니다.
  - 마지막 성공적인 MQPUT 호출(이는 작업 단위의 일부일 수 있음).
  - 큐에서 메시지를 제거한 마지막 성공적인 MQGET 호출(이는 작업 단위의 일부일 수 있음).
  - 큐에서 메시지를 찾아보는 마지막으로 성공한 MQGET 호출(작업 단위의 일부가 될 수 없음).애플리케이션이 작업 단위의 일부로 메시지를 넣거나 가져온 다음 작업 단위를 백아웃하기로 결정한 경우, 그룹 및 세그먼트 정보가 이전 값으로 복원됩니다.
  - MQPUT 호출과 연관된 정보가 첫 번째 성공적인 MQPUT 호출 이전 현재 작업 단위의 큐 핸들에 대한 값으로 복원됩니다.
  - MQGET 호출과 연관된 정보가 현재 작업 단위에서 해당 큐 핸들에 대해 처음으로 성공한 MQGET 호출 전에 가지고 있던 값으로 복원됩니다.



작업 단위가 시작된 이후 애플리케이션을 통해 업데이트되었지만 작업 단위 범위의 외부에 있는 큐에서는 작업 단위가 백업된 경우 그룹 및 세그먼트 정보가 복원되지 않습니다.

작업의 단위가 백아웃될 때 그룹과 세그먼트 정보를 해당 이전 값으로 복구하면 애플리케이션이 여러 작업 단위 전체에 걸쳐 많은 세그먼트로 구성되어 대량 메시지 그룹 또는 대량 논리 메시지를 펼칠 수 있습니다. 그리고 고 작업 단위 중 하나가 실패하면 메시지 그룹 또는 논리 메시지의 정확한 위치에서 재시작하도록 허용합니다.

로컬 큐 관리자에 제한된 큐 스토리지만 있는 경우에는 여러 작업 단위를 사용하면 유용할 수 있습니다. 시스템 장애가 발생한 경우 그러나 애플리케이션은 정확한 위치에 메시지를 넣거나 가져와서 재시작할 수 있기 위한 충분한 정보를 유지보수해야 합니다.

시스템 장애 후 현재 지점에서 재시작하는 방법에 대한 자세한 정보는 [PMOPT \(10자리 부호 있는 정수\)](#)에 설명된 [PMLOGO](#) 옵션 및 [GMOPT \(10자리 부호 있는 정수\)](#)에 설명된 [GMLOGO](#) 옵션을 참조하십시오.

나머지 사용 시 참고사항은 큐 관리자가 작업 단위를 통합할 때만 적용됩니다.

- 작업 단위는 연결 핸들과 동일한 범위를 갖습니다. 특정 작업 단위에 영향을 미치는 모든 IBM MQ 호출은 동일한 연결 핸들을 사용하여 수행해야 합니다. 다른 연결 핸들을 사용하여 실행된 호출(예: 다른 애플리케이션이 발행한 호출)은 다음 작업 단위에 영향을 줍니다. 연결 핸들 범위에 대한 자세한 정보는 [HCONN \(10자리 부호 있는 정수\)](#) - 출력을 참조하십시오.
- 현재 작업 단위의 일부로서 넣거나 검색된 메시지만 이 호출의 영향을 받습니다.
- 작업 단위 내에서 MQGET, MQPUT 또는 MQPUT1 호출을 발행했지만 커밋 또는 백아웃 호출을 발행하지 않고 장시간 실행 중인 애플리케이션이 다른 애플리케이션에서 사용할 수 없는 메시지로 큐를 채울 수 있습니다. 이러한 가능성에 대비하기 위해 관리자는 **MaxUncommittedMsgs** 큐 관리자 속성을 런어웨이 애플리케이션이 큐를 채우는 것을 방지할 만큼 충분히 낮지만 예상 메시징 애플리케이션이 올바르게 작동할 수 있을 만큼 충분히 높은 값으로 설정해야 합니다.

## 매개변수

MQSETMP 호출의 매개변수는 다음과 같습니다.

### HCONN(10자리 부호 있는 정수) - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다.

값이 **HMSG** 매개변수에 지정된 메시지 핸들 작성에 사용된 연결 핸들과 일치해야 합니다.

HCUNAS를 사용하여 메시지 핸들을 작성한 경우, 메시지 핸들의 특성을 설정하는 스레드에 올바른 연결이 설정되어야 합니다. 그렇지 않으면 이유 코드 RC2009와 함께 호출이 실패합니다.

### HMSG(10자리 부호 있는 정수) - 입력

수정할 메시지 핸들입니다. 값은 이전 MQCRTMH 호출에서 리턴합니다.

### SETOPT (MQSMPO) - 입력

메시지 특성의 설정 방법을 제어합니다.

이 구조에서는 애플리케이션이 메시지 특성의 설정 방법을 제어하는 옵션을 지정할 수 있습니다. 구조는 MQSETMP 호출의 입력 매개변수입니다. 자세한 정보는 [MQSMPO](#)의 내용을 참조하십시오.

### PRNAME (MQCHARV) - 입력

설정할 특성의 이름입니다.

특성 이름 사용에 대한 자세한 정보는 [특성 이름 및 특성 이름 제한사항](#)을 참조하십시오.

### PRPDSC (MQPD) - 입출력(I/O)

이 구조는 다음과 같은 특성의 속성을 정의하는 데 사용됩니다.

- 특성이 지원되는 않으면 일어나는 현상
- 특성이 속한 메시지 컨텍스트
- 이동 시 특성이 복사되는 메시지

이 구조에 대한 자세한 정보는 [MQPD](#)를 참조하십시오.

## **TYPE(10자리 부호 있는 정수) - 입력**

설정하는 특성의 데이터 유형. 다음 중 하나가 될 수 있습니다.

### **TYPBOL**

부울. *ValueLength*는 4여야 합니다.

### **TYPBST**

바이트 문자열. *ValueLength*는 0 이상이어야 합니다.

### **TYPI8**

8비트 부호 있는 정수. *ValueLength*는 1이어야 합니다.

### **TYPI16**

16비트 부호 있는 정수. *ValueLength*는 2이어야 합니다.

### **TYPI32**

32비트 부호 있는 정수. *ValueLength*는 4여야 합니다.

### **TYPI64**

64비트 부호 있는 정수. *ValueLength*는 8이어야 합니다.

### **TYPF32**

32비트 부동 소수점 숫자. *ValueLength*는 4여야 합니다.

### **TYPF64**

64비트 부동 소수점 숫자. *ValueLength*는 8이어야 합니다.

### **TYPSTR**

문자열. *ValueLength*는 0 이상이거나 특수 값 VNULL이어야 합니다.

### **TYPNUL**

특성이 존재하지만 값이 없습니다. *ValueLength*는 0이어야 합니다.

## **VALLEN(10자리 부호 있는 정수) - 입력**

*Value* 매개변수에서 특성 값의 길이(바이트)입니다.

널 값이나 문자열 또는 바이트 문자열에 대해서만 0이 유효합니다. 0은 특성이 존재하지만 값에 문자 또는 바이트가 없음을 표시합니다.

*Type* 매개변수를 TYPSTR로 설정한 경우, 값이 0 이상이거나 다음 특수 값이어야 합니다.

### **VNULL**

값이 문자열에 나타난 첫 번째 널로 구분됩니다. 널은 문자열의 일부로 포함되지 않습니다. TYPSTR을 설정하지 않은 경우 이 값이 올바르지 않습니다.

참고: VNULL이 설정된 경우 문자열 종료에 사용된 널 문자는 값의 문자 세트에서 가져온 널입니다.

## **VALUE (1바이트 비트 문자열 x VALLEN) - 입력**

설정할 특성의 값. 버퍼는 값 데이터의 네이처에 적절한 경계에 맞춰야 합니다.

C 프로그래밍 언어에서 매개변수는 pointer-to-void로서 선언됩니다. 임의의 데이터 유형의 주소를 매개변수로 지정할 수 있습니다.

*ValueLength*가 0이면 *Value*가 참조되지 않습니다. 이 경우에 C 또는 System/390 어셈블리로 작성된 프로그램이 전달하는 매개변수 주소는 널(null)일 수 있습니다.

## **CMPCOD(10자리 부호 있는 정수) - 출력**

완료 코드는 다음 중 하나입니다.

### **CCOK**

정상적으로 완료되었습니다.

### **CCFAIL**

호출에 실패했습니다.

## **REASON(10자리 부호 있는 정수) - 출력**

CMPCOD를 규정하는 이유 코드입니다.

CMPCOD가 CCOK일 경우:

**RCNONE**

(0, X'000') 보고할 이유가 없습니다.

If *CMPCOD* is CCWARN:

**RC2421**

(2421, X'0975') 특성을 포함하는 MQRFH2 폴더를 구문 분석할 수 없습니다.

*CMPCOD*가 CCFAIL일 경우:

**RC2204**

(2204, X'089C') 어댑터를 사용할 수 없습니다.

**RC2130**

(2130, X'852') 어댑터 서비스 모듈을 로드할 수 없습니다.

**RC2157**

(2157, X'86D') 1차 및 홈 ASID가 다릅니다.

**RC2004**

(2004, X'07D4') 값 매개변수가 올바르지 않습니다.

**RC2005**

(2005, X'07D5') 값 길이 매개변수가 올바르지 않습니다.

**RC2219**

(2219, X'08AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**RC2460**

(2460, X'099C') 메시지 핸들 포인터가 올바르지 않습니다.

**RC2499**

(2499, X'09C3') 메시지 핸들이 이미 사용 중입니다.

**RC2046**

(2046, X'07FE') 옵션이 올바르지 않거나 일치하지 않습니다.

**RC2482**

(2482, X'09B2') 특성 디스크립터 구조가 올바르지 않습니다.

**RC2442**

(2442, X'098A') 올바르지 않은 특성 이름입니다.

**RC2473**

(2473, X'09A9') 특성 데이터 유형이 올바르지 않습니다.

**RC2472**

(2472, X'09A8') 숫자 형식 오류가 값 데이터에서 발견되었습니다.

**RC2463**

(2463, X'099F') 메시지 특성 설정 옵션 구조가 올바르지 않습니다.

**RC2111**

(2111, X'083F') 특성 이름 코드화 문자 세트 ID가 올바르지 않습니다.

**RC2071**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**RC2195**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

자세한 내용은 [1359 페이지의 『IBM i\(ILE RPG\)의 리턴 코드』](#)의 내용을 참조하십시오.

**RPG 선언**

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQSETMP(HCONN : HMSG : SETOPT :
                        PRNAME : PRPDSC :
                        TYPE : VALLEN : VALUE :
                        CMPCOD : REASON)

```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```
DMQSETMP          PR          EXTPROC('MQSETMP')
D* Connection handle
D HCONN           10I 0 VALUE
D* Message handle
D HMSG            10I 0 VALUE
D* Options that control the action of MQSETMP
D SETOPT          20A
D* Property name
D PRNAME          32A
D* Property descriptor
D PRPDSC          24A
D* Property data type
D TYPE            10I 0 VALUE
D* Length of the Value area
D VALLEN          10I 0 VALUE
D* Property value
D VALUE           *   VALUE
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CompCode
D REASON          10I 0
```

## IBM i IBM i의 MQSTAT(상태 정보 검색)

상태 정보를 검색하려면 MQSTAT 호출을 사용하십시오. 리턴된 상태 정보의 유형은 호출에 지정된 STYPE 값에 의해 판별됩니다.

- [1296 페이지의 『구문』](#)
- [1296 페이지의 『사용법 참고』](#)
- [1296 페이지의 『매개변수』](#)
- [1298 페이지의 『RPG 선언』](#)

### 구문

MQSTAT (HCONN, STYPE, STAT, CMPCOD, REASON)

### 사용법 참고

1. STATAPT 유형을 지정하는 MQSTAT 호출은 이전의 비동기 MQPUT 및 MQPUT1 조작에 대한 정보를 리턴합니다. 호출에서 전달된 MQSTAT 구조는 해당 연결에 대해 처음 기록된 비동기 경고 또는 오류 정보와 함께 완료됩니다. 이후 추가 오류 또는 경고가 이어지면 일반적으로 이 값을 대체하지 않습니다. 그러나, 완료 코드 CCWARN과 함께 오류가 발생하면 완료 코드 CCFAIL과 함께 후속 실패가 대신 리턴됩니다.
2. 연결이 설정되거나 마지막 MQSTAT 호출 이후 발생한 오류가 없으면 CMPCOD로 CCOK, REASON로 RCNONE이 리턴됩니다.
3. 연결 핸들 하에 처리된 비동기 호출의 수는 세 가지 카운터 STSPSC, STSPWC 및 STSPFC를 사용하여 리턴됩니다. 이러한 카운터는 비동기 조작이 성공적으로 처리되거나 경고가 있거나 실패할 때마다 큐 관리자에 의해 증분됩니다. 회계 목적으로 분배 목록에 넣기는 분배 목록당 한 번이 아닌 목적지 큐당 한 번으로 계수됨을 참고하십시오.
4. MQSTAT 호출에 성공하면 이전 오류 정보 또는 수가 재설정됩니다.

### 매개변수

MQSTAT 호출의 매개변수는 다음과 같습니다.

#### Hconn (MQHCONN) - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. Hconn의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

**STYPE(10자리 부호 있는 정수) - 입력**

요청되는 상태 정보의 유형. 유일한 올바른 값은 다음과 같습니다.

**STATAPT**

이전의 비동기 넣기 조작에 대한 정보를 리턴합니다.

**STS (MQSTS) - 입출력(I/O)**

상태 정보 구조. 자세한 내용은 [1175 페이지의 『IBM i의 MQSTS\(상태 보고 구조\)』의 내용을 참조하십시오.](#)

**CMPCOD(10자리 부호 있는 정수) - 출력**

완료 코드는 다음 중 하나입니다.

**CCOK**

정상적으로 완료되었습니다.

**CCFAIL**

호출에 실패했습니다.

**REASON(10자리 부호 있는 정수) - 출력**

CMPCOD를 규정하는 이유 코드입니다.

CMPCOD가 CCOK일 경우:

**RCNONE**

(0, X'000') 보고할 이유가 없습니다.

CMPCOD가 CCFail일 경우:

**RC2374**

(2374, X'946') API 엑시트가 실패했습니다.

**RC2183**

(2183, X'887') API 엑시트를 로드할 수 없습니다.

**RC2219**

(2219, X'8AB') MQI 호출이 이전 호출 완료 전에 입력되었습니다.

**RC2009**

(2009, X'7D9') 큐 관리자에 대한 연결이 유실되었습니다.

**RC2203**

(2203, X'89B') 연결이 종료됩니다.

**RC2018**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**RC2162**

(2162, X'872') 큐 관리자가 중지되고 있습니다.

**RC2102**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**RC2430**

(2430, X'97E') MQSTAT 유형에 오류가 있습니다.

**RC2071**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**RC2424**

(2424, X'978') MQSTS 구조에 오류가 있습니다.

**RC2195**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

**RC2298**

(2298, X'8FA') 요청된 함수는 현재 환경에서 사용할 수 없습니다.

이러한 코드에 대한 자세한 정보는 다음을 참조하십시오.

- [메시지 및 이유 코드](#)

## RPG 선언

```
C*.. 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7
C          CALLP      MQSTAT(HCONN : ETYPE : ERR :
C                               CMPCOD : REASON)
```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```
D.. 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7
DMQSTAT          PR          EXTPROC('MQSTAT')
D* Connection handle
D HCONN          10I 0 VALUE
D* Status information type
D STYPE          10I 0 VALUE
D* Status information
D STATUS          296A
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CompCode
D REASON          10I 0
```

## IBM i IBM i의 MQSUB(구독 등록)

MQSUB 호출은 특정 토픽에 대한 애플리케이션 구독을 등록합니다.

- [1298 페이지의 『구문』](#)
- [1298 페이지의 『사용법 참고』](#)
- [1299 페이지의 『매개변수』](#)
- [1302 페이지의 『RPG 선언』](#)

## 구문

MQSUB (*HCONN*, *SUBDSC*, *HOBJ*, *HSUB*, *CMPCOD*, *REASON*)

## 사용법 참고

- 구독은 사전 정의된 토픽 오브젝트의 축약 이름, 토픽 문자열의 전체 이름을 사용하여 이름 지정된 토픽에 작성되거나 [토픽 문자열 결합](#)에 설명된 대로 두 파트의 연결로 구성됩니다.
- MQSUB 호출 발행 시 액세스 허용 전에 애플리케이션이 실행 중인 사용자 ID에 적절한 레벨의 권한이 있는지 확인하기 위해 큐 관리자가 보안 검사를 수행합니다. 적절한 토픽 오브젝트는 호출에서 제공된 짧은 이름 또는 긴 이름이 제공되는 경우 토픽 계층 구조에서 가장 가까운 짧은 이름 오브젝트 옆에 있습니다. 이 토픽 오브젝트를 대상으로 구독 권한이 설정되었는지, 목적지 큐를 대상으로 출력 권한이 설정되었는지 권한 검사를 수행합니다. SDMAN 옵션을 사용하면 이 토픽 오브젝트와 연관된 관리 큐 이름에 권한 검사가 수행되고, 비관리 큐가 제공되면 **HOBJ** 매개변수로 표시된 큐에 권한 검사가 수행됩니다.
- SOMAN 옵션 사용 시 MQSUB 호출에 리턴된 *HOBJ*를 조회하여 백아웃 임계값, 초과 백아웃 리큐 이름과 같은 속성을 확인할 수 있습니다. 관리 큐의 이름을 조회할 수도 있지만, 이 큐를 직접 열지는 않아야 합니다.
- 구독을 그룹화하면 둘 이상의 그룹이 발행물에 일치하더라도 하나의 구독만 구독 그룹에 전달되게 할 수 있습니다. 구독은 SOGRP 옵션을 사용하여 그룹화하고, 구독을 그룹화하려면 다음을 수행해야 합니다.
  - 동일한 이름 지정 큐(SOMAN 옵션을 사용하지 않음)를 동일한 큐 관리자에 사용 - MQSUB 호출의 **HOBJ** 매개변수로 표시됨
  - 동일한 *SDCID* 공유
  - 동일한 *SDSL* 임

이 속성은 그룹에 있는 것으로 간주되는 구독 세트를 정의하며, 구독이 그룹화된 경우 대체할 수 없는 속성이기도 합니다. *SDSL*을 대체하면 *RC2512*가, 그 외 다른 항목(구독이 그룹화되지 않은 경우 변경 가능)을 대체하면 *RC2515*가 발생합니다.

- *SORES* 옵션을 사용하는 *MQSUB* 호출의 리턴에서 *MQSD*의 필드가 완료됩니다. 구독을 *MQSD*에 적용하기 위해 변경해야 하는 사항과 함께 *SOALT* 옵션을 사용하는 *MQSUB* 호출에 직접 리턴된 *MQSD*가 전달될 수 있습니다. 일부 필드는 표에 명시된 대로 특별한 고려사항이 적용됩니다.

표 752. <i>MQSUB</i> 의 <i>MQSD</i> 출력	
<b>MQSD의 필드 이름</b>	<b>특별 고려사항</b>
액세스 또는 작성 옵션	<i>MQSUB</i> 호출의 리턴에서 이러한 옵션이 설정되지 않습니다. 나중에 <i>MQSUB</i> 호출에 <i>MQSD</i> 를 재사용하는 경우, 필요한 옵션이 명시적으로 설정되어야 합니다.
지속성 옵션, 목적지 옵션, 등록 옵션 및 와일드카드 옵션	이 옵션은 필요에 따라 설정됩니다.
발행물 옵션	이 옵션은 <i>SOCRE</i> 에만 적용할 수 있는 <i>SONEWP</i> 를 제외하고, 필요에 따라 설정됩니다.
기타 옵션	이 옵션은 <i>MQSUB</i> 호출의 리턴에서 변경되지 않습니다. 이는 API 호출이 실행되고 구독에서 저장되지 않는 방법을 제어합니다. <i>MQSD</i> 를 재사용하는 후속 <i>MQSUB</i> 호출에서 필수로 설정해야 합니다.
ObjectName	이 입력 전용 필드는 <i>MQSUB</i> 호출의 리턴에서 변경되지 않습니다.
ObjectString	이 입력 전용 필드는 <i>MQSUB</i> 호출의 리턴에서 변경되지 않습니다. 버퍼가 제공된 경우, 사용된 전체 토픽 이름이 <i>SDRO</i> 필드에 리턴됩니다.
AlternateUserId 및 AlternateSecurityId	이 입력 전용 필드는 <i>MQSUB</i> 호출의 리턴에서 변경되지 않습니다. 이는 API 호출이 실행되고 구독에서 저장되지 않는 방법을 제어합니다. <i>MQSD</i> 를 재사용하는 후속 <i>MQSUB</i> 호출에서 필수로 설정해야 합니다.
SubExpiry	<i>SORES</i> 옵션을 사용하는 <i>MQSUB</i> 호출의 리턴에서 이 파일은 남은 만기 시간이 아닌 원래 구독 만기로 설정됩니다. <i>SOALT</i> 옵션을 사용하여 <i>MQSUB</i> 호출에서 <i>MQSD</i> 를 재사용하는 경우, 구독 만기가 다시 계수를 시작하도록 재설정합니다.
SubName	이 필드는 <i>MQSUB</i> 호출의 입력 필드이고 출력에서 변경되지 않습니다.
SubUserData 및 SelectionString	버퍼가 제공되고 <i>VCHRP</i> 의 버퍼 길이가 양수인 경우, <i>SORES</i> 옵션을 사용하는 <i>MQSUB</i> 호출의 출력에서 이 변수 길이 필드가 리턴됩니다. 버퍼가 제공되지 않으면 길이만 <i>MQCHARV</i> 의 <i>VCHRL</i> 필드에 리턴됩니다. 제공된 버퍼가 필드 리턴에 필요한 공간보다 작으면 <i>VCHRP</i> 바이트만 제공된 버퍼에 리턴됩니다.  나중에 <i>SOALT</i> 옵션을 사용하는 <i>MQSUB</i> 호출에 <i>MQSD</i> 를 재사용하고 버퍼가 제공되지 않지만 0이 아닌 <i>VCHRL</i> 이 제공되는 경우 그리고 해당 길이가 필드의 기존 길이에 일치하는 경우, 필드가 대체되지 않습니다.
SubCorrelId 및 PubAccountingToken	<i>SOSCID</i> 를 사용하지 않으면 큐 관리자에서 <i>SDCID</i> 를 생성합니다. <i>SOSETI</i> 를 사용하지 않는 경우, 큐 관리자에서 <i>SDACC</i> 가 생성됩니다.  이 필드는 <i>SORES</i> 옵션을 사용하여 <i>MQSUB</i> 호출의 <i>MQSD</i> 에서 리턴됩니다. 큐 관리자에 의해 생성되면 생성된 값이 <i>SOCRE</i> 또는 <i>SOALT</i> 옵션을 사용하여 <i>MQSUB</i> 호출에 리턴됩니다.
PubPriority, SubLevel & PubAppIdentityData	이 필드는 <i>MQSD</i> 에 리턴됩니다.
ResObjectString	버퍼가 제공되면 이 출력 전용 필드가 <i>MQSD</i> 에 리턴됩니다.

## 매개변수

*MQSUB* 호출의 매개변수는 다음과 같습니다.

### **HCONN(10자리 부호 있는 정수) - 입력**

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. *HCONN*의 값은 이전 *MQCONN* 또는 *MQCONNX* 호출에 의해 리턴되었습니다.



## SUBDSC (MQSD) - 입출력(I/O)

애플리케이션에 사용 등록된 오브젝트를 식별하는 구조입니다. 1159 페이지의 『IBM i의 MQSD(구독 디스크립터)』의 내용을 참조하십시오.

## HOBJ(10자리 부호 있는 정수) - 입출력(I/O)

이 핸들은 이 구독으로 송신된 메시지를 가져오기 위해 설정된 액세스 권한을 나타냅니다. 이 메시지를 특정 큐에 저장하거나, 특정 큐 없이 스토리지를 관리하도록 큐 관리자에 요청할 수 있습니다.

오브젝트 핸들.

특정 큐를 사용하는 경우, 작성 시간에 구독과 연관되어 있어야 합니다. 두 가지 방법으로 수행할 수 있습니다.

- SDCRT 옵션으로 MQSUB을 호출할 때 이 핸들을 제공합니다. 이 핸들은 호출에서 입력 매개변수로 제공된 경우 OOINP\*, OOOOUT(예: 리포트 큐인 경우) 또는 OOBW 옵션 중 하나 이상을 사용하여 큐의 이전 MQOPEN 호출에서 리턴된 올바른 오브젝트 핸들이어야 합니다. 그렇지 않은 경우에는 RC2019와 함께 호출이 실패합니다. 토픽 오브젝트로 해석되는 알리어스 큐에 대한 오브젝트 핸들일 수는 없습니다. 해당 경우, RC2019와 함께 호출이 실패합니다.
- DEFINE SUB MQSC 명령을 사용하고 해당 명령에 큐 오브젝트의 이름을 제공합니다.

큐 관리자가 이 구독으로 송신한 메시지의 스토리지를 관리하는 경우, SOMAN 옵션을 사용하고 매개변수 값을 HONONE으로 설정하여 구독 작성 시 이를 표시해야 합니다. 큐 관리자는 호출에서 출력 매개변수로 핸들을 리턴하고 리턴된 핸들이 관리 핸들로 인식됩니다. HONONE을 지정하고 SOMAN은 지정하지 않은 경우, RC2019와 함께 호출이 실패합니다.

큐 관리자가 리턴한 관리 핸들을 MQINQ 호출 또는 MQCLOSE에서, 찾아보기 옵션의 유무에 상관없이, MQGET 또는 MQCB 호출에서 사용할 수 있습니다. MQPUT, MQSET 또는 후속 MQSUB에서는 사용할 수 없으며, 이렇게 하려고 하면 MQPUT에는 RC2039, MQSET에는 RC2040, MQSUB에는 RC2038이 표시되면서 실패합니다.

MQSD 구조의 OPTS 필드에 있는 SORES 옵션을 사용하여 이 구독을 재개하는 경우, HONONE이 지정되면 핸들이 이 매개변수로 애플리케이션에 리턴될 수 있습니다. 구독이 관리 핸들을 사용하는지 여부에 따라 이 기능을 사용합니다. 이것은 DEFINE SUB 명령에 정의된 구독 큐에 대한 핸들을 원하는 경우 DEFINE SUB를 사용하여 작성한 구독에 유용할 수 있습니다. 관리 과정에 따라 작성한 구독을 재개하는 경우, OOINPQ 및 OOBW를 사용하여 큐를 엽니다. 다른 옵션이 필요하면 애플리케이션이 구독 큐를 명시적으로 열고 호출에서 오브젝트 핸들을 제공해야 합니다. 큐를 여는 데 문제가 있으면 RC2522와 함께 호출이 실패합니다. HOBJ를 제공한 경우, 이것은 원래 MQSUB 호출의 HOBJ와 같습니다. 즉, MQOPEN 호출에서 리턴된 오브젝트 핸들이 제공되는 경우 이 핸들은 이전에 사용한 것과 동일한 큐에 대한 것이어야 하며, 그렇지 않으면 RC2019와 함께 호출이 실패합니다.

MQSD 구조의 OPTS 필드에 SOALT 옵션을 사용하여 이 구독을 대체하는 경우, 다른 HOBJ가 제공될 수 있습니다. 이 매개변수를 통해 이전에 식별된 큐로 전달된 발행물은 해당 큐에 그대로 있고, HOBJ 매개변수가 다른 큐를 나타내는 경우 해당 메시지를 가져오는 것은 애플리케이션이 담당합니다.

다양한 구독 옵션과 이 매개변수의 사용이 다음 표에 요약되어 있습니다.

표 753. 여러 구독 옵션에서 Hobj 사용		
옵션	Hobj	설명
SOCRT + SOMAN	입력에서 무시됨	큐 관리자가 관리하는 메시지 스토리지와 함께 구독을 작성합니다.
SOCRT	올바른 오브젝트 핸들	특정 큐를 메시지 목적으로 제공하는 구독을 작성합니다.
SORES	HONONE	이전에 작성된 구독(관리 또는 비관리)을 계속하고 큐 관리자가 애플리케이션에 사용될 오브젝트 핸들을 리턴하도록 합니다.
SORES	유효하고 일치하는 오브젝트 핸들	특정 큐를 메시지의 목적으로 사용하는 이전에 작성된 구독을 계속하고 특정 열기 옵션과 함께 오브젝트 핸들을 사용합니다.
SOALT + SOMAN	HONONE	이전에 특정 큐를 사용했던 기존 구독을 관리되는 형태로 대체합니다.



표 753. 여러 구독 옵션에서 <i>Hobj</i> 사용 (계속)		
옵션	<i>Hobj</i>	설명
SOALT	올바른 오브젝트 핸들	특정 큐(관리 큐 또는 다른 특정 큐에 속함)를 사용하도록 기존 구독을 대체합니다.

제공되든 리턴되든 *HOBJ*는 발행물을 수신해야 하는 후속 MQGET 호출에 지정해야 합니다.

MQCLOSE 호출이 발행되거나 핸들 범위를 정의하는 처리 단위가 종료되면 *HOBJ* 핸들의 유효성이 정지합니다. 리턴되는 오브젝트 핸들의 범위는 호출에서 지정된 연결 핸들의 범위와 동일합니다. 핸들 범위에 대한 정보는 *HCONN*을 참조하십시오. *HOBJ* 핸들의 MQCLOSE는 *HSUB* 핸들에 아무런 영향이 없습니다.

#### **HSUB(10자리 부호 있는 정수) - 출력**

이 핸들은 작성된 구독을 나타냅니다. 다음 두 가지 추가 조작에 사용할 수 있습니다.

- 구독할 때 SOPUBR 옵션이 사용되는 경우 구독을 송신하도록 요청하기 위해 후속 MQSUBRQ 호출에 사용할 수 있습니다.
- 작성된 구독을 제거하려면 후속 MQCLOSE 호출에 사용합니다. MQCLOSE 호출이 발행되거나 핸들 범위를 정의하는 처리 단위가 종료되면 *HSUB* 핸들의 유효성이 정지합니다. 리턴되는 오브젝트 핸들의 범위는 호출에서 지정된 연결 핸들의 범위와 동일합니다. *HSUB* 핸들의 MQCLOSE는 *HOBJ* 핸들에 아무런 영향이 없습니다.

이 핸들을 MQGET 또는 MQCB 호출로 전달할 수 없습니다. **HOB**J 매개변수를 사용해야 합니다. 이 핸들을 다른 IBM MQ 호출로 전달하면 RC2019가 표시됩니다.

#### **CMPCOD(10자리 부호 있는 정수) - 출력**

완료 코드는 다음 중 하나입니다.

##### **CCOK**

성공적인 완료

##### **CCWARN**

경고(일부 완료)

##### **CCFAIL**

호출에 실패했습니다.

#### **REASON(10자리 부호 있는 정수) - 출력**

*CMPCOD*를 규정하는 이유 코드입니다.

*CMPCOD*가 CCOK일 경우:

##### **RCNONE**

(0, X'000') 보고할 이유가 없습니다.

*CMPCOD*가 CCFail일 경우:

##### **RC2019**

(2019 X'07E3') 오브젝트 핸들이 올바르지 않습니다.

##### **RC2046**

(2046 X'07FE') 오브젝트가 올바르지 않거나 일관되지 않습니다.

##### **RC2085**

(2085 X'0825') 오브젝트 ID를 찾을 수 없습니다.

##### **RC2161**

(2161 X'0871') 큐 관리자가 정지 중

##### **RC2298**

(2298 X'08FA') 함수가 지원되지 않습니다.

##### **RC2424**

(2424 X'0978') 구독 디스크립터(MQSD)가 올바르지 않습니다.

**RC2425**

(2441 X'979') 토픽 문자열이 올바르지 않습니다.

**RC2428**

(2428 X'097C') 지정된 구독 이름이 기존 구독과 일치하지 않습니다.

**RC2429**

(2429 X'097D') 구독 이름이 있으며 다른 애플리케이션에서 사용 중입니다.

**RC2431**

(2431 X'097F') SubUserData 필드가 올바르지 않습니다.

**RC2432**

(2432 X'0980') 구독이 있습니다.

**RC2434**

(2434 X'0982') 구독 이름이 기존 구독과 일치합니다.

**RC2440**

(2440 X'0988') SubName 필드가 올바르지 않습니다.

**RC2441**

(2441 X'0989') Objectstring 필드가 올바르지 않습니다.

**RC2435**

(2435 X'0983') SDALT를 사용하여 속성을 변경할 수 없거나, 구독이 SDIMM으로 작성되었습니다.

**RC2436**

(2436 X'0984') SODUR 옵션이 올바르지 않습니다.

**RC2459**

(2459, X'99B') 선택 문자열 구문 오류입니다.

**RC2503**

(2503 X'09C7') 구독하는 토픽에 대해 현재 MQSUB 호출이 금지됩니다.

**RC2519**

(2519, X'9D7') 선택 문자열이 MQCHARV 구조 사용 방법 설명에 지정된 내용과 다릅니다.

**RC2551**

(2551, X'9F7') 지정된 선택 문자열을 사용할 수 없습니다.

**RPG 선언**

```

C*..1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQSUB(HCONN : SUBDSC : HOBJ :
C                               HSUB : CMPCOD : REASON)

```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```

D*..1.....2.....3.....4.....5.....6.....7..
DMQSUB          PR          EXTPROC('MQSUB')
D* Connection handle
D HCONN          10I 0 VALUE
D* Subscription descriptor
D SUBDSC          400A
D* Object handle for queue
D HOBJ          10I 0
D* Subscription object handle
D HSUB          10I 0
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CompCode
D REASON          10I 0

```

**IBM i IBM i의 MQSUBRQ(구독 요청)**

MQSUBRQ 호출은 구독에 대한 요청을 작성합니다.

- [1303 페이지의 『구문』](#)

- [1303 페이지의 『사용법 참고』](#)
- [1303 페이지의 『매개변수』](#)
- [1304 페이지의 『RPG 선언』](#)

## 구문

MQSUBRQ (*HCONN, HSUB, ACTION, SUBROPT, CMPCOD, REASON*)

## 사용법 참고

다음 사용법 참고사항은 SRAPUB의 사용에 적용됩니다.

1. 이 verb가 완료되는 경우 지정된 구독과 일치하는 보유된 발행이 구독에 전송되고 구독을 작성한 원래 MQSUB verb에 리턴한 HOBJ를 사용하는 MQGET 또는 MQCB를 사용하여 검색할 수 있습니다.
2. 토픽이 와일드카드가 포함된 구독을 작성한 원래 MQSUB verb에 구독된 경우 둘 이상의 보유된 발행물을 전송할 수 있습니다. 이 호출의 결과로 전송된 발행물 수는 SBROPT 구조의 SRNMP 필드에 기록됩니다.
3. 이 verb가 RC2437의 이유 코드로 완료되면 현재 지정된 토픽에 보유된 발행물이 없습니다.
4. 이 동사가 RC2525 또는 RC2526의 이유 코드로 완료되면 현재 지정된 토픽에 보유된 발행물이 있지만 전달할 수 없음을 의미하는 오류가 발생했습니다.
5. 이 호출을 작성하기 전에 애플리케이션에 토픽에 대한 현재 구독이 있어야 합니다. 구독이 애플리케이션의 이전 인스턴스에서 작성되고 구독에 대해 유효한 핸들을 사용할 수 없는 경우 애플리케이션은 SORES 옵션이 있는 MQSUB를 먼저 호출하여 이 호출에 사용할 핸들을 확보해야 합니다.
6. 발행물은 이 애플리케이션의 현재 구독과 함께 사용할 등록되는 목적지에 전송됩니다. 발행물을 다른 곳으로 전송해야 하는 경우 구독은 먼저 SOALT 옵션이 있는 MQSUB 호출을 사용하여 대체되어야 합니다.

## 매개변수

MQSUBRQ 호출에 다음 매개변수가 있습니다.

### HCONN(10자리 부호 있는 정수) - 입력

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. HCONN의 값은 이전 MQCONN 또는 MQCONNX 호출에 의해 리턴되었습니다.

CICS 용 z/OS 애플리케이션에서 MQCONN 호출을 생략할 수 있으며 HCONN에 다음 값을 지정할 수 있습니다.

#### HCDEFH

기본 연결 핸들

### HSUB(10자리 부호 있는 정수) - 입력

이 핸들은 업데이트가 요청되는 구독을 나타냅니다. HSUB의 값이 이전 MQSUB 호출에서 리턴됩니다.

### ACTION(10자리 부호 있는 정수) - 입력

이 매개변수는 구독에 요청되는 특별한 조치를 제어합니다. 다음 중 하나(하나만)는 지정되어야 합니다.

#### SRAPUB

이 조치는 지정된 토픽에 업데이트 발행을 전송할 것을 요청합니다. 이는 일반적으로 구독자가 구독을 작성할 때 MQSUB 호출에 옵션 SOPUBR을 지정한 경우 사용됩니다. 큐 관리자에게 토픽에 대해 보유된 발행물이 있는 경우 이는 구독자에게 전송됩니다. 그렇지 않은 경우 호출은 실패합니다. 애플리케이션은 보유된 발행물이 전송되는 경우 해당 발행물의 MQIsRetained 메시지 특성으로 표시됩니다.

HSUB 매개변수로 표시된 기존 구독의 토픽에 와일드 카드를 포함할 수 있기 때문에 구독자는 여러 보유된 발행을 수신할 수 있습니다.

### SBROPT(MQSRO) - 입출력(I/O)

이러한 옵션은 MQSUBRQ의 조치를 제어하며 세부사항은 [569 페이지의 『MQSRO - 구독 요청 옵션』](#)의 내용을 참조하십시오.

## CMPCOD(10자리 부호 있는 정수) - 출력

완료 코드는 다음 중 하나입니다.

### CCOK

성공적인 완료

### CCWARN

경고(일부 완료)

### CCFAIL

호출에 실패했습니다.

## Reason(10자리 부호 있는 정수) - 출력

CMPCOD를 규정하는 이유 코드입니다.

CMPCOD가 CCOK인 경우:

### RCNONE

(0, X'000') 보고할 이유가 없습니다.

CMPCOD가 CCFAIL인 경우:

### RC2298

2298 (X'08FA') 요청된 함수를 현재 환경에서 사용할 수 없습니다.

### RC2437

2437 (X'0985') 현재 이 토픽에 대해 저장된 보유된 발행물이 없습니다.

### RC2046

2046 (X'07FE') 올바르지 않은 옵션을 포함하는 옵션 매개변수 또는 필드 또는 올바르지 않은 옵션의 결합입니다.

### RC2161

2161 (X'0871') 큐 관리자 정지 중

### RC2438

2438 (X'0986') MQSUBRQ 호출에서 구독 요청 옵션 MQSRO는 올바르지 않습니다.

## RPG 선언

```
C*.1.....2.....3.....4.....5.....6.....7..
C                                CALLP    MQSUBRQ(HCONN : HSUB : ACTION :
C                                SBROPT : CMPCOD : REASON)
```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```
D*.1.....2.....3.....4.....5.....6.....7..
DMQSUBRQ          PR          EXTPROC('MQSUBRQ')
D* Connection handle
D HCONN          10I 0 VALUE
D* Subscription handle
D HSUB          10I 0 VALUE
D* Action requested on the subscription
D ACTION          10I 0 VALUE
D* Subscription Request Options
D SBROPT          16A
D* Completion code
D CMPCOD          10I 0
D* Reason code qualifying CompCode
D REASON          10I 0
```

## IBM i IBM i의 오브젝트 속성

이 토픽의 컬렉션은 MQINQ 함수 호출의 제목이 될 수 있는 IBM MQ 오브젝트만 나열하고 조회할 수 있는 속성 및 사용될 선택자에 대한 세부사항을 제공합니다.

## 큐의 속성

다양한 유형의 큐 정의와 각각에 지원되는 속성에 대해 알아보려면 이 정보를 사용하십시오.

**큐의 유형:** 큐 관리자는 다음 유형의 큐 정의를 지원합니다.

### 로컬 큐

메시지를 저장하는 물리적 큐입니다. 큐는 로컬 큐 관리자에 존재합니다.

로컬 큐 관리자에 연결된 애플리케이션이 이 유형의 큐에 메시지를 넣거나 큐로부터 메시지를 제거할 수 있습니다. **QType** 큐 속성의 값은 QTLOC입니다.

### 공유 큐

메시지를 저장하는 물리적 큐입니다. 큐는 공유 저장소를 소유한 큐 공유 그룹에 속한 모든 큐 관리자에 액세스할 수 있는 공유 저장소에 존재합니다.

큐 공유 그룹 내의 큐 관리자에 연결된 애플리케이션이 이 유형의 큐에서 메시지를 배치하거나 큐로부터 메시지를 제거할 수 있습니다. 해당 큐는 사실상 로컬 큐와 동일합니다. **QType** 큐 속성의 값은 QTLOC입니다.

- 공유 큐는 z/OS에서만 지원됩니다.

### 클러스터 큐

메시지를 저장하는 물리적 큐입니다. 큐는 로컬 큐 관리자 또는 로컬 큐 관리자와 동일한 클러스터에 속한 하나 이상의 큐 관리자에 존재합니다.

로컬 큐 관리자에 연결된 애플리케이션은 큐의 위치에 상관없이 이 유형의 큐에 메시지를 넣을 수 있습니다. 큐의 인스턴스가 로컬 큐 관리자에 존재하는 경우, 큐는 로컬 큐와 동일하게 작동하며 로컬 큐 관리자에 연결된 애플리케이션은 큐로부터 메시지를 제거할 수 있습니다. **QType** 큐 속성의 값은 QTCLUS입니다.

### 알리어스 큐

물리적 큐가 아닙니다. 로컬 큐의 대체 이름입니다. 알리어스가 해석되는 로컬 큐의 이름은 알리어스 큐 정의의 일부입니다.

로컬 큐 관리자에 연결된 애플리케이션은 알리어스 큐에 메시지를 넣거나 알리어스 큐로부터 메시지를 제거할 수 있습니다. 알리어스가 해석되는 로컬 큐에 메시지를 넣고 제거합니다. **QType** 큐 속성의 값은 QTALS입니다.

### 리모트 큐

물리적 큐가 아닙니다. 리모트 큐 관리자에 존재하는 큐의 로컬 정의입니다. 리모트 큐의 로컬 정의는 리모트 큐 관리자에 메시지를 라우팅하는 방법을 로컬 큐 관리자에 알려주는 정보를 포함합니다.

로컬 큐 관리자에 연결된 애플리케이션은 리모트 큐에 메시지를 넣을 수 있습니다. 메시지를 리모트 큐 관리자에 라우팅하는 데 사용되는 로컬 전송 큐에 메시지를 넣습니다. 애플리케이션이 리모트 큐로부터 메시지를 제거할 수 없습니다. **QType** 큐 속성의 값은 QTREM입니다.

리모트 큐 정의는 다음 용도로도 사용할 수 있습니다.

- 응답 큐 알리어스

이 경우 정의 이름은 응답 대상 큐의 이름입니다. 자세한 정보는 [응답 대상 큐 별명 알리어스](#)를 참조하십시오.

- 큐 관리자 알리어스

이 경우 정의 이름은 큐의 이름이 아니라 큐 관리자의 알리어스입니다. 자세한 정보는 [큐 관리자 알리어스 정의](#)를 참조하십시오.

### 모델 큐

물리적 큐가 아닙니다. 로컬 큐가 생성될 수 있는 큐 속성 세트입니다.

이 유형의 큐에는 메시지를 저장할 수 없습니다.

일부 큐 속성은 모든 유형의 큐에 적용되고, 그 외 큐 속성은 특정 유형의 큐에만 적용됩니다. 속성이 적용되는 큐의 유형은 [1306 페이지의 표 754](#) 및 이후 표에서 "X"로 표시됩니다.

[1306 페이지의 표 754](#)에서는 큐에 특정한 속성을 요약합니다. 속성은 알파벳 순서로 설명합니다.

표에 표시된 속성의 이름은 MQINQ 및 MQSET 호출에 사용되는 이름입니다. 속성을 정의, 대체 또는 표시하기 위해 MQSC 명령이 사용되는 경우, 대체 짧은 이름이 사용됩니다. 자세한 정보는 MQSC 명령의 내용을 참조하십시오.

다음 표에서, 열은 다음과 같이 적용됩니다.

- 로컬 큐에 대한 열은 공유 큐에도 적용됩니다.
- 모델 큐에 대한 열은 모델 큐로부터 작성된 로컬 큐에 상속되는 속성을 나타냅니다.
- 클러스터 큐에 대한 열은 조회 용도로만 또는 조회와 출력 용도로 클러스터 큐를 열 때 조회할 수 있는 속성을 나타냅니다. 조회는 물론 입력, 찾아보기 또는 설정을 한 번 이상 수행하기 위해 클러스터 큐를 연 경우, 로컬 큐에 대한 열이 대신 적용됩니다.

속성	설명	로컬	모델	알리어스	원격	클러스터
<u>AlterationDate</u>	정의가 마지막으로 변경된 날짜	X		X	X	
<u>AlterationTime</u>	정의가 마지막으로 변경된 시간	X		X	X	
<u>BackoutRequeueQName</u>	초과 백아웃 큐 이름	X	X			
<u>BackoutThreshold</u>	백아웃 임계값	X	X			
<u>BaseQName</u>	알리어스가 해석되는 큐 이름			X		
<u>ClusterChannelName</u>	클러스터 송신자 채널 이름	✓	✓			
<u>ClusterName</u>	큐가 속한 클러스터의 이름	X		X	X	
<u>ClusterNameList</u>	큐가 속한 클러스터의 이름을 포함하는 이름 목록 오브젝트의 이름	X		X	X	
<u>CreationDate</u>	큐 작성 날짜	X				
<u>CreationTime</u>	큐 작성 시간	X				
<u>CurrentQDepth</u>	현재 큐 용량	X				
<u>DefBind</u>	기본 바인딩	X		X	X	X
<u>DefinitionType</u>	큐 정의 유형	X	X			
<u>DefInputOpenOption</u>	기본 입력 열기 옵션	X	X			
<u>DefPersistence</u>	기본 메시지 지속성	X	X	X	X	X
<u>DefPriority</u>	기본 메시지 우선순위	X	X	X	X	X
<u>DistLists</u>	분배 목록 지원	X	X			
<u>HardenGetBackout</u>	정확한 백아웃 수의 유지보수 여부	X	X			
<u>InhibitGet</u>	큐에 대한 Get 조작의 허용 여부 제어	X	X	X		
<u>InhibitPut</u>	큐에 대한 Put 조작의 허용 여부 제어	X	X	X	X	X

표 754. 큐의 속성 (계속)						
속성	설명	로컬	모델	알리어스	원격	클러스터
<u>InitiationQName</u>	이니시에이션 큐의 이름	X	X			
<u>MaxMsgLength</u>	최대 메시지 길이(바이트)	X	X			
<u>MaxQDepth</u>	최대 큐 용량	X	X			
<u>MediaLog</u>	지정된 큐의 매체 복원에 필요한 가장 오래된 로그 익스텐트(또는 IBM i의 가장 오래된 저널 수신자)의 ID	✓	✓			
<u>MsgDeliverySequence</u>	메시지 전달 순서	X	X			
<u>OpenInputCount</u>	입력하기 위한 열기 수	X				
<u>OpenOutputCount</u>	출력하기 위한 열기 수	X				
<u>ProcessName</u>	프로세스 이름	X	X			
<u>QDepthHighEvent</u>	큐 용량 상한 이벤트의 생성 여부 제어	X	X			
<u>QDepthHighLimit</u>	큐 용량에 대한 상한	X	X			
<u>QDepthLowEvent</u>	큐 용량 하한 이벤트의 생성 여부 제어	X	X			
<u>QDepthLowLimit</u>	큐 용량에 대한 하한	X	X			
<u>QDepthMaxEvent</u>	큐 가득 참 이벤트의 생성 여부 제어	X	X			
<u>QDesc</u>	큐 설명	X	X	X	X	X
<u>QName</u>	큐 이름	X		X	X	X
<u>QServiceInterval</u>	큐 서비스 간격 대상	X	X			
<u>QServiceIntervalEvent</u>	서비스 간격 높음 이벤트 또는 서비스 간격 확인 이벤트의 생성 여부 제어	X	X			
<u>QType</u>	큐 유형	X		X	X	X
<u>RemoteQMgrName</u>	리모트 큐 관리자의 이름				X	
<u>RemoteQName</u>	리모트 큐의 이름				X	
<u>RetentionInterval</u>	보유 간격	X	X			
<u>Scope</u>	큐에 대한 항목이 셀 디렉토리에 존재하는지 여부 제어	X		X	X	
<u>Shareability</u>	큐 공유 가용성	X	X			
<u>TriggerControl</u>	트리거 제어	X	X			

표 754. 큐의 속성 (계속)						
속성	설명	로컬	모델	알리어스	원격	클러스터
<u>TriggerData</u>	트리거 데이터	X	X			
<u>TriggerDepth</u>	트리거 용량	X	X			
<u>TriggerMsgPriority</u>	트리거에 대한 임계값 메시지 우선순위	X	X			
<u>TriggerType</u>	트리거 유형	X	X			
<u>사용법</u>	큐 용도	X	X			
<u>XmitQName</u>	전송 큐 이름				X	

**IBM i IBM i의 AlterationDate(12바이트 문자열)**

정의가 마지막으로 변경된 날짜.

표 755. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	

정의가 마지막으로 변경된 날짜입니다. 날짜의 형식은 YYYY-MM-DD이며, 12바이트 길이로 만들기 위해 뒤에 두 개의 후미 공백으로 채워집니다(예: 1992-09-23--이며, 여기서 --은 두 개의 공백 문자를 나타냄).

큐 관리자가 작동하면 특정 속성 값(예: *CurrentQDepth*)이 변경됩니다. 이러한 속성에 대한 변경사항은 *AlterationDate*에 영향을 주지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CAALTD 선택자를 사용하십시오. 이 속성의 길이는 LNDATE에서 제공됩니다.

**IBM i IBM i의 AlterationTime(8바이트 문자열)**

정의가 마지막으로 변경된 시간.

표 756. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	

정의가 마지막으로 변경된 시간입니다. 시간 형식은 24시간 클럭을 사용하는 HH.MM.SS이며, 시간이 10 미만이면 앞에 0이 있습니다(예: 09.10.20). 시간은 로컬 시간입니다.

큐 관리자가 작동하면 특정 속성 값(예: *CurrentQDepth*)이 변경됩니다. 이러한 속성에 대한 변경사항은 *AlterationTime*에 영향을 주지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CAALTT 선택자를 사용하십시오. 이 속성의 길이는 LNTIME에서 제공됩니다.

**IBM i IBM i의 BackoutRequeueQName(48바이트 문자열)**

초과 백아웃 큐 이름.



표 757. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

WebSphere Application Server 내부에서 실행되는 애플리케이션과 IBM MQ Application Server Facilities를 사용하는 애플리케이션은 이 속성을 사용하여 백아웃된 메시지가 이동해야 하는 위치를 판별합니다. 그 외 모든 애플리케이션의 경우, 이 값의 조회를 허용하는 것 외에 큐 관리자는 속성 값을 기반으로 어떠한 조치도 취하지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CABRQN 선택자를 사용하십시오. 이 속성의 길이는 LNQN에서 제공됩니다.

### IBM i IBM i의 BackoutThreshold(10자리의 부호 있는 정수)

백아웃 임계값.

표 758. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

WebSphere Application Server 내부에서 실행되는 애플리케이션과 IBM MQ Application Server Facilities를 사용하는 애플리케이션은 이 속성을 사용하여 메시지가 백아웃되어야 하는지 판별합니다. 그 외 모든 애플리케이션의 경우, 이 값의 조회를 허용하는 것 외에 큐 관리자는 속성 값을 기반으로 어떠한 조치도 취하지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IABTHR 선택자를 사용하십시오.

### IBM i IBM i의 BaseQName(48바이트 문자열)

알리어스가 해석되는 큐 이름입니다.

표 759. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
		X		

이는 로컬 큐 관리자에 정의된 큐의 이름입니다. (큐 이름에 대한 자세한 정보는 MQOD의 ODOM 필드에 대한 설명을 참조하십시오. 큐는 다음 유형 중 하나입니다.

#### QTLOC

로컬 큐.

#### QTREM

리모트 큐의 로컬 정의입니다.

#### QTCLUS

클러스터 큐.

이 속성의 값을 판별하려면 MQINQ 호출에서 CABASQ 선택자를 사용하십시오. 이 속성의 길이는 LNQN에서 제공됩니다.

### IBM i IBM i의 BaseType(정수 매개변수 구조)

알리아스가 해석되는 오브젝트의 유형.

표 760. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
		X		

이 속성의 값은 다음 중 하나입니다.

**OTQ**

기본 오브젝트 유형이 큐입니다.

**OTTOP**

기본 오브젝트 유형이 토픽입니다.

**IBM i IBM i의 CFStrucName(12바이트 문자열)**

커플링 기능 구조 이름.

표 761. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

이것은 큐의 메시지가 저장되는 커플링 기능 구조의 이름입니다. 이름의 첫 번째 문자는 A-Z이고, 나머지 문자는 A-Z, 0-9 또는 공백입니다.

커플링 기능에서 구조의 전체 이름은 **QSGName** 큐 관리자 속성 값에 **CFStrucName** 큐 속성의 값을 접미사로 붙여 구합니다.

이 속성은 공유 큐에만 적용됩니다. *QSGDisp*의 값이 *QSGDSH*가 아니면 무시됩니다.

이 속성의 값을 판별하려면 *MQINQ* 호출에서 *CACFSN* 선택자를 사용하십시오. 이 속성의 길이는 *LNCFSN*을 통해 제공됩니다.

**z/OS** 이 속성은 z/OS에서만 지원됩니다.

**ClusterChannelName(20바이트 문자열)**

**ClusterChannelName** 이름은 이 큐를 전송 큐로 사용하는 클러스터-송신자 채널의 일반 이름입니다. 이 속성은 해당 클러스터 전송 큐에서 클러스터 수신자 채널로 메시지를 송신할 클러스터 송신자 채널을 지정합니다.

표 762. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

기본 큐 관리자 구성은 모든 클러스터 송신자 채널이 단일 전송 큐 *SYSTEM.CLUSTER.TRANSMIT.QUEUE*에서 메시지를 전송하기 위한 것입니다. 기본 구성은 큐 관리자 속성 (**DefClusterXmitQueueType**)을 변경하여 수정하여 변경할 수 있습니다. 이 속성의 기본값은 *SCTQ*이며, 값을 *CHANNEL*로 변경할 수 있습니다.

**DefClusterXmitQueueType** 속성을 *CHANNEL*로 설정하면 각 클러스터 송신자 채널의 기본값은 특정 클러스터 전송 큐(*SYSTEM.CLUSTER.TRANSMIT.ChannelName*)를 사용합니다.

수동으로 전송 큐 속성 **ClusterChannelName**을 클러스터 송신자 채널로 설정할 수도 있습니다. 클러스터 송신자 채널을 통해 연결된 큐 관리자를 목적지로 하는 메시지는 클러스터 송신자 채널을 식별하는 전송 큐에 저장되고, 메시지는 기본 클러스터 전송 큐에 저장되지 않습니다. **ClusterChannelName** 속성을 비워 둘 경우 채널이 재시작 시 기본 클러스터 전송 큐로 전환됩니다. 기본 큐는 큐 관리자 **DefClusterXmitQueueType** 속성의 값에 따라 *SYSTEM.CLUSTER.TRANSMIT.ChannelName* 또는 *SYSTEM.CLUSTER.TRANSMIT.QUEUE*입니다.

**ClusterChannelName**에 별표("\*")를 지정하면 전송 큐를 클러스터 송신자 채널 세트와 연관시킬 수 있습니다. 별표는 채널 이름 문자열의 시작 부분이나 끝에 지정하거나 채널 이름 문자열 중간의 원하는 위치에 지정할 수 있습니다. **ClusterChannelName**은(는) 길이가 20자로 제한됩니다. MQ\_CHANNEL\_NAME\_LENGTH.

### IBM i IBM i의 ClusterName(48바이트 문자열)

큐가 속한 클러스터의 이름.

표 763. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	

큐가 속한 클러스터의 이름입니다. 큐가 둘 이상의 클러스터에 속하는 경우, *ClusterNameList*은(는) 클러스터를 식별하는 이름 목록 오브젝트의 이름을 지정하고 *ClusterName*은(는) 공백입니다. *ClusterName* 및 *ClusterNameList* 중 하나 이상은 공백이어야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CACLN 선택자를 사용하십시오. 이 속성의 길이는 LNCLUN에서 제공됩니다.

### IBM i IBM i의 ClusterNameList(48바이트 문자열)

큐가 속한 클러스터의 이름을 포함하는 이름 목록 오브젝트의 이름입니다.

표 764. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	

이 큐가 속한 클러스터의 이름을 포함하는 이름 목록 오브젝트의 이름입니다. 큐가 하나의 클러스터에만 속하는 경우, 이름 목록 오브젝트에는 하나의 이름만 있습니다. 또는 *ClusterName*을 사용하여 클러스터의 이름을 지정할 수 있는데, 이 경우 *ClusterNameList*는 공백입니다. *ClusterName* 및 *ClusterNameList* 중 하나 이상은 공백이어야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CACLNL 선택자를 사용하십시오. 이 속성의 길이는 LNCLN에서 제공됩니다.

### IBM i IBM i의 CreationDate(12바이트 문자열)

큐가 작성된 날짜입니다.

표 765. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X				

큐를 작성한 날짜입니다. 날짜의 형식은 YYYY-MM-DD이며, 12바이트 길이로 만들기 위해 뒤에 두 개의 후미 공백으로 채워집니다(예: 1992-09-23--이며, 여기서 --은 두 개의 공백 문자를 나타냄).

- IBM i에서는 큐의 작성 날짜가 큐를 나타내는 기본 운영 체제 엔티티(파일 또는 사용자 공간)의 작성 날짜와 다를 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CACRTD 선택자를 사용하십시오. 이 속성의 길이는 LNCRTD에서 제공됩니다.

**IBM i IBM i의 CreationTime(8바이트 문자열)**

큐가 작성된 시간입니다.

표 766. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X				

큐가 작성된 시간입니다. 시간 형식은 24시간 클럭을 사용하는 HH.MM.SS이며, 시간이 10 미만이면 앞에 0이 있습니다(예: 09.10.20). 시간은 로컬 시간입니다.

- IBM i에서는 큐의 작성 시간이 큐를 나타내는 기본 운영 체제 엔티티(파일 또는 사용자 공간)의 작성 시간과 다를 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CACRTT 선택자를 사용하십시오. 이 속성의 길이는 LNCRTT에서 제공됩니다.

**IBM i IBM i의 CurrentQDepth(10자리의 부호 있는 정수)**

현재 큐 용량입니다.

표 767. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X				

현재 큐에 있는 메시지 수입니다. MQPUT 호출 중과 MQGET 호출의 백아웃 중에 증가합니다. 비열람 MQGET 호출 중과 MQPUT 호출의 백아웃 중에 감소합니다. 이에 따른 영향으로, 작업 단위 내에서 큐에 넣었지만 아직 커미트되지 않은 메시지가 MQGET 호출로 검색되는 대상이 아니지만 수에 포함됩니다. 마찬가지로, MQGET 호출을 사용하는 작업 단위 내에서 검색되지만 아직 커미트되지 않은 메시지는 제외합니다.

또한 만기 시간이 지났지만 아직 제거하지 않은 메시지도 검색 대상이 아니지만 수에 포함됩니다. [1063 페이지의 『IBM i의 MQMD\(메시지 디스크립터\)』](#)에 설명된 MDEXP 필드를 참조하십시오.

메시지의 작업 단위 처리 및 세그먼트화로 인해 CurrentQDepth가 MaxQDepth를 초과할 수 있습니다. 그러나, 이렇게 해도 메시지의 검색 가능성에는 아무런 영향이 없습니다. 정상적인 방법으로 MQGET 호출을 사용하여 큐의 모든 메시지를 검색할 수 있습니다.

이 속성의 값은 큐 관리자가 조작함에 따라 변동됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IACDEP 선택자를 사용하십시오.

**IBM i IBM i의 DefBind(10자리의 부호 있는 정수)**

기본 바인딩입니다.

표 768. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	X

이 속성은 OOBNDQ가 MQOPEN 호출에 지정되고 큐가 클러스터 큐인 경우 사용되는 기본 바인딩입니다. DefBind의 값은 다음 중 하나입니다.

**BNDOPN**

MQOPEN 호출로 고정된 바인딩.

## BNDNOT

바인딩이 고정되지 않습니다.

## BNDGRP

바인딩이 MQOPEN 호출로 고정되지 않지만, 논리 그룹의 모든 메시지에 대해서는 MQPUT에서 고정됩니다. 이 속성의 값을 판별하려면 MQINQ 호출에서 IADBND 선택자를 사용하십시오.

## IBM i IBM i의 DefinitionType(10자리의 부호 있는 정수)

큐 정의 유형입니다.

표 769. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

큐가 어떻게 정의되었는지 표시합니다. 값은 다음 중 하나입니다.

## QDPRE

사전정의된 영구적 큐.

이 큐는 시스템 관리자가 작성한 영구적 큐입니다. 시스템 관리자만 삭제할 수 있습니다.

사전정의된 큐는 DEFINE MQSC 명령을 사용하여 작성하고 DELETE MQSC 명령을 사용하여 삭제할 수 있습니다. 사전정의된 큐는 모델 큐에서 작성할 수 없습니다.

명령은 운영자 또는 명령 입력 큐에 명령 메시지를 송신하는 권한 있는 사용자가 발행할 수 있습니다(1334 페이지의 『IBM i의 큐 관리자에 대한 속성』에 설명된 **CommandInputQName** 속성 참조).

## QDPERM

동적으로 정의된 영구적 큐입니다.

이 큐는 오브젝트 디스크립터 MQOD에 지정된 모델 큐의 이름으로 MQOPEN 호출을 발행하는 애플리케이션이 작성한 영구적 큐입니다. 모델 큐 정의에서 **DefinitionType** 속성 값이 QDPERM입니다.

이 유형의 큐는 MQCLOSE 호출을 사용하여 삭제할 수 있습니다. 자세한 내용은 1211 페이지의 『IBM i의 MQCLOSE(오브젝트 닫기)』의 내용을 참조하십시오.

영구적 동적 큐의 **QSGDisp** 속성 값은 QSGDQM입니다.

## QDTEMP

동적으로 정의된 임시 큐입니다.

이 큐는 오브젝트 디스크립터 MQOD에 지정된 모델 큐의 이름으로 MQOPEN 호출을 발행하는 애플리케이션이 작성한 임시 큐입니다. 모델 큐 정의에서 **DefinitionType** 속성 값이 QDTEMP입니다.

이 유형의 큐는 작성한 애플리케이션에 의해 닫히면 MQCLOSE 호출을 통해 자동으로 삭제됩니다.

임시 동적 큐의 **QSGDisp** 속성 값은 QSGDQM입니다.

## QDSHAR

동적으로 정의된 공유 큐입니다.

이 큐는 오브젝트 디스크립터 MQOD에 지정된 모델 큐의 이름으로 MQOPEN 호출을 발행하는 애플리케이션이 작성한 공유 영구적 큐입니다. 모델 큐 정의에서 **DefinitionType** 속성 값이 QDSHAR입니다.

이 유형의 큐는 MQCLOSE 호출을 사용하여 삭제할 수 있습니다. 자세한 내용은 1211 페이지의 『IBM i의 MQCLOSE(오브젝트 닫기)』의 내용을 참조하십시오.

공유 동적 큐의 **QSGDisp** 속성 값은 QSGDQH입니다.

모델 큐는 항상 사전정의되기 때문에 모델 큐 정의에서 이 속성이 모델 큐가 어떻게 정의되었는지 표시하지 않습니다. 대신, 모델 큐의 이 속성 값은 MQOPEN 호출을 사용하여 모델 큐 정의에서 작성한 각 동적 큐의 **DefinitionType**을 판별하는 데 사용됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IADEFI 선택자를 사용하십시오.

**IBM i IBM i의 DefInputOpenOption(10자리의 부호 있는 정수)**  
기본 입력 열기 옵션입니다.

표 770. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

입력을 위해 큐를 여는 기본 방법입니다. 큐를 열 때 OOINPQ 옵션이 MQOPEN 호출에 지정된 경우 적용됩니다. 값은 다음 중 하나일 수 있습니다.

**OOINPX**

배타적 액세스를 메시지를 가져오기 위해 큐를 엽니다.

해당 큐가 후속 MQSET 호출에 사용하도록 열립니다. 임의 유형의 입력(OOINPS 또는 OOINPX)을 대상으로 현재 이 애플리케이션 또는 다른 애플리케이션에 의해 큐가 열려 있으면 이유 코드 RC2042가 표시되면서 호출이 실패합니다.

**OOINPS**

공유 액세스를 메시지를 가져오기 위해 큐를 엽니다.

해당 큐가 후속 MQSET 호출에 사용하도록 열립니다. 현재 이 애플리케이션 또는 다른 애플리케이션에 의해 OOINPS를 사용하여 큐가 열려 있으면 호출이 성공합니다. 그러나 큐가 현재 OOINPX를 사용하여 열려 있으면 이유 코드 RC2042가 표시되면서 호출이 실패합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IADINP 선택자를 사용하십시오.

**IBM i IBM i의 DefPersistence(10자리의 부호 있는 정수)**  
기본 메시지 지속성.

표 771. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X	X	X	X

이는 큐에 있는 메시지의 기본 지속성입니다. 메시지를 넣을 때 PEQDEF가 메시지 디스크립터에 지정된 경우 적용됩니다.

큐 이름 해석 경로에 정의가 둘 이상인 경우, 기본 지속성은 MQPUT 또는 MQPUT1 호출 시 경로의 첫 번째 정의에 있는 이 속성의 값에서 가져옵니다. 값은 다음이 될 수 있습니다.

- 알리어스 큐
- 로컬 큐
- 리모트 큐의 로컬 정의
- 큐 관리자 알리어스
- 전송 큐(예: DefXmitQName 큐)

값은 다음 중 하나일 수 있습니다.

**PEPER**

메시지가 지속됩니다.

이는 시스템 장애 또는 큐 관리자 재시작 시 메시지가 지속된다는 의미입니다. 다음 큐에는 지속 메시지를 넣을 수 없습니다.

- 임시 동적 큐

- 공유 큐

지속 메시지는 영구적 동적 큐 및 사전정의된 큐에 배치할 수 있습니다.

#### PENPER

메시지가 지속되지 않습니다.

이는 일반적으로 시스템 장애 또는 큐 관리자 재시작 시 메시지가 지속되지 않는다는 의미입니다. 이것은 큐 관리자를 재시작하는 동안 원본 그대로의 메시지 사본이 보조 기억장치에서 발견되는 경우에도 적용됩니다.

공유 큐라는 특별한 사례에서는 큐 공유 그룹에서 큐 관리자를 재시작해도 비지속 메시지가 지속되지만, 공유 큐에 메시지를 저장하는 데 사용하는 커플링 기능에 장애가 발생하면 지속되지 않습니다.

지속 및 비지속 메시지 둘 다 동일한 큐에 존재할 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IADPER 선택자를 사용하십시오.

#### IBM i IBM i의 DefPriority(10자리의 부호 있는 정수)

기본 메시지 우선순위입니다.

표 772. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X	X	X	X

큐에 있는 메시지의 기본 우선순위입니다. 메시지를 큐에 넣을 때 PRQDEF가 메시지 디스크립터에 지정된 경우 적용됩니다.

큐 이름 해석 경로에 정의가 둘 이상인 경우, 메시지의 기본 우선순위는 Put 조작 시 경로의 첫 번째 정의에 있는 이 속성의 값에서 가져옵니다. 값은 다음이 될 수 있습니다.

- 알리어스 큐
- 로컬 큐
- 리모트 큐의 로컬 정의
- 큐 관리자 알리어스
- 전송 큐(예: DefXmitQName 큐)

메시지를 큐에 넣는 방법은 큐의 **MsgDeliverySequence** 속성 값에 따라 다릅니다.

- **MsgDeliverySequence** 속성이 MSPRIO이면 큐에서 메시지를 넣는 논리 위치가 메시지 디스크립터의 **MDPRI** 필드 값에 따라 결정됩니다.
- **MsgDeliverySequence** 속성이 MSFIFO이면 메시지 디스크립터의 **MDPRI** 필드 값에 상관없이 우선순위가 해석된 큐의 **DefPriority**와 같더라도 메시지를 큐에 넣습니다. 그러나, 메시지를 넣는 애플리케이션이 지정한 값은 **MDPRI** 필드에 있습니다. 자세한 정보는 1305 페이지의 『큐의 속성』에 설명된 **MsgDeliverySequence** 속성을 참조하십시오.

우선순위는 0(최저값) - **MaxPriority** (최고값) 범위에 있습니다. 1334 페이지의 『IBM i의 큐 관리자에 대한 속성』에 설명된 **MaxPriority** 속성을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 IADPRI 선택자를 사용하십시오.

#### IBM i IBM i의 DefReadAhead(10자리의 부호 있는 정수)

클라이언트에 전달된 비지속 메시지의 디폴트 미리 읽기 작동을 지정합니다.

표 773. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X	X		



DefReadAhead는 다음 값 중 하나로 설정할 수 있습니다.

**RAHNO**

애플리케이션이 요청하기 전에 비지속 메시지를 클라이언트에 미리 송신하지 않습니다. 클라이언트가 비정상적으로 종료되면 하나의 비지속 메시지의 최대값을 잃을 수 있습니다.

**RAHYES**

애플리케이션이 요청하기 전에 비지속 메시지를 클라이언트에 미리 송신합니다. 클라이언트가 비정상적으로 종료된 경우나 클라이언트가 송신된 메시지를 모두 이용하지 않는 경우 비지속 메시지가 손실될 수 있습니다.

**RAHDIS**

비지속 메시지의 미리 읽기는 이 큐에 사용되지 않습니다. 클라이언트 애플리케이션이 미리 읽기를 요청했는지 여부와 무관하게 메시지가 클라이언트에 미리 송신되지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IADRAH 선택자를 사용하십시오.

**IBM i IBM i의 DefPResp(10자리의 부호 있는 정수)**

기본 Put 응답 유형(DEFPRESP) 속성은 MQPMO의 PutResponseType이 PMRASQ로 설정된 경우 애플리케이션에 사용되는 값을 정의합니다. 이 속성은 모든 큐 유형에 대해 유효합니다.

표 774. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X	X	X	X

값은 다음 중 하나일 수 있습니다.

**SYNC**

Put 조작은 동기식으로 발행되며 응답을 리턴합니다.

**ASYNCR**

Put 조작이 비동기식으로 실행되며 MQMD 필드의 서브세트를 리턴합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IADPRT 선택자를 사용하십시오.

**IBM i IBM i의 DistLists(10자리의 부호 있는 정수)**

분배 목록 지원.

표 775. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

큐에 분배 목록 메시지를 넣을 수 있는지 여부를 표시합니다. 속성은 메시지 채널 에이전트(MCA)가 채널의 다른 끝에 있는 큐 관리자가 분배 목록을 지원하는지 여부를 큐 로컬 관리자에 알리기 위해 설정합니다. 나중에 언급된 이 큐 관리자("파트너 큐 관리자")는 송신 MCA에 의해 로컬 전송 큐에서 제거된 이후 다음 순서로 메시지를 수신하는 큐 관리자입니다.

속성은 송신 MCA가 파트너 큐 관리자의 수신 MCA에 대한 연결을 설정할 때마다 설정됩니다. 이 방식에서 송신 MCA로 인해 로컬 큐 관리자는 파트너 큐 관리자가 올바르게 처리할 수 있는 메시지만 전송 큐에 넘겨 됩니다.

이 속성은 주로 전송 큐에 사용되지만, 큐에 정의된 사용법에 상관없이 설명된 처리가 수행됩니다(Usage 속성 참조).

값은 다음 중 하나일 수 있습니다.

**DLSUPP**

분배 목록이 지원됩니다.

분배 목록 메시지를 큐에 저장할 수 있고 해당 양식으로 파트너 큐 관리자에 전송할 수 있음을 표시합니다. 이렇게 하면 메시지를 여러 목적지로 송신하는 데 필요한 처리량이 줄어듭니다.



## DLNSUP

분배 목록이 지원되지 않습니다.

파트너 큐 관리자가 분배 목록을 지원하지 않기 때문에 분배 목록 메시지를 큐에 저장할 수 없음을 표시합니다. 애플리케이션이 분배 목록 메시지를 넣고 해당 메시지가 이 큐에 놓이는 경우, 큐 관리자는 대신 분배 목록을 분할하여 개별 메시지를 큐에 놓습니다. 이렇게 하면 메시지를 여러 목적지로 송신하는 데 필요한 처리량이 늘어나지만, 파트너 큐 관리자가 메시지를 올바르게 처리하는지 확인할 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IADIST 선택자를 사용하십시오. 이 속성의 값을 변경하려면 MQSET 호출을 사용하십시오.

## IBM i IBM i의 HardenGetBackout(10자리의 부호 있는 정수)

정확한 백아웃 수의 유지보수 여부입니다.

표 776. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

각 메시지마다 작업 단위 내에서 MQGET 호출로 인해 메시지가 수신되고 나중에 해당 작업 단위가 백아웃된 횟수를 나타내는 수가 보관됩니다. 이 수는 MQGET 호출을 완료한 후 메시지 디스크립터의 *MBOC* 필드에서 확인할 수 있습니다.

큐 관리자를 재시작할 때 메시지 백아웃 수가 지속됩니다. 그러나, 수의 정확성을 보장하기 위해 이 큐의 작업 단위 내에서 MQGET 호출에 의해 메시지가 수신될 때마다 정보가 "기록"(디스크 또는 다른 영구적 스토리지 디바이스에 기록됨)되어야 합니다. 이 작업이 수행되지 않고 MQGET 호출의 백아웃과 함께 큐 관리자 장애가 발생하면 수가 증가하지 않을 수 있습니다.

그러나 작업 단위 내에서 각 MQGET 호출 정보를 기록하면 성능이 저하되므로 수가 정확해야 하는 경우에만 **HardenGetBackout** 속성을 QABH로 설정해야 합니다.

- IBM i에서는 이 속성 설정에 상관없이 메시지 백아웃 수가 항상 기록됩니다.

가능한 값은 다음과 같습니다.

### QABH

백아웃 수를 기억합니다.

기록은 이 큐에 있는 메시지의 백아웃 수가 정확한지 확인하는 데 사용됩니다.

### QABNH

백아웃 수를 기억하지 않습니다.

이 큐에 있는 메시지의 백아웃 수가 정확한지 확인하기 위해 기록을 사용하지 않습니다. 따라서 해당 수는 예상보다 적을 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAHGB 선택자를 사용하십시오.

## IBM i IBM i의 InhibitGet(10자리의 부호 있는 정수)

이 큐에 대해 Get 조작이 허용되는지 여부를 제어합니다.

표 777. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X	X		

알리어스 큐인 경우, MQGET 호출이 성공하려면 Get 조작 시에 알리어스 큐 및 기본 큐 둘 다에 대해 Get 조작이 허용되어야 합니다. 값은 다음 중 하나입니다.

## QAGETI

Get 조작이 금지됩니다.

이유 코드 RC2016과 함께 MQGET 호출이 실패합니다. GMBRWF 또는 GMBRWN을 지정하는 MQGET 호출이 여기에 포함됩니다.

**참고:** 작업 단위 내에서 MQGET 호출 조작이 성공적으로 완료되면 이후에 QAGETI에 대한 **InhibitGet** 속성 값을 변경해도 작업 단위 커미트가 금지되지 않습니다.

## QAGETA

Get 조작이 허용됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAIGET 선택자를 사용하십시오. 이 속성의 값을 변경하려면 MQSET 호출을 사용하십시오.

## IBM i IBM i의 InhibitPut(10자리의 부호 있는 정수)

이 큐에 대해 Put 조작이 허용되는지 여부를 제어합니다.

표 778. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X	X	X	X

큐 이름 해석 경로에 정의가 둘 이상인 경우 MQPUT 또는 MQPUT1 호출이 성공하기 위해 Put 조작 시 경로의 모든 정의(큐 관리자 알리어스 정의 포함)에 대해 Put 조작이 허용되어야 합니다. 값은 다음 중 하나일 수 있습니다.

## QAPUTI

Put 조작이 금지됩니다.

이유 코드 RC2051과 함께 MQPUT 및 MQPUT1 호출이 실패합니다.

**참고:** 작업 단위 내에서 MQPUT 호출 조작이 성공적으로 완료되면 이후에 QAPUTI에 대한 **InhibitPut** 속성 값을 변경해도 작업 단위 커미트가 금지되지 않습니다.

## QAPUTA

Put 조작이 허용됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAIPUT 선택자를 사용하십시오. 이 속성의 값을 변경하려면 MQSET 호출을 사용하십시오.

## IBM i IBM i의 InitiationQName(48바이트 문자열)

이니시에이션 큐의 이름입니다.

표 779. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

로컬 큐 관리자에 정의된 큐의 이름입니다. 큐 유형이 QTLOC여야 합니다. 큐 관리자는 이 속성이 속한 큐에 메시지가 도착하여 애플리케이션을 시작해야 할 때 트리거 메시지를 이니시에이션 큐로 송신합니다. 이니시에이션 큐는 트리거 메시지 수신 이후 적절한 애플리케이션을 시작하는 트리거 모니터 애플리케이션에서 모니터링해야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CAINIQ 선택자를 사용하십시오. 이 속성의 길이는 LNQN에서 제공됩니다.

## IBM i IBM i의 MaxMsgLength(10자리의 부호 있는 정수)

최대 메시지 길이(바이트)입니다.

표 780. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

큐에 둘 수 있는 가장 긴 물리적 메시지 길이의 상한입니다. 그러나, **MaxMsgLength** 큐 속성을 **MaxMsgLength** 큐 관리자 속성에 상관없이 설정할 수 있기 때문에 큐에 넣을 수 있는 가장 긴 물리적 메시지 길이의 실제 상한은 이 두 값 중 작은 값입니다.

큐 관리자가 세그먼트화를 지원하는 경우, 애플리케이션이 MQMD에 MFSEGA 플래그를 지정한 때에만 애플리케이션은 두 개의 **MaxMsgLength** 속성 중 작은 값보다 더 긴 논리 메시지를 넣을 수 있습니다. 해당 플래그를 지정하면 논리 메시지 길이의 상한이 999 999 999바이트이지만, 일반적으로 운영 체제 또는 애플리케이션 실행 환경에서 적용되는 자원 제한으로 인해 한계가 낮아집니다.

너무 긴 메시지를 큐에 넣으려고 하면 다음 이유 코드가 표시되면서 실패합니다.

- 메시지가 큐에 비해 너무 긴 경우 RC2030
- 메시지가 큐 관리자에 비해 너무 크지만 큐에 비해 크지 않은 경우 RC2031

**MaxMsgLength** 속성의 하한은 0입니다. 상한은 환경에 따라 판별됩니다.

- IBM i에서 최대 메시지 길이는 100MB(104 857 600바이트)입니다.

자세한 정보는 1272 페이지의 『IBM i의 MQPUT(메시지 넣기)』에 설명된 **BUFLEN** 매개변수를 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAMLEN 선택자를 사용하십시오.

#### IBM i IBM i의 MaxQDepth(10자리의 부호 있는 정수)

최대 큐 용량입니다.

표 781. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

한 번에 큐에 있을 수 있는 물리적 메시지 수에 대해 정의된 상한입니다. **MaxQDepth** 메시지가 이미 있는 큐에 메시지를 넣으려고 하면 이유 코드 RC2053과 함께 실패합니다.

메시지의 작업 단위 처리 및 세그먼트화로 인해 큐에 있는 물리적 메시지의 실제 수가 **MaxQDepth**를 초과할 수 있습니다. 그러나, 이렇게 해도 메시지의 검색 가능성에는 아무런 영향이 없습니다. 정상적인 방법으로 MQGET 호출을 사용하여 큐의 모든 메시지를 검색할 수 있습니다.

이 속성의 값은 0 이상입니다. 상한은 환경에 따라 판별됩니다.

**참고:** 큐에 **MaxQDepth**보다 적은 메시지가 있더라도 큐에 사용 가능한 스토리지 공간이 소진될 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAMDEP 선택자를 사용하십시오.

#### IBM i IBM i의 MediaLog(10자리의 부호 있는 정수)

특정 큐의 매체 복원에 필요한 로그 익스텐트(또는 IBM i의 저널 수신자)의 ID입니다.

표 782. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

순환 로깅이 사용 중인 큐 관리자에서는 값이 널 문자열로 리턴됩니다.

**IBM i IBM i의 MsgDeliverySequence(10자리의 부호 있는 정수)**  
 메시지 전달 순서입니다.

표 783. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

MQGET 호출에 의해 메시지가 애플리케이션으로 리턴되는 순서를 판별합니다.

**MSFIFO**

메시지는 선입선출(FIFO) 순서로 리턴됩니다.

이는 MQGET 호출이 메시지 우선순위에 상관없이 호출에 지정된 선택 기준을 충족하는 첫 번째 메시지를 리턴함을 의미합니다.

**MSPRIO**

메시지가 우선순위 순서로 리턴됩니다.

이는 MQGET 호출이 호출에 지정된 선택 기준을 충족하는 가장 높은 우선순위 메시지를 리턴함을 의미합니다. 각 우선순위 레벨 내에서 메시지는 선입선출(FIFO) 순서로 리턴됩니다.

큐에 메시지가 있는 동안 관련 속성이 변경되면 전달 순서는 다음과 같습니다.

- MQGET 호출에서 메시지를 리턴하는 순서는 메시지가 큐에 도착할 때 큐에 적용되는 **MsgDeliverySequence** 및 **DefPriority** 속성 값에 따라 결정됩니다.
  - 메시지가 도착할 때 *MsgDeliverySequence*가 MSFIFO이면 우선순위가 *DefPriority*이더라도 메시지가 큐에 놓입니다. 이렇게 해도 메시지의 메시지 디스크립터에 있는 *MDPRI* 필드 값에는 영향을 주지 않습니다. 해당 필드는 메시지를 처음 넣을 때의 값이 그대로 유지됩니다.
  - 메시지가 도착할 때 *MsgDeliverySequence*가 MSPRIO이면 메시지 디스크립터의 *MDPRI* 필드에 지정된 우선순위에 적절한 위치의 큐에 메시지가 놓입니다.

큐에 메시지가 있는 동안 **MsgDeliverySequence** 속성 값이 변경되어도 큐에 있는 메시지의 순서는 변경되지 않습니다.

메시지가 큐에 있는 동안 **DefPriority** 속성이 변경되는 경우, **MsgDeliverySequence** 속성이 MSFIFO로 설정되었더라도 메시지가 반드시 FIFO 순서로 전달되지는 않습니다. 높은 우선순위로 큐에 놓인 메시지가 먼저 전달됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAMDS 선택자를 사용하십시오.

**IBM i IBM i의 OpenInputCount(10자리의 부호 있는 정수)**  
 입력하기 위한 열기 수입니다.

표 784. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X				

현재 MQGET 호출을 사용하여 큐에서 메시지를 제거할 수 있는 핸들의 수입니다. 로컬 큐 관리자에 인식되는 해당 핸들의 합계입니다. 큐가 공유 큐이면 로컬 큐 관리자가 속한 큐 공유 그룹의 다른 큐 관리자에서 큐에 수행된 입력을 위한 열기가 해당 수에 포함되지 않습니다.

해당 수에 입력을 위해 이 큐로 해석되는 알리어스 큐를 연 핸들은 포함됩니다. 입력이 없는 조치를 위해 큐를 연 핸들은 해당 수에 포함되지 않습니다(예: 찾아보기용으로만 연 큐).

이 속성의 값은 큐 관리자가 조작함에 따라 변동됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAIOC 선택자를 사용하십시오.

### IBM i IBM i의 OpenOutputCount(10자리의 부호 있는 정수)

출력하기 위한 열기 수입니다.

표 785. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X				

현재 MQPUT 호출을 사용하여 큐에 메시지를 추가할 수 있는 핸들의 수입니다. 로컬 큐 관리자에 인식되는 해당 핸들의 합계입니다. 리모트 큐 관리자에서 이 큐에 수행된 출력을 위한 열기는 포함되지 않습니다. 큐가 공유 큐이면 로컬 큐 관리자가 속한 큐 공유 그룹의 다른 큐 관리자에서 큐에 수행된 출력을 위한 열기가 해당 수에 포함되지 않습니다.

해당 수에 출력을 위해 이 큐로 해석되는 알리어스 큐를 연 핸들은 포함됩니다. 출력이 없는 조치에 대해 큐를 연 핸들은 해당 수에 포함되지 않습니다(예: 조회용으로만 연 큐).

이 속성의 값은 큐 관리자가 조작함에 따라 변동됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAIOC 선택자를 사용하십시오.

### IBM i IBM i의 ProcessName(48바이트 문자열)

프로세스 이름입니다.

표 786. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

로컬 큐 관리자에 정의된 프로세스 오브젝트의 이름입니다. 프로세스 오브젝트는 큐를 제공할 수 있는 프로그램을 식별합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 ICAPRON 선택자를 사용하십시오. 이 속성의 길이는 LNPRON에서 제공됩니다.

### IBM i IBM i의 QDepthHighEvent(10자리의 부호 있는 정수)

큐 용량 상한 이벤트의 생성 여부를 제어합니다.

표 787. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

큐 용량 상한 이벤트는 애플리케이션이 큐에 메시지를 넣었으며, 이로 인해 큐에 있는 메시지 수가 큐 용량 상한 임계값 이상이 되었음을 나타냅니다(QDepthHighLimit 속성 참조).

**참고:** 이 속성의 값은 동적으로 변경할 수 있습니다.

QDepthHighEvent는 다음 두 개 값 중 하나를 가질 수 있습니다.

#### EVDRDIS

이벤트 보고를 사용하지 않습니다.

## EVRENA

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAQDHE 선택자를 사용하십시오.

### IBM i IBM i의 QDepthHighLimit(10자리의 부호 있는 정수)

큐 용량의 상한입니다.

표 788. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

큐 용량 상한(Queue Depth High) 이벤트를 생성하기 위해 큐 용량을 비교하는 임계값입니다. 이 이벤트는 애플리케이션이 큐에 메시지를 넣었음을 나타내며, 이로 인해 큐에 있는 메시지 수가 큐 용량 상한 임계값 이상이 되었습니다. **QDepthHighEvent** 속성을 참조하십시오.

값은 최대 큐 용량(**MaxQDepth** 속성)의 백분율(0 ~ 100)로 표시됩니다. 기본값은 80입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAQDHL 선택자를 사용하십시오.

### IBM i IBM i의 QDepthLowEvent(10자리의 부호 있는 정수)

큐 용량 하한 이벤트의 생성 여부를 제어합니다.

표 789. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

큐 용량 하한 이벤트는 애플리케이션이 큐에서 메시지를 검색했으며, 이로 인해 큐에 있는 메시지 수가 큐 용량 하한 임계값 이하가 되었음을 나타냅니다(**QDepthLowLimit** 속성 참조).

**참고:** 이 속성의 값은 동적으로 변경할 수 있습니다.

QDepthLowEvent는 다음 값 중 하나를 가질 수 있습니다.

## EVRDIS

이벤트 보고를 사용하지 않습니다.

## EVRENA

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAQDLE 선택자를 사용하십시오.

### IBM i IBM i의 QDepthLowLimit(10자리의 부호 있는 정수)

큐 용량의 하한입니다.

표 790. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

큐 용량 하한(Queue Depth Low) 이벤트를 생성하기 위해 큐 용량을 비교하는 임계값입니다. 이 이벤트는 애플리케이션이 큐에서 메시지를 검색했음을 나타내며 이로 인해 큐에 있는 메시지 수가 큐 용량 하한 임계값 이하가 되었습니다. **QDepthLowEvent** 속성을 참조하십시오.

값은 최대 큐 용량(**MaxQDepth** 속성)의 백분율(0 ~ 100)로 표시됩니다. 기본값은 20입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAQDLL 선택자를 사용하십시오.

### IBM i IBM i의 QDepthMaxEvent(10자리의 부호 있는 정수)

큐 가득 참(Queue Full) 이벤트의 생성 여부를 제어합니다.

표 791. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

큐 가득 참 이벤트는 큐가 가득 차서 즉, 큐 용량이 이미 최대값에 도달해서 큐에 대한 입력이 거부되었음을 나타냅니다.

**참고:** 이 속성의 값은 동적으로 변경할 수 있습니다.

값은 다음 중 하나일 수 있습니다.

#### EVRDIS

이벤트 보고를 사용하지 않습니다.

#### EVRENA

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAQDME 선택자를 사용하십시오.

### IBM i IBM i의 QDesc(64바이트 문자열)

큐 설명입니다.

표 792. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X	X	X	X

주석에 사용할 수 있는 필드입니다. 이 필드의 내용은 큐 관리자에 중요하지 않습니다. 그러나 큐 관리자에서 필드에 표시 가능한 문자만 있도록 요구할 수 있습니다. 널(null) 문자는 포함할 수 없지만, 필요한 경우 오른쪽으로 공백을 채워넣을 수 있습니다. DBCS 설치시 필드는 DBCS 문자(최대 필드 길이 64비트에 따라)를 포함할 수 있습니다.

**참고:** 이 필드에 큐 관리자의 문자 세트에 없는 문자가 포함된 경우(**CodedCharSetId** 큐 관리자 속성에 정의된 대로), 이 필드가 다른 큐 관리자로 송신되면 해당 문자가 올바르게 않게 변환될 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CAQD 선택자를 사용하십시오. 이 속성의 길이는 LNQD에서 제공됩니다.

### IBM i IBM i의 QName(48바이트 문자열)

큐 이름입니다.



표 793. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	X

로컬 큐 관리자에 정의된 큐의 이름입니다. 큐 이름에 대한 자세한 정보는 IBM MQ 오브젝트의 이름을 지정하는 규칙을 참조하십시오. 큐 관리자에 정의된 모든 큐가 같은 큐 네임스페이스를 공유합니다. 따라서, QTLOC 큐와 QTALS 큐는 같은 이름을 가질 수 없습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CAQN 선택자를 사용하십시오. 이 속성의 길이는 LNQN에서 제공됩니다.

### IBM i IBM i의 QServiceInterval(10자리의 부호 있는 정수)

큐 서비스 간격의 대상입니다.

표 794. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

서비스 간격 높음 및 서비스 간격 확인 이벤트를 생성하기 위한 비교에 사용되는 서비스 간격입니다.

**QServiceIntervalEvent** 속성을 참조하십시오.

이 값의 단위는 밀리초이고, 범위는 0 - 999 999 999입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAQSI 선택자를 사용하십시오.

### IBM i IBM i의 QServiceIntervalEvent(10자리의 부호 있는 정수)

서비스 간격 높음 또는 서비스 간격 확인 이벤트의 생성 여부를 제어합니다.

표 795. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

- 검사 결과 최소한 **QServiceInterval** 속성에 의해 지정된 시간 동안 큐에서 검색된 메시지가 없는 것으로 표시되면 서비스 간격 높음 이벤트가 생성됩니다.
- 검사 결과 **QServiceInterval** 속성에 의해 지정된 시간 내에 큐에서 검색된 메시지가 있는 것으로 표시되면 서비스 간격 확인 이벤트가 생성됩니다.

**참고:** 이 속성의 값은 동적으로 변경할 수 있습니다.

이 속성의 값은 다음 중 하나입니다.

#### QSIEHI

큐 서비스 간격 높음 이벤트가 사용 가능합니다.

- 큐 서비스 간격 높음 이벤트를 **사용**하고
- 큐 서비스 간격 확인 이벤트를 **사용 안함**으로 설정합니다.

#### QSIEOK

큐 서비스 간격 확인 이벤트가 사용 가능합니다.

- 큐 서비스 간격 높음 이벤트를 **사용**하고
- 큐 서비스 간격 확인 이벤트를 **사용**합니다.



## QSIENO

사용 가능한 큐 서비스 간격 이벤트가 없습니다.

- 큐 서비스 간격 높음 이벤트를 **사용**하고
- 큐 서비스 간격 확인 이벤트를 **사용 안함**으로 설정합니다.

공유 큐인 경우, 이 속성 값이 무시됩니다. QSIENO 값으로 가정합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAQSIE 선택자를 사용하십시오.

## IBM i IBM i의 QSGDisp(10자리의 부호 있는 정수)

큐 공유 그룹 속성 지정.

표 796. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	

큐의 처리를 지정합니다. 값은 다음 중 하나입니다.

### QSGDQM

큐 관리자 처리.

오브젝트에 큐 관리자 속성 지정 값이 있습니다. 즉, 오브젝트 정의가 로컬 큐 관리자에만 인식되고 큐 공유 그룹의 다른 큐 관리자에는 인식되지 않습니다.

큐 공유 그룹의 각 큐 관리자는 현재 오브젝트와 이름 및 유형이 동일한 오브젝트를 가질 수 있으나 이는 별도의 오브젝트이며 이러한 오브젝트 사이에 아무런 상관 관계가 없습니다. 속성이 서로 동일해야 한다는 제한 사항은 없습니다.

### QSGDCP

복사된 오브젝트 속성 지정 값.

오브젝트는 공유 저장소에 존재하는 마스터 오브젝트 정의의 로컬 사본입니다. 큐 공유 그룹의 각 큐 관리자는 오브젝트의 자체 사본을 가질 수 있습니다. 처음에는 모든 사본이 동일한 속성을 가지지만, MQSC 명령을 사용하면 사본의 속성을 다른 사본의 속성과 다르게 대체할 수 있습니다. 사본의 속성은 공유 저장소 내의 마스터 정의가 대체될 때 다시 동기화됩니다.

### QSGDSH

공유 속성 지정 값.

오브젝트에 공유 속성 지정 값이 있습니다. 즉, 큐 공유 그룹의 모든 큐 관리자에 인식되는 오브젝트의 단일 인스턴스가 공유 저장소에 있습니다. 그룹의 큐 관리자가 이 오브젝트에 액세스할 때에는 오브젝트의 단일 공유 인스턴스에 액세스하게 됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAQSGD 선택자를 사용하십시오.

**z/OS** 이 속성은 z/OS에서만 지원됩니다.

## IBM i IBM i의 QType(10자리의 부호 있는 정수)

큐 유형.

표 797. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	X

이 속성의 값은 다음 중 하나입니다.

**QTALS**

알리어스 큐 정의입니다.

**QTCLUS**

클러스터 큐.

**QTLOC**

로컬 큐.

**QTREM**

리모트 큐의 로컬 정의입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAQTYP 선택자를 사용하십시오.

**IBM i IBM i의 RemoteQMGrName(48바이트 문자열)**

리모트 큐 관리자의 이름입니다.

표 798. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
			X	

큐 *RemoteQName*이 정의된 리모트 큐 관리자의 이름입니다. *RemoteQName* 큐에서 *QSGDisp* 값이 *QSGDCP* 또는 *QSGDSH*이면 *RemoteQMGrName*은 *RemoteQName*을 소유한 큐 공유 그룹의 이름일 수 있습니다.

애플리케이션이 리모트 큐의 로컬 정의를 열면 *RemoteQMGrName*이 공백이 아니거나 로컬 큐 관리자의 이름이 아니어야 합니다. *XmitQName*이 공백이면 *RemoteQMGrName*과 이름이 같은 로컬 큐가 전송 큐로 사용됩니다. *RemoteQMGrName* 이름의 큐가 없으면 **DefXmitQName** 큐 관리자 속성으로 식별된 큐가 사용됩니다.

이 정의가 큐 관리자 알리어스에 사용되는 경우, *RemoteQMGrName*은(는) 별명이 지정된 큐 관리자의 이름입니다. 로컬 큐 관리자의 이름일 수 있습니다. 그렇지 않으면, 열기를 수행할 때 *XmitQName*이 공백인 경우 *RemoteQMGrName*과 이름이 같은 로컬 큐가 있어야 합니다. 이 큐는 전송 큐로 사용됩니다.

이 정의가 응답 대상 알리어스에 사용되는 경우, 이 이름이 *MDRM*이 될 큐 관리자의 이름입니다.

**참고:** 큐 정의가 작성되거나 수정된 경우, 이 속성에 대해 지정된 값에 대해 유효성 검증이 수행되지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 *CARQMN* 선택자를 사용하십시오. 이 속성의 길이는 *LNQMN*에서 제공됩니다.

**IBM i IBM i의 RemoteQName(48바이트 문자열)**

리모트 큐의 이름.

표 799. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
			X	

이 이름은 리모트 큐 관리자 *RemoteQMGrName*에 알려진 큐의 이름입니다.

애플리케이션이 리모트 큐의 로컬 정의를 열면 열기가 발생할 때 *RemoteQName*이(가) 공백이 아니어야 합니다.

큐 관리자 알리어스 정의에 이 정의가 사용되는 경우, 열기가 발생할 때 *RemoteQName*이(가) 공백이어야 합니다.

정의가 응답 대상 알리어스에 사용되는 경우, 이 이름은 *MDRQ*이(가) 될 큐의 이름입니다.

**참고:** 큐 정의가 작성되거나 수정된 경우, 이 속성에 대해 지정된 값에 대해 유효성 검증이 수행되지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CARQN 선택자를 사용하십시오. 이 속성의 길이는 LNQN에서 제공됩니다.

### IBM i IBM i의 RetentionInterval(10자리의 부호 있는 정수)

보유 간격.

표 800. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

큐를 보유해야 하는 시간입니다. 이 시간이 경과한 후 큐가 삭제 대상이 됩니다.

시간은 큐를 작성한 날짜 및 시간부터 시간 단위로 측정됩니다. 큐의 작성 날짜는 *CreationDate*에 기록되고 큐의 작성 시간은 **CreationTime** 속성에 기록됩니다.

이 정보는 보조관리 애플리케이션 또는 운영자가 더 이상 필요하지 않은 큐를 식별하고 삭제할 수 있도록 제공됩니다.

**참고:** 큐 관리자는 이 속성을 기반으로 큐를 삭제하거나 보유 간격이 만기되지 않은 큐의 삭제를 방지하려고 하지 않습니다. 사용자 본인이 필요한 조치를 수행해야 합니다.

영구적 동적 큐의 누적을 방지하려면 실제 보유 간격을 사용해야 합니다(*DefinitionType* 참조). 그러나, 사전 정의된 큐에도 이 속성을 사용할 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IARINT 선택자를 사용하십시오.

### IBM i IBM i의 Scope(10자리의 부호 있는 정수)

큐에 대한 항목이 셀 디렉토리에 존재하는지 여부를 제어합니다.

표 801. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X		X	X	

셀 디렉토리가 설치 가능한 이름 서비스에 의해 제공됩니다. 값은 다음 중 하나일 수 있습니다.

#### SCOQM

큐 관리자 범위.

큐 정의에 큐 관리자 범위가 있습니다. 이는 큐 정의가 큐를 소유하는 큐 관리자를 넘어 확장되지 않음을 의미합니다. 일부 다른 큐 관리자로부터 출력을 위해 큐를 열려면, 소유하는 큐 관리자의 이름을 지정해야 하고, 다른 큐 관리자에 큐의 로컬 정의가 있어야 합니다.

#### SCOCEL

셀 범위

큐 정의에 셀 범위가 있습니다. 이는 큐 정의가 셀의 모든 큐 관리자에서 사용할 수 있는 셀 디렉토리에 있음을 의미합니다. 큐 이름을 지정하는 것만으로 셀의 큐 관리자에서 출력을 위해 큐를 열 수 있습니다. 큐를 소유한 큐 관리자의 이름은 지정하지 않아도 됩니다. 그러나, 로컬 정의가 우선하므로 해당 이름의 큐 로컬 정의가 있는 셀의 큐 관리자에서는 큐 정의를 사용할 수 없습니다.

셀 디렉토리는 LDAP(Lightweight Directory Access Protocol)와 같은 설치 가능한 이름 서비스를 통해 제공됩니다. IBM MQ에서 큐 정의를 DCE 디렉토리(더 이상 지원되지 않음)에 삽입하기 위해 이전에 사용한 DCE(Distributed Computing Environment) 이름 서버를 더 이상 지원하지 않습니다.

모델과 동적 큐에는 셀 범위를 포함할 수 없습니다.

이 값은 셀 디렉토리를 지원하는 이름 서비스가 구성된 경우에만 유효합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IASCOP 선택자를 사용하십시오.

이 속성의 지원에는 다음 제한이 적용됩니다.

- IBM i에서는 속성이 지원되지만 SCOQM만 유효합니다.

### IBM i IBM i의 Shareability(10자리의 부호 있는 정수)

입력에 대한 큐 공유 여부입니다.

표 802. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

입력을 위해 큐를 여러 번 동시에 열 수 있는지 여부를 표시합니다. 값은 다음 중 하나일 수 있습니다.

#### QASHR

큐를 공유할 수 있습니다.

OOINPS 옵션을 사용하면 여러 번 열 수 있습니다.

#### QANSHR

큐를 공유할 수 없습니다.

OOINPS 옵션을 포함한 MQOPEN 호출은 OOINPX로 처리됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IASHAR 선택자를 사용하십시오.

### IBM i IBM i의 TriggerControl(10자리의 부호 있는 정수)

트리거 제어.

표 803. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

큐 제공을 위해 애플리케이션을 시작하도록 트리거 메시지가 이니시에이션 큐에 기록되는지 여부를 제어합니다. 다음 중 하나입니다.

#### TCOFF

트리거 메시지가 필요하지 않습니다.

이 큐에는 트리거 메시지가 기록되지 않습니다. 이 경우, *TriggerType* 값은 관계가 없습니다.

#### TCON

트리거 메시지가 필요합니다.

적절한 트리거 이벤트가 발생하면 이 큐에 대해 트리거 메시지가 기록됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IATRGC 선택자를 사용하십시오. 이 속성의 값을 변경하려면 MQSET 호출을 사용하십시오.

### IBM i IBM i의 TriggerData(64바이트 문자열)

트리거 데이터.

표 804. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

이 큐에 도착하는 메시지로 인해 트리거 메시지가 이니시에이션 큐에 기록되는 경우 큐 관리자가 트리거 메시지에 삽입하는 자유 형식 데이터입니다.

이 데이터의 콘텐츠는 큐 관리자에 중요하지 않습니다. 이니시에이션 큐를 처리하는 트리거 모니터 애플리케이션 또는 트리거 모니터에 의해 시작된 애플리케이션에 의미가 있습니다.

널을 포함할 수 없는 문자열입니다. 필요한 경우, 오른쪽에 공백을 채워 넣어야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CATRGD 선택자를 사용하십시오. 이 속성의 값을 변경하려면 MQSET 호출을 사용하십시오. 이 속성의 길이는 LNTRGD에서 제공합니다.

**IBM i IBM i의 TriggerDepth(10자리의 부호 있는 정수)**  
트리거 용량.

표 805. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

우선순위가 *TriggerMsgPriority* 이상이고 트리거 메시지가 기록되기 전에 큐에 있어야 하는 메시지의 수입니다. *TriggerType*이 TTDPTH로 설정된 경우 적용됩니다. *TriggerDepth*의 값은 1 이상입니다. 그 밖의 경우에는 이 속성이 사용되지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IATRGD 선택자를 사용하십시오. 이 속성의 값을 변경하려면 MQSET 호출을 사용하십시오.

**IBM i IBM i의 TriggerMsgPriority(10자리의 부호 있는 정수)**  
IBM MQ for IBM i의 트리거에 대한 임계값 메시지 우선순위입니다.

표 806. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

메시지가 트리거 메시지 생성의 원인이 되지 않는 메시지 우선순위입니다(즉, 트리거 메시지의 생성 여부를 판별할 때 큐 관리자가 이 메시지를 무시함). *TriggerMsgPriority*는 0(최저값) - *MaxPriority* (최고값. 1334 페이지의 『IBM i의 큐 관리자에 대한 속성』 참조) 범위에 있습니다. 값이 0이면 모든 메시지가 트리거 메시지 생성의 원인이 됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IATRGP 선택자를 사용하십시오. 이 속성의 값을 변경하려면 MQSET 호출을 사용하십시오.

**IBM i IBM i의 TriggerType(10자리의 부호 있는 정수)**  
트리거 유형.

표 807. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

이 큐에 도착한 메시지의 결과로 트리거 메시지가 기록되는 조건을 제어합니다. 값은 다음 중 하나입니다.

#### TTNONE

트리거 메시지가 없습니다.

이 큐에 있는 메시지의 결과로 트리거 메시지가 기록되지 않습니다. 이 값은 *TriggerControl*을 TCOFF로 설정하는 것과 같은 효과가 있습니다.

#### TTFRST

큐 용량이 0 - 1인 경우의 트리거 메시지입니다.

큐에서 우선순위가 *TriggerMsgPriority* 이상인 메시지의 수가 0에서 1로 변경될 때마다 트리거 메시지가 기록됩니다.

#### TTEVRY

모든 메시지에 대한 트리거 메시지입니다.

우선순위가 *TriggerMsgPriority* 이상인 메시지가 큐에 도착할 때마다 트리거 메시지가 기록됩니다.

#### TTDPTH

용량 임계값을 초과할 때의 트리거 메시지.

큐에서 우선순위가 *TriggerMsgPriority* 이상인 메시지의 수가 *TriggerDepth* 이상일 때마다 트리거 메시지가 기록됩니다. 트리거 메시지가 기록된 후 명시적으로 다시 활성화할 때까지 추가 트리거되지 않도록 *TriggerControl*이 TCOFF로 설정됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IATRGT 선택자를 사용하십시오. 이 속성의 값을 변경하려면 MQSET 호출을 사용하십시오.

### IBM i IBM i의 Usage(10자리의 부호 있는 정수)

큐 사용법.

표 808. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
X	X			

큐의 용도를 표시합니다. 값은 다음 중 하나입니다.

#### USNORM

정상적인 사용법입니다.

메시지를 넣고 가져올 때 일반 애플리케이션이 사용하는 큐입니다. 해당 큐는 전송 큐가 아닙니다.

#### USTRAN

전송 큐.

리모트 큐 관리자로 이동하는 메시지를 보관하는 데 사용되는 큐입니다. 일반 애플리케이션이 리모트 큐로 메시지를 송신하면 로컬 큐 관리자가 메시지를 특수 형식으로 적절한 전송 큐에 일시적으로 저장합니다. 그러면 메시지 채널 에이전트가 전송 큐에서 메시지를 읽어오고, 이 메시지를 리모트 큐 관리자로 전송합니다. 전송 큐에 대한 자세한 정보는 [전송 큐](#)의 내용을 참조하십시오.

권한이 있는 애플리케이션만 OOOUT가 메시지를 직접적으로 넣도록 전송 큐를 열 수 있습니다. 유틸리티 애플리케이션만 이 작업을 정상적으로 수행할 것입니다. 메시지 데이터 형식이 올바른지 주의를 기울여야 합니다(1189 페이지의 『IBM i의 MQXQH(전송 큐 헤더)』 참조). 그렇지 않으면, 전송 프로세스 중에 오류가 발생할 수 있습니다. PM\* 컨텍스트 옵션 중 하나가 지정되지 않은 경우, 컨텍스트가 전달되거나 설정되지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAUSAG 선택자를 사용하십시오.

### IBM i IBM i의 XmitQName(48바이트 문자열)

전송 큐 이름.

표 809. 이 속성이 적용되는 큐 유형				
로컬	모델	알리어스	원격	클러스터
			X	

리모트 큐 또는 큐 관리자 알리어스 정의에 대한 열기가 발생할 때 이 속성이 공백이 아닌 경우, 메시지 전달에 사용할 로컬 전송 큐의 이름을 지정합니다.

*XmitQName*이 공백이면 *RemoteQMgrName*과 이름이 같은 로컬 큐가 전송 큐로 사용됩니다.

*RemoteQMgrName* 이름의 큐가 없으면 **DefXmitQName** 큐 관리자 속성으로 식별된 큐가 사용됩니다.

정의를 큐 관리자 알리어스로 사용 중이고 *RemoteQMgrName*이 로컬 큐 관리자의 이름인 경우 이 속성이 무시됩니다. 또한 정의가 응답 대상 큐 알리어스 정의로 사용 되는 경우에도 무시됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CAXQN 선택자를 사용하십시오. 이 속성의 길이는 LNQN에서 제공됩니다.

## 이름 목록에 대한 속성

이 주제는 이름 목록과 관련된 속성을 요약합니다. 속성은 알파벳 순서로 설명합니다.

**참고:** 표시되는 속성의 이름은 MQINQ 및 MQSET 호출과 함께 사용되는 이름입니다.

## 속성 설명

이름 목록 오브젝트는 다음 속성을 가집니다.

### AlterationDate(12바이트 문자열)

정의가 마지막으로 변경된 날짜.

정의가 마지막으로 변경된 날짜입니다. 날짜의 형식은 YYYY-MM-DD이며 길이를 12바이트로 만들기 위해 두 개의 후미 공백으로 채워집니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CAALTD 선택자를 사용하십시오. 이 속성의 길이는 LNDATE에서 제공됩니다.

### AlterationTime(8바이트 문자열)

정의가 마지막으로 변경된 시간.

정의가 마지막으로 변경된 시간입니다. 시간의 형식은 HH.MM.SS입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CAALTT 선택자를 사용하십시오. 이 속성의 길이는 LNTIME에서 제공됩니다.

### NameCount(10자리 부호 있는 정수)

이름 목록에 있는 이름 수.

값이 0보다 크거나 같습니다. 다음 값이 정의됩니다.

#### NCMXNL

이름 목록에 있는 최대 이름 수.

이 속성의 값을 판별하려면 MQINQ 호출에서 IANAMC 선택자를 사용하십시오.

### NamelistDesc(64바이트 문자열)

이름 목록 설명

주석에 사용될 수 있는 필드입니다. 그 값은 정의 프로세스에 의해 설정됩니다. 이 필드의 내용은 큐 관리자에 중요하지 않습니다. 그러나 큐 관리자에서 필드에 표시 가능한 문자만 있도록 요구할 수 있습니다. 널(null) 문자는 포함할 수 없지만, 필요한 경우 오른쪽으로 공백을 채워넣을 수 있습니다. DBCS 설치 시 이 필드는 DBCS 문자(최대 필드 길이 64비트에 따라)를 포함할 수 있습니다.



**참고:** 이 필드에 큐 관리자의 문자 세트에 없는 문자가 포함된 경우(**CodedCharSetId** 큐 관리자 속성에 정의된 대로), 이 필드가 다른 큐 관리자로 송신되면 해당 문자가 올바르게 변환될 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CALSTD 선택자를 사용하십시오.

이 속성의 길이는 LNNLD에서 제공합니다.

### **NamelistName(48바이트 문자열)**

이름 목록 이름.

이는 로컬 큐 관리자에서 정의된 이름 목록의 이름입니다.

각 이름 목록에 큐 관리자에 속하는 기타 이름 목록의 이름과 다른 이름이 있지만, 여기에서 큐와 같이 다른 유형의 기타 큐 관리자 오브젝트의 이름을 복제할 수도 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CALSTN 선택자를 사용하십시오.

이 속성의 길이는 LNNLN에서 제공합니다.

### **Names(48바이트 문자열 x NameCount)**

*NameCount* 이름의 목록.

각 이름은 로컬 큐 관리자에 정의된 오브젝트의 이름입니다. 오브젝트 이름에 대한 자세한 정보는 [Naming IBM MQ 오브젝트](#)를 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 CANAMS 선택자를 사용하십시오.

목록에 있는 각 이름의 길이는 LNOBJN에서 제공합니다.

## **IBM i IBM i의 프로세스 정의에 대한 속성**

이 주제는 프로세스 정의와 관련된 속성을 요약합니다. 속성은 알파벳 순서로 설명합니다.

**참고:** 표시되는 속성의 이름은 MQINQ 및 MQSET 호출과 함께 사용되는 이름입니다. 속성을 정의, 대체 또는 표시하기 위해 MQSC 명령이 사용되는 경우, 대체 짧은 이름이 사용됩니다. 자세한 정보는 [MQSC 명령의 내용](#)을 참조하십시오.

### **속성 설명**

프로세스 정의 오브젝트는 다음 속성을 가집니다.

#### **AlterationDate(12바이트 문자열)**

정의를 마지막으로 변경된 날짜.

정의를 마지막으로 변경된 날짜입니다. 날짜의 형식은 YYYY-MM-DD이며 길이를 12바이트로 만들기 위해 두 개의 후미 공백으로 채워집니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CAALTD 선택자를 사용하십시오. 이 속성의 길이는 LNDATE에서 제공합니다.

#### **AlterationTime(8바이트 문자열)**

정의를 마지막으로 변경된 시간.

정의를 마지막으로 변경된 시간입니다. 시간의 형식은 HH.MM.SS입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CAALTT 선택자를 사용하십시오. 이 속성의 길이는 LNTIME에서 제공합니다.

#### **ApplId(256바이트 문자열)**

애플리케이션 ID입니다.

시작할 애플리케이션을 식별하는 문자열입니다. 이 정보는 이니시에이션 큐에서 메시지를 처리하는 트리거 모니터 애플리케이션에서 사용됩니다. 정보가 트리거 메시지의 일부로서 이니시에이션 큐에 송신됩니다.



*ApplId*의 의미는 트리거 모니터 애플리케이션에 의해 판별됩니다. IBM MQ에서 제공되는 트리거 모니터는 *ApplId*를 실행 가능 프로그램의 이름으로 설정해야 합니다.

널을 포함할 수 없는 문자열입니다. 필요한 경우, 오른쪽에 공백을 채워 넣어야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CAAPPI 선택자를 사용하십시오. 이 속성의 길이는 LNPROA에서 제공합니다.

### ApplType(10자리 부호 있는 정수)

애플리케이션 유형.

트리거 메시지 수신에 대한 응답으로 시작할 프로그램의 네이처를 식별합니다. 이 정보는 이니시에이션 큐에서 메시지를 처리하는 트리거 모니터 애플리케이션에서 사용됩니다. 정보가 트리거 메시지의 일부로서 이니시에이션 큐에 송신됩니다.

*ApplType*은 어떤 값이든 가질 수 있습니다. 표준 유형에는 다음 값을 사용할 수 있고, 사용자 정의 애플리케이션 유형은 ATUFST에서 ATULST 사이의 값으로 제한됩니다.

#### :CICS

CICS 트랜잭션.

#### AT400

IBM i 애플리케이션입니다.

#### ATUFST

사용자 정의된 애플리케이션 유형의 최저값입니다.

#### ATULST

사용자 정의된 애플리케이션 유형의 최고값입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAAPPT 선택자를 사용하십시오.

### EnvData(128바이트 문자열)

환경 데이터.

시작할 애플리케이션에 적용되는 환경 관련 정보를 포함하는 문자열입니다. 이 정보는 이니시에이션 큐에서 메시지를 처리하는 트리거 모니터 애플리케이션에서 사용됩니다. 정보가 트리거 메시지의 일부로서 이니시에이션 큐에 송신됩니다.

*EnvData*의 의미는 트리거 모니터 애플리케이션에 의해 판별됩니다. IBM MQ이(가) 제공하는 트리거 모니터는 시작된 애플리케이션에 전달될 매개변수 목록에 *EnvData*을(를) 추가합니다. 매개변수 목록은 MQTMC2 구조로 구성되며 뒤에 공백이 있고 뒤에 공백이 있는 *EnvData*이(가) 제거됩니다.

널을 포함할 수 없는 문자열입니다. 필요한 경우, 오른쪽에 공백을 채워 넣어야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CAENVD 선택자를 사용하십시오. 이 속성의 길이는 LNPROE에서 제공합니다.

### ProcessDesc(64바이트 문자열)

프로세스 설명.

주석에 사용할 수 있는 필드입니다. 이 필드의 내용은 큐 관리자에 중요하지 않습니다. 그러나 큐 관리자에서 필드에 표시될 수 있는 문자만 포함되어야 할 수도 있습니다. 널(null) 문자는 포함할 수 없지만, 필요한 경우 오른쪽으로 공백을 채워넣을 수 있습니다. DBCS 설치시 필드는 DBCS 문자(최대 필드 길이 64비트에 따라)를 포함할 수 있습니다.

**참고:** 이 필드에 큐 관리자의 문자 세트에 없는 문자가 포함된 경우(**CodedCharSetId** 큐 관리자 속성에 정의된 대로), 이 필드가 다른 큐 관리자로 송신되면 해당 문자가 올바르게 않게 변환될 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CAPROD 선택자를 사용하십시오.

이 속성의 길이는 LNPROD에서 제공합니다.

### ProcessName(48바이트 문자열)

프로세스 이름입니다.

로컬 큐 관리자에 정의된 프로세스 정의의 이름입니다.

각 프로세스 정의에는 큐 관리자에 속하는 기타 프로세스 정의의 이름과 다른 이름이 있습니다. 그러나 프로세스 정의의 이름은 큐와 같이 다른 유형의 기타 큐 관리자 오브젝트의 이름과 동일할 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 ICAPRON 선택자를 사용하십시오.

이 속성의 길이는 LNPRON에서 제공합니다.

### UserData(128바이트 문자열)

사용자 데이터.

시작할 애플리케이션에 적용되는 사용자 정보를 포함하는 문자열입니다. 이 정보는 이니시에이션 큐에서 메시지를 처리하는 트리거 모니터 애플리케이션 또는 트리거 모니터에 의해 시작된 애플리케이션에 사용됩니다. 정보가 트리거 메시지의 일부로 이니시에이션 큐에 송신됩니다.

*UserData*의 의미는 트리거 모니터 애플리케이션에 의해 판별됩니다. IBM MQ에서 제공되는 트리거 모니터는 매개변수 목록의 일부로 *UserData*를 시작된 애플리케이션으로 전달합니다. 매개변수 목록은 MQTMC2 구조(*UserData* 포함)뒤에 하나의 공백이 있고 뒤에 오는 공백이 있는 *EnvData*이(가) 제거됩니다.

널을 포함할 수 없는 문자열입니다. 필요한 경우, 오른쪽에 공백을 채워 넣어야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CAUSRD 선택자를 사용하십시오. 이 속성의 길이는 LNPROU에서 제공합니다.

## IBM i IBM i의 큐 관리자에 대한 속성

큐 관리자 속성의 요약입니다.

일부 큐 관리자 속성은 특정 구현에서 고정되고 나머지는 MQSC 명령 ALTER QMGR을 사용하여 변경할 수 있습니다. DISPLAY QMGR 명령을 사용하는 방법으로도 속성을 표시할 수 있습니다. 대부분의 큐 관리자 속성은 특수 OTQM 오브젝트를 열고 리턴되는 핸들과 함께 MQINQ 호출을 사용하여 조회할 수 있습니다.

다음은 큐 관리자와 관련된 속성을 요약한 표입니다. 속성은 알파벳 순서로 설명합니다.

**참고:** 이 절에 표시된 속성의 이름은 MQINQ 및 MQSET 호출에 사용되는 이름입니다. 속성을 정의, 대체 또는 표시하기 위해 MQSC 명령이 사용되는 경우, 대체 짧은 이름이 사용됩니다. 자세한 정보는 MQSC 명령을 참조하십시오.

속성	설명
<u>AlterationDate</u>	정의를 마지막으로 변경된 날짜
<u>AlterationTime</u>	정의를 마지막으로 변경된 시간
<u>AuthorityEvent</u>	권한 부여(권한 부여되지 않음) 이벤트 생성 여부를 제어합니다.
<u>BridgeEvent</u>	IMS 브릿지 이벤트의 생성 여부 제어
<u>ChannelAutoDef</u>	자동 채널 정의의 허용 여부를 제어합니다.
<u>ChannelAutoDefEvent</u>	채널 자동 정의 이벤트 생성 여부를 제어합니다.
<u>ChannelAutoDefExit</u>	자동 채널 정의에 대한 사용자 엑시트의 이름입니다.
<u>ChannelEvent</u>	채널 이벤트의 생성 여부 제어
<u>ClusterCacheType</u>	클러스터 캐시가 고정된 크기인지 동적으로 변경되는 크기인지 여부 제어
<u>ClusterWorkloadData</u>	클러스터 워크로드 엑시트의 사용자 데이터입니다.
<u>ClusterWorkloadExit</u>	클러스터 워크로드 관리에 대한 사용자 엑시트의 이름입니다.
<u>ClusterWorkloadLength</u>	클러스터 워크로드 엑시트에 전달된 메시지 데이터의 최대 길이입니다.
<u>CodedCharSetId</u>	코드화 문자 세트 ID

표 810. 큐 관리자의 속성 (계속)	
속성	설명
<a href="#">CommandEvent</a>	명령 이벤트 메시지의 큐잉 여부 제어
<a href="#">CommandInputQName</a>	명령 입력 큐 이름입니다.
<a href="#">CommandLevel</a>	명령 레벨
<a href="#">ConfigurationEvent</a>	구성 이벤트
<a href="#">DeadLetterQName</a>	데드-레터 큐의 이름입니다.
<a href="#">DefClusterXmitQueueType</a>	기본 클러스터 전송 큐 유형
<a href="#">DefXmitQName</a>	기본 전송 큐 이름입니다.
<a href="#">DistLists</a>	분배 목록 지원
<a href="#">InhibitEvent</a>	금지(가져오기 금지 및 넣기 금지) 이벤트 생성 여부를 제어합니다.
<a href="#">LocalEvent</a>	로컬 오류 이벤트 생성 여부를 제어합니다.
<a href="#">LoggerEvent</a>	복구 로그 이벤트의 생성 여부 제어
<a href="#">MaxHandles</a>	핸들의 최대 수입입니다.
<a href="#">MaxMsgLength</a>	최대 메시지 길이(바이트)
<a href="#">MaxPriority</a>	최대 우선순위입니다.
<a href="#">MaxUncommittedMsgs</a>	작업 단위 내에서 커밋되지 않은 최대 메시지 수입입니다.
<a href="#">PerformanceEvent</a>	성능 관련 이벤트 생성 여부를 제어합니다.
플랫폼	큐 관리자가 실행 중인 플랫폼입니다.
<a href="#">PubSubMode</a>	발행/구독 엔진 및 큐된 발행/구독 인터페이스의 실행 여부
<a href="#">QMgrDesc</a>	큐 관리자 설명입니다.
<a href="#">QMgrIdentifier</a>	큐 관리자의 내부적으로 생성된 고유 ID
<a href="#">QMgrName</a>	큐 관리자 이름
<a href="#">RemoteEvent</a>	리모트 오류 이벤트 생성 여부를 제어합니다.
<a href="#">RepositoryName</a>	이 큐 관리자가 저장소 서비스를 제공하는 클러스터의 이름입니다.
<a href="#">RepositoryNamelist</a>	이 큐 관리자가 저장소 서비스를 제공하는 클러스터의 이름을 포함하는 이름 목록 오브젝트의 이름입니다.
<a href="#">SSLCRLNamelist</a>	인증 정보 오브젝트의 이름을 포함하는 이름 목록 오브젝트의 이름(참고 1 참조)
<a href="#">SSLEvent</a>	TLS 이벤트의 생성 여부 제어
<a href="#">SSLKeyRepository</a>	TLS 키 저장소의 위치(참고 1 참조)
<a href="#">SSLKeyResetCount</a>	암호화 키를 재협상하기 전에 TLS 대화 안에서 송신 및 수신된 암호화되지 않은 바이트 수를 판별합니다.
<a href="#">StartStopEvent</a>	시작 및 중지 이벤트 생성 여부를 제어합니다.
<a href="#">SyncPoint</a>	동기점 가용성입니다.
<a href="#">TraceRouteRecording</a>	메시지에 대한 라우트 추적 정보의 기록 제어
<a href="#">TreeLifeTime</a>	관리되지 않는 토픽의 지속 시간(초)

표 810. 큐 관리자의 속성 (계속)	
속성	설명
TriggerInterval	트리거 메시지 간격입니다.
<b>참고:</b> 1. 이 속성은 MQINQ 호출을 사용하여 조회할 수 없으며 이 절에서 설명하지 않습니다. 이 속성에 대한 자세한 정보는 큐 관리자 변경의 내용을 참조하십시오.	

### IBM i IBM i의 AlterationDate(12바이트 문자열)

정의가 마지막으로 변경된 날짜.

정의가 마지막으로 변경된 날짜입니다. 날짜의 형식은 YYYY-MM-DD이며 길이를 12바이트로 만들기 위해 두 개의 후미 공백으로 채워집니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CAALTD 선택자를 사용하십시오. 이 속성의 길이는 LNDATE에서 제공됩니다.

### IBM i IBM i의 AlterationTime(8바이트 문자열)

정의가 마지막으로 변경된 시간.

정의가 마지막으로 변경된 시간입니다. 시간의 형식은 HH.MM.SS입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CAALTT 선택자를 사용하십시오. 이 속성의 길이는 LNTIME에서 제공됩니다.

### IBM i IBM i의 AuthorityEvent(10자리의 부호 있는 정수)

(권한이 아니라) 권한 부여 이벤트의 생성 여부를 제어합니다.

AuthorityEvent 속성은 다음 값 중 하나로 설정해야 합니다.

#### EVRDIS

이벤트 보고를 사용하지 않습니다.

#### EVRENA

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAAUTE 선택자를 사용하십시오.

### IBM i IBM i의 BridgeEvent(문자열)

이 속성은 IMS 브릿지 이벤트 메시지를 SYSTEM.ADMIN.CHANNEL.EVENT 큐에 넣는지 여부를 판별합니다. z/OS에서만 지원됩니다.

### IBM i IBM i의 ChannelAutoDef(10자리의 부호 있는 정수)

자동 채널 정의의 허용 여부를 제어합니다.

이 속성은 CTCRVR 및 CTSVCN 유형 채널의 자동 정의를 제어합니다. CTCLSD 채널의 자동 정의가 항상 사용되는 점에 유의하십시오. 값은 다음 중 하나일 수 있습니다.

#### CHADDI

채널 자동 정의 사용 안함.

#### CHADEN

채널 자동 정의 사용.

이 속성의 값을 판별하려면 MQINQ 호출에서 IACAD 선택자를 사용하십시오.

### IBM i IBM i의 ChannelAutoDefEvent(10자리의 부호 있는 정수)

채널 자동 정의 이벤트의 생성 여부를 제어합니다.

이 속성은 CTCVR, CTSVCN 및 CTCLSD 유형의 채널에 적용됩니다. 값은 다음 중 하나일 수 있습니다.

#### **EVREDIS**

이벤트 보고를 사용하지 않습니다.

#### **EVRENA**

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [모니터링 및 성능의 내용](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 IACADE 선택자를 사용하십시오.

#### **IBM i IBM i의 ChannelAutoDefExit(20바이트 문자열)**

자동 채널 정의에 대한 사용자 엑시트의 이름입니다.

이 이름이 공백이 아니고 *ChannelAutoDef*의 값이 CHADEN이면 큐 관리자가 채널 정의를 작성하려고 할 때마다 엑시트가 호출됩니다. 이 속성은 CTCVR, CTSVCN 및 CTCLSD 유형의 채널에 적용됩니다. 그러면 엑시트는 다음 중 하나를 수행할 수 있습니다.

- 변경 없이 채널 정의 작성을 진행할 수 있습니다.
- 작성된 채널 정의의 속성을 수정합니다.
- 채널 작성을 완전히 차단합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 ICACADX 선택자를 사용하십시오. 이 속성의 길이는 LNEXTN에서 제공합니다.

#### **IBM i IBM i의 ChannelEvent(문자열)**

채널 이벤트 메시지의 생성 여부를 지정합니다.

이 속성에 따라 채널 이벤트 메시지를 SYSTEM.ADMIN.CHANNEL.EVENT 큐에 넣는지 여부, 그리고 넣을 경우 어떤 유형의 메시지를 넣는지(예: '시작된 채널', '중지된 채널', '활성화되지 않은 채널') 결정됩니다. 이 속성을 구현하기 전에 채널 이벤트 메시지가 큐에 보관되는 것을 방지하는 유일한 방법은 대상 큐를 삭제하는 것입니다.

이 속성은 또한 IMS 브릿지 이벤트만 수집할 수 있도록 합니다(채널 이벤트를 끌 수 있으므로 동일한 큐에서 가져오고 넣지 않음). 채널 이벤트를 수집하지 않고도 수집 가능한 TLS 이벤트에도 동일하게 적용됩니다.

이 속성은 중요한 이벤트만 수집할 수 있도록 합니다(예: 정상적으로 시작 및 중지할 때가 아니라 채널에 오류가 발생할 때).

ChannelEvent 속성의 값은 다음 중 하나일 수 있습니다.

- EVREXP(RC2279, RC2283, RC2284, RC2295, RC2296 채널 이벤트만 생성됨).
- EVRENA(모든 채널 이벤트가 생성됨, 즉 EVREXP에 의해 생성된 이벤트 외에 RC2282 및 RC2283 이벤트도 생성됨).
- EVREDIS(채널 이벤트가 생성되지 않음, 큐 관리자 초기 기본값임).

이 속성의 값을 판별하려면 MQINQ 호출에서 IACHNE 선택자를 사용하십시오.

#### **IBM i IBM i의 ClusterCacheType(32바이트 문자열)**

클러스터 캐시가 고정된 크기인지 동적으로 변경되는 크기인지 여부를 제어합니다.

호출 시 클러스터 워크로드 엑시트에 전달되는 사용자 정의 32바이트 문자열입니다. 엑시트에 전달할 데이터가 없으면 문자열이 공백입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CACLWD 선택자를 사용하십시오.

#### **IBM i IBM i의 ClusterWorkloadData(32바이트 문자열)**

클러스터 워크로드 엑시트에 대한 사용자 데이터입니다.

호출 시 클러스터 워크로드 엑시트에 전달되는 사용자 정의 32바이트 문자열입니다. 엑시트에 전달할 데이터가 없으면 문자열이 공백입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CACLWD 선택자를 사용하십시오.

## IBM i IBM i의 ClusterWorkloadExit(20바이트 문자열)

클러스터 워크로드 관리에 필요한 사용자 엑시트의 이름입니다.

이 이름이 공백이 아니면 메시지를 클러스터 큐에 넣거나 클러스터 송신자 큐 간에 이동할 때마다 엑시트가 호출됩니다. 그러면 엑시트는 큐 관리자에서 선택한 큐 인스턴스를 메시지의 목적지로 허용하거나 다른 큐 인스턴스를 선택할 수 있습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CACLWX 선택자를 사용하십시오. 이 속성의 길이는 LNEXTN에서 제공됩니다.

## IBM i IBM i의 ClusterWorkloadLength(10자리의 부호 있는 정수)

클러스터 워크로드 엑시트에 전달되는 메시지 데이터의 최대 길이입니다.

클러스터 워크로드 엑시트에 전달되는 메시지 데이터의 최대 길이입니다. 엑시트에 전달된 데이터의 실제 길이는 다음 중 가장 작은 값입니다.

- 메시지의 길이.
- 큐 관리자의 **MaxMsgLength** 속성.
- **ClusterWorkloadLength** 속성.

이 속성의 값을 판별하려면 MQINQ 호출에서 IACLWL 선택자를 사용하십시오.

## IBM i IBM i의 CodedCharSetId(10자리의 부호 있는 정수)

코드화 문자 세트 ID.

오브젝트 이름, 큐 작성 날짜 및 시간 등 MQI에 정의된 모든 문자열 필드에 대해 큐 관리자가 사용하는 문자 세트를 정의합니다. 이 문자 세트는 오브젝트 이름에 유효한 문자의 1바이트 문자를 포함한 세트여야 합니다. 메시지로 전달되는 애플리케이션 데이터에는 적용되지 않습니다. 값은 환경에 따라 달라집니다.

- IBM i에서 값은 큐 관리자를 처음 작성할 때 환경에 설정된 항목입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IACCSI 선택자를 사용하십시오.

## IBM i IBM i의 CommandEvent(정수)

명령이 발행될 때 메시지를 로컬 큐에 넣는지 여부를 제어합니다.

명령이 발행될 때마다 메시지를 새 이벤트 큐, SYSTEM.ADMIN.COMMAND.EVENT에 기록하는지 여부를 제어합니다. 이 기능은 명령 추적 알림과 문제점 진단에 유용합니다. CommandEvent 큐 관리자 속성을 조회하려면 새 속성 선택자 iacev를 다음 값 중 하나와 함께 사용하십시오.

- EVRENA - 모든 성공적인 명령에 대해 명령 이벤트 메시지가 생성되고 큐에 메시지를 넣습니다.
- EVND - DISPLAY (MQSC) 명령과 Inquire (PCF) 명령을 제외한 모든 성공적인 명령에 대해 명령 이벤트 메시지가 생성되고 큐에 메시지를 넣습니다.
- EVRDIS - 명령 이벤트 메시지가 생성되지 않거나 큐에 넣지 않습니다(큐 관리자의 초기 기본값임).

이 속성의 값을 판별하려면 MQINQ 호출에서 CMDEV 선택자를 사용하십시오.

## IBM i IBM i의 CommandInputQName(48바이트 문자열)

명령 입력 큐 이름입니다.

CommandInputQName은 로컬 큐 관리자에 정의된 명령 입력 큐의 이름입니다. 권한이 부여된 사용자가 명령을 송신할 수 있는 큐입니다. 큐의 이름은 환경에 따라 다릅니다.

- IBM i에서 큐의 이름은 SYSTEM.ADMIN.COMMAND.QUEUE 및 PCF 명령만 전송할 수 있습니다. 그러나, CMESC 유형의 PCF 명령 안에 MQSC 명령이 있는 경우 MQSC 명령을 이 큐로 송신할 수 있습니다. 이스케이프 명령에 대한 자세한 정보는 [이스케이프](#)의 내용을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 CACMDQ 선택자를 사용하십시오. 이 속성의 길이는 LNQN에서 제공됩니다.

## IBM i IBM i의 CommandLevel(10자리의 부호 있는 정수)

명령 레벨. 이 매개변수는 큐 관리자가 지원하는 시스템 제어 명령의 레벨을 표시합니다.

레벨은 다음 값 중 하나입니다.

### CML800

시스템 제어 명령의 레벨 800입니다.

이 값은 다음 애플리케이션에서 리턴합니다.

- IBM MQ for IBM i
  - 버전 8.0

### CML900

시스템 제어 명령의 레벨 900입니다.

이 값은 다음 애플리케이션에서 리턴합니다.

- IBM MQ for IBM i
  - 버전 9.0

### CML910

시스템 제어 명령의 레벨 910입니다.

이 값은 다음 애플리케이션에서 리턴합니다.

- IBM MQ for IBM i
  - 버전 9.1

### CML920

시스템 제어 명령의 레벨 920.

이 값은 다음 애플리케이션에서 리턴합니다.

- IBM MQ for IBM i
  - 버전 9.2

### CML930

시스템 제어 명령의 레벨 930.

이 값은 다음 애플리케이션에서 리턴합니다.

- IBM MQ for IBM i
  - 버전 9.3

**CommandLevel** 속성의 특정 값에 해당하는 시스템 제어 명령 세트가 **Platform** 속성 값에 따라 다릅니다. 둘 다 지원되는 시스템 제어 명령을 결정하는 데 사용해야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IACMDL 선택자를 사용하십시오.

## IBM i IBM i의 ConfigurationEvent

구성 이벤트가 생성되고 SYSTEM.ADMIN.CONFIG.EVENT 큐 기본 오브젝트로 송신되는지 여부를 제어합니다.

ConfigurationEvent 속성의 값은 다음 중 하나입니다.

- EVRENA
- EVRDIS

ConfigurationEvent 속성을 EVRENA로 설정하고 특정 명령이 runmqsc 또는 PCF에 의해 성공적으로 발행되면 구성 이벤트가 생성되어 SYSTEM.ADMIN.CONFIG.EVENT 큐로 송신됩니다. 다음 명령에 대한 이벤트는 대체 명령이 관련 오브젝트를 변경하지 않더라도 발행됩니다. 구성 이벤트가 생성 및 송신되는 명령은 다음과 같습니다.

- DEFINE/ALTER AUTHINFO
- DEFINE/ALTER CHANNEL



- DEFINE/ALTER NAMELIST
- DEFINE/ALTER PROCESS
- DEFINE/ALTER QLOCAL(임시 동적 큐가 아닌 경우)
- DEFINE/ALTER QMODEL/QALIAS/QREMOTE
- DELETE AUTHINFO
- DELETE CHANNEL
- DELETE NAMELIST
- DELETE PROCESS
- DELETE QLOCAL(임시 동적 큐가 아닌 경우)
- DELETE QMODEL/QALIAS/QREMOTE
- ALTER QMGR(CONFIGEV 속성이 사용 불가능하고 사용하도록 변경하지 않은 경우 외)
- REFRESH QMGR
- 임시 동적 큐에 대한 것을 제외한 MQSET 호출

다음 환경에서는 이벤트가 생성되지 않습니다(사용되는 경우).

- 명령 또는 MQSET 호출이 실패합니다.
- 큐 관리자가 이벤트 메시지를 이벤트 큐에 넣을 수 없습니다. 명령이 여전히 성공적으로 완료되어야 합니다.
- 임시 동적 큐.
- 내부 속성 변경이 직접적으로 또는 암시적으로 완료되었습니다(MQSET 또는 명령 사용 안함). 이 동작은 TRIGGER, CURDEPTH, IPPROCS, OPPROCS, QDPHIEV, QDPLOEV, QDPMAXEV, QSVCI EV에 영향을 미칩니다.
- 구성 이벤트 큐가 변경되면 새로 고침 요청 시 해당 변경사항에 대한 이벤트 메시지가 생성됩니다.
- REFRESH/RESET CLUSTER과 RESUME/SUSPEND QMGR 명령을 통해 클러스터가 변경됩니다.
- 큐 관리자를 작성 또는 삭제합니다.

### IBM i IBM i의 DeadLetterQName(48바이트 문자열)

데드 레터(미배달 메시지) 큐의 이름입니다.

로컬 큐 관리자에 정의된 큐의 이름입니다. 메시지를 올바른 목적지로 라우트할 수 없는 경우 메시지를 이 큐로 송신합니다.

예를 들어, 다음의 경우에 메시지가 이 큐에 넣어집니다.

- 큐 관리자에 아직 정의되지 않은 큐로 예정된 메시지가 이 큐 관리자에 도달한 경우
- 메시지가 큐 관리자에 도달했으나, 그 메시지를 수신하기로 예정된 큐가 다음과 같은 이유로 수신할 수 없는 경우
  - 큐가 가득 참
  - 넣기 요청이 금지됨
  - 송신 노드에는 메시지를 큐에 넣을 수 있는 권한이 없습니다.

애플리케이션도 메시지를 데드-레터 큐에 넣을 수 있습니다.

보고 메시지는 일반 메시지와 동일한 방식으로 처리됩니다. 보고 메시지를 목적지 큐로 전달할 수 없으면(일반적으로 원래 메시지의 메시지 디스크립터의 MDRQ 필드에 지정된 큐) 보고 메시지가 데드 레터(미배달 메시지) 큐에 놓입니다.

**참고:** 만기 시간(1063 페이지의 『IBM i의 MQMD(메시지 디스크립터)』에 설명된 MDEXP 필드 참조)이 지난 메시지를 제거하면 이 큐에 해당 메시지가 이 큐로 전송되지 **않습니다**. 그러나 만기 보고 메시지(ROEXP)는 여전히 송신 애플리케이션의 요청에 따라 생성되고 MDRQ 큐로 송신됩니다.

넣기 요청을 발행한 애플리케이션에 MQPUT 또는 MQPUT1 호출이 리턴한 이유 코드와 함께 문제점 알림이 동기적으로 제공된 경우, 메시지를 데드 레터(미배달 메시지) 큐에 넣지 않습니다(예: 넣기 요청이 금지된 로컬 큐에 메시지 넣음).



때에 따라 데드 레터(미배달 메시지) 큐의 메시지는 애플리케이션 메시지 데이터에 MQDLH 구조가 접두부로 붙습니다. 이 구조에는 메시지를 데드 레터(미배달 메시지) 큐에 넣은 이유를 표시하는 추가 정보가 있습니다. 이 구조의 자세한 정보는 1021 페이지의 『IBM i의 MQDLH (데드-레터 헤더)』의 내용을 참조하십시오.

이 큐는 **Usage** 속성이 USNORM인 로컬 큐여야 합니다.

데드 레터(미배달 메시지) 큐가 큐 관리자에서 지원되지 않거나 정의되지 않은 경우, 이름이 모두 공백입니다. 모든 IBM MQ 큐 관리자는 데드 레터(미배달 메시지) 큐를 지원하지만 기본적으로 정의되지 않습니다.

데드 레터(미배달 메시지) 큐가 정의되지 않았거나 가득 차거나 몇 가지 다른 이유로 사용할 수 없으면 메시지 채널 에이전트를 통해 여기로 전송된 메시지가 대신 전송 큐에 보유됩니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CADLQ 선택자를 사용하십시오. 이 속성의 길이는 LNQN에서 제공됩니다.

### **DefClusterXmitQueueType(10자리 부호 있는 정수)**

DefClusterXmitQueueType 속성은 클러스터-송신자 채널이 기본적으로 메시지를 가져오고 클러스터-수신자 채널로 메시지를 보내기 위해 선택하는 전송 큐를 제어합니다.

DefClusterXmitQueueType의 값은 MQCLXQ\_SCTQ 또는 MQCLXQ\_CHANNEL입니다.

#### **MQCLXQ\_SCTQ**

모든 클러스터 송신자 채널은 SYSTEM.CLUSTER.TRANSMIT.QUEUE에서 메시지를 전송합니다. 전송 큐에 있는 메시지의 correlID가 메시지의 목적지가 될 클러스터 송신자 채널을 식별합니다.

SCTQ는 큐 관리자가 정의될 때 설정됩니다.

#### **MQCLXQ\_CHANNEL**

각 클러스터 송신자 채널이 다른 전송 큐에서 메시지를 보냅니다. 각 전송 큐는 모델 큐 SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE에서 영구 동적 큐로 작성됩니다.

큐 관리자 속성 DefClusterXmitQueueType이 CHANNEL로 설정된 경우, 기본 구성이 개별 클러스터 전송 큐와 연관된 클러스터 송신자 채널로 변경됩니다. 전송 큐는 모델 큐 SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE에서 작성된 영구적 동적 큐입니다. 각 전송 큐는 하나의 클러스터 송신자 채널과 연관됩니다. 하나의 클러스터 송신자 채널이 하나의 클러스터 전송 큐를 제공할 경우 전송 큐에는 한 클러스터의 한 큐 관리자에 대한 메시지만 포함됩니다. 클러스터에 있는 개별 큐 관리자가 하나의 클러스터 큐만 포함하도록 클러스터를 구성할 수 있습니다. 이 경우 큐 관리자에서 각 클러스터 큐로 전달되는 메시지 트래픽은 메시지에서 다른 큐로 개별적으로 송신됩니다.

값을 조회하려면 MQINQ를 호출하거나 MQIA\_DEF\_CLUSTER\_XMIT\_Q\_TYPE 선택자를 설정하여 큐 관리자 조회(MQCMD\_INQUIRE\_Q\_MGR) PCF 명령을 전송하십시오. 값을 변경하려면 MQIA\_DEF\_CLUSTER\_XMIT\_Q\_TYPE 선택자를 설정하여 큐 관리자 변경(MQCMD\_CHANGE\_Q\_MGR) PCF 명령을 전송하십시오.

#### **관련 참조**

큐 관리자 변경

큐 관리자 조회

1246 페이지의 『IBM i의 MQINQ(오브젝트 속성 조회)』

MQINQ 호출은 오브젝트의 속성을 포함한 문자열 세트 및 정수 배열을 리턴합니다.

### **IBM i IBM i의 DefXmitQName(48바이트 문자열)**

기본 전송 큐 이름.

사용하는 전송 큐에 대한 표시가 없는 경우, 리모트 큐 관리자에 메시지를 전송하는 데 사용되는 전송 큐의 이름입니다.

기본 전송 큐 이름이 없으면 이름이 완전히 공백입니다. 이 속성의 초기값은 공백입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CADXQN 선택자를 사용하십시오. 이 속성의 길이는 LNQN에서 제공됩니다.

## ▶ IBM i IBM i의 DistLists(10자리의 부호 있는 정수)

분배 목록 지원.

로컬 큐 관리자가 MQPUT 및 MQPUT1 호출에서 분배 목록을 지원하는지 여부를 표시합니다. 값은 다음 중 하나일 수 있습니다.

### DLSUPP

분배 목록이 지원됩니다.

### DLNSUP

분배 목록이 지원되지 않습니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IADIST 선택자를 사용하십시오.

## ▶ IBM i IBM i의 InhibitEvent(10자리의 부호 있는 정수)

금지(Get 금지 및 Put 금지) 이벤트의 생성 여부를 제어합니다.

값은 다음 중 하나일 수 있습니다.

### EVRDIS

이벤트 보고를 사용하지 않습니다.

### EVRENA

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [모니터링 및 성능의 내용](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAINHE 선택자를 사용하십시오.

## ▶ IBM i IBM i의 LocalEvent(10자리의 부호 있는 정수)

로컬 오류 이벤트의 생성 여부를 제어합니다.

값은 다음 중 하나입니다.

### EVRDIS

이벤트 보고를 사용하지 않습니다.

### EVRENA

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 IALCLE 선택자를 사용하십시오.

## ▶ IBM i IBM i의 LoggerEvent(10자리의 부호 있는 정수)

복구 로거 이벤트의 생성 여부를 제어합니다.

값은 다음 중 하나일 수 있습니다.

### ENABLED

로거 이벤트가 생성됩니다.

### DISABLED

로거 이벤트가 생성되지 않습니다. 큐 관리자의 초기 기본값입니다.

이벤트에 대한 자세한 정보는 [모니터링 및 성능의 내용](#)을 참조하십시오.

## ▶ IBM i IBM i의 MaxHandles(10자리의 부호 있는 정수)

최대 핸들 수.

하나의 태스크가 동시에 사용할 수 있는 열기 핸들의 최대 수입니다. 단일 큐(또는 큐가 아닌 오브젝트)에 대해 성공한 각 MQOPEN 호출에 한 개의 핸들이 사용됩니다. 오브젝트를 닫으면 해당 핸들을 다시 사용할 수 있게 됩니다. 그러나, 분배 목록을 연 경우에는 분배 목록에 있는 각 큐에 별도의 핸들이 할당되므로 MQOPEN 호출이 분배 목록에 있는 큐의 수만큼 많은 핸들을 사용합니다. *MaxHandles*에 적절한 값을 결정할 때 이러한 점을 고려해야 합니다.

MQPUT1 호출이 처리의 일부로 MQOPEN 호출을 수행합니다. 따라서 MQPUT1은 MQOPEN만큼 많은 핸들을 사용하지만, 해당 핸들은 MQPUT1 호출의 지속 기간 동안에만 사용됩니다.

값의 범위는 1 - 999 999 999입니다. IBM i에서 기본값은 256입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAMHND 선택자를 사용하십시오.

### IBM i IBM i의 MaxMsgLength(10자리의 부호 있는 정수)

최대 메시지 길이(바이트)입니다.

큐 관리자 처리할 수 있는 가장 긴 물리적 메시지의 길이입니다. 그러나, **MaxMsgLength** 큐 관리자 속성을 **MaxMsgLength** 큐 속성에 상관없이 설정할 수 있기 때문에 큐에 넣는 가장 긴 물리적 메시지는 이 두 값 중 작은 값입니다.

큐 관리자가 세그먼트화를 지원하는 경우, 애플리케이션이 MQMD에 MFSEGA 플래그를 지정한 때에만 애플리케이션은 두 개의 **MaxMsgLength** 속성 중 작은 값보다 더 긴 논리 메시지를 넣을 수 있습니다. 해당 플래그를 지정하면 논리 메시지 길이의 상한이 999 999 999바이트이지만, 일반적으로 운영 체제 또는 애플리케이션 실행 환경에서 적용되는 자원 제한으로 인해 한계가 낮아집니다.

**MaxMsgLength** 속성에 대한 하한은 32KB(32 768바이트)입니다. IBM i에서 최대 메시지 길이는 100MB(104 857 600바이트)입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAMLEN 선택자를 사용하십시오.

### IBM i IBM i의 MaxPriority(10자리의 부호 있는 정수)

최대 우선순위입니다.

큐 관리자에 지원되는 최대 메시지 우선순위입니다. 우선순위의 범위는 0(최저) - *MaxPriority* (최고)입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAMPRI 선택자를 사용하십시오.

### IBM i IBM i의 MaxUncommittedMsgs(10자리의 부호 있는 정수)

작업 단위 내에서 커밋되지 않은 메시지의 최대 수입니다.

작업 단위 내에 있을 수 있는 커밋되지 않은 메시지의 최대 수입니다. 커밋되지 않은 메시지의 수는 현재 작업 단위가 시작한 이후 다음 항목의 합계입니다.

- PMSYP 옵션으로 애플리케이션이 넣은 메시지
- GMSYP 옵션으로 애플리케이션이 검색한 메시지
- PMSYP 옵션으로 메시지 넣기에 대해 큐 관리자가 생성한 트리거 메시지 및 COA 보고 메시지
- GMSYP 옵션으로 검색한 메시지에 대해 큐 관리자가 생성한 COD 보고 메시지

다음 메시지는 커밋되지 않음으로 계수되지 않습니다.

- 작업 단위 외부에서 애플리케이션이 넣거나 검색한 메시지
- 작업 단위 외부에서 메시지 넣기 또는 검색의 결과로 큐 메시지가 생성한 트리거 메시지 또는 COA/COD 보고 메시지
- 큐 관리자가 생성한 만기 보고 메시지(만기 보고 메시지를 유발하는 호출이 GMSYP를 지정한 경우에도)
- 큐 관리자가 생성한 이벤트 메시지(이벤트 메시지를 유발하는 호출이 PMSYP 또는 GMSYP를 지정한 경우에도)

#### 참고:

1. 예외 보고 메시지는 Message Channel Agent(MCA) 또는 애플리케이션에서 생성되므로, 애플리케이션이 일반 메시지를 넣거나 검색하듯이 동일한 방법으로 처리됩니다.
2. PMSYP 옵션으로 메시지 또는 세그먼트를 넣으면 넣기로 인해 실제 발생하는 물리적 메시지 수에 상관없이 커밋되지 않은 메시지의 수가 하나씩 증가합니다. (큐 관리자가 메시지 또는 세그먼트를 다시 나누어야 하는 경우 둘 이상의 물리적 메시지가 발생할 수 있습니다.)
3. PMSYP 옵션으로 분배 목록을 넣으면 생성되는 각 물리적 메시지에 대해 커밋되지 않은 메시지의 수가 하나씩 증가합니다. 이 수는 분배 목록에 있는 목적지의 수만큼 적거나 많습니다.

이 속성에 대한 하한은 1이고 상한은 999 999 999입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAMUNC 선택자를 사용하십시오.

### **IBM i** **IBM i의 PerformanceEvent(10자리의 부호 있는 정수)**

성능 관련 이벤트의 생성 여부를 제어합니다.

PerformanceEvent는 다음 값 중 하나를 가질 수 있습니다.

#### **EVRDIS**

이벤트 보고를 사용하지 않습니다.

#### **EVRENA**

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 IAPFME 선택자를 사용하십시오.

### **IBM i** **IBM i의 Platform(10자리의 부호 있는 정수)**

큐 관리자가 실행되고 있는 플랫폼입니다.

이 속성은 큐 관리자가 실행되고 있는 운영 체제를 표시합니다. 값은 다음과 같습니다.

#### **PL400**

IBM i.

### **IBM i** **IBM i의 PubSubMode(10자리의 부호 있는 정수)**

발행/구독 엔진 및 큐에 있는 발행/구독 인터페이스가 실행 중이므로 애플리케이션이 API(Application Programming Interface) 및 큐에 있는 발행/구독 인터페이스로 모니터링되는 큐를 사용하여 발행 또는 구독할 수 있는지 여부입니다.

값은 다음 중 하나일 수 있습니다.

#### **PSMCP**

발행/구독 엔진이 실행 중입니다. 따라서 API(Application Programming Interface)를 사용하여 발행/구독할 수 있습니다. 큐에 있는 발행/구독 인터페이스가 실행 중이 아니므로 큐에 있는 발행/구독 인터페이스로 모니터링하는 큐에 넣은 메시지가 적용되지 않습니다. 이 설정은 이 큐 관리자를 사용하는 WebSphere Message Broker V6 또는 이전 버전과의 호환성을 위해 사용하는데, 그 이유는 큐에 보관된 발행/구독 인터페이스가 일반적으로 읽는 동일한 큐를 읽어야 하기 때문입니다.

#### **PSMDS**

발행/구독 엔진 및 큐 발행/구독 인터페이스가 실행 중이지 않습니다. 따라서 API(Application Programming Interface)를 사용하여 발행/구독할 수 없습니다. 큐된 발행/구독 인터페이스에서 모니터링하는 큐에 넣은 발행/구독 메시지가 처리되지 않습니다.

#### **PSMEN**

발행/구독 엔진 및 큐에 있는 발행/구독 인터페이스가 실행 중입니다. 따라서 API(Application Programming Interface) 및 큐에 있는 발행/구독 인터페이스에서 모니터링하는 큐를 사용하여 발행/구독할 수 있습니다. 이것이 큐 관리자의 초기 기본값입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 PSMODE 선택자를 사용하십시오.

### **IBM i** **IBM i의 QMgrDesc(64바이트 문자열)**

큐 관리자 설명입니다.

주석에 사용할 수 있는 필드입니다. 이 필드의 내용은 큐 관리자에 중요하지 않습니다. 그러나 큐 관리자에서 필드에 표시될 수 있는 문자만 포함되어야 할 수도 있습니다. 널(null) 문자는 포함할 수 없지만, 필요한 경우 오른쪽으로 공백을 채워넣을 수 있습니다. DBCS 설치 시 이 필드는 DBCS 문자(최대 필드 길이 64비트에 따라)를 포함할 수 있습니다.

**참고:** 이 필드에 큐 관리자의 문자 세트에 없는 문자가 포함된 경우(CodedCharSetId 큐 관리자 속성에 정의된 대로), 이 필드가 다른 큐 관리자로 송신되면 해당 문자가 올바르게 않게 변환될 수 있습니다.

IBM i에서 기본값은 공백입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CAQMD 선택자를 사용하십시오. 이 속성의 길이는 LNQMD에서 제공됩니다.

### IBM i IBM i의 QMgrIdentifier(48바이트 문자열)

큐 관리자의 내부적으로 생성된 고유 ID입니다.

이 속성은 큐 관리자의 내부적으로 생성된 고유 이름입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CAQMID 선택자를 사용하십시오. 이 속성의 길이는 LNQMID에서 제공됩니다.

### IBM i IBM i의 QMgrName(48바이트 문자열)

큐 관리자 이름.

로컬 큐 관리자의 이름, 즉 애플리케이션이 연결되는 큐 관리자의 이름입니다.

이름의 처음 12자는 고유 메시지 ID를 구성하는 데 사용됩니다(1063 페이지의 『IBM i의 MQMD(메시지 디스크립터)』에 설명된 MDMID 필드 참조). 따라서 메시지 ID가 큐 관리자 네트워크에서 고유하기 위해 상호 통신할 수 있는 큐 관리자 이름의 처음 12자가 서로 달라야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CAQMN 선택자를 사용하십시오. 이 속성의 길이는 LNQMN에서 제공됩니다.

### IBM i IBM i의 RemoteEvent(10자리의 부호 있는 정수)

리모트 오류 이벤트의 생성 여부를 제어합니다.

값은 다음 중 하나입니다.

#### EVRDIS

이벤트 보고를 사용하지 않습니다.

#### EVRENA

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 IARMTE 선택자를 사용하십시오.

### IBM i IBM i의 RepositoryName(48바이트 문자열)

큐 관리자가 저장소 서비스를 제공하는 클러스터의 이름입니다.

이 큐 관리자가 저장소 관리자 서비스를 제공하는 클러스터의 이름입니다. 큐 관리자가 둘 이상의 클러스터에 이 서비스를 제공하면 *RepositoryNameList*는 클러스터를 식별하는 이름 목록 오브젝트의 이름을 지정하고, *RepositoryName*은 공백입니다. *RepositoryName* 및 *RepositoryNameList* 중 하나 이상은 공백이어야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CARPN 선택자를 사용하십시오. 이 속성의 길이는 LNQMN에서 제공됩니다.

### IBM i IBM i의 RepositoryNameList(48바이트 문자열)

이 큐 관리자가 저장소 서비스를 제공하는 클러스터의 이름을 포함하는 이름 목록 오브젝트의 이름입니다.

이 큐 관리자가 저장소 관리자 서비스를 제공하는 클러스터의 이름을 포함하는 이름 목록 오브젝트의 이름입니다. 큐 관리자가 하나의 클러스터에만 이 서비스를 제공하는 경우, 이름 목록 오브젝트에 하나의 이름만 있습니다. 또는, *RepositoryName*을 사용하여 클러스터의 이름을 지정할 수 있는데, 이 경우 *RepositoryNameList*는 공백입니다. *RepositoryName* 및 *RepositoryNameList* 중 하나 이상은 공백이어야 합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 CARPNL 선택자를 사용하십시오. 이 속성의 길이는 LNNLN에서 제공됩니다.

### IBM i IBM i의 SSLEvent(문자열)

TLS 이벤트의 생성 여부를 판별합니다.



값은 다음 중 하나입니다.

- EVRENA (MQINQ/PCF/config 이벤트) ENABLED (MQSC): TLS 이벤트가 생성됩니다(즉, RC2371 이벤트가 생성됨).
- EVRDIS (MQINQ/PCF/config 이벤트) DISABLED (MQSC): TLS 이벤트가 생성되지 않습니다. 이것이 큐 관리자의 초기 기본값입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IASSLE 선택자를 사용하십시오.

### **IBM i IBM i의 SSLKeyResetCount(정수)**

비밀 키를 재협상하기 전에 TLS 대화 안에서 송신 및 수신된 암호화되지 않은 바이트의 합계를 판별합니다. 바이트수에 메시지 채널 에이전트(MCA)가 송신하는 제어 정보가 포함됩니다.

이 값은 이 큐 관리자에서 통신을 시작하는 TLS 채널 MCA에만 사용됩니다(즉, 송신자 및 수신자 채널 페어링에서 송신자 채널 MCA).

이 속성의 값이 0보다 크고 채널 하트비트가 채널에 사용되는 경우, 채널 하트비트에 이어 데이터를 송신 또는 수신하기 전에도 비밀 키가 재협상됩니다. 재협상에 성공할 때마다 그 후에 다음 비밀 키 재협상이 재설정될 때까지 바이트수입니다.

값의 범위는 0 - 999 999 999입니다. 이 속성의 값이 0이면 비밀 키가 재협상되지 않음을 나타냅니다. TLS 비밀 키 재설정 계수를 1B - 32KB 범위로 지정하는 경우 TLS 채널은 32KB의 비밀 키 재설정 계수를 사용합니다. 이것은 TLS 비밀 키 재설정 값이 작은 경우에 발생하는 초과 키 재설정의 처리 비용을 피하기 위함입니다.

SSL 서버가 IBM MQ 큐 관리자이고 비밀 키 재설정과 채널 하트비트가 사용되는 경우, 각채널 비트하트 직후 재협상이 발생합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IASSRC 선택자를 사용하십시오.

### **IBM i IBM i의 StartStopEvent(10자리의 부호 있는 정수)**

시작/중지 이벤트의 생성 여부를 제어합니다.

이 속성의 값은 다음 중 하나입니다.

#### **EVRDIS**

이벤트 보고를 사용하지 않습니다.

#### **EVRENA**

이벤트 보고를 사용합니다.

이벤트에 대한 자세한 정보는 [이벤트 모니터링](#)을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 IASSE 선택자를 사용하십시오.

### **IBM i IBM i의 SyncPoint(10자리의 부호 있는 정수)**

동기점 사용 가능성입니다.

로컬 큐 관리자가 작업 단위 및 MQGET, MQPUT, MQPUT1 호출과의 동기점 조정을 지원하는지 여부입니다.

#### **SPAVL**

작업 단위 및 동기점 조정을 사용할 수 있습니다.

#### **SPNAVL**

작업 단위 및 동기점 조정이 사용 불가능합니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IASYNC 선택자를 사용하십시오.

### **IBM i IBM i의 TraceRouteRecording(10자리의 부호 있는 정수)**

메시지 정보가 큐 관리자를 거쳐 이동할 때 기록되는지 여부를 제어합니다.

값은 다음 중 하나입니다.

- RECD: 추적 라우트 메시지에 추가할 수 없음
- RECDQ: 메시지를 지정된 고정 큐에 넣음
- RECDM: 메시지 사용을 판별함(초기 기본 설정임)

추적 라우트 메시지가 시스템에 남아 있지 않게 하려면 만기 값을 0보다 큰 값으로 설정하고 RODISC 보고 옵션을 지정하십시오. 보고 메시지 또는 응답 메시지가 시스템에 남아 있지 않게 하려면 보고 옵션 ROPDAE를 설정하십시오. 자세한 정보는 1365 페이지의 『IBM i의 보고 옵션 및 메시지 플래그』의 내용을 참조하십시오.

이 속성의 값을 판별하려면 MQINQ 호출에서 IATRGI 선택자를 사용하십시오.

### IBM i IBM i의 TreeLifeTime(10자리의 부호 있는 정수)

비관리 토픽의 지속 시간(초)입니다.

비관리 토픽은 애플리케이션에서 관리 노드로 존재하지 않는 토픽 문자열에 발행하거나 해당 토픽 문자열로 구독할 때 작성되는 토픽입니다. 이 비관리 노드에 더 이상 활성 구독이 없으면 이 매개변수는 이 노드를 제거하기 전에 큐 관리자가 대기할 시간을 판별합니다. 지속 가능 구독에 사용 중인 비관리 토픽만이 큐 관리자가 재생된 후 남습니다.

0 - 604,000 범위의 값을 지정하십시오. 0 값은 큐 관리자가 비관리 토픽을 제거하지 않음을 의미합니다. 큐 관리자의 초기 기본 값은 1800입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IATRLFT 선택자를 사용하십시오.

### IBM i IBM i의 TriggerInterval(10자리의 부호 있는 정수)

트리거 메시지 간격입니다.

트리거 메시지의 수를 제한하는 데 사용하는 시간 간격(밀리초)입니다. 이 속성은 *TriggerType*이 TTFIRST인 경우에만 관련이 있습니다. 이 경우, 적절한 메시지가 큐에 도착하고 큐가 이전에 비어 있었던 경우에만 트리거 메시지가 정상적으로 생성됩니다. 그러나 특정 상황에서는 큐가 비어 있지 않았어도 TTFIRST 트리거로 추가 트리거 메시지가 생성될 수 있습니다. 이러한 추가 트리거 메시지는 각 *TriggerInterval* 밀리초보다 더 자주 생성되지 않습니다.

트리거에 대한 자세한 정보는 [트리거링 채널](#)의 내용을 참조하십시오.

값의 범위는 0 - 999 999 999입니다. 기본값은 999,999,999입니다.

이 속성의 값을 판별하려면 MQINQ 호출에서 IATRGI 선택자를 사용하십시오.

## 애플리케이션

이 정보에서는 IBM MQ for IBM i for RPG에 제공되는 샘플 프로그램에 대해 설명합니다. 또한, 작성하는 프로그램에서 실행 가능한 애플리케이션을 빌드하는 방법을 학습하십시오.

### 애플리케이션 빌드

IBM i 발행물은 작성되는 프로그램에서 실행 가능한 애플리케이션을 빌드하는 방법에 대해 설명합니다. 이 주제에서는 IBM i에서 실행할 IBM MQ for IBM i 애플리케이션을 빌드할 때 수행해야 하는 추가 태스크 및 표준 태스크에 대한 변경사항을 설명합니다.

MQI 호출을 소스 코드에 코딩하는 것 외에, RPG 언어에 대한 IBM MQ for IBM i 복사 파일을 포함하도록 적절한 언어 명령문을 추가해야 합니다. 이 파일의 콘텐츠에 익숙해져야 합니다. 해당 이름과 콘텐츠에 대한 간단한 설명이 다음 텍스트에서 제공됩니다.

### IBM i IBM MQ 파일 복사 IBM i

IBM MQ for IBM i는 RPG 프로그래밍 언어에서 애플리케이션 작성을 도와주는 복사 파일을 제공합니다. 이 파일은 WebSphere Development 도구 세트(5722 WDS) ILE RPG 4 컴파일러에 사용하기에 적합합니다.

채널 엑시트 작성을 돕기 위해 IBM MQ for IBM i가 제공하는 복사 파일에 대한 설명이 [메시징 채널에 대한 채널 엑시트 프로그램](#)에 나와 있습니다.

RPG용 IBM MQ for IBM i 복사 파일의 이름에는 접두부 CMQ가 붙습니다. G 또는 H의 접미부가 있습니다. 이름 지정된 상수를 포함하는 개별 사본 파일과 각 구조에 대한 하나의 파일이 있습니다. 복사 파일이 968 페이지의 『언어 고려사항』에 나열되어 있습니다.

**참고:** ILE RPG/400의 경우, 해당 파일은 파일의 멤버로 제공됩니다. 라이브러리 QMQM의 QRPGLSRC입니다.

구조 선언은 DS 명령문을 포함하지 않습니다. 이렇게 하면 애플리케이션이 선언의 나머지 부분에 복사하기 위해 DS 명령문을 코딩하고 /COPY 명령문을 사용하여 데이터 구조(또는 다중 발생 데이터 구조)를 선언할 수 있습니다.

ILE RPG/400의 경우 명령문은 다음과 같습니다.

```
D*..1.....2.....3.....4.....5.....6.....7
D* Declare an MQMD data structure
D MQMD DS
D/COPY CMQMDG
```

## 실행할 프로그램 준비

실행 가능 IBM MQ for IBM i 애플리케이션을 작성하려면 작성한 소스 코드를 컴파일해야 합니다.

ILE RPG/400에 이 작업을 수행하려면 일반 IBM i 명령, CRTRPGMOD 및 CRTPGM을 사용할 수 있습니다.

\*MODULE를 작성한 후, CRTPGM 명령에 BNDSRVPGM(QMQM/LIBMQM)을 지정해야 합니다. 여기에는 프로그램의 다양한 IBM MQ 프로시저가 포함됩니다.

컴파일을 수행할 때는 복사 파일(QMQM)이 들어 있는 라이브러리가 라이브러리 목록에 있는지 확인하십시오.

클라이언트 모드를 포함하여 프로그래밍 고려사항에 대한 자세한 정보는 968 페이지의 『언어 고려사항』의 내용을 참조하십시오.

## IBM i 외부 동기점 관리자에 대한 인터페이스

IBM MQ for IBM i는 기본 IBM i 커밋 제어를 외부 동기점 조정자로 사용합니다.

IBM i의 커밋 제어 기능에 대한 자세한 정보는 IBM i 프로그래밍: 백업 및 복구 안내서를 참조하십시오.

IBM i 커밋 제어 기능을 시작하려면 STRCMTCTL 시스템 명령을 사용하십시오. 커밋 제어를 종료하려면 ENDCMTCTL 시스템 명령을 사용하십시오.

**참고:** 커밋 정의 범위의 기본값은 \*ACTGRP입니다. IBM MQ for IBM i에는 \*JOB으로 정의되어야 합니다. 예를 들면, 다음과 같습니다.

```
STRCMTCTL LCKLVL(*ALL) CMTSCOPE(*JOB)
```

커밋 제어를 시작한 후 MQPUT, MQPUT1 또는 MQGET을 호출하고 PMSYP 또는 GMSYP를 지정하는 경우, IBM MQ for IBM i가 자체적으로 커밋 정의에 API 커밋 자원으로 추가됩니다. 이는 일반적으로 작업에서 최초의 호출입니다. 특정 커밋 정의에 따라 등록된 API 커밋 자원이 있는 동안은 해당 정의에 대한 커밋 제어를 종료할 수 없습니다.

현재 작업 단위에 보류 중인 MQI 작업이 없는 경우, 큐 관리자와 연결을 끊으면 IBM MQ for IBM i에서는 등록을 API 커밋 자원으로 제거합니다.

현재 작업 단위에 보류 중인 MQPUT, MQPUT1 또는 MQGET 조작이 있는 동안 큐 관리자로부터 연결을 끊은 경우에는 IBM MQ for IBM i가 여전히 API 커밋 자원으로 등록되어 있으므로 다음 커밋 또는 롤백에 대한 알림을 받습니다. 다음 동기점에 도달하면, IBM MQ는 필요에 따라 변경사항을 커밋하거나 롤백합니다. 애플리케이션이 활성 작업 단위 동안 큐 관리자의 연결을 끊었다가 다시 연결하고 동일한 작업 단위 내에서 추가 MQGET 및 MQPUT 조작을 수행할 수 있습니다(보류 중인 연결 끊기).

해당 커밋 정의에 대한 ENDCMTCTL 시스템 명령을 실행하려고 시도하면, 보류 중인 변경이 활성화되었음을 나타내는 메시지 CPF8355가 발행됩니다. 이 메시지는 작업이 끝날 때에도 작업 로그에 표시됩니다. 이러한 상황을 피하려면 모든 보류 중인 IBM MQ 조작을 커밋 또는 롤백하고 큐 관리자와 연결을 끊어야 합니다. 따라서 성공적으로 완료하기 위해 ENDCMTCTL 앞에 COMMIT 또는 ROLLBACK 명령을 사용하여 커밋 제어 종료를 사용하도록 설정해야 합니다.

IBM i 커밋 제어가 동기점 조정자로 사용되면 MQCMIT, MQBACK 및 MQBEGIN 호출이 발행되지 않을 수 있습니다. 이 함수의 호출이 이유 코드 RC2012와 함께 실패합니다.

작업 단위를 커밋하거나 롤백하려면 커밋 제어를 지원하는 프로그래밍 언어 중 하나를 사용하십시오. 예를 들면, 다음과 같습니다.



- CL 명령: COMMIT 및 ROLLBACK
- ILE C 프로그래밍 함수: \_Rcommit 및 \_Rollback
- RPG/400: COMMIT 및 ROLBK
- COBOL/400®: COMMIT 및 ROLLBACK

### **CICS for IBM i 애플리케이션의 동기점**

IBM MQ for IBM i는 CICS와 함께 작업 단위에 참여합니다. CICS 애플리케이션에서 MQI를 사용하여 현재 작업 단위 내부에 메시지를 넣고 가져올 수 있습니다.

EXEC CICS SYNCPOINT 명령을 사용하여 IBM MQ for IBM i 조작을 포함하는 동기점을 설정할 수 있습니다. 이전 동기점까지 모든 변경사항을 백아웃하려면 EXEC CICS SYNCPOINT ROLLBACK 명령을 사용할 수 있습니다.

CICS 애플리케이션에서 PMSYP 또는 GMSYP 옵션 세트와 함께 MQPUT, MQPUT1 또는 MQGET을 사용하는 경우, IBM MQ for IBM i가 API 커밋 자원으로 해당 등록을 제거할 때까지 CICS 로그오프할 수 없습니다. 따라서 큐 관리자와 연결을 끊기 전에 보류 중인 Put 또는 Get 조작을 커밋하거나 백업해야 합니다. 그러면 CICS에서 로그오프할 수 있습니다.

### **IBM i의 샘플 프로그램**

이 주제에서는 IBM MQ for IBM i for RPG에서 제공되는 샘플 프로그램에 대해 설명합니다. 샘플은 MQI(Message Queue Interface)의 일반적인 사용을 보여줍니다.

샘플은 일반 프로그래밍 기술을 보여주려는 목적이 아니므로 제작 프로그램에서 원하는 몇 가지 오류 검사가 생략되어 있습니다. 그러나, 이러한 샘플은 사용자의 고유 메시지 큐잉 프로그램을 만드는 기반으로 사용하기에 적합합니다.

모든 샘플에 대한 소스 코드는 제품과 함께 제공됩니다. 이 소스에는 프로그램에서 보여준 메시지 큐잉 기술을 설명하는 주석이 포함되어 있습니다.

ILE 샘플 프로그램 세트가 하나 있습니다.

#### **1. MQI에 대해 프로토타입 호출을 사용하는 프로그램(정적 바인드 호출)**

소스가 QMQMSAMP/QRPGLESRC에 있습니다. 멤버에는 AMQ3xxx4라는 이름이 지정되고, 여기서 xxx는 샘플 함수를 나타냅니다. 복사 멤버는 QMQM/QRPGLESRC에 존재합니다. 각 멤버 이름에는 G 또는 H라는 접미부가 있습니다.

1349 페이지의 표 811에 IBM MQ for IBM i에서 제공되는 샘플 프로그램의 전체 목록과 지원되는 프로그래밍 언어별 프로그램의 이름이 나와 있습니다. 해당 이름은 모두 접두부 AMQ로 시작하고 이름의 4번째 문자가 프로그래밍 언어를 나타내는 점에 유의하십시오.

표 811. 샘플 프로그램의 이름	
	RPG(ILE)
Put 샘플	AMQ3PUT4
찾아보기 샘플	AMQ3GBR4
Get 샘플	AMQ3GET4
요청 샘플	AMQ3REQ4
에코 샘플	AMQ3ECH4
조회 샘플	AMQ3INQ4
설정 샘플	AMQ3SET4
트리거 모니터 샘플	AMQ3TRG4
트리거 서버 샘플	AMQ3SRV4

뿐만 아니라, IBM MQ for IBM i 샘플 옵션에는 특정 샘플 프로그램과 관리 태스크를 표시하는 샘플 CL 프로그램에 대한 입력으로 사용할 수 있는 샘플 데이터 파일 AMQSDATA가 포함되어 있습니다. CL 샘플은 IBM i관리에서 설명합니다. 샘플 CL 프로그램을 사용하여 이 주제에 설명된 샘플 프로그램에 사용할 큐를 작성할 수 있습니다.

샘플 프로그램 실행 방법에 대한 정보는 1350 페이지의 『IBM i에서 샘플 프로그램 준비 및 실행』의 내용을 참조하십시오.

### IBM i의 샘플 프로그램에 나타나는 기능

IBM MQ for IBM i 샘플 프로그램에 나타나는 기술 설명 표

일부 기술은 둘 이상의 샘플 프로그램에서 발생하지만 하나의 프로그램만 테이블에 나열됩니다. 모든 샘플은 MQOPEN 및 MQCLOSE 호출을 사용하여 큐를 열고 닫으므로 이 기술은 표에 별도로 나열되지 않습니다.

표 812. MQI 사용을 보여주는 샘플 프로그램	
기술	RPQ(ILE)
MQCONN 및 MQDISC 호출 사용	AMQ3ECH4 또는 AMQ3INQ4
암시적으로 연결 및 연결 끊기	AMQ3PUT4
MQPUT 호출을 사용한 메시지 넣기	AMQ3PUT4
MQPUT1 호출을 사용한 단일 메시지 넣기	AMQ3ECH4 또는 AMQ3INQ4
요청 메시지에 응답	AMQ3INQ4
메시지 가져오기(대기 없음)	AMQ3GBR4
메시지 가져오기(시간 제한이 있는 대기)	AMQ3GET4
메시지 가져오기(데이터 변환 사용)	AMQ3ECH4
큐 찾아보기	AMQ3GBR4
공유 입력 큐 사용	AMQ3INQ4
독점 입력 큐 사용	AMQ3REQ4
MQINQ 호출 사용	AMQ3INQ4
MQSET 호출 사용	AMQ3SET4
응답 대상 큐 사용	AMQ3REQ4
예외 메시지 요청	AMQ3REQ4
잘린 메시지 허용	AMQ3GBR4
해석된 큐 이름 사용	AMQ3GBR4
처리 트리거	AMQ3SRV4 또는 AMQ3TRG4

**참고:** 모든 샘플 프로그램은 처리 결과를 포함한 스푼 파일을 생성합니다.

### IBM i에서 샘플 프로그램 준비 및 실행

IBM MQ for IBM i 샘플 프로그램을 실행하기 전에 다른 IBM MQ for IBM i 애플리케이션에서 하는 것처럼 컴파일해야 합니다. 이렇게 하려면 IBM i 명령 CRTRPGMOD 및 CRTPGM을 사용합니다.

AMQ3xxx4 프로그램을 작성할 때 CRTPGM 명령에 BNDSRVPGM(QMQM/LIBMQM)을 지정해야 합니다. 이 작업에는 프로그램에 있는 다양한 IBM MQ 프로시저가 포함됩니다.

샘플 프로그램은 라이브러리 QMQMSAMP에서 QRPGLSRC 멤버로 제공됩니다. 라이브러리 QMQM에서 제공되는 복사 파일을 사용하므로 컴파일할 때 이 라이브러리가 라이브러리 목록에 있는지 확인하십시오. 샘플은 복사 파일에 선언되는 많은 변수를 사용하지 않기 때문에 RPG 컴파일러에서 정보 메시지를 제공합니다.

## 샘플 프로그램 실행

샘플 실행 시 고유의 큐를 사용하거나, AMQSAMP4를 컴파일하고 실행하여 몇 가지 샘플 큐를 작성할 수 있습니다. 이 프로그램의 소스는 라이브러리 QMQMSAMP의 파일 QCLSRC에서 제공됩니다. CRTCLPGM 명령을 사용하여 컴파일할 수 있습니다.

샘플 프로그램 중 하나를 호출하려면 다음 명령을 사용하십시오.

```
CALL PGM(QMQMSAMP/AMQ3PUT4) PARM('Queue_Name','Queue_Manager_Name')
```

Queue\_Name 및 Queue\_Manager\_Name이 48자여야 하는 경우, Queue\_Name 및 Queue\_Manager\_Name에 필요한 수만큼 공백을 채우면 됩니다.

조회 및 설정 샘플 프로그램의 경우, AMQSAMP4로 작성된 샘플 정의 때문에 이 샘플의 C 버전이 트리거될 수 있습니다. RPG 버전을 트리거하려면 프로세스 정의 SYSTEM.SAMPLE.ECHOPROCESS 및 SYSTEM.SAMPLE.INQPROCESS, SYSTEM.SAMPLE.SETPROCESS를 변경해야 합니다. CHGMQMPRC 명령 (MQ 프로세스 변경(CHGMQMPRC)에 설명됨)을 사용하여 이 작업을 수행하거나, 대체 정의를 사용해 AMQSAMP4를 편집 및 실행할 수 있습니다.

## IBM i의 Put 샘플 프로그램

Put 샘플 프로그램, AMQ3PUT4는 MQPUT 호출을 사용하여 메시지를 큐에 넣습니다.

프로그램을 시작하려면, 프로그램을 호출하고 프로그램 매개변수로 대상 큐의 이름을 지정하십시오. 프로그램에서 고정 메시지 세트를 큐에 넣습니다. 이 메시지는 프로그램 소스 코드 끝의 데이터 블록에서 가져온 것입니다. 샘플 Put 프로그램은 라이브러리 QMQMSAMP에서 AMQ3PUT4입니다.

이 예제 프로그램을 사용하는 경우, 명령은 다음과 같습니다.

```
CALL PGM(QMQMSAMP/AMQ3PUT4) PARM('Queue_Name','Queue_Manager_Name')
```

Queue\_Name 및 Queue\_Manager\_Name이 48자여야 하는 경우, Queue\_Name 및 Queue\_Manager\_Name에 필요한 수만큼 공백을 채우면 됩니다.

## Put 샘플 프로그램의 디자인

프로그램은 OOOOUT 옵션과 MQOPEN 호출을 사용하여 메시지를 넣을 대상 큐를 엽니다. 그 결과가 스푼 파일의 출력입니다. 큐를 열 수 없으면 프로그램에서 MQOPEN 호출이 리턴한 이유 코드를 포함하는 오류 메시지를 기록합니다. 프로그램을 단순하게 유지하기 위해 이 호출 및 후속 MQI 호출에서 프로그램은 여러 옵션의 기본값을 사용합니다.

소스 코드에 있는 각 데이터 행에 대해, 텍스트를 버퍼로 읽어오고 MQPUT 호출을 사용하여 해당 행의 텍스트를 포함한 데이터그램 메시지를 작성합니다. 프로그램은 입력의 끝에 도달하거나 MQPUT 호출에 실패할 때까지 계속됩니다. 프로그램이 입력의 끝에 도달하는 경우 MQCLOSE 호출을 사용하여 큐를 닫습니다.

## IBM i의 찾아보기 샘플 프로그램

찾아보기 샘플 프로그램, AMQ3GBR4는 MQGET 호출을 사용하여 큐에서 메시지를 찾습니다.

프로그램을 호출하면 프로그램은 사용자가 지정한 큐에서 모든 메시지의 사본을 검색합니다. 해당 메시지는 큐에 그대로 있습니다. 제공된 큐 SYSTEM.SAMPLE.LOCAL을 사용할 수 있습니다. 먼저 Put 샘플 프로그램을 실행하여 일부 메시지를 큐에 넣습니다. 큐 SYSTEM.SAMPLE.ALIAS를 사용할 수 있는데, 이는 동일한 로컬 큐에 대한 알리아스 이름입니다. 프로그램은 큐의 끝에 도달하거나 MQI 호출에 실패할 때까지 계속됩니다.

RPG 프로그램 호출 명령의 예는 다음과 같습니다.

```
CALL PGM(QMQMSAMP/AMQ3GBR4) PARM('Queue_Name','Queue_Manager_Name')
```

Queue\_Name 및 Queue\_Manager\_Name이 48자여야 하는 경우, Queue\_Name 및 Queue\_Manager\_Name에 필요한 수만큼 공백을 채우면 됩니다. 따라서 SYSTEM.SAMPLE.LOCAL을 대상 큐로 사용하는 경우, 29개 공백 문자가 필요합니다.

## 찾아보기 샘플 프로그램의 디자인

이 프로그램은 OOBROW 옵션과 MQOPEN 호출을 사용하여 대상 큐를 엽니다. 큐를 열 수 없으면 프로그램에서 MQOPEN 호출이 리턴한 이유 코드를 포함하는 오류 메시지를 스펴 파일에 기록합니다.

큐의 각 메시지의 경우 프로그램은 큐에서 메시지를 복사하기 위해 MQGET 호출을 사용한 후 메시지에 포함된 데이터를 표시합니다. MQGET 호출은 이러한 옵션을 사용합니다.

### GMBRWN

MQOPEN 호출 후, 찾아보기 커서가 큐의 첫 번째 메시지 앞에 논리적으로 배치되므로 이 옵션을 사용하면 첫 호출 시 첫 번째 메시지가 리턴됩니다.

### GMNWT

큐 위의 메시지가 없는 경우 프로그램은 대기하지 않습니다.

### GMATM

MQGET 호출은 고정된 크기의 버퍼를 지정합니다. 메시지가 이 버퍼보다 긴 경우 프로그램은 메시지가 잘렸다는 경과와 함께 잘린 메시지를 표시합니다.

각 MQGET 호출 이후 MQMD 구조의 MDMID 및 MDCID 필드를 어떻게 지워야 하는지가 프로그램에 표시됩니다. 이 호출이 해당 필드를 검색한 메시지에 있는 값으로 설정하기 때문입니다. 이러한 필드를 지운다는 것은 연속적인 MQGET 호출이 메시지가 큐에 보유되는 순서대로 메시지를 검색한다는 것을 의미합니다.

큐의 끝까지 프로그램이 계속됩니다. 따라서 MQGET 호출은 RC2033(메시지 없음) 이유 코드를 리턴하고 프로그램에 경고 메시지가 표시됩니다. MQGET 호출이 실패하면 스펴 파일에 이유 코드를 포함한 오류 메시지가 기록됩니다.

그런 다음 프로그램은 MQCLOSE 호출을 사용하여 큐를 닫습니다.

## IBM i의 Get 샘플 프로그램

Get 샘플 프로그램, AMQ3GET4는 MQGET 호출을 사용하여 큐에서 메시지를 가져옵니다.

프로그램이 호출되면 지정된 큐에서 메시지를 제거합니다. 제공된 큐 SYSTEM.SAMPLE.LOCAL을 사용할 수 있습니다. 먼저 Put 샘플 프로그램을 실행하여 일부 메시지를 큐에 넣습니다. SYSTEM.SAMPLE.ALIAS 큐를 사용할 수 있는데, 이는 동일한 로컬 큐에 대한 알리어스 이름입니다. 프로그램은 큐가 비거나 MQI 호출이 실패할 때까지 계속됩니다.

RPG 프로그램 호출 명령의 예는 다음과 같습니다.

```
CALL PGM(QMQMSAMP/AMQ3GET4) PARM('Queue_Name', 'Queue_Manager_Name')
```

여기서 Queue\_Name 및 Queue\_Manager\_Name의 길이는 48자여야 하며 Queue\_Name 및 Queue\_Manager\_Name에 필요한 수만큼 공백을 채우면 됩니다. 따라서 SYSTEM.SAMPLE.LOCAL을 대상 큐로 사용하는 경우, 29개 공백 문자가 필요합니다.

## Get 샘플 프로그램의 디자인

프로그램이 메시지를 가져올 대상 큐를 엽니다. OOINPQ 옵션과 MQOPEN 호출을 사용합니다. 큐를 열 수 없으면 프로그램에서 MQOPEN 호출이 리턴한 이유 코드를 포함하는 오류 메시지를 스펴 파일에 기록합니다.

큐의 각 메시지에 대해, 프로그램은 MQGET 호출을 사용하여 큐에서 메시지를 제거한 다음 메시지에 있는 데이터를 표시합니다. MQGET 호출은 GMWT 옵션을 사용하여 15초의 대기 간격(GMWT)을 지정하므로 큐에 메시지가 없으면 이 시간 동안 프로그램이 대기합니다. 이 간격의 만기 전에 메시지가 도착하지 않으면 호출이 실패하고 RC2033(메시지 없음) 이유 코드를 리턴합니다.

각 MQGET 호출 이후 MQMD 구조의 MDMID 및 MDCID 필드를 어떻게 지워야 하는지가 프로그램에 표시됩니다. 이 호출이 해당 필드를 검색한 메시지에 있는 값으로 설정하기 때문입니다. 이러한 필드를 지운다는 것은 연속적인 MQGET 호출이 메시지가 큐에 보유되는 순서대로 메시지를 검색한다는 것을 의미합니다.

MQGET 호출은 고정된 크기의 버퍼를 지정합니다. 메시지가 이 버퍼보다 더 긴 경우 호출에 실패하고 프로그램은 중지됩니다.

MQGET 호출이 RC2033(메시지 없음) 이유 코드를 리턴하거나 MQGET 호출이 실패할 때까지 프로그램이 계속됩니다. 호출에 실패하는 경우 프로그램은 이유 코드를 포함하는 오류 메시지를 표시합니다.

그런 다음 프로그램은 MQCLOSE 호출을 사용하여 큐를 닫습니다.

### IBM i의 요청 샘플 프로그램

요청 샘플 프로그램, AMQ3REQ4는 클라이언트/서버 처리를 보여줍니다. 샘플은 서버 프로그램이 처리한 요청 메시지를 큐에 넣는 클라이언트입니다. 서버 프로그램이 응답 메시지를 응답 대상 큐에 넣을 때까지 대기합니다.

요청 샘플은 MQPUT 호출을 사용하여 일련의 요청 메시지를 큐에 넣습니다. 이 메시지가 응답 대상 큐로 SYSTEM.SAMPLE.REPLY를 지정합니다. 프로그램에서 응답 메시지를 대기했다가 표시합니다. 서버 애플리케이션이 대상 큐(서버 큐라고 함)를 처리하고 있거나 해당 목적으로 애플리케이션이 트리거된 경우(조회 및 설정 샘플 프로그램이 트리거되도록 설계됨)에만 응답이 송신됩니다. 샘플은 첫 번째 응답이 도착할 때까지 5분(서버 애플리케이션의 트리거 시간 허용), 후속 응답에 대해 15초 대기하지만, 응답을 가져오지 않고도 종료될 수 있습니다.

프로그램을 시작하려면, 프로그램을 호출하고 프로그램 매개변수로 대상 큐의 이름을 지정하십시오. 프로그램에서 고정 메시지 세트를 큐에 넣습니다. 이 메시지는 프로그램 소스 코드 끝의 데이터 블록에서 가져온 것입니다.

### 요청 샘플 프로그램의 디자인

프로그램이 메시지를 넣을 수 있도록 서버 큐를 엽니다. OOOUT 옵션과 MQOPEN 호출을 사용합니다. 큐를 열 수 없는 경우 프로그램은 MQOPEN 호출에서 리턴한 이유 코드가 포함된 오류 메시지를 표시합니다.

그런 다음 응답 메시지를 가져올 수 있도록 프로그램은 SYSTEM.SAMPLE.REPLY라는 응답 대상 큐를 엽니다. 이를 위해 OOINPX 옵션과 MQOPEN 호출을 사용합니다. 큐를 열 수 없는 경우 프로그램은 MQOPEN 호출에서 리턴한 이유 코드가 포함된 오류 메시지를 표시합니다.

입력의 각 행에 대해 프로그램은 텍스트를 버퍼로 해석하고 MQPUT 호출을 사용하여 해당 행의 텍스트를 포함하는 요청 메시지를 작성합니다. 이 호출에서 프로그램은 ROEXCD 보고 옵션을 사용하여 요청 메시지에 관해 발송된 보고 메시지에 메시지 데이터의 처음 100바이트가 포함되도록 요청합니다. 프로그램은 입력의 끝에 도달하거나 MQPUT 호출에 실패할 때까지 계속됩니다.

그런 다음 프로그램은 MQGET 호출을 사용하여 큐에서 응답 메시지를 제거하고 응답에 포함된 데이터를 표시합니다. MQGET 호출은 GMWT 옵션을 사용하여 첫 번째 응답에 대해 5분(서버 애플리케이션의 트리거 시간 허용), 후속 응답에 대해 15초의 대기 간격(GMWI)을 지정합니다. 큐에 메시지가 없는 경우 프로그램은 이 기간 동안 대기합니다. 이 간격의 만기 전에 메시지가 도착하지 않으면 호출이 실패하고 RC2033(메시지 없음) 이유 코드를 리턴합니다. 또한 호출이 GMATM 옵션을 사용하므로 선언된 버퍼 크기보다 긴 메시지는 잘립니다.

각 MQGET 호출 이후 MQMD 구조의 MDMID 및 MDCOD 필드를 어떻게 지워야 하는지가 프로그램에 표시됩니다. 이 호출이 해당 필드를 검색한 메시지에 있는 값으로 설정하기 때문입니다. 이러한 필드를 지운다는 것은 연속적인 MQGET 호출이 메시지가 큐에 보유되는 순서대로 메시지를 검색한다는 것을 의미합니다.

MQGET 호출이 RC2033(메시지 없음) 이유 코드를 리턴하거나 MQGET 호출이 실패할 때까지 프로그램이 계속됩니다. 호출에 실패하는 경우 프로그램은 이유 코드를 포함하는 오류 메시지를 표시합니다.

그런 다음 프로그램이 MQCLOSE 호출을 사용하여 서버 큐와 응답 대상 큐를 닫습니다. 1353 페이지의 표 813은 조회 및 설정 샘플 프로그램 실행에 필요한 에코 샘플 프로그램의 변경사항을 보여줍니다.

**참고:** 에코 샘플 프로그램에 대한 자세한 정보가 참고로 제공됩니다.

프로그램 이름	SYSTEM/SAMPLE 큐	프로그램 시작
에코	ECHO	AMQ3ECH4
조회	INQ	AMQ3INQ4
설정	SET	AMQ3SET4

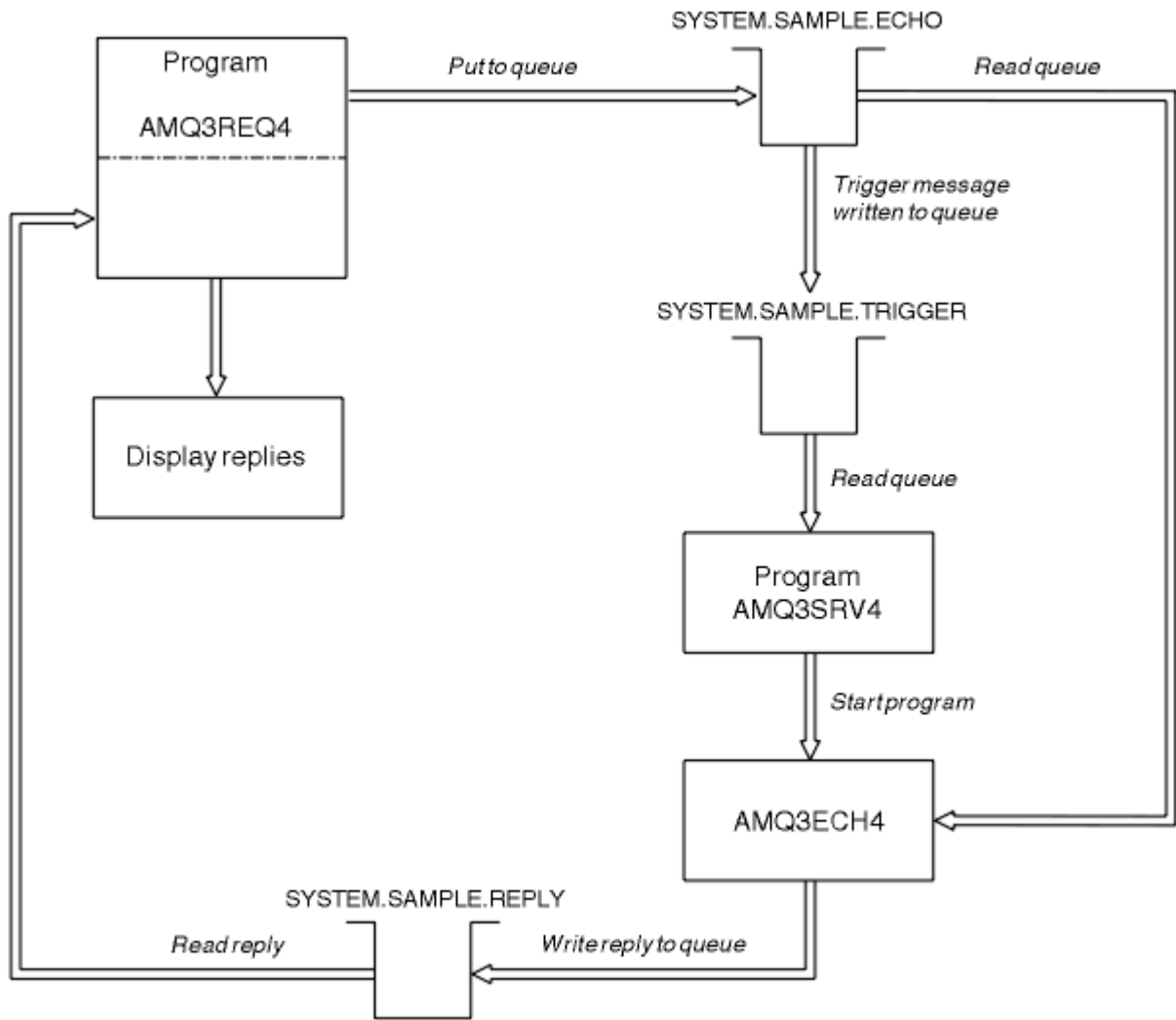


그림 9. 샘플 클라이언트/서버(에코) 프로그램 흐름도

#### IBM i IBM i에서 요청 샘플로 트리거 사용

트리거를 사용하여 샘플을 실행하려면 하나의 작업에서 필요한 이니시에이션 큐를 대상으로 트리거 서버 프로그램, AMQ3SRV4를 시작한 다음 다른 작업에서 AMQ3REQ4를 시작하십시오.

요청 샘플 프로그램에서 메시지를 보내는 경우 트리거 서버가 준비되었음을 의미합니다.

#### 참고:

1. 샘플은 SYSTEM.SAMPLE.ECHO, SYSTEM.SAMPLE.INQ 또는 SYSTEM.SAMPLE.SET 로컬 큐에 대한 이니시에이션 큐로 SYSTEM SAMPLE TRIGGER 큐를 사용합니다. 또는, 고유의 이니시에이션 큐를 정의할 수 있습니다.
2. AMQSAMP4가 작성한 샘플 정의 때문에 샘플의 C 버전이 트리거됩니다. RPG 버전을 트리거하려면 프로세스 정의 SYSTEM.SAMPLE.ECHOPROCESS 및 SYSTEM.SAMPLE.INQPROCESS, SYSTEM.SAMPLE.SETPROCESS를 변경해야 합니다. CHGMQMPRC 명령(자세한 정보는 MQ 프로세스 변경(CHGMQMPRC) 참조)을 사용하여 이 작업을 수행하거나, 고유 AMQSAMP4 버전을 편집 및 실행할 수 있습니다.
3. QMQMSAMP/QRPGLESRC에 제공되는 소스에서 트리거 서버 프로그램을 컴파일해야 합니다.

실행할 트리거 프로세스에 따라 AMQ3REQ4는 이들 샘플 서버 큐 중 하나에 요청 메시지를 넣도록 지정하는 매개변수를 사용하여 호출해야 합니다.

- SYSTEM.SAMPLE.ECHO (Echo 샘플 프로그램의 경우)
- SYSTEM.SAMPLE.INQ (Inquire 샘플 프로그램의 경우)



- SYSTEM.SAMPLE.SET (Set 샘플 프로그램의 경우)

SYSTEM.SAMPLE.ECHO 프로그램의 흐름도가 1354 페이지의 [그림 9](#)에 나와 있습니다. 예를 사용하는 경우, 이 서버에 RPG 프로그램 요청을 발행하는 명령은 다음과 같습니다.

```
CALL PGM(QMQMSAMP/AMQ3REQ4) PARM('SYSTEM.SAMPLE.ECHO
+ 30 blank characters','Queue_Manager_Name')
```

큐 이름과 큐 관리자 이름의 길이는 48자여야 하기 때문입니다.

**참고:** 이 샘플 큐에는 FIRST 트리거 유형이 있으므로 Request 샘플을 실행하기 전에 큐에 이미 메시지가 있는 경우 서버 애플리케이션은 송신된 메시지에 의해 트리거되지 않습니다.

추가 예를 시도하려는 경우 다음 변형을 시도할 수 있습니다.

- 작업을 대신 제출하려면 AMQ3SRV4 대신 AMQ3TRG4를 사용하십시오. 작업 제출 지연 가능성 때문에 과정을 따르기가 쉽지 않을 수 있습니다.
- SYSTEM.SAMPLE.INQ 및 SYSTEM.SAMPLE.SET 샘플 큐를 사용하십시오. 예제 데이터 파일을 사용하는 경우, 이 서버에 RPG 프로그램 요청을 발행하는 명령은 다음과 같습니다.

```
CALL PGM(QMQMSAMP/AMQ3INQ4) PARM('SYSTEM.SAMPLE.INQ
+ 31 blank characters')
CALL PGM(QMQMSAMP/AMQ3SET4) PARM('SYSTEM.SAMPLE.SET
+ 31 blank characters')
```

큐 이름의 길이는 48자여야 하기 때문입니다.

이러한 샘플 큐에도 FIRST 트리거 유형이 있습니다.

## IBM i의 에코 샘플 프로그램

에코 샘플 프로그램은 응답 큐로 송신된 메시지를 리턴합니다. 프로그램의 이름은 AMQ3ECH4입니다.

트리거 프로세스가 작동하려면 사용할 에코 샘플 프로그램이 SYSTEM.SAMPLE.ECHO 큐에 도착하는 메시지에 의해 트리거되는지 확인해야 합니다. 이렇게 하려면 사용할 에코 샘플 프로그램의 이름을 프로세스 정의 SYSTEM.SAMPLE.ECHOPROCESS의 *AppId* 필드에 지정하십시오. (이 경우, [IBM i](#) 관리에 설명된 CHGMQMPRC 명령을 사용할 수 있습니다.) 샘플 큐의 트리거 유형이 FIRST이므로, 요청 샘플을 실행하기 전에 이미 메시지가 큐에 있는 경우 사용자가 송신한 메시지로 인해 에코 샘플이 트리거되지 않습니다.

정의를 올바르게 설정한 경우, 먼저 하나의 작업에서 AMQ3SRV4를 시작한 다음 다른 작업에서 AMQ3REQ4를 시작하십시오. AMQ3SRV4 대신 AMQ3TRG4를 사용할 수 있지만, 작업 제출 지연 가능성 때문에 과정을 따르기가 쉽지 않을 수 있습니다.

요청 샘플 프로그램을 사용하여 메시지를 SYSTEM.SAMPLE.ECHO 큐에 보냅니다. Echo 샘플 프로그램은 요청 메시지에 데이터를 포함하는 응답 메시지를 요청 메시지에 지정된 응답 큐로 보냅니다.

## 에코 샘플 프로그램의 설계

프로그램이 트리거되는 경우, 이는 MQCONN 호출을 사용하여 기본 큐 관리자에 명시적으로 연결됩니다. IBM i에서는 이 기능이 필요하지 않지만, 다른 플랫폼에서 소스 코드를 변경하지 않고도 동일한 프로그램을 사용할 수 있습니다.

그런 다음 프로그램은 시작 시 전달된 트리거 메시지 구조에 지정된 큐를 엽니다. (명확성을 위해 이를 요청 큐라고 합니다.) 프로그램은 MQOPEN 호출을 사용하여 공유 입력에 대한 이 큐를 엽니다.

프로그램은 MQGET 호출을 사용하여 이 큐에서 메시지를 제거합니다. 이 호출은 GMATM 및 GMWT 옵션을 사용하여 대기 간격은 5초입니다. 프로그램은 요청 메시지인지 확인하기 위해 각 메시지의 디스크립터를 테스트합니다. 요청 메시지가 아닌 경우 프로그램은 메시지를 제거하고 경고 메시지를 표시합니다.

요청 큐에서 제거된 각 요청 메시지를 대상으로, 프로그램은 MQPUT 호출을 사용하여 응답 메시지를 응답 대상 큐에 넣습니다. 이 메시지는 요청 메시지의 콘텐츠를 포함합니다.

요청 큐에 남아 있는 메시지가 없는 경우 프로그램은 해당 큐를 닫고 큐 관리자의 연결을 끊습니다.

또한 이 프로그램은 이 상황에 대한 샘플이 제공되지 않더라도 IBM i 이외의 플랫폼에서 큐로 송신된 메시지에 응답할 수 있습니다. 에코(ECHO) 프로그램이 작동하려면 다음을 수행합니다.

- 프로그램을 작성하고, *Format, Encoding* 및 *CCSID* 필드를 올바르게 지정하여 텍스트 요청 메시지를 송신합니다.

ECHO 프로그램은 필요한 경우 큐 관리자에게 메시지 데이터 변환을 수행하도록 요청합니다.

- 작성한 프로그램이 응답을 위해 비슷한 변환을 제공하지 않는 경우, IBM MQ for IBM i 송신 채널에 CONVERT(\*YES)를 지정하십시오.

### IBM i의 조회 샘플 프로그램

조회 샘플 프로그램, AMQ3INQ4는 MQINQ 호출을 사용하여 큐의 몇 가지 속성을 조회합니다.

이 프로그램은 트리거된 프로그램으로 실행되므로 유일한 입력은 MQTMC(트리거 메시지) 구조입니다. 이 구조에는 조회될 속성이 있는 대상 큐의 이름이 포함되어 있습니다.

트리거 프로세스가 작동하려면 조회 샘플 프로그램이 SYSTEM.SAMPLE.INQ 큐에 도착하는 메시지에 의해 트리거되는지 확인해야 합니다. 이렇게 하려면 SYSTEM.SAMPLE.INQPROCESS 프로세스 정의의 *AppLId* 필드에 조회 샘플 프로그램의 이름을 지정하십시오. (이를 위해 MQ 프로세스 변경(CHGMQMPRC)에 설명된 CHGMQMPRC 명령을 사용할 수 있습니다.) 샘플 큐의 트리거 유형은 FIRST이므로, Request 샘플을 실행하기 전에 큐에 이미 메시지가 있으면 보내는 메시지에 의해 Inquire 샘플이 트리거되지 않습니다.

정의를 올바르게 설정한 경우, 먼저 하나의 작업에서 AMQ3SRV4를 시작한 다음 다른 작업에서 AMQ3REQ4를 시작하십시오. AMQ3SRV4 대신 AMQ3TRG4를 사용할 수 있지만, 작업 제출 지연 가능성 때문에 과정을 따르기가 쉽지 않을 수 있습니다.

각각 큐 이름만 포함한 요청 메시지를 SYSTEM.SAMPLE.INQ 큐로 송신하려면 요청 샘플 프로그램을 사용하십시오. 각 요청 메시지를 대상으로, 조회 샘플 프로그램은 요청 메시지에 지정된 큐 정보를 포함한 응답 메시지를 송신합니다. 응답은 요청 메시지에 지정된 응답 대상 큐로 송신됩니다.

### 조회 샘플 프로그램의 디자인

프로그램이 트리거되는 경우, 이는 MQCONN 호출을 사용하여 기본 큐 관리자에 명시적으로 연결됩니다. IBM i에서는 필요하지 않지만, 이러한 설계 덕분에 다른 플랫폼에서 소스 코드를 변경하지 않고도 동일한 프로그램을 사용할 수 있습니다.

그런 다음 프로그램은 시작 시 전달된 트리거 메시지 구조에 지정된 큐를 엽니다. (명확성을 위해 이를 요청 큐라고 합니다.) 프로그램은 MQOPEN 호출을 사용하여 공유 입력에 대한 이 큐를 엽니다.

프로그램은 MQGET 호출을 사용하여 이 큐에서 메시지를 제거합니다. 이 호출은 GMATM 및 GMWT 옵션을 사용하며 대기 간격은 5초입니다. 프로그램은 요청 메시지인지를 확인하기 위해 각 메시지의 디스크립터를 테스트합니다. 아닌 경우, 프로그램은 메시지를 제거하고, 경고 메시지를 표시합니다.

요청 큐에서 제거된 각 요청 메시지를 대상으로, 프로그램은 데이터에 있는 큐(대상 큐라고 하겠음)의 이름을 읽어오고 OOINQ 옵션과 MQOPEN 호출을 사용하여 해당 큐를 엽니다. 그럼 다음 MQINQ 호출을 사용하여 대상 큐의 **InhibitGet**, **CurrentQDepth** 및 **OpenInputCount** 속성 값을 조회합니다.

MQINQ 호출이 성공하면 프로그램은 MQPUT 호출을 사용하여 응답 메시지를 응답 대상 큐에 넣습니다. 이 메시지는 세 가지 속성 값을 포함합니다.

MQOPEN 또는 MQINQ 호출이 실패하면 프로그램은 MQPUT 호출을 사용하여 보고 메시지를 응답 대상 큐에 넣습니다. 이 보고 메시지의 메시지 디스크립터의 *MDFB* 필드에 실패한 항목에 따라 MQOPEN 또는 MQINQ 호출에서 리턴된 이유 코드가 있습니다.

MQINQ 호출 후 프로그램은 MQCLOSE 호출을 사용하여 대상 큐를 닫습니다.

요청 큐에 남아 있는 메시지가 없는 경우 프로그램은 해당 큐를 닫고 큐 관리자의 연결을 끊습니다.

### IBM i의 설정 샘플 프로그램

설정 샘플 프로그램, AMQ3SET4는 MQSET 호출을 사용하여 큐에서 Put 조작이 큐의 **InhibitPut** 속성을 변경하지 못하도록 금지합니다.

이 프로그램은 트리거된 프로그램으로 실행되므로 유일한 입력은 조회할 속성을 포함한 대상 큐의 이름이 있는 MQTMC(트리거 메시지) 구조입니다.



트리거 프로세스가 작동하려면 설정 샘플 프로그램이 SYSTEM.SAMPLE.SET 큐에 도착하는 메시지에 의해 트리거되는지 확인해야 합니다. 이렇게 하려면 SYSTEM.SAMPLE.SETPROCESS 프로세스 정의의 *AppId* 필드에 설정 샘플 프로그램의 이름을 지정하십시오. (이 경우, IBM i 관리에 설명된 CHGMQMPRC 명령을 사용할 수 있습니다.) 샘플 큐의 트리거 유형이 FIRST이므로, 요청 샘플을 실행하기 전에 이미 메시지가 큐에 있는 경우 사용자가 송신한 메시지로 인해 설정 샘플이 트리거되지 않습니다.

정의를 올바르게 설정한 경우, 먼저 하나의 작업에서 AMQ3SRV4를 시작한 다음 다른 작업에서 AMQ3REQ4를 시작하십시오. AMQ3SRV4 대신 AMQ3TRG4를 사용할 수 있지만, 작업 제출 지연 가능성 때문에 과정을 따르기가 쉽지 않을 수 있습니다.

각각 큐 이름만 포함한 요청 메시지를 SYSTEM.SAMPLE.SET 큐로 송신하려면 요청 샘플 프로그램을 사용하십시오. 각 요청 메시지를 대상으로, 설정 샘플 프로그램은 Put 조작이 지정된 큐에서 금지되었는지 확인 내용을 포함한 응답 메시지를 송신합니다. 응답은 요청 메시지에 지정된 응답 대상 큐로 송신됩니다.

## 설정 샘플 프로그램의 디자인

프로그램이 트리거되는 경우, 이는 MQCONN 호출을 사용하여 기본 큐 관리자에 명시적으로 연결됩니다. IBM i에서는 필요하지 않지만, 다른 플랫폼에서 소스 코드를 변경하지 않고도 동일한 프로그램을 사용할 수 있습니다.

그런 다음 프로그램은 시작 시 전달된 트리거 메시지 구조에 지정된 큐를 엽니다. (명확성을 위해 이를 요청 큐라고 합니다.) 프로그램은 MQOPEN 호출을 사용하여 공유 입력에 대한 이 큐를 엽니다.

프로그램은 MQGET 호출을 사용하여 이 큐에서 메시지를 제거합니다. 이 호출은 GMATM 및 GMWT 옵션을 사용하여 대기 간격은 5초입니다. 프로그램은 요청 메시지인지 확인하기 위해 각 메시지의 디스크립터를 테스트합니다. 요청 메시지가 아닌 경우 프로그램은 메시지를 제거하고 경고 메시지를 표시합니다.

요청 큐에서 제거된 각 요청 메시지를 대상으로, 프로그램은 데이터에 있는 큐(대상 큐라고 하겠음)의 이름을 읽어오고 OOSSET 옵션과 MQOPEN 호출을 사용하여 해당 큐를 엽니다. 그런 다음 MQSET 호출을 사용하여 대상 큐의 **InhibitPut** 속성 값을 QAPUTI로 설정합니다.

MQSET 호출이 성공하면 프로그램은 MQPUT 호출을 사용하여 응답 메시지를 응답 대상 큐에 넣습니다. 이 메시지에는 PUT inhibited 문자열이 포함됩니다.

MQOPEN 또는 MQSET 호출이 실패하면 프로그램은 MQPUT 호출을 사용하여 보고 메시지를 응답 대상 큐에 넣습니다. 이 보고 메시지의 메시지 디스크립터의 *MDFB* 필드에 실패한 항목에 따라 MQOPEN 또는 MQSET 호출에서 리턴된 이유 코드가 있습니다.

MQSET 호출 후 프로그램은 MQCLOSE 호출을 사용하여 대상 큐를 닫습니다.

요청 큐에 남아 있는 메시지가 없는 경우 프로그램은 해당 큐를 닫고 큐 관리자의 연결을 끊습니다.

## IBM i의 트리거 샘플 프로그램

IBM MQ for IBM i는 ILE/RPG로 작성된 두 가지 트리거 샘플 프로그램을 제공합니다.

프로그램은 다음과 같습니다.

### AMQ3TRG4

IBM i 환경을 위한 트리거 모니터입니다. 애플리케이션이 시작되도록 IBM i 작업을 제출하지만, 이는 각 트리거 메시지와 연관된 추가 처리 비용이 있음을 의미합니다.

### AMQ3SRV4

이는 IBM i 환경을 위한 트리거 서버입니다. 각 트리거 메시지의 경우 이 서버는 지정된 애플리케이션을 시작하기 위해 고유 작업에서 시작 명령을 실행합니다. 트리거 서버는 CICS 트랜잭션을 호출할 수 있습니다.

이 샘플의 C 언어 버전은 라이브러리 QMQM에서 AMQSTRG4 및 AMQSERV4라는 실행 가능 프로그램으로 제공됩니다.

### IBM i의 AMQ3TRG4 샘플 트리거 모니터

AMQ3TRG4는 트리거 모니터입니다. 한 개 매개변수를 사용하는데, 바로 제공할 이니시에이션 큐의 이름입니다. AMQSAMP4는 샘플 프로그램을 시도하는 경우 사용할 수 있는 샘플 이니시에이션 큐 SYSTEM.SAMPLE.TRIGGER를 정의합니다.

AMQ3TRG4는 이니시에이션 큐에서 가져온 유효한 트리거 메시지마다 IBM i 작업을 제출합니다.

## 트리거 모니터의 설계

트리거 모니터는 이니시에이션 큐를 열고, 큐에서 메시지를 가지고오, 무제한 대기 간격을 지정합니다.

트리거 모니터는 IBM i 작업을 제출하여 트리거 메시지에 지정된 애플리케이션을 시작하고, MQTMC(트리거 메시지의 문자 버전) 구조를 전달합니다. 트리거 메시지의 환경 데이터가 작업 제출 매개변수로 사용됩니다.

마지막으로 프로그램은 이니시에이션 큐를 닫습니다.

### AMQ3SRV4 샘플 트리거 서버

AMQ3SRV4는 트리거 서버입니다. 한 개 매개변수를 사용하는데, 바로 제공할 이니시에이션 큐의 이름입니다. AMQSAMP4는 샘플 프로그램을 시도하는 경우 사용할 수 있는 샘플 이니시에이션 큐 SYSTEM.SAMPLE.TRIGGER를 정의합니다.

각 트리거 메시지에 대해, AMQ3SRV4는 고유 작업에서 시작 명령을 실행하여 지정된 애플리케이션을 시작합니다.

예 트리거 큐를 사용하여 발행할 명령은 다음과 같습니다.

```
CALL PGM(QMQM/AMQ3SRV4) PARM('Queue Name')
```

여기서 Queue Name의 길이는 48자여야 하며 큐 이름에 필요한 수만큼 공백을 채우면 됩니다. 따라서 SYSTEM.SAMPLE.TRIGGERL을 대상 큐로 사용하는 경우, 28개 공백 문자가 필요합니다.

## 트리거 서버의 디자인

트리거 서버의 설계는 다음 사항을 제외하고 트리거 모니터의 설계와 비슷합니다.

- CICS 및 IBM i 애플리케이션 허용
- 트리거 메시지의 환경 데이터를 사용하지 않음
- IBM i 작업을 제출하지 않고 자체 작업에서 IBM i 애플리케이션을 호출합니다 (또는 STRCICSUSR을 사용하여 CICS 애플리케이션 시작).
- 공유 입력을 위해 이니시에이션 큐를 열기 때문에 많은 트리거 서버를 동시에 실행할 수 있음

**참고:** AMQ3SRV4에 의해 시작된 프로그램은 MQDISC 호출이 트리거 서버를 중지하므로 해당 호출을 사용하지 않아야 합니다. AMQ3SRV4에 의해 시작된 프로그램이 MQCONN 호출을 사용하면 RC2002 이유 코드가 표시됩니다.

### IBM i의 트리거링 샘플 프로그램 종료

sysrequest 옵션 2(ENDRQS)에 의해 또는 트리거 큐에서 가져오기를 금지하여 트리거 모니터 프로그램을 종료할 수 있습니다.

샘플 트리거 큐를 사용하는 경우, 명령은 다음과 같습니다.

```
CHGMQM QNAME('SYSTEM.SAMPLE.TRIGGER') GETENBL(*NO)
```

**참고:** 이 큐에서 트리거를 다시 시작하려면 다음 명령을 입력해야 합니다.

```
CHGMQM QNAME('SYSTEM.SAMPLE.TRIGGER') GETENBL(*YES)
```

## IBM i에서 리모트 큐를 사용하여 샘플 실행

연결된 메시지 큐 관리자에서 샘플을 실행하여 리모트 큐잉을 보여줄 수 있습니다.

프로그램 AMQSAMP4는 OTHER라는 리모트 큐 관리자를 사용하는 리모트 큐의 로컬 정의(SYSTEM.SAMPLE.REMOTE)를 제공합니다. 이 샘플 정의를 사용하려면 OTHER를 사용할 두 번째 메시지 큐 관리자의 이름으로 변경하십시오. 또한 두 개의 메시지 큐 관리자 사이에 메시지 채널을 설정해야 합니다. 이 작업 방법에 대한 자세한 정보는 [메시징 채널에 대한 채널 엑시트 프로그램의 내용을 참조하십시오.](#)

요청 샘플 프로그램은 송신하는 메시지의 *MDRM* 필드에 고유의 로컬 큐 관리자 이름을 넣습니다. 조회 및 설정 샘플은 처리하는 요청 메시지의 *MDRQ* 및 *MDRM* 필드에 지정된 큐 및 메시지 큐 관리자로 응답 메시지를 송신합니다.

## IBM i(ILE RPG)의 리턴 코드

이 정보는 MQI 및 MQAI와 연관된 리턴 코드를 설명합니다.

다음과 연관된 리턴 코드:

- PCF(프로그래밍 가능 명령 형식) 명령은 프로그래밍 가능 명령 형식 참조에 나와 있습니다.
- C++ 호출은 C++ 사용에 나와 있습니다.

각 호출마다, 큐 관리자 또는 엑시트 루틴은 호출의 성공 또는 실패를 나타내는 완료 코드 및 이유 코드를 리턴합니다.

특별히 명시된 경우를 제외하고, 오류를 확인하는 특정 순서에 따라 애플리케이션이 달라지지 않아야 합니다. 둘 이상의 완료 코드 또는 이유 코드가 호출에서 발생할 수 있는 경우, 보고되는 특정 오류는 구현에 따라 다릅니다.

## IBM i(ILE RPG)의 완료 코드

완료 코드 매개변수(*CMPCOD*)는 호출자가 호출이 성공적으로 완료되었는지, 부분적으로 완료되었는지 또는 실패했는지 여부를 빨리 알 수 있도록 합니다.

### CCOK

(다른 플랫폼에서는 MQCC\_OK)

정상적으로 완료되었습니다.

호출이 완전히 완료되었습니다. 모든 출력 매개변수가 설정되었습니다. 이 경우 **REASON** 매개변수의 값은 항상 RCNONE입니다.

### CCWARN

(다른 플랫폼에서는 MQCC\_WARN)

경고(일부 완료).

호출이 부분적으로 완료되었습니다. *CMPCOD* 및 *REASON* 출력 매개변수 외에 일부 출력 매개변수가 설정될 수 있습니다. **REASON** 매개변수는 부분적인 완료에 대한 추가 정보를 제공합니다.

### CCFAIL

(다른 플랫폼에서는 MQCC\_FAIL)

호출에 실패했습니다.

호출 처리가 완료되지 않았으며 큐 관리자의 상태가 정상적으로 변경되지 않습니다. 예외가 특별히 명시됩니다. *CMPCOD* 및 *REASON* 출력 매개변수가 설정되었습니다. 다른 매개변수는 명시된 것을 제외하면 변경되지 않습니다.

이유는 애플리케이션 프로그램의 결함 또는 프로그램 외부의 어떤 상황에 따른 결과(예: 사용자 권한 취소) 때문일 수 있습니다. **REASON** 매개변수는 오류에 관한 추가 정보를 제공합니다.

## IBM i(ILE RPG)의 이유 코드

이유 코드 매개변수(*REASON*)가 완료 코드 매개변수(*CMPCOD*)를 규정합니다.

특별한 보고 이유가 없는 경우 RCNONE가 리턴됩니다. 성공적인 호출은 CCOK 및 RCNONE를 리턴합니다.

완료 코드가 CCWARN 또는 CCFAIL인 경우, 큐 관리자는 항상 규정 이유를 보고합니다. 각 호출 설명에 자세한 내용이 제공되어 있습니다.

사용자 엑시트 루틴이 완료 코드와 이유 코드를 설정하는 경우, 이 규칙을 준수해야 합니다. 또한, 사용자 엑시트에서 정의한 특별한 이유 값은 큐 관리자에서 정의한 값과 충돌하지 않도록 0(영)보다 작아야 됩니다. 엑시트는 이러한 항목이 적절한 큐 관리자에 의해 이미 정의된 이유를 설정할 수 있습니다.

이유 코드가 다음 위치에도 있을 수 있습니다.

- MQDLH 구조의 *DLREA* 필드
- MQMD 구조의 *MDFB* 필드

전체 이유 코드 목록은 [API 완료 및 이유 코드](#)를 참조하십시오.

이 목록에서 IBM i 이유 코드를 찾으려면 앞에서 "RC"를 제거하십시오(예: RC2002가 2002가 됨). 또한 여기 있는 완료 코드는 다른 플랫폼에서와 똑같이 표시됩니다.

표 814. IBM i 및 기타 플랫폼의 이유 코드 이름	
IBM i	다른 플랫폼
CCOK	MQCC_OK
CCWARN	MQCC_WARN
CCFAIL	MQCC_FAIL

## IBM i(ILE RPG)에 대해 MQI 옵션 유효성 검증 규칙

이 주제는 MQOPEN, MQPUT, MQPUT1, MQGET 또는 MQCLOSE 호출에서 RC2046 이유 코드를 생성하는 상황에 대한 정보를 제공합니다.

### IBM i의 MQOPEN 호출

MQOPEN 호출의 옵션:

- 다음 항목 중 하나 이상을 지정해야 합니다.

- OOBROW
- OOINPQ
- OOINPX
- OOINPS
- OOINQ
- OOOUT
- OOSET

- 다음 중 하나만 허용됩니다.

- OOINPQ
- OOINPX
- OOINPS

- 다음 중 하나만 허용됩니다.

- OOBNDQ
- OOBNDN
- OOBNDQ

**참고:** 앞에 나열된 옵션은 상호 배타적입니다. 그러나, OOBNDQ의 값이 0(영)이기 때문에 다른 두 바인드 옵션 중 하나로 지정하면 이유 코드 RC2046이 나타나지 않습니다. OOBNDQ는 프로그램 문서를 보조하기 위해 제공됩니다.

- OOSAVA가 지정되면 OOINP\* 옵션 중 하나도 지정해야 합니다.
- OOSET\* 또는 OOPAS\* 옵션 중 하나가 지정되면 OOOUT도 지정해야 합니다.

### IBM i의 MQPUT 호출

메시지 넣기 옵션:

- PMSYP 및 PMNSYP의 결합은 허용되지 않습니다.
- 다음 중 하나만 허용됩니다.
  - PMDEFC
  - PMNOC
  - PMPASA
  - PMPASI
  - PMSETA
  - PMSETI
- PMALTU는 허용되지 않습니다(MQPUT1 호출에서만 유효함).

## IBM i의 MQPUT1 호출

메시지 넣기 옵션의 경우, 다음 옵션을 제외하고, MQPUT 호출과 규칙이 동일합니다.

- PMALTU는 허용됩니다.
- PMLOGO는 허용되지 않습니다.

## IBM i의 MQGET 호출

메시지 가져오기 옵션:

- 다음 옵션 중 하나만 허용됩니다.
  - GMNSYP
  - GMSYP
  - GMPSYP
- 다음 옵션 중 하나만 허용됩니다.
  - GMBRWF
  - GMBRWC
  - GMBRWN
  - GMMUC
- GMSYP는 다음 옵션과 허용되지 않습니다.
  - GMBRWF
  - GMBRWC
  - GMBRWN
  - GMLK
  - GMUNLK
- GMPSYP는 다음 옵션과 허용되지 않습니다.
  - GMBRWF
  - GMBRWC
  - GMBRWN
  - GMCMPM
  - GMUNLK
- GMLK가 지정되면 다음 옵션 중 하나도 지정해야 합니다.
  - GMBRWF
  - GMBRWC

- GMBRWN
- GMUNLK가 지정되면 다음 옵션만 허용됩니다.
  - GMNSYP
  - GMNWT

## IBM i의 MQCLOSE 호출

- MQCLOSE 호출의 옵션에 대한 것입니다. CODEL 및 COPURG의 결합은 허용되지 않습니다.
- 다음 중 하나만 허용됩니다.
  - COKPSB
  - CORMSB

## IBM i의 MQSUB 호출

MQSUB 호출의 옵션:

- 다음 옵션 중 하나 이상을 지정해야 합니다.
- 다음 옵션 중 하나 이상을 지정해야 합니다.
  - SOALT
  - SORES
  - SOCRT
- 다음 중 하나만 허용됩니다.
  - SODUR
  - SONDUR

**참고:** 앞에 나열된 옵션은 상호 배타적입니다. 그러나, SONDUR의 값이 0(영)이기 때문에 SODUR로 지정해도 이유 코드 RC2046이 나타나지 않습니다. SONDUR은 프로그램 문서를 보조하기 위해 제공됩니다.

- SOGRP 및 SOMAN의 결합은 허용되지 않습니다.
- SOGRP를 사용하려면 SOSCID를 지정해야 합니다.
- SOAUID SOFUID 옵션 중 하나만 허용됩니다.
- SONEWP 및 SOPUBR의 결합은 허용되지 않습니다.
- SONEWP는 SOCRT와 결합 시에만 허용됩니다.
- 다음 중 하나만 허용됩니다.
  - SOWCHR
  - SOWTOP

## IBM i의 시스템 인코딩

메시지 디스크립터의 *MDENC* 필드 구조에 대해 알아보려면 이 정보를 사용하십시오.

메시지 디스크립터에 대한 자세한 정보는 [1063 페이지의 『IBM i의 MQMD\(메시지 디스크립터\)』](#)의 내용을 참조하십시오.

*MDENC* 필드는 4개의 별도 하위 필드로 나뉘는 32비트 정수입니다. 이 하위 필드는 다음을 식별합니다.

- 2진 정수에 사용된 인코딩
- 팩형 10진수 정수에 사용된 인코딩
- 부동 소수점 숫자에 사용된 인코딩
- 예약 비트

각 하위 필드는 하위 필드에 해당되는 위치에 1비트가 있고 다른 곳에는 0비트가 있는 비트 마스크로 식별됩니다. 비트는 숫자로 지정되며, 비트 0은 가장 중요한 비트이고 비트 31은 가장 중요하지 않은 비트입니다. 다음 마스크가 정의됩니다.

#### **KOIMSK**

2진 정수 인코딩을 위한 마스크.

이 하위 필드는 *MDENC* 필드에서 28 ~ 31의 비트 위치를 차지합니다.

#### **ENDMSK**

팩형 10진수 정수 인코딩을 위한 마스크.

이 하위 필드는 *MDENC* 필드에서 24 ~ 27의 비트 위치를 차지합니다.

#### **ENFMSK**

부동 소수점 인코딩을 위한 마스크.

이 하위 필드는 *MDENC* 필드에서 20 ~ 23의 비트 위치를 차지합니다.

#### **ENRMSK**

예약 비트를 위한 마스크.

이 하위 필드는 *MDENC* 필드에서 0 ~ 19의 비트 위치를 차지합니다.

### **IBM i IBM i의 2진 정수 인코딩**

2진 정수 인코딩에 대한 올바른 값입니다.

2진 정수 인코딩에 대해 올바른 값은 다음과 같습니다.

#### **ENIUND**

정의되지 않은 정수 인코딩.

2진 정수가 정의되지 않은 인코딩을 사용하여 표시됩니다.

#### **ENINOR**

일반 정수 인코딩.

2진 정수가 전통적인 방법으로 표시됩니다.

- 숫자의 LSB(Least Significant Byte)에는 숫자의 임의 바이트의 최상위 주소가 있으며, MSB(Most Significant Byte)에는 최하위 주소가 있습니다.
- 각 바이트의 LSB(Least Significant Bit)는 다음 상위 주소를 포함한 바이트 옆에 있으며, 각 바이트의 MSB(Most Significant Bit)는 다음 하위 주소를 포함한 바이트 옆에 있습니다.

#### **ENIREV**

역방향 정수 인코딩.

2진 정수가 ENINOR과 동일한 방식으로 표시되지만, 바이트가 역순으로 배열됩니다. 각 바이트 내 비트는 ENINOR과 동일한 방식으로 배열됩니다.

### **IBM i IBM i의 팩형 10진수 정수 인코딩**

팩형 10진수 정수 인코딩에 대한 올바른 값입니다.

팩형 10진수 정수 인코딩에 대해 올바른 값은 다음과 같습니다.

#### **ENDUND**

정의되지 않은 팩형 10진수 인코딩.

팩형 10진수 정수가 정의되지 않은 인코딩을 사용하여 표시됩니다.

#### **ENDNOR**

일반 팩형 10진수 인코딩.

팩형 10진수 정수가 전통적인 방법으로 표시됩니다.



- 인쇄 가능한 숫자 형식의 각 10진수는 X'0' - X'9' 범위의 단일 16진수 숫자를 통해 팩형 10진수로 표시됩니다. 각 16진수가 4비트를 차지하므로 팩형 10진수의 각 바이트는 인쇄 가능한 숫자 형식에서 2개의 10진수를 나타냅니다.
- 팩형 10진수 숫자의 LSB(Least Significant Byte)는 최하위 10진수 숫자를 포함한 바이트입니다. 해당 바이트 내에서 최상위 4비트는 최하위 10진수 숫자를 포함하고 최하위 4비트는 부호를 포함합니다. 부호는 X'C'(양수), X'D'(음수) 또는 X'F'(부호 없음)입니다.
- 숫자의 LSB(Least Significant Byte)에는 숫자의 임의 바이트의 최상위 주소가 있으며, MSB(Most Significant Byte)에는 최하위 주소가 있습니다.
- 각 바이트의 LSB(Least Significant Bit)는 다음 상위 주소를 포함한 바이트 옆에 있으며, 각 바이트의 MSB(Most Significant Bit)는 다음 하위 주소를 포함한 바이트 옆에 있습니다.

#### ENDREV

역방향 팩형 10진수 인코딩.

팩형 10진수 정수는 ENDNOR과 동일한 방식으로 표시되지만, 바이트가 역순으로 배열됩니다. 각 바이트 내 비트는 ENDNOR과 동일한 방식으로 배열됩니다.

#### IBM i IBM i의 부동 소수점 인코딩

부동 소수점 인코딩에 대한 올바른 값입니다.

부동 소수점 인코딩에 대해 올바른 값은 다음과 같습니다.

#### ENFUND

정의되지 않은 부동 소수점 인코딩.

부동 소수점 숫자가 정의되지 않은 인코딩을 사용하여 표시됩니다.

#### ENFNOR

정상 IEEE(The Institute of Electrical and Electronics Engineers) float 인코딩.

부동 소수점 숫자가 표준 IEEE 부동 소수점 형식을 사용하여 표시되고, 바이트가 다음과 같이 배열됩니다.

- 가수(mantissa)의 LSB(Least Significant Byte)에는 숫자의 임의 바이트의 최상위 주소가 있으며, 지수를 포함한 바이트에는 최하위 주소가 있습니다.
- 각 바이트의 LSB(Least Significant Bit)는 다음 상위 주소를 포함한 바이트 옆에 있으며, 각 바이트의 MSB(Most Significant Bit)는 다음 하위 주소를 포함한 바이트 옆에 있습니다.

IEEE float 인코딩의 세부사항을 IEEE Standard 754에서 찾을 수 있습니다.

#### ENFREV

역방향 IEEE float 인코딩.

부동 소수점 숫자가 ENFNOR과 동일한 방식으로 표시되지만, 바이트가 역순으로 배열됩니다. 각 바이트 내 비트는 ENFNOR과 동일한 방식으로 배열됩니다.

#### ENF390

System/390 아키텍처 float 인코딩.

부동 소수점 숫자가 표준 System/390 부동 소수점 형식을 사용하여 표시됩니다. System/370에도 사용됩니다.

#### IBM i IBM i의 구성 인코딩

MQMD에서 MDENC 필드 값을 구성하려면 필수 인코딩을 설명하는 관련 상수를 추가해야 합니다.

ENI\* 인코딩 중 하나만 END\* 인코딩 중 하나 그리고 ENF\* 인코딩 중 하나와 결합하도록 하십시오.

#### IBM i IBM i의 분석 인코딩

MDENC 필드에는 하위 필드가 있으며 이 때문에 정수, 팩형 10진수 또는 float 인코딩을 조사해야 하는 애플리케이션은 이 주제에 설명된 기술을 사용해야 합니다.



## 산술 사용

다음 단계는 정수 산술을 사용하여 수행해야 합니다.

1. 필요한 인코딩 유형에 따라 다음 값 중 하나를 선택하십시오.

- 2진 정수 인코딩의 경우 1
- 팩형 10진수 정수 인코딩의 경우 16
- 부동 소수점 인코딩의 경우 256

값 A를 호출하십시오.

2. MDENC 필드의 값을 A로 나누십시오. 결과 B를 호출합니다.

3. B를 16으로 나누십시오. 결과 C를 호출합니다.

4. C에 16을 곱하고 B에서 빼십시오. 결과 D를 호출합니다.

5. D에 A를 곱합니다. 결과 E를 호출합니다.

6. E는 필수 인코딩으로, 해당 인코딩 유형에 사용 가능한 각 값과 같은지 테스트할 수 있습니다.

## IBM i IBM i의 시스템 아키텍처 인코딩 요약

시스템 아키텍처에 대한 인코딩을 요약한 표입니다.

시스템 아키텍처 인코딩이 [1365 페이지의 표 815](#)에 나와 있습니다.

표 815. 시스템 아키텍처 인코딩 요약			
시스템 아키텍처	2진 정수 인코딩	팩형 10진수 정수 인코딩	부동 소수점 인코딩
IBM i	정상	정상	IEEE 정상
Intel x86	역방향	역방향	IEEE 역방향
PowerPC	정상	정상	IEEE 정상
System/390	정상	정상	System/390

## IBM i IBM i의 보고 옵션 및 메시지 플래그

이 주제는 MQGET, MQPUT 및 MQPUT1 호출에 지정된 메시지 디스크립터 MQMD에 속한 MDREP 및 MDMFL 필드를 설명합니다.

메시지 디스크립터에 대한 자세한 정보는 [1063 페이지의 『IBM i의 MQMD\(메시지 디스크립터\)』](#)의 내용을 참조하십시오. 이 정보는 다음을 설명합니다.

- 보고 필드의 구조 및 큐 관리자의 처리 방법
- 애플리케이션의 보고 필드 분석 방법
- 메시지 플래그 필드의 구조

### 보고 필드의 구조

MDREP 필드는 3개의 별도 하위 필드로 나뉘는 32비트 정수입니다.

이러한 하위 필드는 다음을 식별합니다.

- 로컬 큐 관리자가 인식하지 못하면 거부되는 보고 옵션
- 로컬 큐 관리자가 인식하지 못해도 항상 허용되는 보고 옵션
- 다른 특정 조건을 충족하는 경우에만 허용되는 보고 옵션

각 하위 필드는 하위 필드에 해당되는 위치에 1비트가 있고 다른 곳에는 0비트가 있는 비트 마스크로 식별됩니다. 하위 필드에서는 비트가 반드시 인접하지 않다는 점에 유의하십시오. 비트는 숫자로 지정되며, 비트 0은 가장 중요한 비트이고 비트 31은 가장 중요하지 않은 비트입니다. 다음 마스크가 하위 필드를 식별하기 위해 정의됩니다.

## RORUM

거부된 지원되지 않는 보고 옵션에 대한 마스크.

이 마스크는 로컬 큐 관리자에서 지원되지 않는 보고 옵션 때문에 완료 코드 CCFAIL 및 이유 코드 RC2061이 표시되면서 MQPUT 또는 MQPUT1 호출이 실패할 경우 *MDREP* 필드에서 비트 위치를 식별합니다.

이 하위 필드는 비트 위치 3 및 11 ~ 13을 차지합니다.

## ROAUM

허용된 지원되지 않는 보고 옵션에 대한 마스크.

이 마스크는 로컬 큐 관리자에서 지원되지 않는 보고 옵션이 그럼에도 MQPUT 또는 MQPUT1 호출에서 허용되는 경우 *MDREP* 필드에서 비트 위치를 식별합니다. 이 경우, 완료 코드 CCWARN 및 이유 코드 RC2104가 리턴됩니다.

이 하위 필드는 0 ~ 2, 4 ~ 10 및 24 ~ 31의 비트 위치를 차지합니다.

다음 보고 옵션은 이 하위 필드에 포함됩니다.

- ROCMTC
- RODLQ
- RODISC
- ROEXC
- ROEXCD
- ROEXCF
- ROEXP
- ROEXPD
- ROEXPF
- RONAN
- RONMI
- RONONE
- ROPAN
- ROPCI
- ROPMI

## ROAUXM

특정 환경에서만 허용된 지원되지 않는 보고 옵션에 대한 마스크.

이 마스크는 로컬 큐 관리자에서 지원되지 않는 보고 옵션이 그럼에도 다음 조건을 모두 충족하면 MQPUT 또는 MQPUT1 호출에서 허용되는 경우 *MDREP* 필드에서 비트 위치를 식별합니다.

- 메시지가 리모트 큐 관리자로 이동합니다.
- 애플리케이션이 로컬 전송 큐에 직접 메시지를 넣지 않습니다(즉, MQOPEN 또는 MQPUT1 호출에 지정된 오브젝트 디스크립터의 *ODON* 및 *ODMN* 필드로 식별된 큐는 로컬 전송 큐가 아님).

이 조건을 충족하면 완료 코드 CCWARN 및 이유 코드 RC2104가 리턴되고 그렇지 않으면 CCFAIL과 이유 코드 RC2061이 리턴됩니다.

이 하위 필드는 비트 위치 14 ~ 23을 차지합니다.

다음 보고 옵션은 이 하위 필드에 포함됩니다.

- ROCOA
- ROCOAD
- ROCOAF
- ROCOD
- ROCODD
- ROCODF

큐 관리자가 인식하지 못하는 *MDREP* 필드에 지정된 옵션이 있으면 큐 관리자가 비트 단위의 AND 연산자를 사용하여 각 하위 필드를 번갈아 검사하고 해당 하위 필드의 마스크와 *MDREP* 필드를 결합합니다. 해당 조작의 결과가 0이 아닌 경우, 이전에 설명된 완료 코드 및 이유 코드가 리턴됩니다.

CCWARN이 리턴되면, 다른 경고 조건이 있을 때 어떤 리턴 코드가 리턴되는지 정의되지 않았습니다.

로컬 큐 관리자에 인식되지 않는 보고 옵션을 지정하고 허용하는 기능은 리모트 큐 관리자에 인식 및 처리될 보고 옵션으로 메시지를 송신해야 하는 경우 유용합니다.

## IBM i IBM i에서 보고 필드 분석

*MDREP* 필드는 하위 필드를 포함합니다. 이 때문에 일부 애플리케이션은 메시지의 송신자가 특정 보고서를 요청했는지 확인해야 합니다. 해당 애플리케이션이 이 토픽에 설명된 기술을 사용해야 합니다.

### 산술 사용

다음 단계는 정수 산술을 사용하여 수행해야 합니다.

1. 확인할 보고서의 유형에 따라 다음 값 중 하나를 선택하십시오.

- COA 보고서의 경우 ROCOA
- COD 보고서의 경우 ROCOD
- 예외 보고서의 경우 ROEXC
- 만기 보고서의 경우 ROEXP

값 A를 호출하십시오.

2. *MDREP* 필드를 A로 나눕니다. 결과 B를 호출합니다.

3. B를 8로 나눕니다. 결과 C를 호출합니다.

4. C에 8을 곱하고 B에서 뺍니다. 결과 D를 호출합니다.

5. D에 A를 곱합니다. 결과 E를 호출합니다.

6. 보고서 유형에 사용 가능한 각 값과 같은지 E를 테스트합니다.

예를 들어, A가 ROEXC이면 메시지 송신자가 지정한 내용을 판별하기 위해 다음 각 항목과 같은지 E를 테스트합니다.

- RONONE
- ROEXC
- ROEXCD
- ROEXCF

테스트는 애플리케이션 논리에 가장 편리한 임의의 순서로 수행할 수 있습니다.

다음 의사 코드는 예외 보고 메시지에 대한 이 기술을 설명합니다.

```
A = ROEXC
B = Report/A
C = B/8
D = B - C*8
E = D*A
```

비슷한 메소드를 사용하여 ROPMI 또는 ROPCI 옵션에 대해 테스트할 수 있습니다. 이 두 상수 중 적절한 항목을 값 A로 선택한 다음 이전에 설명한 대로 진행하되, 이전 단계의 값 8을 값 2로 대체합니다.

## IBM i IBM i에서 메시지 플래그 필드의 구조

*MDMFL* 필드는 3개의 별도 하위 필드로 나뉘는 32비트 정수입니다.

이러한 하위 필드는 다음을 식별합니다.

- 로컬 큐 관리자가 인식하지 못하면 거부되는 메시지 플래그

- 로컬 큐 관리자가 인식하지 못해도 항상 허용되는 메시지 플래그
- 다른 특정 조건을 충족하는 경우에만 허용되는 메시지 플래그

**참고:** *MDMFL*의 모든 하위 필드는 큐 관리자에 사용하도록 예약됩니다.

각 하위 필드는 하위 필드에 해당되는 위치에 1비트가 있고 다른 곳에는 0비트가 있는 비트 마스크로 식별됩니다. 비트는 숫자로 지정되며, 비트 0은 가장 중요한 비트이고 비트 31은 가장 중요하지 않은 비트입니다. 다음 마스크가 하위 필드를 식별하기 위해 정의됩니다.

#### **MFRUM**

거부된 지원되지 않는 메시지 플래그에 대한 마스크.

이 마스크는 로컬 큐 관리자가 지원되지 않는 메시지 플래그 때문에 완료 코드 *CCFAIL* 및 이유 코드 *RC2249*가 표시되면서 *MQPUT* 또는 *MQPUT1* 호출이 실패할 경우 *MDMFL* 필드에서 비트 위치를 식별합니다.

이 하위 필드는 비트 위치 20 ~ 31을 차지합니다.

다음 메시지 플래그가 이 하위 필드에 포함됩니다.

- *MFLMIG*
- *MFLSEG*
- *MFMIIG*
- *MFSEG*
- *MFSEGA*
- *MFSEGI*

#### **MFAUM**

허용된 지원되지 않는 메시지 플래그에 대한 마스크.

이 마스크는 로컬 큐 관리자가 지원되지 않는 메시지 플래그가 그럼에도 *MQPUT* 또는 *MQPUT1* 호출에서 허용되는 경우 *MDMFL* 필드에서 비트 위치를 식별합니다. 완료 코드는 *CCOK*입니다.

이 하위 필드는 비트 위치 0 ~ 11을 차지합니다.

#### **MFAUXM**

특정 환경에서만 허용된 지원되지 않는 메시지 플래그에 대한 마스크.

이 마스크는 로컬 큐 관리자가 지원되지 않는 메시지 플래그가 그럼에도 다음 조건을 모두 충족하면 *MQPUT* 또는 *MQPUT1* 호출에서 허용되는 경우 *MDMFL* 필드에서 비트 위치를 식별합니다.

- 메시지가 리모트 큐 관리자로 이동합니다.
- 애플리케이션이 로컬 전송 큐에 직접 메시지를 넣지 않습니다(즉, *MQOPEN* 또는 *MQPUT1* 호출에 지정된 오브젝트 디스크립터의 *ODON* 및 *ODMN* 필드로 식별된 큐는 로컬 전송 큐가 아님).

이 조건을 충족하면 완료 코드 *CCOK*가 리턴되고 그렇지 않으면 *CCFAIL* 및 이유 코드 *RC2249*가 리턴됩니다.

이 하위 필드는 비트 위치 12 ~ 19를 차지합니다.

큐 관리자가 인식하지 못하는 *MDMFL* 필드에 지정된 플래그가 있으면 큐 관리자가 비트 단위의 AND 연산자를 사용하여 각 하위 필드를 번갈아 검사하고 해당 하위 필드의 마스크와 *MDMFL* 필드를 결합합니다. 해당 조작의 결과가 0이 아닌 경우, 이전에 설명된 완료 코드 및 이유 코드가 리턴됩니다.

## **IBM i** IBM i의 데이터 변환

이 주제에서는 데이터 변환 엑시트에 대한 인터페이스 및 데이터 변환 요구 시 큐 관리자가 수행되는 처리를 설명합니다.

데이터 변환 엑시트가 *MQGET* 호출 처리의 일부로 호출됩니다. 애플리케이션 메시지 데이터를 수신 애플리케이션에서 필요한 표현으로 변환하는 데 사용됩니다. 애플리케이션 메시지 데이터의 변환은 선택사항이며, *GMCONV* 옵션을 *MQGET* 호출에서 지정해야 합니다.

다음은 데이터 변환의 측명에 대해 설명합니다.

- GMCONV 옵션에 응답하여 큐 관리자에서 수행된 처리. [1369 페이지의 『IBM i의 변환 처리』](#)의 내용을 참조하십시오.
- 내장 형식 처리 시 큐 관리자에서 사용된 규칙 처리. 이 규칙은 사용자 작성 엑시트에 대해서도 권장됩니다. [1370 페이지의 『IBM i의 처리 규칙』](#)의 내용을 참조하십시오.
- 보고 메시지 변환에 대한 특별한 고려사항. [1373 페이지의 『IBM i에서 보고 메시지의 변환』](#)의 내용을 참조하십시오.
- 데이터 변환 엑시트에 전달된 매개변수. [1383 페이지의 『IBM i의 MQCONVX\(데이터 변환 엑시트\)』](#)의 내용을 참조하십시오.
- 다른 표현 간에 문자 데이터를 변환하기 위해 엑시트에서 사용될 수 있는 호출. [1379 페이지의 『IBM i의 MQXCNCV\(문자 변환\)』](#)의 내용을 참조하십시오.
- 엑시트에 특정적인 데이터 구조 매개변수. [1374 페이지의 『IBM i의 MQDXP\(데이터 변환 엑시트 매개변수\)』](#)의 내용을 참조하십시오.

## IBM i IBM i의 변환 처리

이 정보는 GMCONV 옵션에 응답하여 큐 관리자에서 수행된 처리에 대해 설명합니다.

GMCONV 옵션이 MQGET 호출에 지정되어 있으면 큐 관리자가 다음 조치를 수행하며, 애플리케이션에 메시지가 리턴됩니다.

1. 다음 중 하나 이상에 해당하면 변환할 필요가 없습니다.

- 메시지 데이터가 MQGET 호출을 발행하는 애플리케이션에 필요한 문자 세트 및 인코딩에 이미 있습니다. 애플리케이션은 호출을 발행하기 전에 MQGET 호출의 **MSGDSC** 매개변수에 있는 **MDCSI** 및 **MDENC** 필드를 필요한 값으로 설정해야 합니다.
- 메시지 데이터의 길이는 0입니다.
- MQGET 호출의 **BUFFER** 매개변수의 길이는 0입니다.

이러한 경우 MQGET 호출을 발행하는 애플리케이션으로 변환하지 않고 메시지가 리턴됩니다. **MSGDSC** 매개변수의 **MDCSI** 및 **MDENC** 값은 메시지의 제어 정보에 있는 값으로 설정되며 호출은 다음 완료 코드 및 이유 코드 조합 중 하나로 완료됩니다.

완료 코드  
이유 코드

**CCOK**  
RCNONE

**CCWARN**  
RC2079

**CCWARN**  
RC2080

다음 단계는 메시지 데이터의 문자 세트 또는 인코딩이 **MSGDSC** 매개변수의 해당 값과 다른 경우에만 수행되며, 변환되는 데이터가 있습니다.

1. 메시지의 제어 정보에 있는 **MDFMT** 필드의 값이 **FMNONE**이면 완료 코드 **CCWARN** 및 이유 코드 **RC2110**이 표시되면서 메시지가 변환 없이 리턴됩니다.

다른 모든 경우에 변환 처리가 계속됩니다.

2. 메시지가 큐에서 제거되고 **BUFFER** 매개변수와 같은 크기의 임시 버퍼에 놓입니다. 찾아보기 조작의 경우, 메시지가 큐에서 제거되지 않고 임시 버퍼에 복사됩니다.

3. 메시지가 버퍼에 맞게 잘린 경우, 다음이 수행됩니다.

- **GMATM** 옵션을 지정하지 않은 경우, 메시지가 변환되지 않은 채로 리턴되고 완료 코드 **CCWARN**과 이유 코드 **RC2080**이 나타납니다.
- **GMATM** 옵션이 지정된 경우, 완료 코드가 **CCWARN**으로, 이유 코드가 **RC2079**로 설정되고 변환 처리가 계속됩니다.

4. 메시지가 잘림 없이 버퍼에 수용될 수 있거나 GMATM 옵션이 지정된 경우, 다음이 수행됩니다.
- 형식이 내장 형식이면 버퍼가 큐 관리자의 데이터 변환 서비스에 전달됩니다.
  - 형식이 내장 형식이 아니면 버퍼가 형식과 동일한 이름의 사용자 작성 엑시트에 전달됩니다. 엑시트가 없으면 완료 코드 CCWARN 및 이유 코드 RC2110이 표시되면서 메시지가 변환 없이 리턴됩니다.
- 오류가 발생하지 않으면 데이터 변환 서비스 또는 사용자 작성 엑시트에서 변환된 메시지 및 MQGET 호출을 발행하는 애플리케이션에 리턴되는 완료 코드 및 이유 코드가 출력됩니다.
5. 변환에 성공하면 큐 관리자가 변환된 메시지를 애플리케이션으로 리턴합니다. 이 경우, MQGET 호출에서 리턴된 완료 코드와 이유 코드는 일반적으로 다음 조합 중 하나입니다.

**완료 코드**  
**이유 코드**

**CCOK**  
RCNONE

**CCWARN**  
RC2079

그러나, 사용자 작성 엑시트에 의해 변환이 수행된 경우에는 변환에 성공하더라도 다른 이유 코드가 리턴될 수 있습니다.

변환이 실패하는 경우 (어떤 이유로든), 큐 관리자는 변환되지 않은 메시지를 애플리케이션에 리턴하며, **MSGDSC** 매개변수의 **MDCSI** 및 **MDENC** 필드는 메시지의 제어 정보에 있는 값으로 설정되고 완료 코드는 CCWARN입니다.

## IBM i IBM i의 처리 규칙

내장 형식을 변환할 때 큐 관리자는 이 주제에 설명된 처리 규칙을 따릅니다.

큐 관리자가 적용하지 않더라도 사용자 작성 엑시트에 이러한 규칙을 적용하는 것이 좋습니다. 큐 관리자에 의해 변환되는 내장 형식은 다음과 같습니다.

- FMADMN
- FMMDE
- FMCICS
- FMPCF
- FMCMD1
- FMRMH
- FMCMD2
- FMRFH
- FMDLH
- FMRFH2
- FMDH
- FMSTR
- FMEVNT
- FMTM
- FMIMS
- FMXQH
- FMIMVS

1. 변환 중에 메시지가 확장되어 **BUFFER** 매개변수의 크기를 초과하면 다음이 수행됩니다.

- GMATM 옵션을 지정하지 않은 경우, 메시지가 변환되지 않은 채로 리턴되고 완료 코드 CCWARN과 이유 코드 RC2120이 나타납니다.

- GMATM 옵션이 지정된 경우, 메시지가 잘리고 완료 코드가 CCWARN으로, 이유 코드가 RC2079로 설정되며 변환 처리가 계속됩니다.
2. (변환 이전 또는 이후에) 잘림이 발생하면 **BUFFER** 매개변수에 리턴되는 올바른 바이트 수가 버퍼 길이 미만일 수 있습니다.
- 예를 들어, 4바이트 정수 또는 DBCS 문자가 버퍼의 끝에 걸쳐 발생할 수 있습니다. 정보의 미완료 요소는 변환되지 않으므로 리턴된 메시지의 바이트에 올바른 정보가 포함되지 않습니다. 변환 전에 잘린 메시지가 변환하는 동안 축소되는 경우에도 이 동작이 발생할 수 있습니다.
- 리턴된 올바른 바이트 수가 버퍼 길이 미만이면 버퍼 끝에 있는 미사용 바이트가 널로 설정됩니다.
3. 배열 또는 문자열이 버퍼 끝에 걸쳐 있으면 최대한 많은 데이터가 변환됩니다. 특정 배열 요소 또는 미완료 상태의 DBCS 문자만 변환되지 않고 앞의 배열 요소 또는 문자는 변환됩니다.
4. (변환 이전이나 도중에) 잘림이 발생하면 **DATLEN** 매개변수에 리턴되는 길이는 잘림 이전 변환되지 않은 메시지의 길이입니다.
5. 문자열이 1바이트 문자 세트(SBCS), 2바이트 문자 세트(DBCS) 또는 멀티바이트 문자 세트(MBCS) 간에 변환할 때 문자열이 확장되거나 축소될 수 있습니다.
- PCF 형식 FMADMN, FMEVNT 및 FMPCF에서 MQCFST 및 MQCFSL 구조의 문자열이 변환 이후 문자열을 수용하기 위해 필요에 따라 확장 또는 축소됩니다.
- 문자열 목록 구조 MQCFSL의 경우, 목록의 문자열이 각기 다른 크기로 확장 또는 축소될 수 있습니다. 이 경우, 큐 관리자는 더 짧은 문자열을 공백으로 채워 변환 후 가장 긴 문자열과 동일한 길이가 되도록 합니다.
- FMRMH 형식에서 RMSEO, RMSNO, RMDEO 및 RMDNO 필드를 통해 처리되는 문자열은 변환 이후 문자열을 수용하기 위해 필요에 따라 확장 또는 축소됩니다.
  - FMRFH 형식에서 RFNVS 필드는 변환 이후 이름-값 쌍을 수용하기 위해 필요에 따라 확장 또는 축소됩니다.
  - 필드 크기가 고정된 구조에서 큐 관리자는 중요한 정보가 손실되지 않는 경우 문자열이 고정 필드 내에서 확장 또는 축소될 수 있도록 허용합니다. 이러한 점에서 필드의 첫 번째 널 문자 뒤의 후미 공백 및 문자는 중요하지 않은 것으로 처리됩니다.
    - 문자열이 확장되지만 필드에서 변환된 문자열을 수용하기 위해 중요하지 않은 문자만 제거해야 하는 경우, 변환에 성공하고 CCOK 및 이유 코드 RCNONE이 표시되면서 호출이 완료됩니다(다른 오류가 없다고 가정함).
    - 문자열이 확장되지만 필드에 맞게 변환된 문자열에서 중요한 문자를 제거해야 하는 경우, 메시지가 변환 없이 리턴되고 CCWARN 및 이유 코드 RC2190이 표시되면서 호출이 완료됩니다.
- 참고:** GMATM 옵션의 지정 여부에 상관없이 이유 코드 RC2190가 나타나면 이러한 상황이 발생합니다.
- 문자열이 축소되면 큐 관리자가 필드의 길이에 맞게 문자열을 공백으로 채웁니다.
6. 사용자 데이터가 뒤에 오는 하나 이상의 IBM MQ 헤더 구조로 구성된 메시지의 경우, 헤더 구조의 하나 이상을 변환하고 메시지의 나머지는 변환하지 않을 수 있습니다. 그러나, 두 가지 예외가 있는데, 각 헤더 구조의 MDCSI 및 MDENC 필드는 항상 헤더 구조를 따르는 데이터의 문자 세트 및 인코딩을 올바르게 표시합니다.
- 두 가지 예외는 MQCIH 및 MQIIH 구조로, 해당 구조에서 MDCSI 및 MDENC 필드의 값은 중요하지 않습니다. 해당 구조의 경우, 이 구조를 따르는 데이터가 MQCIH 또는 MQIIH 구조 자체와 동일한 문자 세트 및 인코딩에 있습니다.
7. 검색 중인 메시지의 제어 정보 또는 **MSGDSC** 매개변수에 있는 MDCSI 또는 MDENC 필드가 정의되지 않았거나 지원되지 않는 값을 지정하는 경우, 정의되지 않았거나 지원되지 않는 값을 메시지 변환에 사용할 필요가 없으면 큐 관리자가 오류를 무시할 수 있습니다.
- 예를 들어, 메시지의 MDENC 필드는 지원되지 않는 Float 인코딩을 지정하지만 메시지에 정수 데이터만 있거나 (소스 및 대상 Float 인코딩이 동일하기 때문에) 변환할 필요가 없는 부동 소수점 데이터만 있는 경우, 오류가 진단되거나 진단되지 않을 수 있습니다.
- 오류가 진단되면 완료 코드 CCWARN 및 RC2111, RC2112, RC2113, RC2114 또는 RC2115, RC2116, RC2117, RC2118 이유 코드 (해당되는 경우) 중 하나와 함께 메시지가 변환되지 않은 상태로 리턴됩니다. **MSGDSC** 매개변수의 MDCSI 및 MDENC 필드는 메시지의 제어 정보에 있는 값으로 설정됩니다.

오류가 진단되지 않고 변환이 성공적으로 완료되면 **MSGDSC** 매개변수의 MDCSI 및 MDENC 필드에 리턴된 값은 MQGET 호출을 발행하는 애플리케이션에서 지정한 값입니다.

8. 모든 경우에 메시지가 변환되지 않은 애플리케이션으로 리턴되면 완료 코드는 CCWARN으로 설정되고 **MSGDSC** 매개변수의 MDCSI 및 MDENC 필드는 변환되지 않은 데이터에 적합한 값으로 설정됩니다. FMNONE에 대해서도 동일한 동작이 수행됩니다.

메시지가 잘려야 하지 않는 한, **REASON** 매개변수가 변환을 수행할 수 없는 이유를 나타내는 코드로 설정됩니다. 잘림 관련 이유 코드가 변환 관련된 이유 코드보다 우선합니다. (잘린 메시지가 변환되었는지 판별하려면 **MSGDSC** 매개변수의 MDCSI 및 MDENC 필드에서 리턴된 값을 확인하십시오.)

오류가 진단되면 일반 이유 코드 RC2119 또는 특정 이유 코드가 리턴됩니다. 리턴되는 이유 코드는 기본 데이터 변환 서비스의 진단 기능에 따라 다릅니다.

9. 완료 코드 CCWARN이 리턴되고 둘 이상의 이유 코드가 관련이 있으면 우선순위 순서는 다음과 같습니다.
  - a. 다음 이유 코드가 다른 항목보다 우선합니다.
    - RC2079
  - b. 우선순위에서 다음과 같은 이유가 뒤따릅니다.
    - RC2110
  - c. 남은 이유 코드에서 우선순위의 순서가 정의되어 있지 않습니다.

#### 10. MQGET 호출 완료 시:

- 다음 이유 코드는 메시지가 성공적으로 변환되었음을 표시합니다.
  - RCNONE
- 다음 이유 코드는 메시지가 성공적으로 변환되었을 수 있음 을 표시합니다 (**MSGDSC** 매개변수의 MDCSI 및 MDENC 필드를 확인하여 찾으십시오).
  - RC2079
- 기타 모든 이유 코드는 메시지가 변환되지 않았음을 표시합니다.

다음 처리는 내장 형식에 고유합니다. 사용자 정의 형식에 적용할 수 없습니다.

#### 1. 다음 형식을 제외됩니다.

- FMADMN
- FMEVNT
- FMIMVS
- FMPCF
- FMSTR

내장 형식은 큐 이름에 유효한 문자로 SBCS 문자를 포함하지 않은 문자 세트(부터) 변환할 수 없습니다. 이러한 변환을 수행하려고 하면 완료 코드 CCWARN 및 적절한 이유 코드 RC2111 또는 RC2115가 표시되면서 메시지가 변환 없이 리턴됩니다.

유니코드 문자 세트 UTF-16은 큐 이름에 올바른 문자로 SBCS 문자를 포함하지 않은 문자 세트의 예입니다.

2. 내장 형식의 메시지 데이터가 잘리는 경우 문자열 길이 또는 요소나 구조의 개수가 포함된 메시지 내의 필드는 애플리케이션에 리턴되는 데이터 길이를 반영하도록 조정되지 않습니다. 메시지 데이터 내의 해당 필드에 대해 리턴된 값은 잘림 전에 메시지에 적용할 수 있는 값입니다.

잘린 FMADMN 메시지와 같은 메시지를 처리할 때 애플리케이션이 리턴된 데이터 끝을 넘어 데이터에 액세스하려고 하지 않도록 주위를 기울여야 합니다.

3. 형식 이름이 FMDLH이면 메시지 데이터가 MQDLH 구조로 시작하고 0바이트 이상의 애플리케이션 메시지 데이터가 뒤에 올 수 있습니다. 형식, 문자 세트 및 애플리케이션 메시지 데이터의 인코딩은 메시지 시작 부분에 있는 MQDLH 구조의 DLFMT, DLCSI 및 DLENC 필드를 통해 정의됩니다. MQDLH 구조 및 애플리케이션 메시지 데이터의 문자 세트와 인코딩이 서로 다르므로 MQDLH 구조 및 애플리케이션 메시지 데이터의 하나, 다른 하나 또는 모두 변환이 필요할 수 있습니다.



큐 관리자는 필요에 따라 먼저 MQDLH 구조를 변환합니다. 변환에 성공하거나 MQDLH 구조가 변환을 요구하지 않으면 큐 관리자가 MQDLH 구조의 DLCSI 및 DLENC 필드를 확인하여 애플리케이션 메시지 데이터의 변환이 필요하지 알아봅니다. 변환이 필요한 경우 큐 관리자는 MQDLH 구조의 DLFMT 필드에 지정된 이름을 사용하여 사용자 작성 엑시트를 호출하거나 변환 자체를 수행합니다(DLFMT가 내장 형식의 이름인 경우).

MQGET 호출이 CCWARN 완료 코드를 리턴하고 이유 코드가 변환 실패를 표시하는 값 중 하나이면 다음 중 하나가 적용됩니다.

- MQDLH 구조를 변환할 수 없습니다. 이 경우, 애플리케이션 메시지 데이터도 변환되지 않습니다.
- MQDLH 구조가 변환되었지만, 애플리케이션 메시지 데이터는 변환되지 않았습니다.

애플리케이션은 이전에 적용된 값을 판별하기 위해 MSGDSC 매개변수의 MDCSI 및 MDENC 필드에서 리턴된 값과 MQDLH 구조의 값을 검사할 수 있습니다.

4. 형식 이름이 FMXQH이면 메시지 데이터가 MQXQH 구조로 시작하고 0바이트 이상의 추가 데이터가 뒤에 나올 수 있습니다. 이 추가 데이터는 일반적으로 애플리케이션 메시지 데이터(0 길이일 수 있음)이지만, 추가 데이터 시작 부분에 하나 이상의 IBM MQ 헤더 구조가 더 있을 수도 있습니다.

MQXQH 구조는 큐 관리자의 문자 세트 및 인코딩에 있어야 합니다. MQXQH 구조 뒤에 오는 데이터의 형식, 문자 세트 및 인코딩은 MQXQH 내에 포함된 MQMD 구조의 MDFMT, MDCSI 및 MDENC 필드에 의해 제공됩니다. 각 후속 IBM MQ 헤더 구조에 대해, 구조의 MDFMT, MDCSI 및 MDENC 필드는 해당 구조를 따르는 데이터를 설명하고 해당 데이터는 다른 IBM MQ 헤더 구조 또는 애플리케이션 메시지 데이터입니다.

FMXQH 메시지에 GMCONV 옵션이 지정된 경우, 애플리케이션 메시지 데이터 및 특정 MQ 헤더 구조가 변환되지만 MQXQH 구조의 데이터는 변환되지 않습니다. 따라서 MQGET 호출에서 리턴 시:

- MSGDSC 매개변수의 MDFMT, MDCSI 및 MDENC 필드 값은 애플리케이션 메시지 데이터가 아닌 MQXQH 구조의 데이터를 설명합니다. 따라서 이 값은 MQGET 호출을 발행한 애플리케이션에 의해 지정된 값과 동일하지 않습니다.

이로 인해 GMCONV 옵션을 지정하여 전송 큐에서 메시지를 반복적으로 가져오는 애플리케이션은 각 MQGET 호출 전에 MSGDSC 매개변수의 MDCSI 및 MDENC 필드를 애플리케이션 메시지 데이터에 필요한 값으로 재설정해야 합니다.

- 마지막 MQ 헤더 구조의 MDFMT, MDCSI, 및 MDENC 필드 값이 애플리케이션 메시지 데이터를 설명합니다. 다른 IBM MQ 헤더 구조가 없는 경우, MQXQH 구조 내 MQMD 구조에 있는 이 필드에 애플리케이션 메시지 데이터가 설명됩니다. 변환에 성공하면 해당 값은 MQGET 호출을 발행한 애플리케이션에 의해 MSGDSC 매개변수에 지정된 값과 동일합니다.

메시지가 분배 목록 메시지이면, MQXQH 구조 뒤에 MQDH 구조(또한 MQOR 및 MQPMR 레코드)가 오고 다시 0개 이상의 추가 IBM MQ 헤더 구조와 0바이트 이상의 애플리케이션 메시지 데이터가 뒤에 올 수 있습니다. MQXQH 구조처럼 MQDH 구조는 큐 관리자의 문자 세트 및 인코딩에 있어야 하며, GMCONV 옵션을 지정하지 않았더라도 MQGET 호출에서 변환되지 않습니다.

이전에 설명한 MQXQH 및 MQDH 구조의 처리는 기본적으로 메시지 채널 에이전트가 전송 큐에서 메시지를 가져올 때 사용할 목적으로 제공됩니다.

## IBM i IBM i에서 보고 메시지의 변환

보고 메시지는 원본 메시지의 송신자가 지정한 보고 옵션에 따라 다양한 분량의 애플리케이션 메시지 데이터를 포함할 수 있습니다.

특히, 보고 메시지는

1. 애플리케이션 메시지 데이터를 포함할 수 없습니다.
2. 원본 메시지에 있는 애플리케이션 메시지 데이터의 일부를 포함할 수 있습니다.  
원본 메시지의 송신자가 RO\*D를 지정하고 메시지가 100바이트보다 길면 이 상황이 발생합니다.
3. 원본 메시지에 있는 애플리케이션 메시지 데이터의 전부를 포함할 수 있습니다.

원본 메시지의 송신자가 RO\*F를 지정하거나 RO\*D를 지정하고 메시지가 100바이트 이하이면 이 상황이 발생합니다.

큐 관리자 또는 메시지 채널 에이전트가 보고 메시지를 생성할 때 원본 메시지에서 보고 메시지의 제어 정보에 있는 *MDFMT* 필드에 형식 이름을 복사합니다. 따라서 보고 메시지의 형식 이름이 보고 메시지에 있는 길이와 다른 데이터의 길이를 암시할 수 있습니다(사례 1 및 2에 설명됨).

보고 메시지를 검색할 때 *GMCONV* 옵션이 지정된 경우:

- 이전에 설명된 사례 1에서는 (보고 메시지에 데이터가 없기 때문에) 데이터 변환 엑시트가 호출되지 않습니다.
- 이전에 설명된 사례 3에서는 형식 이름이 메시지 데이터의 길이를 정확하게 암시합니다.
- 그러나, 이전에 설명된 사례 2에서는 형식 이름에 암시된 길이보다 짧은 메시지를 변환하기 위해 데이터 변환 엑시트가 호출됩니다.

또한, 엑시트에 전달된 이유 코드는 일반적으로 *RCNONE*입니다(즉, 이유 코드가 메시지가 잘린 것을 표시하지 않음). 이는 *MQGET* 호출에 대한 응답으로 수신자의 큐 관리자가 아닌 보고 메시지의 송신자에 의해 메시지 데이터가 잘렸기 때문에 발생합니다.

이러한 가능성 때문에 데이터 변환 엑시트는 형식 이름을 사용하여 전달된 데이터의 길이를 추론하지 않아야 합니다. 대신 엑시트는 제공된 데이터 길이를 확인해야 하며 형식 이름으로 내재된 길이보다 작은 데이터를 변환할 준비가 되어 있어야 합니다. 데이터가 성공적으로 변환될 수 있으면 완료 코드 *CCOK* 및 이유 코드 *RCNONE*이 엑시트에 의해 리턴되어야 합니다. 변환할 메시지 데이터의 길이가 엑시트에 **INLEN** 매개변수로 전달됩니다.

## 제품별 프로그래밍 인터페이스

보고 메시지에 발생한 활동 정보를 포함되어 있으면 활동 보고서로 인식됩니다. 활동의 예는 다음과 같습니다.

- 채널을 따라 큐에서 메시지를 송신하는 MCA
- 채널에서 메시지를 수신하여 큐에 넣는 MCA
- 전달 불가능한 메시지를 큐잉하는 MCA 데드 레터
- 큐에서 메시지를 가져오고 제거하는 MCA
- 메시지를 큐에 다시 놓는 데드 레터 핸들러
- PCF 요청을 처리하는 명령 서버 - 발행 요청을 처리하는 브로커
- 큐에서 메시지를 가져오는 사용자 애플리케이션 - 큐에서 메시지를 찾아보는 사용자 애플리케이션

큐 관리자를 포함한 애플리케이션은 보고서 헤더 뒤의 활동 보고서에 일부 메시지 데이터를 추가할 수 있습니다. 일부가 송신된 경우 제공되어야 하는 데이터 양은 고정되어 있지 않으며, 애플리케이션에 의해 결정됩니다. 리턴되는 정보는 활동 보고서를 처리하는 애플리케이션에 유용해야 합니다. 큐 관리자 활동 보고서는 초기 메시지에 포함된 표준 IBM MQ 헤더 구조(시작 'MQH')를 함께 리턴합니다. 예를 들어, 여기에는 원본 메시지에 있는 *MQRFH2* 헤더가 포함됩니다. 또한 큐 관리자는 연관된 PCF 매개변수가 아닌 발견된 *MQCFH* 헤더를 리턴합니다. 이렇게 하면 모니터링 애플리케이션은 메시지의 내용이 무엇인지 알 수 있습니다.

## IBM i IBM i의 MQDXP(데이터 변환 엑시트 매개변수)

데이터 변환 엑시트 매개변수 블록.

### 개요

**목적:** MQDXP 구조는 MQGET 호출 처리의 일부로 메시지 데이터를 변환하기 위해 엑시트를 호출할 때 큐 관리자가 데이터 변환 엑시트에 전달하는 매개변수입니다. 데이터 변환 엑시트에 대한 자세한 정보는 MQCONVX 호출 설명을 참조하십시오.

**문자 세트 및 인코딩:** MQDXP의 문자 데이터는 로컬 큐 관리자의 문자 세트입니다. **CodedCharSetId** 큐 관리자 속성을 통해 제공됩니다. MQDXP의 숫자 데이터는 고유 시스템 인코딩입니다. ENNAT를 통해 제공됩니다.

**사용법:** MQDXP의 *DXLEN*, *DXCC*, *DXREA* 및 *DXRES* 필드만 엑시트에 의해 변경될 수 있습니다. 다른 필드의 변경사항은 무시됩니다. 그러나 변환 중인 메시지가 논리 메시지의 일부만을 포함하는 세그먼트일 경우 *DXLEN* 필드를 변경할 수 없습니다.

엑시트에서 큐 관리자로 제어가 되돌아갈 때 큐 관리자가 MQDXP에서 리턴되는 값을 검사합니다. 리턴되는 값이 올바르지 않으면 큐 관리자는 엑시트가 *DXRES*에 *XRFAIL*을 리턴한 것처럼 처리를 계속합니다. 그러나 큐 관리자

는 이 경우 엑시트에서 리턴한 *DXCC* 및 *DXREA* 필드의 값을 무시하고, 엑시트에 입력 시 해당 필드의 값을 대신 사용합니다. MQDXP에서는 다음 값으로 인해 아래와 같은 처리가 발생합니다.

- *DXRES* 필드가 *XROK* 및 *XRFAIL*이 아님
- *DXCC* 필드가 *CCOK* 및 *CCWARN*이 아님
- *DXLEN* 필드가 0보다 작거나, 변환되는 메시지가 논리 메시지의 일부만 포함한 세그먼트인 경우 *DXLEN* 필드가 변경됩니다.
- [1375 페이지의 『필드』](#)
- [1378 페이지의 『RPG declaration \(copy file CMQDXPH\)』](#)

## 필드

MQDXP 구조에는 다음 필드가 포함됩니다. 필드에 대해서는 **알파벳순**으로 설명합니다.

### DXAOP(10자리 부호 있는 정수)

애플리케이션 옵션.

MQGET 호출을 발행하는 애플리케이션이 지정한 MQGMO 구조의 *GMOPT* 필드 사본입니다. GMATM 옵션이 지정되었는지 확인하기 위해 엑시트가 조사해야 할 수도 있습니다.

엑시트의 입력 필드입니다.

### DXCC(10자리 부호 있는 정수)

완료 코드.

엑시트가 호출되면 여기에 MQGET 호출을 발행한 애플리케이션에 리턴될 완료 코드가 포함됩니다. 단, 엑시트가 아무 것도 수행하지 않는 경우에만 적용됩니다. 메시지가 잘리거나 메시지에 변환이 필요하지만 아직 수행되지 않았기 때문에 항상 *CCWARN*입니다.

엑시트의 출력에서 이 필드는 MQGET 호출의 **CMPCOD** 매개변수에 있는 애플리케이션에 리턴되는 완료 코드를 포함합니다. *CCOK* 및 *CCWARN*만 유효합니다. 엑시트가 출력에 이 필드를 설정하는 방법은 *DXREA* 필드의 설명을 참조하십시오.

이는 엑시트의 입출력(I/O) 필드입니다.

### DXCSI(10자리 부호 있는 정수)

애플리케이션에 필요한 문자 세트.

MQGET 호출을 발행하는 애플리케이션에 필요한 문자 세트의 코드화 문자 세트 ID입니다. 자세한 정보는 MQMD 구조의 *MDCSI* 필드를 참조하십시오. 애플리케이션이 MQGET 호출에 특수 값 *CSQM*을 지정하는 경우, 큐 관리자가 엑시트를 호출하기 전에 이 값을 큐 관리자에 사용된 문자 세트의 실제 문자 세트 ID로 변경합니다.

변환에 성공하면 엑시트가 이를 메시지 디스크립터 내의 *MDCSI* 필드에 복사해야 합니다.

엑시트의 입력 필드입니다.

### DXENC(10자리 부호 있는 정수)

애플리케이션에 필요한 숫자 인코딩.

MQGET 호출을 발행하는 애플리케이션에 필요한 숫자 인코딩입니다. 자세한 정보는 MQMD 구조의 *MDENC* 필드를 참조하십시오.

변환에 성공하면 엑시트가 이를 메시지 디스크립터 내의 *MDENC* 필드에 복사해야 합니다.

엑시트의 입력 필드입니다.

### DXHCN(10자리 부호 있는 정수)

연결 핸들입니다.

MQXCNCV 호출에서 사용될 수 있는 연결 핸들입니다. 이 핸들이 MQGET 호출을 발행한 애플리케이션이 지정한 핸들과 반드시 동일할 필요는 없습니다.

## DXLEN(10자리 부호 있는 정수)

메시지 데이터의 길이(바이트).

엑시트가 호출되면 이 필드에는 애플리케이션 메시지 데이터의 원본 길이가 포함됩니다. 애플리케이션에서 제공하는 버퍼에 맞게 메시지가 잘린 경우, 엑시트에 제공되는 메시지의 크기가 *DXLEN*의 값보다 작습니다. 엑시트에 제공되는 메시지의 크기는 발생한 잘림에 상관없이 항상 엑시트의 **INLEN** 매개변수를 통해 제공됩니다.

잘림은 엑시트에 대한 입력에 RC2079 값이 있는 *DXREA* 필드로 표시됩니다.

대부분의 변환에서는 이러한 길이 변경이 필요하지 않지만 필요하면 엑시트가 자르기를 수행할 수 있습니다. 엑시트에 의해 설정된 값은 MQGET 호출의 **DATLEN** 매개변수에서 애플리케이션에 리턴됩니다. 그러나 변환 중인 메시지가 논리 메시지의 일부만을 포함하는 세그먼트일 경우 이 길이를 변경할 수 없습니다. 그 이유는 길이를 변경하면 논리 메시지에서 이후 세그먼트의 오프셋이 올바르게 맞지 않기 때문입니다.

엑시트가 데이터의 길이를 변경하려는 경우 큐 관리자가 이미 변경되지 않은 데이터의 길이를 기반으로 메시지 데이터가 애플리케이션의 버퍼에 맞는지 결정한 점에 유의하십시오. 이 결정에 따라 메시지가 큐에서 제거되는지(또는 찾아보기 요청에 대해 찾아보기 커서가 이동되는지), 변환으로 발생한 데이터 길이의 변경에 영향을 받지 않는지 판별합니다. 이런 이유 때문에 변환 엑시트가 애플리케이션 메시지 데이터의 길이를 변경하지 않도록 권장합니다.

문자 변환이 길이 변경을 암시하는 경우, 필요에 따라 후미 공백을 자르거나 공백으로 채워서 문자열이 동일한 길이(바이트)의 다른 문자열로 변환될 수 있습니다.

메시지에 애플리케이션 메시지 데이터가 포함되지 않으면 엑시트가 호출되지 않습니다. 따라서 *DXLEN*이 항상 0보다 큼니다.

이는 엑시트의 입출력(I/O) 필드입니다.

## DXREA(10자리 부호 있는 정수)

*DXCC*을(를) 규정하는 이유 코드.

엑시트가 호출되면 여기에 MQGET 호출을 발행한 애플리케이션에 리턴될 이유 코드가 포함됩니다. 단, 엑시트가 아무 것도 수행하지 않는 경우에만 적용됩니다. 가능한 값 중에서 RC2079는 애플리케이션이 제공한 버퍼에 맞게 메시지가 잘렸음을 나타내며 RC2119는 메시지에 변환이 필요하나 아직 변환되지 않았음을 나타냅니다.

엑시트의 출력에서 이 필드는 MQGET 호출의 **REASON** 매개변수에 애플리케이션에 리턴되는 이유 코드를 포함합니다. 권장사항은 다음과 같습니다.

- *DXREA*에서 엑시트에 대한 입력에 RC2079 값이 있는 경우, 변환의 성공 또는 실패에 상관없이 *DXREA* 및 *DXCC* 필드는 대체되지 않아야 합니다.

(*DXCC* 필드가 CCOK가 아닌 경우, 메시지를 검색하는 애플리케이션이, 요청된 값이 있는 메시지 디스크립터에서 리턴되는 *MDENC* 및 *MDCSI* 값을 비교하여 변환 실패를 식별할 수 있습니다. 대조적으로, 애플리케이션은 버퍼 크기에 맞는 메시지에서 잘린 메시지를 구별하지 못합니다. 이런 이유로 인해 변환 실패를 표시하는 모든 이유보다 우선적으로 RC2079가 리턴되어야 합니다.

- *DXREA*에서 엑시트에 대한 입력에 기타 값이 있는 경우:

- 변환이 성공하면 *DXCC*가 CCOK로 설정되고 *DXREA*가 RCNONE으로 설정되어야 합니다.
- 변환이 실패하거나 메시지가 확장되어 버퍼에 맞도록 잘라야 하는 경우, *DXCC*가 CCWARN으로 설정되고(또는 변경되지 않은 상태로 유지) *DXREA*가 다음 목록의 값 중 하나로 설정되어 실패의 네이처를 표시해야 합니다.

변환 이후 메시지가 버퍼에 비해 너무 크면 MQGET 호출을 발행한 애플리케이션이 GMATM 옵션을 지정한 경우에만 메시지를 잘라야 합니다.

- 해당 옵션을 지정한 경우, 이유 RC2079가 리턴되어야 합니다.
- 해당 옵션을 지정하지 않은 경우, 이유 코드 RC2120이 표시되면서 메시지가 변환되지 않은 채 리턴되어야 합니다.

다음 목록의 이유 코드는 변환이 실패한 이유를 표시하기 위해 엑시트에 사용하도록 권장됩니다. 그러나 엑시트는 적절하다고 판단되면 RC\* 코드 세트에서 다른 값을 리턴할 수 있습니다. 또한 엑시트가 MQGET 호출을 발행하여 애플리케이션에 전달하려는 조건을 표시하기 위해 RC0900 ~ RC0999의 값 범위가 엑시트에 사용하도록 할당됩니다.

**참고:** 메시지를 성공적으로 변환하지 못한 경우, 큐 관리자가 변환되지 않은 메시지를 리턴하도록 하기 위해 엑시트가 DXRES 필드에서 XRFAIL을 리턴해야 합니다. 이는 DXREA 필드에서 리턴되는 이유 코드에 상관없이 적용됩니다.

**RC0900**

(900, X'384') 애플리케이션 정의 이유 코드에 대한 최저값입니다.

**RC0999**

(999, X'3E7') 애플리케이션 정의 이유 코드에 대한 최고값입니다.

**RC2120**

(2120, X'848') 변환된 데이터가 버퍼에 비해 너무 큼니다.

**RC2119**

(2119, X'847') 메시지 데이터가 변환되지 않았습니다.

**RC2111**

(2111, X'83F') 소스 코드화 문자 세트 ID가 올바르지 않습니다.

**RC2113**

(2113, X'841') 메시지의 팩형 10진수 인코딩이 인식되지 않습니다.

**RC2114**

(2114, X'842') 메시지의 부동 소수점 인코딩이 인식되지 않습니다.

**RC2112**

(2112, X'840') 소스 정수 인코딩이 인식되지 않습니다.

**RC2115**

(2115, X'843') 대상 코드화 문자 세트 ID가 올바르지 않습니다.

**RC2117**

(2117, X'845') 수신자가 지정한 팩형 10진수 인코딩을 인식할 수 없습니다.

**RC2118**

(2118, X'846') 수신자에 의해 지정되는 부동 소수점 인코딩이 인식되지 않습니다.

**RC2116**

(2116, X'844') 대상 정수 인코딩이 인식되지 않습니다.

**RC2079**

(2079, X'81F') 잘린 메시지가 리턴되었습니다(처리 완료됨).

이는 엑시트의 입출력(I/O) 필드입니다.

**DXRES(10자리 부호 있는 정수)**

엑시트로부터의 응답.

성공을 표시하기 위해 엑시트에 의해 설정되고, 그 밖의 경우에는 변환의 설정입니다. 다음 중 하나여야 합니다.

**XROK**

변환에 성공했습니다.

엑시트가 이 값을 지정하는 경우에는 큐 관리자가 MQGET 호출을 발행한 애플리케이션에 다음을 리턴합니다.

- 엑시트의 출력에서 DXCC 필드의 값
- 엑시트의 출력에서 DXREA 필드의 값
- 엑시트의 출력에서 DXLEN 필드의 값
- 엑시트의 출력 버퍼 OUTBUF의 콘텐츠. 리턴되는 바이트의 수는 엑시트의 OUTLEN 매개변수 및 엑시트의 출력에서 DXLEN 필드의 값 중 작은 값입니다.

엑시트의 메시지 디스크립터 매개변수에 있는 *MDENC* 및 *MDCSI* 필드가 둘 다 변경되지 않으면 큐 관리자가 다음을 리턴합니다.

- 엑시트에 대한 입력에서 MQDXP 구조의 *MDENC* 및 *MDCSI* 필드 값

엑시트의 메시지 디스크립터 매개변수에 있는 *MDENC* 및 *MDCSI* 필드 중 하나 또는 둘 다가 변경된 경우, 큐 관리자가 다음을 리턴합니다.

- 엑시트의 출력에서 엑시트의 메시지 디스크립터 매개변수에 있는 *MDENC* 및 *MDCSI* 필드 값
- 

#### **XRFAIL**

변환에 실패했습니다.

엑시트가 이 값을 지정하는 경우에는 큐 관리자가 MQGET 호출을 발행한 애플리케이션에 다음을 리턴합니다.

- 엑시트의 출력에서 *DXCC* 필드의 값
- 엑시트의 출력에서 *DXREA* 필드의 값
- 엑시트에 대한 입력에서 *DXLEN* 필드의 값
- 엑시트의 입력 버퍼 *INBUF*의 콘텐츠. 리턴되는 바이트 수는 **INLEN** 매개변수를 통해 제공됩니다.

엑시트가 *INBUF*를 대체한 경우, 결과가 정의되지 않습니다.

*DXRES*는 엑시트의 출력 필드입니다.

#### **DXSID(4바이트 문자열)**

구조 ID입니다.

값은 다음과 같아야 합니다.

#### **DXSIDV**

데이터 변환 엑시트 매개변수 구조의 ID.

엑시트의 입력 필드입니다.

#### **DXVER(10자리 부호 있는 정수)**

구조 버전 번호입니다.

값은 다음과 같아야 합니다.

#### **DXVER1**

데이터 변환 엑시트 매개변수 구조의 버전 번호.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### **DXVERC**

데이터 변환 엑시트 매개변수 구조의 현재 버전.

**참고:** 이 구조의 새 버전을 도입해도 기존 부분의 레이아웃은 변경되지 않습니다. 따라서 엑시트는 *DXVER* 필드가 해당 엑시트가 사용해야 하는 필드를 포함한 가장 낮은 버전보다 높거나 같은지 확인해야 합니다.

엑시트의 입력 필드입니다.

#### **DXXOP(10자리 부호 있는 정수)**

예약됨

이 필드는 예약된 필드이며 해당 값은 0입니다.

### **RPG declaration (copy file CMQDXPH)**

```
D*..1.....2.....3.....4.....5.....6.....7..  
D* MQDXP Structure
```

```

D*
D* Structure identifier
D DXSID 1 4
D* Structure version number
D DXVER 5 8I 0
D* Reserved
D DXXOP 9 12I 0
D* Application options
D DXAOP 13 16I 0
D* Numeric encoding required by application
D DXENC 17 20I 0
D* Character set required by application
D DXCSI 21 24I 0
D* Length in bytes of message data
D DXLEN 25 28I 0
D* Completion code
D DXCC 29 32I 0
D* Reason code qualifying DXCC
D DXREA 33 36I 0
D* Response from exit
D DXRES 37 40I 0
D* Connection handle
D DXHCN 41 44I 0

```

## IBM i IBM i의 MQXCNCV(문자 변환)

MQXCNCV 호출은 문자 간에 변환합니다.

이 호출은 IBM MQ 프레임워크 인터페이스 중 하나인 IBM MQ 데이터 변환 인터페이스(DCI)의 일부입니다. 참고: 이 호출은 데이터 변환 엑시트에서만 사용할 수 있습니다.

- [1379 페이지의 『구문』](#)
- [1379 페이지의 『매개변수』](#)
- [1383 페이지의 『RPG 호출\(ILE\)』](#)

### 구문

**MQXCNCV HCONN, OPTS, SRCCSI, SRCLEN, SRCBUF, TGTCSI, TGTLEN, TGTBUF, DATLEN, CMPCOD, REASON)**

### 매개변수

MQXCNCV 호출의 매개변수는 다음과 같습니다.

#### HCONN(10자리 부호 있는 정수) - 입력

연결 핸들입니다.

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. 일반적으로 MQDXP 구조의 DXHCN 필드에 있는 데이터 변환 엑시트에 전달되는 핸들이어야 합니다. 이 핸들이 MQGET 호출을 발행한 애플리케이션에서 지정한 핸들과 반드시 동일하지는 않습니다.

IBM i에서 HCONN에 다음 특수 값을 지정할 수 있습니다.

#### HCDEFH

기본 연결 핸들

#### OPTS(10자리 부호 있는 정수) - 입력

MQXCNCV의 조치를 제어하는 옵션입니다.

이 섹션의 뒷부분에 설명된 옵션을 0개 이상 지정할 수 있습니다. 둘 이상의 옵션이 필요하면 값을 추가할 수 있습니다(동일한 상수를 두 번 이상 추가하지 않음).

**기본 변환 옵션:** 다음 옵션은 기본 문자 변환의 사용을 제어합니다.

## DCCDEF

기본 변환.

이 옵션은 호출에 지정된 문자 세트 중 하나 또는 둘 다 지원되지 않는 경우 기본 문자 변환을 사용할 수 있도록 지정합니다. 이로 인해 기본값 변환은 문자열 변환 시 지정된 문자 세트를 대략적으로 나타내는 설치 지정 기본값 문자 세트를 큐 관리자가 사용할 수 있습니다.

**참고:** 대략적인 문자 세트를 사용하여 문자열을 변환하면 일부 문자가 제대로 변환되지 않는 결과가 발생할 수 있습니다. 지정된 문자 세트 및 기본 문자 세트 모두에 공통된 문자열의 문자를 사용하여 이를 피할 수 있습니다.

큐 관리자가 설치되거나 재시작될 때 기본 문자 세트는 구성 옵션에 의해 정의됩니다.

DCCDEF를 지정하지 않는 경우, 큐 관리자가 지정된 문자 세트만 사용하여 문자열을 변환하며, 문자 세트의 하나 또는 둘 다가 지원되지 않으면 호출이 실패합니다.

**패딩 옵션:** 다음 옵션을 사용하면 변환된 문자열에 대상 버퍼에 맞도록 큐 관리자가 변환된 문자열을 공백으로 채우거나 중요하지 않은 후미 문자를 제거할 수 있습니다.

## DCCFIL

대상 버퍼를 채웁니다.

이 옵션은 대상 버퍼가 완전히 채워지는 식으로 변환이 발생하도록 요청합니다.

- 변환될 때 문자열이 축소되는 경우 대상 버퍼를 채우기 위해 후미 문자 공백이 추가됩니다.
- 변환 시 문자열이 확장되면 변환된 문자열이 대상 버퍼에 맞도록 중요하지 않은 후미 문자가 제거됩니다. 이 작업이 성공적으로 완료되면 CCOK 및 이유 코드 RCNONE이 표시되면서 호출이 완료됩니다.

중요하지 않은 후미 문자가 너무 적으면 그에 맞는 크기의 문자열이 대상 버퍼에 놓이고 CCWARN 및 이유 코드 RC2120이 표시되면서 호출이 완료됩니다.

중요하지 않은 문자는 다음과 같습니다.

- 후미 문자 공백
- 문자열에서 첫 번째 널 문자 뒤에 오는 문자(첫 번째 널 문자 자체 제외)
- 문자열 TGTCSI 및 TGTLEN 가 대상 버퍼가 올바른 문자로 완전히 설정될 수 없는 경우 호출은 CCFAIL 및 이유 코드 RC2144로 실패합니다. 이는 TGTCSI 가 순수 DBCS 문자 세트(예: UTF-16) 이지만 TGTLEN 가 홀수 바이트 길이를 지정하는 경우에 발생할 수 있습니다.
- TGTLEN은 SRCLEN보다 작거나 클 수 있습니다. MQXCNVV의 리턴에서 DATLEN은 TGTLEN과 동일한 값을 갖습니다.

이 옵션을 지정하지 않은 경우:

- 필요에 따라 대상 버퍼 내에서 문자열을 축소하거나 확장할 수 있습니다. 중요하지 않은 후미 문자는 추가되거나 제거되지 않습니다.
- 변환된 문자열이 대상 버퍼에 맞으면 CCOK 및 이유 코드 RCNONE이 표시되면서 호출이 완료됩니다.
- 변환된 문자열이 대상 버퍼에 비해 너무 크면 그에 맞는 크기의 문자열이 대상 버퍼에 놓이고 CCWARN 및 이유 코드 RC2120이 표시되면서 호출이 완료됩니다. 이 경우 TGTLEN 바이트 미만이 리턴될 수 있습니다.
- TGTLEN은 SRCLEN보다 작거나 클 수 있습니다. MQXCNVV의 리턴에서 DATLEN은 TGTLEN보다 작거나 같습니다.

**인코딩 옵션:** 소스 및 대상 문자열의 정수 인코딩을 지정하는 데 다음 옵션을 사용할 수 있습니다. 해당 문자 세트 ID가 기본 스토리지의 문자 세트 표현이 2진 정수에 사용된 인코딩에 종속됨을 나타내는 경우에만 관련 인코딩이 사용됩니다. 이는 특정 멀티바이트 문자 세트(예: UTF-16 문자 세트)에만 영향을 미칩니다.

문자 세트가 단일 바이트 문자 세트(SBCS) 또는 정수 인코딩에 종속되지 않은 주 기억장치의 표현이 포함된 멀티바이트 문자 세트인 경우 인코딩은 무시됩니다.

DCCT\* 값 중 하나와 결합하여 DCCS\* 값 중 하나만 지정해야 합니다.

## DCCSNA

소스 인코딩은 환경 및 프로그래밍 언어의 기본값입니다.



**DCCSNO**

소스 인코딩은 정상입니다.

**DCCSRE**

소스 인코딩은 역순됩니다.

**DCCSUN**

소스 인코딩은 정의되지 않습니다.

**DCCTNA**

대상 인코딩은 환경 및 프로그래밍 언어의 기본값입니다.

**DCCTNO**

대상 인코딩은 정상입니다.

**DCCTRE**

대상 인코딩은 역순됩니다.

**DCCTUN**

대상 인코딩은 정의되지 않습니다.

이전에 정의된 인코딩 값을 OPTS 필드에 직접적으로 추가할 수 있습니다. 그러나 소스 또는 대상 인코딩을 MQMD나 다른 구조의 MDENC 필드에서 가져온 경우 다음 처리를 수행해야 합니다.

1. 정수 인코딩은 float 및 팩형 10진수 인코딩을 제거하여 MDENC 필드에서 추출되어야 합니다. 이 작업 방법에 대한 자세한 정보는 1364 페이지의 『IBM i의 분석 인코딩』의 내용을 참조하십시오.
2. OPTS 필드에 추가하기 전에 1단계에서 생성된 정수 인코딩에 적절한 요인을 곱해야 합니다. 요인은 다음과 같습니다.

**DCCSFA**

소스 인코딩의 요인

**DCCTFA**

대상 인코딩의 요인

값을 지정하지 않으면 인코딩 옵션이 정의되지 않음(DCC\*UN)으로 기본 설정됩니다. 대부분의 경우, 이는 MQXCNCV 호출의 성공적인 완료에 영향을 주지 않습니다. 그러나 해당 문자 세트가 인코딩에 종속된 표현을 포함한 멀티바이트 문자 세트인 경우(예: UTF-16 문자 세트), 호출에 실패하고 이유 코드 RC2112 또는 RC2116이 표시됩니다(적절한 경우).

**기본 옵션:** 이전에 설명한 옵션을 지정하지 않은 경우, 다음 옵션을 사용할 수 있습니다.

**DCCNON**

옵션이 지정되지 않았습니다.

OOBNDQ는 프로그램 문서를 보조하기 위해 정의됩니다. 이 옵션은 기타와 함께 사용하기 위한 용도가 아니지만, 해당 값이 0이므로 해당 사용을 감지할 수 없습니다.

**SRCCSI(10자리 부호 있는 정수) - 입력**

변환 전 문자열의 코드화 문자 세트 ID.

SRCBUF에서 입력 문자열의 코드화 문자 세트 ID입니다.

**SRCLLEN(10자리 부호 있는 정수) - 입력**

변환 전 문자열의 길이.

SRCBUF의 입력 문자열의 길이(바이트)입니다. 0 이상이어야 합니다.

**SRCBUF(1바이트 문자열 x SRCLLEN) - 입력**

변환되는 문자열.

문자 간에 변환되는 문자열을 포함한 버퍼입니다.

**TGTCSI(10자리 부호 있는 정수) - 입력**

변환 후 문자열의 코드화 문자 세트 ID.

SRCBUF가 변환되는 문자 세트의 코드화 문자 세트 ID입니다.

**TGTLEN(10자리 부호 있는 정수) - 입력**

출력 버퍼의 길이.

출력 버퍼 TGTBUF의 길이(바이트)입니다. 0 이상이어야 합니다. SRCLEN보다 작거나 클 수 있습니다.

**TGTBUF(1바이트 문자열 x TGTLEN) - 출력**

변환 후 문자열.

TGTCSI에 정의된 문자 세트로 변환된 후 문자열입니다. 변환된 문자열은 변환되지 않은 문자열보다 짧거나 길 수 있습니다. **DATLEN** 매개변수는 리턴되는 유효 바이트의 수를 표시합니다.

**DATLEN(10자리 부호 있는 정수) - 출력**

출력 문자열의 길이.

출력 버퍼 TGTBUF에 리턴된 문자열의 길이입니다. 변환된 문자열은 변환되지 않은 문자열보다 짧거나 길 수 있습니다.

**CMPCOD(10자리 부호 있는 정수) - 출력**

완료 코드.

다음 중 하나입니다.

**CCOK**

정상적으로 완료되었습니다.

**CCWARN**

경고(일부 완료).

**CCFAIL**

호출에 실패했습니다.

**REASON(10자리 부호 있는 정수) - 출력**

CMPCOD을(를) 규정하는 이유 코드.

CMPCOD가 CCOK일 경우:

**RCNONE**

(0, X'000') 보고할 이유가 없습니다.

If CMPCOD is CCWARN:

**RC2120**

(2120, X'848') 변환된 데이터가 버퍼에 비해 너무 큽니다.

CMPCOD가 CCFAIL일 경우:

**RC2010**

(2010, X'7DA') 데이터 길이 매개변수가 올바르지 않습니다.

**RC2150**

(2150, X'866') DBCS 문자열이 올바르지 않습니다.

**RC2018**

(2018, X'7E2') 연결 핸들이 유효하지 않습니다.

**RC2046**

(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.

**RC2102**

(2102, X'836') 사용 가능한 시스템 자원이 충분하지 않습니다.

**RC2145**

(2145, X'861') 소스 버퍼 매개변수가 올바르지 않습니다.

**RC2111**

(2111, X'83F') 소스 코드화 문자 세트 ID가 올바르지 않습니다.

**RC2112**

(2112, X'840') 소스 정수 인코딩이 인식되지 않습니다.

**RC2143**

(2143, X'85F') 소스 길이 매개변수가 올바르지 않습니다.

**RC2071**

(2071, X'817') 사용 가능한 스토리지가 충분하지 않습니다.

**RC2146**

(2146, X'862') 대상 버퍼 매개변수가 올바르지 않습니다.

**RC2115**

(2115, X'843') 대상 코드화 문자 세트 ID가 올바르지 않습니다.

**RC2116**

(2116, X'844') 대상 정수 인코딩이 인식되지 않습니다.

**RC2144**

(2144, X'860') 대상 길이 매개변수가 올바르지 않습니다.

**RC2195**

(2195, X'893') 예상치 못한 오류가 발생했습니다.

이러한 이유 코드에 대한 자세한 정보는 [1359 페이지](#)의 『IBM i(ILE RPG)의 리턴 코드』의 내용을 참조하십시오.

**RPG 호출(ILE)**

```

C*.1.....2.....3.....4.....5.....6.....7..
C          CALLP      MQXCNV(CONN : OPTS : SRCCSI :
C                      SRCLEN : SRCBUF : TGTCSI :
C                      TGTLEN : TGTBUF : DATLEN :
C                      CMPCOD : REASON)

```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```

D*.1.....2.....3.....4.....5.....6.....7..
DMQXCNV      PR          EXTPROC('MQXCNV')
D* Connection handle
D HCONN      10I 0 VALUE
D* Options that control the action of MQXCNV
D OPTS      10I 0 VALUE
D* Coded character set identifier of string before conversion
D SRCCSI    10I 0 VALUE
D* Length of string before conversion
D SRCLEN    10I 0 VALUE
D* String to be converted
D SRCBUF    * VALUE
D* Coded character set identifier of string after conversion
D TGTCSI    10I 0 VALUE
D* Length of output buffer
D TGTLEN    10I 0 VALUE
D* String after conversion
D TGTBUF    * VALUE
D* Length of output string
D DATLEN    10I 0
D* Completion code
D CMPCOD    10I 0
D* Reason code qualifying CMPCOD
D REASON    10I 0

```

**IBM i IBM i의 MQCONVX(데이터 변환 엑시트)**

이 호출 정의는 데이터 변환 엑시트로 전달되는 매개변수에 대해 설명합니다.

큐 관리자에서 MQCONVX라는 시작점을 제공하지 않았습니다(사용 시 참고사항 [1385 페이지](#)의 『11』 참조).

이 정의는 IBM MQ 프레임워크 인터페이스 중 하나인 IBM MQ 데이터 변환 인터페이스(DCI)의 일부입니다.

- [1384 페이지](#)의 『구문』

- [1384 페이지의 『사용법 참고』](#)
- [1385 페이지의 『매개변수』](#)
- [1386 페이지의 『RPG 호출\(ILE\)』](#)

## 구문

**MQCONVX (MQDXP, MQMD, INLEN, INBUF, OUTLEN, OUTBUF)**

## 사용법 참고

1. 데이터 변환 엑시트는 MQGET 호출을 처리하는 동안 제어를 수신하는 사용자 작성 엑시트입니다. 데이터 변환 엑시트에 의해 수행되는 함수는 엑시트 제공자가 정의합니다. 그러나 엑시트는 여기에 그리고 연관된 매개변수 구조 MQDXP에 설명된 규칙을 준수해야 합니다.

데이터 변환 엑시트에 사용할 수 있는 프로그래밍 언어는 환경에 따라 판별됩니다.

2. 엑시트는 다음 명령문이 모두 true인 경우에만 호출됩니다.

- MQGET 호출에 GMCONV 옵션이 지정됩니다.
- 메시지 디스크립터에서 *MDFMT* 필드는 FMNONE이 아닙니다.
- 메시지가 아직 필요한 표현이 아닙니다. 즉, 메시지의 *MDCSI* 및 *MDENC* 중 하나 또는 둘 다가 MQGET 호출에 제공된 메시지 디스크립터에 애플리케이션을 통해 지정된 값과 다릅니다.
- 큐 관리자가 아직 변환을 성공적으로 완료하지 않았습니다.
- 애플리케이션의 버퍼 길이가 0보다 큽니다.
- 메시지 데이터의 길이는 0보다 큽니다.
- MQGET 조작 중에 지금까지 표시된 이유 코드는 RCNONE 또는 RC2079입니다.

3. 엑시트를 작성할 때는 잘린 메시지를 변환할 수 있도록 허용하는 방식으로 엑시트를 코딩하는 것을 고려해야 합니다. 잘린 메시지는 다음과 같이 발생할 수 있습니다.

- 수신 애플리케이션이 메시지보다 작은 버퍼를 제공하지만, MQGET 호출에 GMATM 옵션을 지정합니다. 이 경우 엑시트에 대한 입력의 **MQDXP** 매개변수에 있는 *DXREA* 필드의 값은 RC2079입니다.
- 메시지 송신자는 송신 전에 잘립니다. 예를 들어, 이 동작은 보고 메시지에서 발생합니다(자세한 정보는 [1373 페이지의 『IBM i에서 보고 메시지의 변환』](#) 참조). 이 경우 엑시트에 대한 입력에서 **MQDXP** 매개변수의 *DXREA* 필드에 RCNONE값이 있습니다(수신 애플리케이션이 메시지에 대해 충분히 큰 버퍼를 제공한 경우).

따라서 메시지의 잘림 여부를 결정하는 데 엑시트에 대한 입력에 있는 *DXREA* 필드 값을 항상 사용할 수 있는 것은 아닙니다.

잘린 메시지의 특성은 **INLEN** 매개변수의 엑시트에 제공된 길이가 메시지 디스크립터의 *MDFMT* 필드에 있는 형식 이름이 암시한 길이보다 작다는 점입니다. 따라서 엑시트는 데이터 변환을 시도하기 전에 **INLEN**의 값을 검사해야 합니다. 엑시트는 형식 이름이 암시하는 전체 데이터 양이 제공되었다고 가정하지 않아야 합니다.

엑시트가 잘린 메시지를 변환하기 위해 작성되지 않고 **INLEN**이 예상된 값 미만인 경우, 엑시트는 *DXCC* 필드를 *CCWARN*으로, *DXREA* 필드를 RC2110으로 설정하여 **MQDXP** 매개변수의 *DXRES* 필드에서 *XRFAIL*을 리턴해야 합니다.

엑시트가 잘린 메시지를 변환하도록 작성된 경우, 엑시트가 최대한 많은 데이터를 변환해야 하며(다음 사용시 참고사항 참조) *INBUF*의 끝을 벗어나는 데이터를 조사하거나 변환하려고 하지 않도록 주의해야 합니다. 변환이 성공적으로 완료되면 엑시트는 **MQDXP** 매개변수의 *DXREA* 필드를 변경하지 않은 채로 두어야 합니다. 이렇게 하면 수신자의 큐 관리자로 인해 메시지가 잘린 경우 RC2079를, 메시지 송신자에 의해 메시지가 잘린 경우 RCNONE을 리턴합니다.

또한 메시지가 *OUTBUF*보다 큰 지점으로 변환하는 동안 메시지를 확장할 수도 있습니다. 이 경우 엑시트는 메시지를 자르는지 여부를 결정해야 합니다. **MQDXP** 매개변수의 *DXAOP* 필드는 수신 애플리케이션이 GMATM 옵션을 지정했는지 여부를 표시합니다.

4. 일반적으로 *INBUF*에서 엑시트에 제공된 메시지의 모든 데이터가 변환되거나 모두 변환되지 않는 것이 좋습니다. 그러나 변환 이전이나 변환하는 동안 메시지가 잘리는 경우는 이에 대한 예외입니다. 이 경우에는 버퍼 끝에 미완료 항목이 있을 수 있습니다(예: 2바이트 중 1바이트 문자, 4바이트 중 3바이트 정수). 이 상황에서는 미완료 항목을 생략하고 *OUTBUF*의 미사용 바이트를 널로 설정하는 것이 좋습니다. 그러나 배열 또는 문자열 내에서 완료 요소나 문자는 변환되어야 합니다.
5. 엑시트가 처음 필요한 경우, 큐 관리자가 (확장과 별도로) 형식과 이름이 같은 오브젝트를 로드하려고 합니다. 로드된 오브젝트에는 해당 형식 이름의 메시지를 처리하는 엑시트가 있어야 합니다. 모든 환경에서 요구하지는 않지만, 엑시트 이름 및 엑시트를 포함한 오브젝트의 이름을 같게 하는 것이 좋습니다.
6. 애플리케이션이 큐 관리자에 연결된 이후 *MDFMT*를 사용하는 첫 번째 메시지를 수신하려고 할 때 새 엑시트 사본이 로드됩니다. 큐 관리자가 이전에 로드된 사본을 제거한 경우에도 새 사본이 로드될 수 있습니다. 따라서, 엑시트가 정적 스토리지를 사용하여 엑시트의 호출 간에 정보를 전달하려고 하지 않아야 합니다. 엑시트가 두 호출 간에 언로드될 수 있습니다.
7. 큐 관리자에 지원되는 내장 형식 중 하나와 동일한 이름의 사용자 제공 엑시트가 있으면 사용자 제공 엑시트가 내장 변환 루틴을 대체하지 않습니다. 이러한 엑시트가 호출되는 유일한 환경은 다음과 같습니다.
  - 내장 변환 루틴이 관련 *MDCSI* 또는 *MDENC*로(부터) 변환을 처리할 수 없습니다. 또는
  - 내장 변환 루틴이 데이터를 변환하지 못했습니다(예: 변환 불가능한 필드 또는 문자가 있기 때문에).
8. 엑시트의 범위는 환경에 따라 다릅니다. 다른 형식과의 충돌 위험을 최소화하기 위해 *MDFMT* 이름을 선택해야 합니다. 형식 이름을 정의하는 애플리케이션을 식별하는 문자로 시작하는 것이 좋습니다.
9. 데이터 변환 엑시트가 *MQGET* 호출을 실행한 프로그램의 엑시트와 같은 환경에서 실행됩니다. 환경에는 주소 공간과 사용자 프로파일을 포함합니다(해당 경우). 프로그램은 메시지 변환을 지원하지 않는 목적지 큐 관리자에 메시지를 송신하는 메시지 채널 에이전트일 수 있었습니다. 큐 관리자의 환경에서 실행되지 않기 때문에 엑시트는 큐 관리자의 무결성에 타격을 주지 않습니다.
10. 엑시트에 사용할 수 있는 유일한 *MQI* 호출은 *MQXCNCV*입니다. 다른 *MQI* 호출을 사용하려고 하지만 오류 코드 *RC2219* 또는 다른 예측 불가능한 오류가 표시되면서 실패합니다.
11. 큐 관리자에서 *MQCONVX*라는 시작점을 제공하지 않았습니다. 모든 환경에서 요구사항은 아니지만, 엑시트의 이름이 형식 이름과 같아야 합니다(*MQMD*의 *MDFMT* 필드에 있는 이름).

## 매개변수

*MQCONVX* 호출의 매개변수는 다음과 같습니다.

### **MQDXP (MQDXP) - 입출력(I/O)**

데이터 변환 엑시트 매개변수 블록.

이 구조는 엑시트 호출과 관련된 정보를 포함합니다. 엑시트는 이 구조에 정보를 설정하여 변환 출력을 표시합니다. 이 구조의 필드에 대한 자세한 정보는 [1374 페이지의 『IBM i의 MQDXP\(데이터 변환 엑시트 매개변수\)』](#)의 내용을 참조하십시오.

### **MQMD (MQMD) - 입출력(I/O)**

메시지 디스크립터.

엑시트에 대한 입력에서 변환이 수행되지 않는 경우 애플리케이션에 리턴되는 메시지 디스크립터입니다. 따라서 *INBUF*에 있는 변환되지 않은 메시지의 *MDFMT*, *MDENC* 및 *MDCSI*를 포함합니다.

**참고:** 엑시트에 전달된 *MQMD* 매개변수는 항상 엑시트를 호출한 큐 관리자에 지원되는 *MQMD*의 최신 버전입니다. 엑시트가 각기 다른 환경 간에 이동 가능한 경우, 엑시트가 *MQMD*의 *MDVER* 필드를 검사하여 엑시트가 액세스해야 하는 필드가 구조에 있는지 확인해야 합니다.

IBM i에서 엑시트에 버전-2 *MQMD*가 전달됩니다.

변환에 성공한 경우 출력에서 엑시트는 *MDENC* 및 *MDCSI* 필드를 애플리케이션이 요청하는 값으로 변경해야 합니다. 이러한 변경사항이 애플리케이션에 다시 반영됩니다. 엑시트로 인한 구조의 변경사항은 무시됩니다. 애플리케이션에 반영되지 않습니다.

엑시트가 MQDXP 구조의 DXRES 필드에 XROK를 리턴하지만 메시지 디스크립터에서 MDENC 또는 MDCSI 필드를 변경하지 않는 경우, 큐 관리자는 해당 필드에 대해 엑시트 입력에서 MQDXP 구조의 관련 필드에 있는 값을 리턴합니다.

### INLEN(10자리 부호 있는 정수) - 입력

INBUF의 길이(바이트).

입력 버퍼 INBUF의 길이이고, 엑시트에 의해 처리되는 바이트 수를 지정합니다. INLEN은 변환 전 메시지 데이터의 길이 및 MQGET 호출에서 애플리케이션이 제공한 버퍼의 길이 중 작은 값입니다.

값은 항상 0보다 큽니다.

### INBUF(1바이트 비트 문자열 x INLEN) - 입력

변환되지 않은 메시지가 있는 버퍼.

여기에는 변환 전 메시지 데이터가 포함됩니다. 엑시트가 데이터를 변환할 수 없으면 큐 관리자는 엑시트가 완료된 이후 이 버퍼의 콘텐츠를 애플리케이션으로 리턴합니다.

참고: 엑시트가 INBUF을 대체하지 않아야 합니다. 이 매개변수가 대체되면 결과가 정의되지 않습니다.

### OUTLEN(10자리 부호 있는 정수) - 출력

OUTBUF의 길이(바이트).

출력 버퍼 OUTBUF의 길이이며, MQGET 호출에서 애플리케이션이 제공하는 버퍼의 길이와 같습니다.

값은 항상 0보다 큽니다.

### OUTBUF(1바이트 비트 문자열 x OUTLEN) - 출력

변환된 메시지가 있는 버퍼.

엑시트의 출력에서 변환이 성공한 경우 (MQDXP 매개변수의 DXRES 필드에 있는 X대한민국값으로 표시됨), OUTBUF 는 요청된 표시로 애플리케이션에 전달될 메시지 데이터를 포함합니다. 변환에 성공하지 못하면 엑시트가 이 버퍼에 수행한 변경사항이 무시됩니다.

## RPG 호출(ILE)

```
C*.1.....2.....3.....4.....5.....6.....7..
C          CALLP      exitname(MQDXP : MQMD : INLEN :
C                               INBUF : OUTLEN : OUTBUF)
```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```
D*.1.....2.....3.....4.....5.....6.....7..
Dexitname      PR          EXTPROC('exitname')
D* Data-conversion exit parameter block
D MQDXP                44A
D* Message descriptor
D MQMD                  364A
D* Length in bytes of INBUF
D INLEN                 10I 0 VALUE
D* Buffer containing the unconverted message
D INBUF                 *   VALUE
D* Length in bytes of OUTBUF
D OUTLEN                10I 0 VALUE
D* Buffer containing the converted message
D OUTBUF                *   VALUE
```

제품별 프로그래밍 인터페이스의 끝

## 사용자 엑시트, API 엑시트 및 설치 가능 서비스 참조

사용자 엑시트, API 엑시트 및 설치 가능한 서비스 애플리케이션 개발에 유용한 이 섹션의 정보를 사용하십시오.

- [1387 페이지의 『MQIEP 구조』](#)
- [1390 페이지의 『데이터 변환 엑시트 참조』](#)
- [1394 페이지의 『MQ\\_PUBLISH\\_EXIT - 발행 엑시트』](#)
- [1402 페이지의 『채널 엑시트 호출 및 데이터 구조』](#)
- [1487 페이지의 『API 엑시트 참조』](#)
- [1546 페이지의 『설치 가능 서비스 인터페이스 참조 정보』](#)

#### 관련 개념

[사용자 엑시트, API 엑시트 및 IBM MQ 설치 가능 서비스](#)

#### 관련 태스크

[큐 관리자 기능 확장](#)

## MQIEP 구조

MQIEP 구조는 엑시트를 작성할 수 있는 각 함수 호출의 시작점을 포함합니다.

### 필드

#### StrucId

유형: MQCHAR4 - 입력

구조 ID입니다. 값은 다음과 같습니다.

#### **MQIEP\_STRUC\_ID**

#### 버전

유형: MQLONG - 입력

구조 버전 번호입니다. 값은 다음과 같습니다.

#### **MQIEP\_VERSION\_1**

버전 1 구조 버전 번호.

#### **MQIEP\_CURRENT\_VERSION**

구조의 현재 버전.

#### StrucLength

유형: MQLONG

MQIEP 구조의 크기(바이트). 값은 다음과 같습니다.

#### **MQIEP\_LENGTH\_1**

#### 플래그

유형: MQLONG

함수 주소에 대한 정보를 제공합니다. 라이브러리가 스레드 처리되었는지 표시하는 플래그는 라이브러리가 클라이언트 라이브러리인지 또는 서버 라이브러리인지를 표시하는 플래그와 함께 사용할 수 있습니다.

라이브러리 정보를 지정하지 않으려면 다음 값을 사용합니다.

#### **MQIEPF\_NONE**

공유 라이브러리가 스레드인지 또는 비스레드인지 여부를 지정하려면 다음 값 중 하나를 사용합니다.

#### **MQIEPF\_NON\_THREADED\_LIBRARY**

비스레드 공유 라이브러리

#### **MQIEPF\_THREADED\_LIBRARY**

스레드 공유 라이브러리

공유 라이브러리가 클라이언트 공유 라이브러리인지 또는 서버 공유 라이브러리인지 여부를 지정하려면 다음 값 중 하나를 사용합니다.

#### **MQIEPF\_CLIENT\_LIBRARY**

클라이언트 공유 라이브러리

## **MQIEPF\_LOCAL\_LIBRARY**

서버 공유 라이브러리

### **예약됨**

유형: MQPTR

### **MQBACK\_Call**

유형: PMQ\_BACK\_CALL

MQBACK 호출의 주소.

### **MQBEGIN\_Call**

유형: PMQ\_BEGIN\_CALL

MQBEGIN 호출의 주소.

### **MQBUFMH\_Call**

유형: PMQ\_BUFMH\_CALL

MQBUFMH 호출의 주소.

### **MQCB\_Call**

유형: PMQ\_CB\_CALL

MQCB 호출의 주소.

### **MQCLOSE\_Call**

유형: PMQ\_CLOSE\_CALL

MQCLOSE 호출의 주소.

### **MQCMIT\_Call**

유형: PMQ\_CMIT\_CALL

MQCMIT 호출의 주소.

### **MQCONN\_Call**

유형: PMQ\_CONN\_CALL

MQCONN 호출의 주소.

### **MQCONNX\_Call**

유형: PMQ\_CONNX\_CALL

MQCONNX 호출의 주소.

### **MQCRTMH\_Call**

유형: PMQ\_CRTMH\_CALL

MQCRTMH 호출의 주소.

### **MQCTL\_Call**

유형: PMQ\_CTL\_CALL

MQCTL 호출의 주소.

### **MQDISC\_Call**

유형: PMQ\_DISC\_CALL

MQDISC 호출의 주소.

### **MQDLTMH\_Call**

유형: PMQ\_DLTMH\_CALL

MQDLTMH 호출의 주소.

### **MQDLTMP\_Call**

유형: PMQ\_DLTMP\_CALL

MQDLTMP 호출의 주소.



**MQGET\_Call**

유형: PMQ\_GET\_CALL

MQGET 호출의 주소.

**MQINQ\_Call**

유형: PMQ\_INQ\_CALL

MQINQ 호출의 주소.

**MQINQMP\_Call**

유형: PMQ\_INQMP\_CALL

MQINQMP 호출의 주소.

**MQMHBUF\_Call**

유형: PMQ\_MHBUF\_CALL

MQMHBUF 호출의 주소.

**MQOPEN\_Call**

유형: PMQ\_OPEN\_CALL

MQOPEN 호출의 주소.

**MQPUT\_Call**

유형: PMQ\_PUT\_CALL

MQPUT 호출의 주소.

**MQPUT1\_Call**

유형: PMQ\_PUT1\_CALL

MQPUT1 호출의 주소.

**MQSET\_Call**

유형: PMQ\_SET\_CALL

MQSET 호출의 주소.

**MQSETMP\_Call**

유형: PMQ\_SETMP\_CALL

MQSETMP 호출의 주소.

**MQSTAT\_Call**

유형: PMQ\_STAT\_CALL

MQSTAT 호출의 주소.

**MQSUB\_Call**

유형: PMQ\_SUB\_CALL

MQSUB 호출의 주소.

**MQSUBRQ\_Call**

유형: PMQ\_SUBRQ\_CALL

MQSUBRQ 호출의 주소.

**MQXCNVC\_Call**

유형: PMQ\_XCNVC\_CALL

MQXCNVC 호출의 주소.

**MQXCLWLN\_Call**

유형: PMQ\_XCLWLN\_CALL

MQXCLWLN 호출의 주소.

**MQXDX\_Call**

유형: PMQ\_XDX\_CALL

MQXDX 호출의 주소.

### MQXEP\_Call

유형: PMQ\_XEP\_CALL

MQXEP 호출의 주소.

### MQZEP\_Call

유형: PMQ\_ZEP\_CALL

MQZEP 호출의 주소.

## C 선언

```

struct tagMQIEP {
    MQCHAR4      StrucId;          /* Structure identifier */
    MQLONG       Version;         /* Structure version number */
    MQLONG       StrucLength;     /* Structure length */
    MQLONG       Flags;          /* Flags */
    MQPTR        Reserved;       /* Reserved */
    PMQ_BACK_CALL MQBACK_Call;   /* Address of MQBACK */
    PMQ_BEGIN_CALL MQBEGIN_Call; /* Address of MQBEGIN */
    PMQ_BUFMH_CALL MQBUFMH_Call; /* Address of MQBUFMH */
    PMQ_CB_CALL   MQCB_Call;     /* Address of MQCB */
    PMQ_CLOSE_CALL MQCLOSE_Call; /* Address of MQCLOSE */
    PMQ_CMIT_CALL MQCMIT_Call;   /* Address of MQCMIT */
    PMQ_CONN_CALL MQCONN_Call;   /* Address of MQCONN */
    PMQ_CONNX_CALL MQCONNX_Call; /* Address of MQCONNX */
    PMQ_CRTMH_CALL MQCRTMH_Call; /* Address of MQCRTMH */
    PMQ_CTL_CALL  MQCTL_Call;    /* Address of MQCTL */
    PMQ_DISC_CALL MQDISC_Call;   /* Address of MQDISC */
    PMQ_DLTMH_CALL MQDLTMH_Call; /* Address of MQDLTMH */
    PMQ_DLTMP_CALL MQDLTMP_Call; /* Address of MQDLTMP */
    PMQ_GET_CALL  MQGET_Call;    /* Address of MQGET */
    PMQ_INQ_CALL  MQINQ_Call;    /* Address of MQINQ */
    PMQ_INQMP_CALL MQINQMP_Call; /* Address of MQINQMP */
    PMQ_MHBUF_CALL MQMHBUF_Call; /* Address of MQMHBUF */
    PMQ_OPEN_CALL MQOPEN_Call;  /* Address of MQOPEN */
    PMQ_PUT_CALL  MQPUT_Call;    /* Address of MQPUT */
    PMQ_PUT1_CALL MQPUT1_Call;   /* Address of MQPUT1 */
    PMQ_SET_CALL  MQSET_Call;    /* Address of MQSET */
    PMQ_SETMP_CALL MQSETMP_Call; /* Address of MQSETMP */
    PMQ_STAT_CALL MQSTAT_Call;   /* Address of MQSTAT */
    PMQ_SUB_CALL  MQSUB_Call;    /* Address of MQSUB */
    PMQ_SUBBRQ_CALL MQSUBBRQ_Call; /* Address of MQSUBBRQ */
    PMQ_XCLWLN_CALL MQXCLWLN_Call; /* Address of MQXCLWLN */
    PMQ_XCNVC_CALL MQXCNVC_Call; /* Address of MQXCNVC */
    PMQ_XDX_CALL  MQXDX_Call;    /* Address of MQXDX */
    PMQ_XEP_CALL  MQXEP_Call;    /* Address of MQXEP */
    PMQ_ZEP_CALL  MQZEP_Call;    /* Address of MQZEP */
};

```



## 데이터 변환 엑시트 참조

z/OS의 경우, 어셈블러 언어로 데이터 변환 엑시트를 작성해야 합니다. 그 외 플랫폼의 경우, C 프로그래밍 언어를 사용하는 것이 좋습니다.

데이터 변환 엑시트 프로그램을 작성하는 데 도움이 되도록 다음 자원이 제공됩니다.

- 스켈레톤 소스 파일
- 문자 변환 호출
- 데이터 유형 구조에서 데이터 변환을 수행하는 코드 단편을 작성하는 유틸리티. 이 유틸리티는 C 입력만 받아 들입니다. z/OS에서는 어셈블러 코드를 생성합니다.

프로그램 작성 프로시저는 다음을 참조하십시오.






-  [IBM MQ for IBM i에 대한 데이터 변환 종료 프로그램 작성](#)
-  [IBM MQ for z/OS에 대한 데이터 변환 종료 프로그램 작성](#)

- IBM MQ for AIX or Linux 시스템에 대한 데이터 변환 종료 작성
- IBM MQ for Windows의 데이터 변환 엑시트 작성

## 스켈레톤 소스 파일

데이터 변환 엑시트 프로그램을 작성할 때 시작점으로 사용할 수 있습니다.

제공되는 파일이 [1391 페이지의 표 816](#)에 나와 있습니다.

표 816. 스켈레톤 소스 파일	
플랫폼	파일
 AIX	amqsvfc0.c
 IBM i	QMQMSAMP/QCSRC(AMQSVFC4)
 Linux	amqsvfc0.c
 Windows 시스템	amqsvfc0.c
 z/OS	CSQ4BAX8 ( <a href="#">1391 페이지의 『1』</a> ) CSQ4BAX9 ( <a href="#">1391 페이지의 『2』</a> ) CSQ4CAX9 ( <a href="#">1391 페이지의 『3』</a> )
<b>참고:</b> 1. MQXCVNC 호출을 설명합니다. 2. CICS를 제외한 모든 환경에 사용하도록 유틸리티가 생성한 코드 단편의 랩퍼입니다. 3. CICS 환경에 사용하도록 유틸리티가 생성한 코드 단편의 랩퍼입니다.	

## 문자 변환 호출

문자 메시지 데이터를 문자 간에 변환하려면 데이터 변환 엑시트 프로그램 내에서 MQXCNCV (문자 변환 호출) 호출을 사용하십시오. 특정 멀티바이트 문자 세트(예: UTF-16 문자 세트)의 경우 적절한 옵션을 사용해야 합니다.




엑시트 내에서 다른 MQI 호출을 작성할 수 없습니다. 이러한 호출을 작성하려고 하면 이유 코드 MQRC\_CALL\_IN\_PROGRESS가 표시되면서 실패합니다.

MQXCNCV 호출 및 적절한 옵션에 자세한 정보는 [880 페이지의 『MQXCNCV - 문자 변환』](#)의 내용을 참조하십시오.

## 변환-엑시트 코드 작성을 위한 유틸리티

변환-엑시트 코드 작성에 대해 자세히 알아보려면 이 정보를 사용하십시오.

변환-엑시트 코드를 작성하는 명령은 다음과 같습니다.

 IBM i	CVTMQMDTA (IBM MQ 데이터 유형 변환)
 ALW AIX, Linux, and Windows 시스템	crtmqcvx (IBM MQ 변환-엑시트 작성)
 z/OS	CSQUCVX

플랫폼별 명령이 데이터 변환 엑시트 프로그램에 사용하도록 데이터 유형 구조에서 데이터 변환을 수행하는 코드 단편을 생성합니다. 이 명령은 하나 이상의 C 언어 구조 정의가 포함된 파일을 가져옵니다. **z/OS** z/OS에서는 그런 다음 어셈블러 코드 단편과 변환 함수를 포함한 데이터 세트를 생성합니다. 그 외 플랫폼에서는 각 구조 정의를 변환하기 위해 C 함수를 포함한 파일을 생성합니다. z/OS에서는 유틸리티가 LE/370 런타임 라이브러리 SCEERUN에 액세스해야 합니다.

## z/OS에서 CSQUCVX 유틸리티 호출

**z/OS**

1392 페이지의 [그림 10](#)은 CSQUCVX 유틸리티를 호출하는 데 사용하는 JCL의 예를 보여줍니다.

```
//CVX EXEC PGM=CSQUCVX
//STEPLIB DD DISP=SHR,DSN=thlqua1.SCSQANLE
// DD DISP=SHR,DSN=thlqua1.SCSQLOAD
// DD DISP=SHR,DSN=le370qua1.SCEERUN
//SYSPRINT DD SYSOUT=*
//CSQUINP DD DISP=SHR,DSN=MY.MQSERIES.FORMATS(MSG1)
//CSQUOUT DD DISP=OLD,DSN=MY.MQSERIES.EXITS(MSG1)
```

그림 10. CSQUCVX 유틸리티를 호출하는 샘플 JCL

## z/OS 데이터 정의 명령문

**z/OS**

CSQUCVX 유틸리티에는 [1392 페이지의 표 817](#)에 표시된 다음 DD 이름을 포함한 DD 명령문이 필요합니다.

표 817. 데이터 정의 명령문 이름 및 설명	
DD 명령문	설명
SYSPRINT	보고 및 오류 메시지에 대한 데이터 세트 또는 인쇄 스펙 클래스를 지정합니다.
CSQUINP	변환할 데이터 구조의 정의를 포함한 분할 데이터 세트를 지정합니다.
CSQUOUT	변환 코드 단편이 작성되는 분할 데이터 세트를 지정합니다. 논리 레코드 길이(LRECL)는 80이고 레코드 형식(RECFM)은 FB이어야 합니다.

## AIX, Linux, and Windows 시스템의 오류 메시지

crtmqcvx 명령은 AMQ7953 ~ AMQ7970 범위에서 메시지를 리턴합니다.

이러한 메시지는 [메시지 및 이유 코드 IBM MQ](#) 메시지에 나열됩니다.

두 가지 주요 오류 유형은 다음과 같습니다.

- 처리를 계속할 수 없는 경우, 구문 오류와 같은 주요 오류입니다.  
입력 파일의 오류 행 번호를 알려주는 메시지가 화면에 표시됩니다. 출력 파일이 부분적으로 작성되었을 수 있습니다.
- 문제점이 발견되었지만 구조 구문 분석을 계속할 수 있다는 메시지가 표시되는 경우, 기타 오류입니다.  
출력 파일이 작성되고 발생한 문제점에 대한 오류 정보가 포함되어 있습니다. 문제점 수정을 위한 개입 없이 생성된 코드가 컴파일러에 허용되지 않도록 이 오류 정보에는 접두부 #error가 붙습니다.

## 올바른 구문

유틸리티의 입력 파일은 C 언어 구문을 따라야 합니다.

C에 익숙하지 않는 경우, 이 주제에 나온 [C](#) 예를 참조하십시오.

또한, 다음 규칙에 유의하십시오.

- typedef는 구조체(struct) 키워드 앞에서만 인식됩니다.
- 구조 선언에서는 구조 태그가 필요합니다.
- 빈 대괄호 []를 사용하여 메시지 끝에 가변 길이 배열 또는 문자열을 지정할 수 있습니다.
- 문자열의 다차원 배열은 지원되지 않습니다.
- 인식되는 추가 데이터 유형은 다음과 같습니다.

- MQBOOL
- MQBYTE
- MQCHAR
- MQFLOAT32
- MQFLOAT64
- MQSHORT
- MQLONG
- MQINT8
- MQUINT8
- MQINT16
- MQUINT16
- MQINT32
- MQUINT32
- MQINT64
- MQUINT64

MQCHAR 필드는 변환된 코드 페이지이지만, MQBYTE, MQINT8 및 MQUINT8은 변환 없이 유지됩니다. 인코딩이 다르면 MQSHORT, MQLONG, MQINT16, MQUINT16, MQINT32, MQUINT32, MQINT64, MQUINT64, MQFLOAT32, MQFLOAT64 및 MQBOOL이 그에 맞게 변환됩니다.

- 다음 데이터 유형은 사용하지 마십시오.

- double
- 포인터
- 비트 필드

이는 변환 엑시트 코드를 작성하는 유틸리티가 이러한 데이터 유형의 변환 기능을 제공하지 않기 때문입니다. 이 문제를 극복하려면 사용자 고유의 루틴을 작성하고 엑시트에서 호출합니다.

기타 유의 사항:

- 입력 데이터 세트에 순서 번호를 사용하지 마십시오.
- 고유의 변환 루틴을 제공하려는 필드가 있으면 이를 MQBYTE로 선언하고 생성된 CMQXCFBA 매크로를 고유의 변환 코드로 바꾸십시오.

## C 예

```
struct TEST { MQLONG    SERIAL_NUMBER;  
             MQCHAR    ID[5];  
             MQINT16   VERSION;  
             MQBYTE    CODE[4];  
             MQLONG    DIMENSIONS[3];  
             MQCHAR    NAME[24];  
             } ;
```

이것은 다른 프로그래밍 언어에서 다음 선언에 해당합니다.

## COBOL

```
10 TEST.  
15 SERIAL-NUMBER PIC S9(9) BINARY.  
15 ID PIC X(5).  
15 VERSION PIC S9(4) BINARY.  
* CODE IS NOT TO BE CONVERTED  
15 CODE PIC X(4).  
15 DIMENSIONS PIC S9(9) BINARY OCCURS 3 TIMES.  
15 NAME PIC X(24).
```

## System/390

```
TEST EQU *  
SERIAL_NUMBER DS F  
ID DS CL5  
VERSION DS H  
CODE DS XL4  
DIMENSIONS DS 3F  
NAME DS CL24
```

## PL/I

### z/OS에서만 지원됨

```
DCL 1 TEST,  
2 SERIAL_NUMBER FIXED BIN(31),  
2 ID CHAR(5),  
2 VERSION FIXED BIN(15),  
2 CODE CHAR(4), /* not to be converted */  
2 DIMENSIONS(3) FIXED BIN(31),  
2 NAME CHAR(24);
```

## MQ\_PUBLISH\_EXIT - 발행 엑시트

MQ\_PUBLISH\_EXIT 호출은 구독자에게 전달되는 메시지를 검사하고 대체할 수 있습니다.

### 목적

다음과 같이 발행 엑시트를 사용하여 구독자에게 전달되는 메시지를 검사 및 대체할 수 있습니다.

- 각 구독자에게 발행된 메시지의 콘텐츠 조사
- 각 구독자에게 발행된 메시지의 콘텐츠 수정
- 메시지를 넣은 큐 대체
- 구독자에 대한 메시지 전달 중지

IBM MQ for z/OS에서는 이 엑시트를 사용할 수 없습니다.

### 구문

**MQ\_PUBLISH\_EXIT** (*ExitParms*, *PubContext*, *SubContext*)

### 매개변수

#### **ExitParms (MQPSXP) - Input/Output**

*ExitParms*에는 엑시트 호출에 대한 정보가 포함되어 있습니다.

#### **PubContext (MQPBC) - Input**

*PubContext*에는 발행물의 발행자에 대한 컨텍스트 정보가 포함되어 있습니다.

#### **SubContext (MQSBC) - Input/Output**

*SubContext*에는 발행물을 수신하는 구독자에 대한 컨텍스트 정보가 포함되어 있습니다.

## MQPSXP - 발행 엑시트 데이터 구조

MQPSXP 구조는 발행 엑시트로 전달되거나 발행 엑시트에서 리턴되는 정보를 설명합니다.

1395 페이지의 표 818에 구조의 필드가 요약되어 있습니다.

표 818. MQPSXP의 필드	
필드	설명
<u>StrucID</u>	구조 ID
<u>Version</u>	구조 버전 번호
<u>ExitId</u>	호출되는 엑시트의 유형
<u>ExitReason</u>	엑시트를 호출하는 이유
<u>ExitResponse</u>	엑시트의 응답
<u>ExitResponse2</u>	엑시트로부터의 2차 응답
<u>Feedback</u>	피드백 코드
<u>ExitUserArea</u>	엑시트 사용자 영역
<u>ExitData</u>	엑시트 데이터
<u>QMGrName</u>	로컬 큐 관리자의 이름
<u>Hconn</u>	연결 핸들
<u>MsgDescPtr</u>	메시지 디스크립터(MQMD)의 주소
<u>MsgHandle</u>	메시지 특성에 대한 핸들(MQHMSG)
<u>MsgInPtr</u>	입력 메시지의 주소
<u>MsgInLength</u>	입력 메시지의 길이
<u>MsgOutPtr</u>	출력 메시지의 주소
<u>MsgOutLength</u>	출력 메시지의 길이
<u>pEntryPoints</u>	MQIEP 구조의 주소

### 필드

#### **StrucID (MQCHAR4)**

*StrucID*는 구조 ID입니다. 값은 다음과 같습니다.

##### **MQPSXP\_STRUCID**

MQPSXP\_STRUCID는 발행 엑시트 매개변수 구조의 ID입니다. C 프로그램 언어의 경우 MQPSXP\_STRUC\_ID\_ARRAY 상수도 정의됩니다. MQPSXP\_STRUC\_ID와 값이 동일하지만 문자열이 아닌 문자 배열입니다.

*StrucID*는 엑시트에 대한 입력 필드입니다.

#### **Version (MQLONG)**

*Version*은 구조 버전 번호입니다. 값은 다음과 같습니다.

##### **MQPSXP\_VERSION\_1**

MQPSXP\_VERSION\_1은 버전 1 발행 엑시트 매개변수 구조입니다. MQPSXP\_CURRENT\_VERSION 상수도 동일한 값으로 정의됩니다.

*Version*은 엑시트에 대한 입력 필드입니다.

#### **ExitId (MQLONG)**

*ExitId*는 호출되는 엑시트의 유형입니다. 값은 다음과 같습니다.

## **MQXT\_PUBLISH\_EXIT**

발행 엑시트입니다.

*ExitId*는 엑시트에 대한 입력 필드입니다.

### **ExitReason (MQLONG)**

*ExitReason*은 엑시트를 호출하는 이유입니다. 가능한 값은 다음과 같습니다.

#### **MQXR\_INIT**

초기화를 위해 이 연결에 대한 엑시트가 호출됩니다. 이 엑시트는 필요한 자원(예: 주 기억장치)을 확보하여 초기화할 수 있습니다.

#### **MQXR\_TERM**

엑시트가 중지될 예정이므로 이 연결에 대한 엑시트가 호출됩니다. 엑시트는 초기화된 이후에 확보한 자원(예: 주 기억장치)을 모두 비워야 합니다.

#### **MQXR\_PUBLICATION**

구독자의 메시지 큐로 발행물을 넣기 전에 큐 관리자에서 엑시트를 호출합니다. 이 엑시트는 메시지를 변경하며 메시지에 큐를 넣거나 발행물을 정지하지 않습니다.

*ExitReason*는 엑시트에 대한 입력 필드입니다.

### **ExitResponse (MQLONG)**

처리를 계속하는 방법을 지정하려면 엑시트에서 *ExitResponse*를 설정합니다. *ExitResponse*는 다음 값 중 하나입니다.

#### **MQXCC\_OK**

처리를 정상적으로 계속하려면 MQXCC\_OK를 설정합니다. *ExitReason* 값에 대한 응답으로 MQXCC\_OK를 설정합니다.

*ExitReason*의 값이 MQXR\_PUBLICATION인 경우 MQSBC 구조의 *DestinationQName* 및 *DestinationQMGrName* 필드는 메시지가 송신될 목적지를 식별합니다.

#### **MQXCC\_FAILED**

발행 조작을 중지하려면 MQXCC\_FAILED를 설정합니다. 완료 코드 MQCC\_FAILED 및 이유 코드 2557(09FD)(RC2557): MQRC\_PUBLISH\_EXIT\_ERROR는 엑시트에서 리턴 시 설정됩니다.

#### **MQXCC\_SUPPRESS\_FUNCTION**

메시지의 정상 처리를 중지하려면 MQXCC\_SUPPRESS\_FUNCTION을 설정합니다. *ExitReason*의 값이 MQXR\_PUBLICATION인 경우에만 MQXCC\_SUPPRESS\_FUNCTION을 설정하십시오.

메시지의 메시지 디스크립터에서 *Report* 필드의 MQRO\_DISCARD\_MSG 옵션에 따라 큐 관리자가 계속 메시지를 처리합니다.

- MQRO\_DISCARD\_MSG 옵션이 지정되지 않은 경우 메시지가 구독자에게 전달되지 않습니다.
- MQRO\_DISCARD\_MSG 옵션이 지정되지 않은 경우 메시지를 데드-레터 큐에 넣습니다. 데드-레터 큐가 없거나 데드-레터 큐에 메시지를 넣을 수 없는 경우 발행물이 구독자에게 전달되지 않습니다. 다른 구독자에게 발행물이 전달되는 것은 PMSGDLV 및 NPMSGDLV 토픽 오브젝트 속성 값에 따라 달라집니다. 이러한 속성에 대한 자세한 정보는 [DEFINE TOPIC](#) 명령의 매개변수 설명을 참조하십시오.

*ExitResponse*는 엑시트의 출력 필드입니다.

### **ExitResponse2 (MQLONG)**

*ExitResponse2*는 나중에 사용할 수 있도록 예약되었습니다.

### **Feedback (MQLONG)**

*Feedback*은 엑시트가 *ExitResponse*에서 MQXCC\_SUPPRESS\_FUNCTION을 리턴하는 경우에 사용되는 피드백 코드입니다.

엑시트에 대한 입력에서 *Feedback* 값은 항상 MQFB\_NONE입니다. 엑시트가 MQXCC\_SUPPRESS\_FUNCTION을 리턴하는 경우 큐 관리자가 데드-레터 큐에 메시지를 넣을 때 메시지에 사용되는 값으로 *Feedback*을 설정합니다. 엑시트의 리턴에서 *Feedback*의 원래 값이 MQFB\_NONE인 경우 큐 관리자는 *Feedback*을 MQFB\_STOPPED\_BY\_PUBSUB\_EXIT로 설정합니다.

*Feedback*은 엑시트에 대한 입출력(I/O) 필드입니다.



### **ExitUserArea (MQBYTE16)**

*ExitUserArea*는 엑시트에 사용할 수 있는 필드입니다. 각 연결마다 별도의 *ExitUserArea*가 있습니다. *ExitUserArea*의 길이는 MQ\_EXIT\_USER\_AREA\_LENGTH에서 제공됩니다.

*ExitReason* 필드는 엑시트를 처음 호출할 때 MQXR\_INIT 값을 갖습니다. *ExitUserArea*는 연결을 위해 처음 엑시트를 호출할 때 MQXUA\_NONE으로 초기화됩니다. 이후의 *ExitUserArea* 변경사항은 여러 엑시트 호출에서 유지됩니다.

*ExitUserArea*은 엑시트에 대한 입출력(I/O) 필드입니다.

### **ExitData (MQCHAR32)**

*ExitData*는 큐 관리자의 초기화 파일에 있는 스탠자의 **PublishExitData** 매개변수로 정의되는 고정 엑시트 데이터입니다. 필드의 전체 길이에 맞게 데이터에 공백을 채워 넣습니다. 초기화 파일에 고정 엑시트 데이터가 정의되지 않으면 *ExitData*는 공백입니다. *ExitData*의 길이는 MQ\_EXIT\_DATA\_LENGTH에서 제공됩니다.

*ExitData*는 엑시트에 대한 입력 필드입니다.

### **QMgrName (MQCHAR48)**

*QMgrName*은 로컬 큐 관리자의 이름입니다. 이름은 필드의 전체 길이에 맞게 공백으로 채워집니다. 이 필드의 길이는 MQ\_Q\_MGR\_NAME\_LENGTH에서 제공합니다.

*QMgrName*는 엑시트에 대한 입력 필드입니다.

### **Hconn (MQHCONN)**

*Hconn*은 큐 관리자에 대한 연결을 표시하는 핸들입니다. 메시지 특성에 대해 작업하는 경우에만 *Hconn*을 MQSETMP, MQINQMMP 또는 MQDLTMP 메시지 특성 함수 호출에 대한 매개변수로 사용하십시오.

*Hconn*는 엑시트에 대한 입력 필드입니다.

### **MsgDescPtr (PMQMD)**

*MsgDescPtr*은 처리되는 메시지의 메시지 디스크립터(MQMD) 주소이고 MQPUT 호출에서 리턴된 MQMD의 사본입니다. 엑시트는 메시지 디스크립터 콘텐츠를 변경할 수 있습니다. 메시지 디스크립터 콘텐츠를 변경할 때는 항상 주의해야 합니다. 특히 MQSBC 구조의 *SubType* 필드 값이 MQSUBTYPE\_PROXY인 경우 메시지 디스크립터의 *CorrelId* 필드를 변경해서는 안 됩니다.

*ExitReason*이 MQXR\_INIT 또는 MQXR\_TERM이면 메시지 디스크립터가 엑시트에 전달되지 않습니다. 이 경우, *MsgDescPtr*은 널 포인터입니다.

*MsgDescPtr*는 엑시트에 대한 입력 필드입니다.

### **MsgHandle (MQHMSG)**

*MsgHandle*은 메시지 특성에 대한 핸들입니다. 메시지 특성에 대해 작업하는 경우에만 *MsgHandle*을 MQSETMP, MQINQMMP 또는 MQDLTMP 메시지 특성 함수 호출과 함께 사용하십시오.

*MsgHandle*는 엑시트에 대한 입력 필드입니다.

### **MsgInPtr (PMQVOID)**

*MsgInPtr*은 입력 메시지 데이터의 주소입니다. *MsgInPtr*에 의해 처리되는 버퍼의 콘텐츠는 엑시트로 수정할 수 있습니다. *MsgOutPtr*의 내용을 참조하십시오.

*MsgInPtr*는 엑시트에 대한 입력 필드입니다.

### **MsgInLength (MQLONG)**

*MsgInLength*는 엑시트로 전달되는 메시지 데이터의 길이(바이트)입니다. 데이터 주소는 *MsgInPtr*에서 제공됩니다.

*MsgInLength*는 엑시트에 대한 입력 필드입니다.

### **MsgOutPtr (PMQVOID)**

*MsgOutPtr*은 엑시트에서 리턴되는 메시지 데이터가 포함된 버퍼의 주소입니다. 엑시트 입력에서 *MsgOutPtr*은 널입니다. 엑시트의 리턴에서 값이 계속 널인 경우 큐 관리자는 *MsgInLength*로 지정된 길이로 *MsgInPtr*에 지정된 메시지를 송신합니다.

엑시트가 메시지 데이터를 수정하는 경우 다음과 같은 프로시저 중 하나를 사용하십시오.

- 데이터 길이가 변경되지 않은 경우 *MsgInPtr*을 통해 지정된 버퍼에서 데이터가 수정될 수 있습니다. 이 경우 *MsgOutPtr* 및 *MsgOutLength*를 변경하지 마십시오.
- 데이터 길이가 원래 길이보다 짧은 경우 *MsgInPtr*을 통해 지정된 버퍼에서 데이터가 수정될 수 있습니다. 이 경우 *MsgOutPtr*은 입력 메시지 버퍼의 주소로 설정하고, *MsgOutLength*는 새 메시지 데이터 길이로 설정해야 합니다.
- 수정된 데이터가 원래 데이터보다 길거나 원래 데이터의 길이와 동일한 경우 엑시트는 새 메시지 버퍼를 확보해야 합니다. 수정된 데이터를 복사하십시오. *MsgOutPtr*을 새 버퍼 주소로 설정하고 *MsgOutLength*를 새 메시지 데이터 길이로 설정합니다. 이 엑시트는 다음에 엑시트가 호출될 때 *MsgOutPtr*을 통해 처리된 버퍼를 비웁니다.

**참고:** *MsgOutPtr*은 이전에 확보된 메시지 버퍼의 주소가 아니라 항상 엑시트에 대한 입력에서 널 포인터입니다. 이전에 확보된 버퍼를 비우려면 엑시트는 해당 주소 및 길이를 저장해야 합니다. *ExitUserArea*에, 또는 *ExitUserArea*에 해당 주소가 저장되어 있는 제어 블록에 정보를 저장하십시오.

*MsgOutPtr*은 엑시트에 대한 입출력(I/O) 필드입니다.

### MsgOutLength (MQLONG)

*MsgOutLength*는 엑시트에서 리턴된 메시지 데이터의 길이 (바이트)입니다. 엑시트에 대한 입력에서 이 필드는 항상 0입니다. *MsgOutPtr*이 널인 경우 엑시트의 리턴에서 이 필드는 무시됩니다. 메시지 데이터 수정에 대한 자세한 정보는 *MsgOutPtr*을 참조하십시오.

*MsgOutLength*은 엑시트에 대한 입출력(I/O) 필드입니다.

### pEntryPoints (PMQIEP)

*pEntryPoints*는 MQI 및 DCI 호출을 작성할 수 있는 MQIEP 구조의 주소입니다.

## C 언어 선언 - MQPSXP

```
typedef struct tagMQPSXP {
    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;   /* Reserved */
    MQLONG       Feedback;        /* Feedback code */
    MQBYTE16     ExitUserArea;    /* Exit user area */
    MQCHAR32     ExitData;        /* Exit data */
    MQCHAR48     QMgrName;        /* Name of local queue manager */
    MQHCONN      Hconn;          /* Connection handle */
    MQHMSG       MsgHandle;       /* Handle to message properties */
    PMQMD        MsgDescPtr;      /* Address of message descriptor */
    PMQVOID      MsgInPtr;        /* Address of input message data */
    MQLONG       MsgInLength;     /* Length of input message data */
    PMQVOID      MsgOutPtr;       /* Address of output message data */
    MQLONG       MsgOutLength;    /* Length of output message data */
    /* Ver:1 */
    PMQIEP       pEntryPoints;    /* Address of the MQIEP structure */
    /* Ver:2 */
} MQPSXP;
```

## MQPBC - 발행물 엑시트 데이터 구조

MQPBC 구조는 발행 엑시트로 전달되는, 발행물의 발행자와 관련된 컨텍스트 정보를 포함합니다.

1398 페이지의 표 819에 구조의 필드가 요약되어 있습니다.

표 819. MQPBC의 필드	
필드	설명
<i>StrucID</i>	구조 ID

표 819. MQPBC의 필드 (계속)	
필드	설명
<u>Version</u>	구조 버전 번호
<u>PubTopicString</u>	발행 토픽 문자열
<u>MsgDescPtr</u>	메시지 디스크립터(MQMD)의 주소

## 필드

### StrucID (MQCHAR4)

StrucID는 구조 ID입니다. 값은 다음과 같습니다.

#### MQPBC\_STRUCID

MQPBC\_STRUCID는 발행물 컨텍스트 구조의 ID입니다. C 프로그래밍 언어의 경우, MQPBC\_STRUC\_ID\_ARRAY 상수도 정의됩니다. MQPBC\_STRUC\_ID와 값이 동일하지만 문자열이 아닌 문자 배열입니다.

StrucID는 엑시트에 대한 입력 필드입니다.

### Version (MQLONG)

Version은 구조 버전 번호입니다. 값은 다음과 같습니다.

#### MQPBC\_VERSION\_1

MQPBC\_VERSION\_1은 버전 1 발행 엑시트 매개변수 구조입니다.

#### MQPBC\_VERSION\_2

MQPBC\_VERSION\_2는 버전 2 발행 엑시트 매개변수 구조입니다. MQPBC\_CURRENT\_VERSION 상수도 동일한 값으로 정의됩니다.

Version는 엑시트에 대한 입력 필드입니다.

### PubTopicString (MQCHARV)

PubTopicString은 발행되는 토픽 문자열입니다.

PubTopicString는 엑시트에 대한 입력 필드입니다.

### MsgDescPtr (PMQMD)

MsgDescPtr은 처리되는 메시지의 메시지 디스크립터(MQMD) 사본의 주소입니다.

MsgDescPtr는 엑시트에 대한 입력 필드입니다.

## C 언어 선언 - MQPBC

```
typedef struct tagMQPBC {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;          /* Structure version number */
    MQCHARV    PubTopicString;   /* Publish topic string */
    PMQMD      MsgDescPtr;       /* Address of message descriptor */
} MQPBC;
```

## MQSBC - 구독 컨텍스트 데이터 구조

MQSBC 구조는 발행 엑시트로 전달되는, 발행물 수신 구독자와 관련된 컨텍스트 정보를 포함합니다.

1399 페이지의 표 820에 구조의 필드가 요약되어 있습니다.

표 820. MQSBC의 필드	
필드	설명
<u>StrucID</u>	구조 ID

표 820. MQSBC의 필드 (계속)	
필드	설명
<i>Version</i>	구조 버전 번호
<i>DestinationQMgrName</i>	목적지 큐 관리자의 이름
<i>DestinationQName</i>	목적지 큐의 이름
<i>SubType</i>	구독의 유형
<i>SubOptions</i>	구독 옵션
<i>ObjectName</i>	오브젝트 이름
<i>ObjectString</i>	오브젝트 문자열
<i>SubTopicString</i>	구독 토픽 문자열
<i>SubName</i>	구독 이름
<i>SubId</i>	Subscription ID
<i>SelectionString</i>	선택된 문자열의 주소
<i>SubLevel</i>	구독 레벨
<i>PSPProperties</i>	발행/구독 특성

## 필드

### **StrucID (MQCHAR4)**

구조 ID입니다. 값은 다음과 같습니다.

#### **MQSBC\_STRUCID**

MQSBC\_STRUCID는 발행 엑시트 매개변수 구조의 ID입니다. C 프로그래밍 언어의 경우, MQSBC\_STRUC\_ID\_ARRAY 상수도 정의됩니다. MQSBC\_STRUC\_ID\_ARRAY는 MQSBC\_STRUC\_ID와 값이 동일하지만 문자열이 아닌 문자 배열입니다.

*StrucID*는 엑시트에 대한 입력 필드입니다.

### **Version (MQLONG)**

구조 버전 번호입니다. 값은 다음과 같습니다.

#### **MQSBC\_VERSION\_1**

버전 1 발행 엑시트 매개변수 구조입니다. MQSBC\_CURRENT\_VERSION 상수도 동일한 값으로 정의됩니다.

*Version*는 엑시트에 대한 입력 필드입니다.

### **DestinationQMgrName (MQCHAR48)**

*DestinationQMgrName*은 메시지가 송신되는 큐 관리자의 이름입니다. 이름은 필드의 전체 길이에 맞게 공백으로 채워집니다. 엑시트를 통해 이름을 대체할 수 있습니다. 이 필드의 길이는 MQ\_Q\_MGR\_NAME\_LENGTH에서 제공합니다.

*DestinationQMgrName*은 엑시트에 대한 입출력(I/O) 필드입니다. [참고](#)를 참조하십시오.

### **DestinationQName (MQCHAR48)**

*DestinationQName*은 메시지가 송신되는 큐의 이름입니다. 이름은 필드의 전체 길이에 맞게 공백으로 채워집니다. 엑시트를 통해 이름을 대체할 수 있습니다. 이 필드의 길이는 MQ\_Q\_NAME\_LENGTH에서 제공합니다.

*DestinationQName*은 엑시트에 대한 입출력(I/O) 필드입니다. [참고](#)를 참조하십시오.

**SubType (MQLONG)**

*SubType*은 구독의 작성 방식을 표시합니다. 올바른 값은 MQSUBTYPE\_API, MQSUBTYPE\_ADMIN 및 MQSUBTYPE\_PROXY입니다. [구독 상태\(응답\) 조회](#)의 내용을 참조하십시오.

*SubType*는 엑시트에 대한 입력 필드입니다.

**SubOptions (MQLONG)**

*SubOptions*는 구독 옵션입니다. 이 필드에 사용 가능한 값에 대한 설명은 [554 페이지의 『MQSD에 대한 옵션 \(MQLONG\)』](#)의 내용을 참조하십시오.

*SubOptions*는 엑시트에 대한 입력 필드입니다.

**ObjectName (MQCHAR48)**

*ObjectName*은 로컬 큐 관리자에 정의된 토픽 오브젝트의 이름입니다. 이 필드의 길이는 MQ\_TOPIC\_NAME\_LENGTH에서 제공합니다. 오브젝트 이름은 큐 관리자가 토픽 문자열을 연관시킨 관리 토픽 오브젝트의 이름입니다. 구독자가 구독의 일환으로 토픽 오브젝트를 제공했다라도 *ObjectName*은 다른 토픽 오브젝트일 수 있습니다. 구독과 토픽 오브젝트의 연관은 *SubTopicString*의 전체 해석에 따라 다릅니다.

*ObjectName*는 엑시트에 대한 입력 필드입니다.

**ObjectString (MQCHARV)**

*ObjectString*은 구독한 발행물의 전체 토픽 문자열입니다. 원래 구독 문자열에 와일드카드가 있으면 해석됩니다. [562 페이지의 『MQSD의 경우 ObjectString \(MQCHARV\)』](#)에 설명된 *ObjectString* 필드의 MQSD 구독과 다릅니다. 해당 항목은 와일드카드를 포함할 수 있으며 구독자가 제공한 오브젝트 이름이 없습니다.

*ObjectString*는 엑시트에 대한 입력 필드입니다.

**SubTopicString (MQCHARV)**

*SubTopicString*은 구독자가 제공하는 완전한 토픽 문자열입니다. *SubTopicString*은 토픽 오브젝트에 정의된 토픽 문자열과 토픽 문자열의 조합입니다. 구독자가 토픽 오브젝트, 토픽 문자열 또는 둘 다를 제공해야 합니다. 구독자가 토픽 문자열을 제공하면 와일드카드가 포함될 수 있습니다.

*SubTopicString*는 엑시트에 대한 입력 필드입니다.

**SubName (MQCHARV)**

*SubName*은 구독자가 제공한 구독 이름 또는 생성된 이름입니다.

*SubName*는 엑시트에 대한 입력 필드입니다.

**SubId (MQBYTE 24)**

*SubId*는 고유 내부 구독 ID입니다.

*SubId*는 엑시트에 대한 입력 필드입니다.

**SelectionString (MQCHARV)**

*SelectionString*은 토픽의 메시지를 구독할 때 사용하는 선택 기준입니다. [선택자를 참조](#)하십시오.

*SelectionString*는 엑시트에 대한 입력 필드입니다.

**SubLevel (MQLONG)**

*SubLevel*은 구독과 연관된 인터셉션 레벨입니다. 자세한 정보는 [565 페이지의 『MQSD의 경우 SubLevel \(MQLONG\)』](#)의 내용을 참조하십시오.

*SubLevel*는 엑시트에 대한 입력 필드입니다.

**PSPProperties (MQLONG)**

*PSPProperties*는 발행/구독 특성입니다. 발행/구독 관련 메시지 특성을 이 구독으로 송신된 메시지에 추가하는 방법을 지정합니다. 가능한 값은 MQPSPROP\_NONE, MQPSPROP\_COMPAT, MQPSPROP\_RFH2, MQPSPROP\_MSGPROP입니다. 이러한 값에 대한 설명은 [선택적 매개변수 \(구독 변경, 복사 및 작성\)](#)를 참조하십시오.

*PSPProperties*는 엑시트에 대한 입력 필드입니다.

**참고:** 권한 검사는 발행 엑시트에 전달되기 전에 *DestinationQMGrName* 및 *DestinationQName*의 원래 값에 대해서만 수행됩니다. 엑시트가 *DestinationQMGrName* 또는 *DestinationQName*을 변경하여 목적지 큐를 변경하는 경우 새 권한 검사가 수행되지 않습니다.

## C 언어 선언 - MQSBC

```
typedef struct tagMQSBC {
    MQCHAR4      StructId;          /* Structure identifier */
    MQLONG       Version;           /* Structure version number */
    MQCHAR48     DestinationQMGrName; /* Destination queue manager */
    MQCHAR48     DestinationQName;  /* Destination queue name */
    MQLONG       SubType;           /* Type of subscription */
    MQLONG       SubOptions;        /* Subscription options */
    MQCHAR48     ObjectName;        /* Object name */
    MQCHARV      ObjectString;      /* Object string */
    MQCHARV      SubTopicString;    /* Subscription topic string */
    MQCHARV      SubName;           /* Subscription name */
    MQBYTE24     SubId;             /* Subscription identifier */
    MQCHARV      SelectionString;   /* Subscription selection string */
    MQLONG       SubLevel;          /* Subscription level */
    MQLONG       PSPProperties;      /* Publish/subscribe properties */
} MQSBC;
```

## 채널 엑시트 호출 및 데이터 구조

이 주제 컬렉션에서는 특수 IBM MQ 호출 및 채널 엑시트 프로그램 작성 시 사용할 수 있는 데이터 구조에 대한 참조 정보를 제공합니다.

이 정보는 제품별 프로그래밍 인터페이스 정보입니다. 다음 프로그래밍 언어로 IBM MQ 사용자 엑시트를 작성할 수 있습니다.

표 821. IBM MQ 사용자 엑시트: 플랫폼 및 프로그래밍 언어	
플랫폼	프로그래밍 언어
IBM MQ for z/OS	어셈블러 및 C(z/OS C/C++ 프로그래밍 안내서에 설명된, 시스템 엑시트를 위한 C 시스템 프로그래밍 환경을 준수해야 함)
IBM MQ for IBM i	ILE C, ILE COBOL 및 ILE RPG
다른 모든 IBM MQ 플랫폼	C

또한 Java 및 JMS 애플리케이션에서만 사용하기 위해 Java 에서 사용자 엑시트를 작성할 수도 있습니다. 채널 엑시트를 작성하고 IBM MQ classes for Java와 함께 사용하는 데 대한 자세한 정보는 [IBM MQ classes for Java](#) 에서 [채널 엑시트 사용](#)을 참조하고 IBM MQ classes for JMS의 경우 [IBM MQ classes for JMS](#)에서 [채널 엑시트 사용](#)을 참조하십시오.

TAL 또는 Visual Basic에서는 IBM MQ 사용자 엑시트를 작성할 수 없습니다. 그러나, IBM MQ MQI client 프로그램에서 MQCONNX 호출에 사용할 수 있도록 MQCD 구조의 선언이 Visual Basic에서 제공됩니다.

이어지는 설명에서 대부분의 경우 매개변수는 크기가 고정되지 않은 배열 또는 문자열입니다. 이 매개변수에는 숫자 상수를 표시하기 위해 소문자 "n"이 사용됩니다. 해당 매개변수에 대한 선언이 코드화되면, "n"이 필요한 숫자 값으로 바뀌어야 합니다. 이 설명에 사용된 규칙에 관한 자세한 정보는 [234 페이지의 『기본 데이터 유형』](#)의 내용을 참조하십시오.

## 데이터 정의 파일

IBM MQ에서는 각 지원되는 프로그래밍 언어별로 데이터 정의 파일이 제공됩니다. 해당 파일에 대한 자세한 정보는 [복사, 헤더, 포함 및 모듈 파일을 참조하십시오](#).

## MQ\_CHANNEL\_EXIT - 채널 엑시트

MQ\_CHANNEL\_EXIT 호출에서는 메시지 채널 에이전트가 호출한 각 채널 엑시트로 전달되는 매개변수를 설명합니다.

큐 관리자에서 MQ\_CHANNEL\_EXIT라는 시작점을 제공하지 않으며, 채널 엑시트의 이름이 채널 정의 MQCD에 제공되므로 MQ\_CHANNEL\_EXIT 이름은 특별한 의미를 가지지 않습니다.

채널 엑시트는 5가지 유형이 있습니다.

- 채널 보안 엑시트
- 채널 메시지 엑시트
- 채널 송신 엑시트
- 채널 수신 엑시트
- 채널 메시지 재시도 엑시트

매개변수는 각 엑시트 유형마다 비슷하며 특별한 언급이 없으면 여기에 나온 설명이 모든 항목에 적용됩니다.

## 구문

**MQ\_CHANNEL\_EXIT** (*ChannelExitParms, ChannelDefinition, DataLength, AgentBufferLength, AgentBuffer, ExitBufferLength, ExitBufferAddr*)

## 매개변수

MQ\_CHANNEL\_EXIT 호출의 매개변수는 다음과 같습니다.

### ChannelExitParms (MQCXP) - 입출력(I/O)

채널 엑시트 매개변수 블록.

이 구조에는 엑시트 호출과 관련된 추가 정보가 들어 있습니다. 엑시트는 이 구조에 정보를 설정하여 MCA의 처리 방법을 표시합니다.

### ChannelDefinition (MQCD) - 입출력(I/O)

채널 정의.

이 구조는 채널의 동작을 제어하기 위해 관리자가 설정한 매개변수를 포함합니다.

### DataLength (MQLONG) - 입출력(I/O)

데이터 길이.

데이터는 엑시트의 유형에 따라 달라집니다.

- 채널 보안 엑시트의 경우, 엑시트가 호출될 때 *ExitReason*이 MQXR\_SEC\_MSG이면 이 매개변수는 *AgentBuffer* 필드에 보안 메시지의 길이를 포함합니다. 메시지가 없는 경우, 0입니다. *ExitResponse*을 MQXCC\_SEND\_SEC\_MSG 또는 MQXCC\_SEND\_AND\_REQUEST\_SEC\_MSG에 설정한 경우 엑시트는 송신할 임의 보안 메시지의 길이로 이 필드를 설정해야 합니다. 메시지 데이터는 *AgentBuffer* 또는 *ExitBufferAddr*에 있습니다.

보안 메시지의 내용은 전적으로 보안 엑시트에 따라 달라집니다.

- 채널 메시지 엑시트의 경우, 엑시트가 호출될 때 이 매개변수는 메시지의 길이(전송 큐 헤더 포함)를 포함합니다. 엑시트는 진행할 *AgentBuffer* 또는 *ExitBufferAddr*의 메시지 길이로 이 필드를 설정해야 합니다. 이 필드는 전송 큐 헤더(MQXQH)의 길이보다 같거나 커야 합니다.
- 채널 송신 또는 채널 수신 엑시트의 경우, 엑시트가 호출될 때 이 매개변수는 전송의 길이를 포함합니다. 엑시트는 진행할 *AgentBuffer* 또는 *ExitBufferAddr*의 전송 길이로 이 필드를 설정해야 합니다.

보안 엑시트가 메시지를 송신하고 채널의 다른 쪽 끝에 보안 엑시트가 없거나 다른 쪽 끝이 MQXCC\_OK의 *ExitResponse*를 설정하는 경우, MQXR\_SEC\_MSG 및 널 응답(*DataLength*=0)과 함께 시작 엑시트가 다시 호출됩니다.

### AgentBufferLength (MQLONG) - 입력

에이전트 버퍼의 길이.

이 매개변수는 호출 시 *DataLength*보다 클 수 있습니다.



채널 메시지, 송신 및 수신 엑시트의 경우, 호출 시 사용되지 않은 공간을 엑시트가 사용하여 데이터를 확장할 수 있습니다. 이를 수행한 경우에는 **DataLength** 매개변수가 엑시트에 의해 적절히 설정되어야 합니다.

C 프로그래밍 언어에서 이 매개변수는 주소로 전달됩니다.

### AgentBuffer (MQBYTE x AgentBufferLength) - 입출력(I/O)

에이전트 버퍼.

이 매개변수의 내용은 엑시트 유형에 따라 다릅니다.

- 채널 보안 엑시트의 경우, 엑시트 호출 시 *ExitReason*이 MQXR\_SEC\_MSG이면, 보안 메시지가 여기에 포함됩니다. 보안 메시지를 다시 송신하기 위해, 엑시트는 이 버퍼 또는 자체 버퍼(*ExitBufferAddr*)를 사용할 수 있습니다.
- 채널 메시지 엑시트의 경우, 엑시트 호출 시 이 매개변수에는 다음이 포함됩니다.

- 메시지 디스크립터(메시지의 컨텍스트 정보 포함)를 포함한 전송 큐 헤더(MQXQH), 바로 이어
- 메시지 데이터

메시지가 진행될 경우, 엑시트는 다음 중 하나를 수행할 수 있습니다.

- 버퍼의 내용을 그대로 둠
- 보관된 내용을 수정함(*DataLength*에 데이터의 새 길이를 리턴하며, 이 값이 *AgentBufferLength* 보다 크지 않아야 함)
- 필요한 부분을 변경하면서 내용을 *ExitBufferAddr*로 복사

엑시트가 전송 큐 헤더에 수행하는 변경사항은 검사하지 않지만, 수정이 잘못되면 메시지를 목적지에 넣지 못할 수 있습니다.

- 채널 송신 또는 수신 엑시트의 경우, 엑시트 호출 시 전송 데이터가 여기에 포함됩니다. 이 엑시트는 다음 중 하나를 수행할 수 있습니다.
  - 버퍼의 내용을 그대로 둠
  - 보관된 내용을 수정함(*DataLength*에 데이터의 새 길이를 리턴하며, 이 값이 *AgentBufferLength* 보다 크지 않아야 함)
  - 필요한 부분을 변경하면서 내용을 *ExitBufferAddr*로 복사
- 엑시트가 데이터의 처음 8바이트를 변경하지 않아야 합니다.

### ExitBufferLength (MQLONG) - 입출력(I/O)

엑시트 버퍼의 길이.

엑시트의 첫 호출 시, 이 매개변수는 0으로 설정됩니다. 그러면 호출 시 엑시트가 어떤 값을 전달하든 이 값은 다음 번에 호출할 때 엑시트에 표시됩니다. 값은 MCA에 사용되지 않습니다.

**참고:** 이 매개변수는 포인터 데이터 유형을 지원하지 않는 프로그래밍 언어로 작성된 엑시트에서는 사용하지 말아야 합니다.

### ExitBufferAddr (MQPTR) - 입출력(I/O)

엑시트 버퍼의 주소.

이 매개변수는 엑시트에 의해 관리되는 스토리지의 버퍼 주소에 대한 포인터입니다. 여기서 에이전트의 버퍼가 충분히 크지 않거나 또는 이렇게 하는 것이 엑시트에 더 편리하면 리턴 메시지 또는 전송 데이터(엑시트의 유형에 따라 다름)를 선택할 수 있습니다.

엑시트의 첫 호출 시, 엑시트로 전달된 주소는 널(null)입니다. 그러면 호출 시 엑시트가 어떤 주소를 전달하든 이 주소는 다음 번에 호출할 때 엑시트에 표시됩니다.

*ExitBufferAddr*이 널인 경우 사용되는 데이터는 *AgentBuffer* 매개변수에서 가져옵니다.

*ExitBufferAddr*이 널이 아닌 경우 사용되는 데이터는 *ExitBufferAddr* 매개변수가 지정하는 버퍼에서 가져옵니다.

**참고:** 이 매개변수는 포인터 데이터 유형을 지원하지 않는 프로그래밍 언어로 작성된 엑시트에서는 사용하지 말아야 합니다.



## C 호출

```
exitname (&ChannelExitParms, &ChannelDefinition,  
&DataLength, &AgentBufferLength, AgentBuffer,  
&ExitBufferLength, &ExitBufferAddr);
```

엑시트에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCXP  ChannelExitParms; /* Channel exit parameter block */  
MQCD   ChannelDefinition; /* Channel definition */  
MQLONG DataLength; /* Length of data */  
MQLONG AgentBufferLength; /* Length of agent buffer */  
MQBYTE AgentBuffer[n]; /* Agent buffer */  
MQLONG ExitBufferLength; /* Length of exit buffer */  
MQPTR  ExitBufferAddr; /* Address of exit buffer */
```

## COBOL 호출

```
CALL 'exitname' USING CHANNELEXITPARMS, CHANNELDEFINITION,  
DATALENGTH, AGENTBUFFERLENGTH, AGENTBUFFER,  
EXITBUFFERLENGTH, EXITBUFFERADDR.
```

엑시트에 전달된 매개변수는 다음과 같이 선언됩니다.

```
** Channel exit parameter block  
01 CHANNELEXITPARMS.  
COPY CMQCXPV.  
** Channel definition  
01 CHANNELDEFINITION.  
COPY CMQCDV.  
** Length of data  
01 DATALENGTH PIC S9(9) BINARY.  
** Length of agent buffer  
01 AGENTBUFFERLENGTH PIC S9(9) BINARY.  
** Agent buffer  
01 AGENTBUFFER PIC X(n).  
** Length of exit buffer  
01 EXITBUFFERLENGTH PIC S9(9) BINARY.  
** Address of exit buffer  
01 EXITBUFFERADDR POINTER.
```

## RPG 호출(ILE)

```
C*.1.....2.....3.....4.....5.....6.....7..  
C CALLP exitname(MQCXP : MQCD : DATLEN :  
C ABUFL : ABUF : EBUFL :  
C EBUF)
```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```
D*.1.....2.....3.....4.....5.....6.....7..  
Dexitname PR EXTPROC('exitname')  
D* Channel exit parameter block  
D MQCXP 160A  
D* Channel definition  
D MQCD 1328A  
D* Length of data  
D DATLEN 10I 0  
D* Length of agent buffer  
D ABUFL 10I 0  
D* Agent buffer  
D ABUF * VALUE  
D* Length of exit buffer  
D EBUFL 10I 0  
D* Address of exit buffer  
D EBUF *
```

## System/390 어셈블러 호출

```
CALL EXITNAME, (CHANNEEXITPARMS, CHANNELDEFINITION, DATALENGTH, X  
AGENTBUFFERLENGTH, AGENTBUFFER, EXITBUFFERLENGTH, X  
EXITBUFFERADDR)
```

엑시트에 전달된 매개변수는 다음과 같이 선언됩니다.

CHANNEEXITPARMS	CMQCPA	,	Channel exit parameter block
CHANNELDEFINITION	CMQCDA	,	Channel definition
DATALENGTH	DS	F	Length of data
AGENTBUFFERLENGTH	DS	F	Length of agent buffer
AGENTBUFFER	DS	CL(n)	Agent buffer
EXITBUFFERLENGTH	DS	F	Length of exit buffer
EXITBUFFERADDR	DS	F	Address of exit buffer

### 사용법 참고

1. 엑시트 제공자가 채널 엑시트에 의해 수행되는 기능을 정의합니다. 그러나 엑시트는 여기에 정의된 규칙 및 연관된 제어 블록 MQCXP에 정의된 규칙을 따라야 합니다.
2. 채널 엑시트로 전달된 **ChannelDefinition** 매개변수는 여러 가지 버전 중 하나일 수 있습니다. 자세한 정보는 MQCD 구조의 *Version* 필드를 참조하십시오.
3. *Version* 필드가 MQCD\_VERSION\_1보다 큰 값으로 설정된 MQCD 구조를 채널 엑시트가 수신하면, 엑시트는 MQCD의 *ConnectionName* 필드를 *ShortConnectionName* 필드보다 우선 사용해야 합니다.
4. 일반적으로 채널 엑시트는 메시지 데이터의 길이를 변경할 수 있습니다. 길이 변경은 엑시트가 메시지에 데이터를 추가하거나, 메시지에서 데이터를 제거하거나, 메시지를 압축 또는 암호화하는 결과로 일어날 수 있습니다. 그러나 메시지가 논리 메시지의 일부만을 포함한 세그먼트일 경우, 특별한 제한이 적용됩니다. 특히 보완적인 송신 및 수신 엑시트 조치의 결과로 메시지 길이에서 어떠한 변경도 없어야 합니다.

예를 들어, 송신 엑시트가 메시지를 압축하여 메시지를 줄일 수 있지만, 보완 수신 엑시트는 메시지 길이에 어떤 변경이 없도록 이를 압축 해제하여 메시지의 원래 길이를 복원해야 합니다.

세그먼트의 길이를 변경하면 메시지의 이후 세그먼트의 오프셋이 올바르게 않게 되고 이로 인해 세그먼트가 완전한 논리 메시지를 구성했는지 확인하는 큐 관리자의 기능에 문제가 생기므로 이러한 제한이 발생합니다.

### MQ\_CHANNEL\_AUTO\_DEF\_EXIT - 채널 자동 정의 엑시트

The MQ\_CHANNEL\_AUTO\_DEF\_EXIT 호출은 메시지 채널 에이전트가 호출한 채널 자동 정의 엑시트로 전달되는 매개변수를 설명합니다.

큐 관리자에서는 MQ\_CHANNEL\_AUTO\_DEF\_EXIT라는 시작점을 제공하지 않으며, 자동 정의 엑시트의 이름이 큐 관리자에 제공되므로 MQ\_CHANNEL\_AUTO\_DEF\_EXIT 이름은 특별한 의미를 가지지 않습니다.

### 구문

**MQ\_CHANNEL\_AUTO\_DEF\_EXIT (ChannelExitParms, ChannelDefinition)**

### 매개변수

MQ\_CHANNEL\_AUTO\_DEF\_EXIT 호출의 매개변수는 다음과 같습니다.

#### ChannelExitParms (MQCXP) - 입출력(I/O)

채널 엑시트 매개변수 블록.

이 구조에는 엑시트 호출과 관련된 추가 정보가 들어 있습니다. 엑시트는 이 구조에 정보를 설정하여 MCA의 처리 방법을 표시합니다.

#### ChannelDefinition (MQCD) - 입출력(I/O)

채널 정의.

이 구조에는 자동으로 작성된 채널의 동작을 제어하기 위해 관리자가 설정한 매개변수가 포함됩니다. 엑시트는 이 구조에 정보를 설정하여 관리자가 설정한 기본 동작을 수정합니다.

다음 MQCD 필드는 엑시트로 대체되지 않아야 합니다.

- *ChannelName*
- *ChannelType*
- *StrucLength*
- *Version*

다른 필드를 변경한 경우에는 엑시트가 설정한 값이 올바른 값이어야 합니다. 값이 올바르지 않으면, 오류 메시지가 오류 로그 파일에 기록되거나 콘솔에 표시됩니다(환경에 따라 적절하게).



**주의:** 채널 자동 정의(CHAD) 엑시트에 의해 작성된 자동 정의된 채널은 인증서 레이블을 설정할 수 없습니다. TLS 데이터 교환은 채널이 작성되는 시간에 발생하기 때문입니다. 인바운드 채널의 경우 CHAD 엑시트에 인증서 레이블을 설정하는 것은 아무런 효과가 없습니다.

## C 호출

```
exitname (&ChannelExitParms, &ChannelDefinition);
```

엑시트에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCXP ChannelExitParms; /* Channel exit parameter block */
MQCD ChannelDefinition; /* Channel definition */
```

## COBOL 호출

```
CALL 'exitname' USING CHANNELEXITPARMS, CHANNELDEFINITION.
```

엑시트에 전달된 매개변수는 다음과 같이 선언됩니다.

```
** Channel exit parameter block
01 CHANNELEXITPARMS.
   COPY CMQCXPV.
** Channel definition
01 CHANNELDEFINITION.
   COPY CMQCDV.
```

## RPG 호출(ILE)

```
C*..1.....2.....3.....4.....5.....6.....7..
C                                CALLP      exitname(MQCXP : MQCD)
```

호출에 대한 프로토타입 정의는 다음과 같습니다.

```
D*..1.....2.....3.....4.....5.....6.....7..
Dexitname          PR              EXTPROC('exitname')
D* Channel exit parameter block
D MQCXP                                160A
D* Channel definition
D MQCD                                1328A
```

## System/390 어셈블러 호출

```
CALL EXITNAME,(CHANNELEXITPARMS,CHANNELDEFINITION)
```

엑시트에 전달된 매개변수는 다음과 같이 선언됩니다.

```
CHANNELEXITPARMS  CMQXPA  , Channel exit parameter block
CHANNELDEFINITION CMQCDA  , Channel definition
```


## 사용법 참고

1. 엑시트 제공자가 채널 엑시트에 의해 수행되는 기능을 정의합니다. 그러나 엑시트는 여기에 정의된 규칙 및 연관된 제어 블록 MQCXP에 정의된 규칙을 따라야 합니다.
2. 채널 자동 정의 엑시트에 전달된 **ChannelExitParms** 매개변수는 MQCXP 구조입니다. 전달된 MQCXP의 버전은 엑시트가 실행되는 환경에 따라 다릅니다. 자세한 정보는 [1448 페이지의 『MQCXP - 채널 엑시트 매개변수』](#)에서 *Version* 필드의 설명을 참조하십시오.
3. 채널 자동 정의 엑시트에 전달된 **ChannelDefinition** 매개변수는 MQCD 구조입니다. 전달된 MQCD의 버전은 엑시트가 실행되는 환경에 따라 다릅니다. 자세한 정보는 [1409 페이지의 『MQCD - 채널 정의』](#)에서 *Version* 필드의 설명을 참조하십시오.

## MQXWAIT - 엑시트에서 대기

MQXWAIT 호출은 이벤트가 발생할 때까지 대기합니다. z/OS에서는 채널 엑시트에서만 사용할 수 있습니다.

MQXWAIT을 사용하면 채널 엑시트가 대기를 일으키는 어떤 작업을 수행하는 경우 발생할 수 있는 성능 문제점을 방지하는 데 도움이 됩니다. MQXWAIT이 대기하는 이벤트는 MVS ECB(이벤트 제어 블록)로 알 수 있습니다. ECB는 MQXWD 제어 블록 설명에 설명되어 있습니다.

 MQXWAIT 사용 및 채널 엑시트 프로그램 작성에 대한 자세한 정보는 [z/OS에서 채널 엑시트 프로그램 작성을 참조하십시오.](#)

## 구문

**MQXWAIT (Hconn, WaitDesc, CompCode, Reason)**

## 매개변수

MQXWAIT 호출의 매개변수는 다음과 같습니다.

### Hconn (MQHCONN) - 입력

연결 핸들입니다.

이 핸들은 큐 관리자에 대한 연결을 나타냅니다. Hconn의 값은 엑시트의 동일 또는 이전 호출에서 발행된 이전 MQCONN 호출에 의해 리턴되었습니다.

### WaitDesc (MQXWD) - 입출력(I/O)

대기 디스크립터.

이 매개변수는 대기하는 이벤트를 설명합니다. 이 구조의 필드에 대한 자세한 정보는 [1461 페이지의 『MQXWD - 엑시트 대기 디스크립터』](#)의 내용을 참조하십시오.

### CompCode(MQLONG) - 출력

완료 코드.

다음 코드 중 하나입니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_FAILED

호출에 실패했습니다.

### Reason(MQLONG) - 출력

CompCode을(를) 규정하는 이유 코드.

CompCode이(가) MQCC\_OK인 경우:

**MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

**MQRC\_ADAPTER\_NOT\_AVAILABLE**

(2204, X'89C') 어댑터를 사용할 수 없습니다.

**MQRC\_OPTIONS\_ERROR**

(2046, X'7FE') 옵션이 유효하지 않거나 일관적이지 않습니다.

**MQRC\_XWAIT\_CANCELED**

(2107, X'83B') MQXWAIT 호출이 취소됩니다.

**MQRC\_XWAIT\_ERROR**

(2108, X'83C') MQXWAIT 호출이 올바르지 않습니다.

## C 호출

```
MQXWAIT (Hconn, &WaitDesc, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONN  Hconn;      /* Connection handle */
MQXWD    WaitDesc;  /* Wait descriptor */
MQLONG   CompCode;  /* Completion code */
MQLONG   Reason;    /* Reason code qualifying CompCode */
```

## System/390 어셈블러 호출

```
CALL MQXWAIT,(HCONN,WAITDESC,COMPCODE,REASON)
```

매개변수를 다음과 같이 선언합니다.

```
HCONN      DS      F      Connection handle
WAITDESC   CMQXWDA ,      Wait descriptor
COMPCODE   DS      F      Completion code
REASON     DS      F      Reason code qualifying COMPCODE
```

## MQCD - 채널 정의

MQCD 구조는 채널 실행을 제어하는 매개변수를 포함합니다. 이 구조는 메시지 채널 에이전트(MCA)로부터 호출된 각 채널 엑시트에 전달됩니다.

채널 엑시트에 대한 자세한 정보는 1402 페이지의 『MQ CHANNEL EXIT - 채널 엑시트』의 내용을 참조하십시오. 이 주제에 나온 설명은 메시지 채널 및 MQI 채널 모두와 관련이 있습니다.

## 엑시트 이름 필드

엑시트가 호출되면 *SecurityExit*, *MsgExit*, *SendExit*, *ReceiveExit* 및 *MsgRetryExit*의 관련 필드에 현재 호출되는 엑시트의 이름이 포함됩니다. 이들 필드에 있는 이름의 의미는 MCA가 실행 중인 환경에 따라 달라집니다. 명시된 경우를 제외하고, 필드 내의 이름은 중간에 공백 없이 왼쪽으로 정렬됩니다. 필드 길이에 맞게 이름에 공백을 채워 넣습니다. 다음 설명에서, 대괄호([ ])는 옵션 정보를 나타냅니다.

## AIX and Linux

엑시트 이름은 동적으로 로드할 수 있는 모듈 또는 라이브러리의 이름으로, 뒤에 해당 라이브러리에 상주하는 함수의 이름이 붙습니다. 함수 이름은 괄호로 묶어야 합니다. 라이브러리 이름 앞에 선택적으로 디렉토리 경로를 지정할 수 있습니다.

```
[ path ] library ( function )
```

이름은 최대 128자로 제한됩니다.

### **z/OS**

LINK 또는 LOAD 매크로의 EP 매개변수에서 스펙에 유효한 로드 모듈의 엑시트 이름. 이름은 최대 8자로 제한됩니다.

### **Windows**

엑시트 이름은 동적-링크 라이브러리의 이름으로, 해당 라이브러리에 상주하는 함수의 이름이 접미부로 붙습니다. 함수 이름은 괄호로 묶어야 합니다. 라이브러리 이름 앞에 다음과 같이 선택적으로 디렉토리 경로 및 드라이브를 지정할 수 있습니다.

```
[d:][ path ] library ( function )
```

이름은 최대 128자로 제한됩니다.

### **IBM i**

엑시트 이름은 뒤에 10바이트 라이브러리 이름이 오는 10바이트 프로그램 이름입니다. 이름의 길이가 10바이트보다 짧으면, 각 이름 뒤에 공백을 채워 넣어 10바이트를 만듭니다. 라이브러리 이름은 완전한 이름이 필요한 경우인 채널 자동 정의 엑시트를 호출할 때를 제외하고는 \*LIBL이 될 수 있습니다.

## **채널 엑시트에서 MQCD 필드 변경**

채널 엑시트는 MQCD의 필드를 변경할 수 있습니다. 변경된 값은 MQCD에 남아 있으며 엑시트 체인의 나머지 엑시트 및 채널 인스턴스를 공유하는 대화로 전달됩니다. 변경된 MQCD 또한 채널의 지속 시간 동안 정상적인 처리를 위해 MCA에 사용됩니다.

다음 MQCD 필드는 엑시트로 대체되지 않아야 합니다.

- ChannelName
- ChannelType
- StrucLength
- 버전

### **관련 참조**

[1410 페이지의 『필드』](#)

이 주제는 MQCD 구조의 모든 필드를 나열하고 각 필드를 설명합니다.

[1435 페이지의 『C 선언』](#)

이 선언은 MQCD 구조에 대한 C 선언입니다.

[1437 페이지의 『COBOL 선언』](#)

이 선언은 MQCD 구조에 대한 COBOL 선언입니다.

[1439 페이지의 『RPG 선언\(ILE\)』](#)

이 선언은 MQCD 구조에 대한 RPG 선언입니다.

[1442 페이지의 『System/390 어셈블러 선언』](#)

이 선언은 MQCD 구조에 대한 System/390 어셈블러 선언입니다.

[1443 페이지의 『Visual Basic 선언』](#)

이 선언은 MQCD 구조의 Visual Basic 선언입니다.

[1445 페이지의 『채널 엑시트에서 MQCD 필드 변경』](#)

채널 엑시트는 MQCD의 필드를 변경할 수 있습니다. 그러나, 나열된 상황을 제외하고는 일반적으로 이러한 변경 사항이 적용되지 않습니다.

### **필드**

이 주제는 MQCD 구조의 모든 필드를 나열하고 각 필드를 설명합니다.

*BatchDataLimit (MQLONG)*

이 필드는 동기점을 확보하기 전에 채널을 통해 송신할 수 있는 데이터 양의 한계(킬로바이트)입니다.

동기점은 한계에 도달하게 되는 메시지가 채널 전체에 플로우된 후에 확보됩니다.

다음 중 한 가지 조건이 충족되면 배치가 종료됩니다.

- **BatchSize** 메시지가 전송되었습니다.
- **BatchDataLimit** 바이트가 전송되었습니다.
- 전송 큐가 비어 있고 **BatchInterval**을 초과했습니다.

값의 범위는 0 - 999999여야 합니다. 기본값은 5000입니다.

이 속성에서 값이 0이면 이 채널을 통해 배치에 적용되는 데이터 한계가 없음을 의미합니다.

이 매개변수는 *ChannelType*이(가) MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_CLUSRCVR 또는 MQCHT\_CLUSSDR인 채널에만 적용됩니다.

엑시트의 입력 필드입니다. *Version*이 MQCD\_VERSION\_11 미만이면 이 필드가 존재하지 않습니다.

#### *BatchHeartbeat (MQLONG)*

이 필드는 채널에 대해 배치 하트비트를 트리거하는 데 사용되는 시간 간격을 지정합니다.

배치 하트비트는 송신자 채널이 인다우트 상태가 되기 전에 리모트 채널 인스턴스가 여전히 활성 상태인지 판별할 수 있도록 합니다. 배치 하트비트는 송신자 채널이 지정된 시간 간격 내에 리모트 채널 인스턴스와 통신하지 않은 경우 발생합니다.

값의 범위는 0 - 999 999이며, 단위는 밀리초입니다. 값 0은 배치 하트비트가 사용 가능하지 않음을 나타냅니다.

이 필드는 *ChannelType*이 MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_CLUSSDR 또는 MQCHT\_CLUSRCVR인 채널만 관련이 있습니다.

엑시트의 입력 필드입니다. *Version*이 MQCD\_VERSION\_7 미만이면 이 필드가 존재하지 않습니다.

#### *BatchInterval (MQLONG)*

이 필드는 현재 배치에서 *BatchSize*보다 적은 메시지가 전송된 경우 채널이 배치를 열린 상태로 유지하는 대략적 시간(밀리초)을 지정합니다.

*BatchInterval*이 0보다 크면 다음 이벤트 중 가장 먼저 발생하는 이벤트에 의해 배치가 종료됩니다.

- *BatchSize* 메시지가 송신되었습니다. 또는
- 배치 시작 이후 *BatchInterval* 밀리초가 경과되었습니다.

*BatchInterval*이 0이면 다음 이벤트 중 가장 먼저 발생하는 이벤트에 의해 배치가 종료됩니다.

- *BatchSize* 메시지가 송신되었습니다. 또는
- 전송 큐가 비어 있습니다.

*BatchInterval*은 0 - 999 999 999 범위에 있어야 합니다.

이 필드는 *ChannelType*이 MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_CLUSSDR 또는 MQCHT\_CLUSRCVR인 채널에만 관련됩니다.

엑시트의 입력 필드입니다. *Version*이 MQCD\_VERSION\_4 미만이면 필드가 표시되지 않습니다.

#### *BatchSize (MQLONG)*

이 필드는 채널을 동기화하기 전에 채널을 통해 송신할 수 있는 최대 메시지 수를 지정합니다.

이 필드는 *ChannelType*이 MQCHT\_SVRCONN 또는 MQCHT\_CLNTCONN인 채널과는 관련이 없습니다.

#### *CertificateLabel (MQCHAR64)*

이 필드는 사용 중인 인증서 레이블의 세부사항을 제공합니다.

IBM MQ에서 *CertificateLabel* 필드의 기본값을 공백으로 초기화합니다.

이 속성은 런타임에 기본값으로 해석되고, 역호환 가능합니다.

예를 들어, *CertificateLabel* 필드에 기본값인 공백을 사용하거나 11 이전 버전의 MQCD를 지정하는 경우 이 필드가 무시됩니다.

이 필드의 길이는 MQ\_CERT\_LABEL\_LENGTH에서 제공합니다.

### *ChannelMonitoring (MQLONG)*

이 필드는 채널에 대한 모니터링 데이터 콜렉션의 현재 레벨을 지정합니다.

이 필드는 ChannelType이 MQCHT\_CLNTCONN인 채널과는 관련이 없습니다.

다음 값 중 하나입니다.

- MQMON\_OFF
- MQMON\_LOW
- MQMON\_MEDIUM
- MQMON\_HIGH

엑시트의 입력 필드입니다. *Version*이 MQCD\_VERSION\_8 미만이면 필드가 표시되지 않습니다.

### *ChannelName (MQCHAR20)*

이 필드는 채널 정의 이름을 지정합니다.

통신하려면 리모트 시스템에 동일한 이름의 채널 정의가 있어야 합니다.

이름에는 다음 문자만을 사용해야 합니다.

- 대문자 A - Z
- 소문자 a-z
- 숫자 0 - 9
- 마침표(.)
- 정방향 슬래시(/)
- 밑줄(\_)
- 퍼센트 부호(%)

그리고 오른쪽에 공백을 채워 넣어야 하며, 선두 문자 또는 임베드된 공백은 허용되지 않습니다.

이 필드의 길이는 MQ\_CHANNEL\_NAME\_LENGTH에서 제공합니다.

### *ChannelStatistics (MQLONG)*

이 필드는 채널에 대한 통계 데이터 콜렉션의 현재 레벨을 지정합니다.

이 필드는 ChannelType이 MQCHT\_CLNTCONN 또는 MQCHT\_SVRCONN인 채널과는 관련이 없습니다.

다음 값 중 하나입니다.

- MQMON\_OFF
- MQMON\_LOW
- MQMON\_MEDIUM
- MQMON\_HIGH

엑시트의 입력 필드입니다. *Version*이 MQCD\_VERSION\_8 미만이면 필드가 표시되지 않습니다.

### *ChannelType (MQLONG)*

이 필드는 채널의 유형을 지정합니다.

다음 값 중 하나입니다.

#### **MQCHT\_SENDER**

송신자입니다.

#### **MQCHT\_SERVER**

서버.

#### **MQCHT\_RECEIVER**

수신자입니다.

#### **MQCHT\_REQUESTER**

요청자입니다.



## **MQCHT\_CLNTCONN**

클라이언트 연결.

## **MQCHT\_SVRCONN**

서버 연결(클라이언트에서 사용).

## **MQCHT\_CLUSSDR**

클러스터 송신자.

## **MQCHT\_CLUSRCVR**

클러스터 수신자.

### *ClientChannelWeight (MQLONG)*

이 필드는 사용된 클라이언트 연결 채널 정의에 영향을 주는 가중치를 지정합니다.

둘 이상의 적절한 정의가 사용 가능할 때 해당 가중치에 따라 임의로 클라이언트 채널 정의를 선택할 수 있도록 ClientChannelWeight 속성이 사용됩니다. 클라이언트가 별표로 시작하는 큐 관리자 이름을 지정하여 큐 관리자 그룹에 대한 연결을 요청하는 MQCONN을 발행하고 클라이언트 채널 정의 테이블(CCDT)에서 둘 이상의 적절한 채널 정의가 사용 가능할 때, 사용할 정의는 적용 가능한 ClientChannelWeight(0) 정의가 영문자순으로 먼저 선택되는 가중치에 따라 임의로 선택됩니다.

0 - 99 범위의 값을 지정하십시오. 기본값은 0입니다.

0 값은 로드 밸런싱이 수행되지 않으며 적용 가능한 정의가 알파벳순으로 선택됨을 나타냅니다. 로드 밸런싱을 사용하려면 1 - 99 범위의 값을 선택하십시오. 여기서 1은 가장 낮은 가중치이고 99는 가장 높은 가중치입니다. 0이 아닌 가중치가 적용되는 둘 이상의 채널 간 메시지 분배는 해당 가중치 비율에 비례합니다. 예를 들어 ClientChannelWeight 값이 2, 4 및 14인 세 개의 채널은 대략 시간의 10%, 20% 및 70%로 선택됩니다. 이러한 분배가 보장되지는 않습니다.

이 속성은 클라이언트-연결 채널 유형에 대해서만 유효합니다.

엑시트의 입력 필드입니다. *Version*이 MQCD\_VERSION\_9 미만이면 필드가 표시되지 않습니다.

### *ClusterPtr (MQPTR)*

이 필드는 클러스터 이름 목록의 주소를 지정합니다.

*ClustersDefined*가 0보다 크면 이 주소가 클러스터 이름 목록의 주소입니다. 채널은 나열된 각 클러스터에 속합니다.

이 필드는 *ChannelType*이 MQCHT\_CLUSSDR 또는 MQCHT\_CLUSRCVR인 채널에만 해당됩니다.

엑시트의 입력 필드입니다. 이 필드는 *Version*이 MQCD\_VERSION\_5 미만인 경우에는 표시되지 않습니다.

### *ClustersDefined (MQLONG)*

이 필드는 채널이 속한 클러스터의 수를 지정합니다.

이 필드는 *ClusterPtr*이 가리키는 클러스터 이름의 수입입니다. 0 이상입니다.

이 필드는 *ChannelType*이 MQCHT\_CLUSSDR 또는 MQCHT\_CLUSRCVR인 채널에만 해당됩니다.

엑시트의 입력 필드입니다. 이 필드는 *Version*이 MQCD\_VERSION\_5 미만인 경우에는 표시되지 않습니다.

### *CLWLChannelPriority (MQLONG)*

이 필드는 클러스터 워크로드 채널 우선순위를 지정합니다.

워크로드 관리자 선택 알고리즘은 순위를 기반으로 선택된 목적지 세트 중에서 우선순위가 가장 높은 목적지를 선택합니다. 두 개의 목적지 큐 관리자를 사용할 수 있는 경우, 하나의 큐 관리자가 다른 큐 관리자에 장애 복구되게 하려면 이 속성을 사용합니다. 우선순위가 가장 높은 큐 관리자로 모든 메시지가 이동한 다음 그 끝에 도달하면 다음으로 우선순위가 높은 큐 관리자로 메시지가 이동합니다.

값의 범위는 0-9입니다. 기본값은 0입니다.

엑시트의 입력 필드입니다. *Version*이 MQCD\_VERSION\_8 미만이면 이 필드가 존재하지 않습니다.

자세한 정보는 [큐 관리자 클러스터 구성의 내용](#)을 참조하십시오.

### *CLWLChannelRank (MQLONG)*

이 필드는 클러스터 워크로드 채널 순위를 지정합니다.

워크로드 관리자 선택 알고리즘은 순위가 가장 높은 목적지를 선택합니다. 마지막 목적지가 다른 클러스터의 큐 관리자이면 (인접한 클러스터의 교차점에서) 중간 게이트웨이 큐 관리자의 순위를 설정할 수 있으므로 선택 알고리즘이 최종 목적지에 가까운 목적지 큐 관리자를 올바르게 선택합니다.

값의 범위는 0-9입니다. 기본값은 0입니다.

엑시트의 입력 필드입니다. *Version*이 MQCD\_VERSION\_8 미만이면 이 필드가 존재하지 않습니다.

자세한 정보는 [큐 관리자 클러스터 구성](#)의 내용을 참조하십시오.

### *CLWLChannelWeight (MQLONG)*

이 필드는 클러스터 워크로드 채널 가중치를 지정합니다.

클러스터 워크로드 채널 가중치.

워크로드 관리자 선택 알고리즘은 더 많은 메시지가 특정 시스템으로 송신될 수 있도록 채널의 "가중치" 속성을 사용하여 목적지 선택을 변경합니다. 예를 들어, 대규모 UNIX 서버의 채널에 소규모 데스크탑 PC의 다른 채널보다 더 큰 "가중치"를 지정할 수 있으며, 선택 알고리즘은 PC보다 더 자주 UNIX 서버를 선택합니다.

값의 범위는 1 - 99입니다. 기본값은 50입니다.

엑시트의 입력 필드입니다. *Version*이 MQCD\_VERSION\_8 미만이면 이 필드가 존재하지 않습니다.

자세한 정보는 [큐 관리자 클러스터 구성](#)의 내용을 참조하십시오.

### *ConnectionAffinity (MQLONG)*

이 필드는 동일한 큐 관리자 이름을 통해 여러 번 연결하는 클라이언트 애플리케이션이 동일한 클라이언트 채널을 사용하는지 여부를 지정합니다.

여러 개의 적용 가능한 채널 정의가 사용 가능할 때 이 속성을 사용하십시오.

값은 다음 중 하나입니다.

#### **MQCAFTY\_PREFERRED**

클라이언트 채널 정의 테이블(CCDT)을 읽는 프로세스의 첫 번째 연결은 적용 가능한 CLNTWGHT(0) 정의가 처음이며 알파벳순으로 가중치에 따라 적용 가능한 정의의 목록을 작성합니다. 프로세스의 각 연결은 목록의 첫 번째 정의를 사용하여 연결을 시도합니다. 연결에 실패하는 경우 다음 정의가 사용됩니다. CLNTWGHT 값이 0이 아닌 실패한 정의는 목록의 끝으로 이동됩니다. CLNTWGHT(0) 정의는 목록의 처음에 남아서 각 연결에 첫 번째로 선택됩니다.

호스트 이름이 동일한 각 클라이언트 프로세스는 항상 동일한 목록을 작성합니다.

C, C++ 또는 .NET 프로그래밍 프레임워크(완전히 관리되는 .NET 포함)에서 작성된 클라이언트 애플리케이션의 경우, 목록 작성 이후 CCDT가 수정되면 목록이 업데이트됩니다.

이 값이 기본값입니다.

#### **MQCAFTY\_NONE**

CCDT를 읽는 프로세스의 첫 번째 연결이 적용 가능한 정의의 목록을 작성합니다. 프로세스의 모든 연결이 알파벳순에서 첫 번째로 선택되는 적용 가능한 CLNTWGHT(0) 정의의 가중치에 기반하여 적용 가능한 정의를 선택합니다.

C, C++ 또는 .NET 프로그래밍 프레임워크(완전히 관리되는 .NET 포함)에서 작성된 클라이언트 애플리케이션의 경우, 목록 작성 이후 CCDT가 수정되면 목록이 업데이트됩니다.

이 속성은 클라이언트-연결 채널 유형에 대해서만 유효합니다.

엑시트의 입력 필드입니다. *Version*이 MQCD\_VERSION\_9 미만이면 필드가 표시되지 않습니다.

### *ConnectionName (MQCHAR264)*

이 필드는 채널의 연결 이름을 지정합니다.

클러스터 수신자 채널 (지정된 경우) CONNAME은 로컬 큐 관리자와 관련되어 있으며, 다른 채널은 대상 큐 관리자와 관련되어 있습니다. 사용자가 지정하는 값은 사용할 전송 프로토콜(*TransportType*)에 따라 다릅니다.

- MQXPT\_LU62의 경우, 파트너 논리 단위의 완전한 이름입니다.
- MQXPT\_NETBIOS의 경우, 리모트 시스템에 정의된 NetBIOS 이름입니다.
- MQXPT\_TCP의 경우, 호스트 이름, IPv4 점분리 십진수 또는 IPv6 16진 형식에 지정된 리모트 시스템의 네트워크 주소 또는 클러스터 수신기 채널의 로컬 시스템입니다.
- MQXPT\_SPX의 경우, 4바이트 네트워크 주소, 6바이트 노드 주소 및 2바이트 소켓 번호로 구성된 SPX 스타일 주소입니다.

채널 정의 시 이 필드는 *ChannelType*이 MQCHT\_SVRCONN 또는 MQCHT\_RECEIVER인 채널과는 관련이 없습니다. 그러나 채널 정의를 엑시트로 전달하면 이 필드에는 채널 유형이 무엇이든 파트너의 주소가 포함됩니다.

이 필드의 길이는 MQ\_CONN\_NAME\_LENGTH에서 제공합니다. *Version*이 MQCD\_VERSION\_2 미만이면 이 필드가 표시되지 않습니다.

#### *DataConversion (MQLONG)*

이 필드는 수신 메시지 채널 에이전트가 이 변환을 수행할 수 없는 경우 송신 메시지 채널 에이전트가 애플리케이션 메시지 데이터의 변환을 시도하는지 여부를 지정합니다.

이 필드는 논리 메시지의 세그먼트가 아닌 메시지에만 적용됩니다. MCA는 세그먼트인 메시지를 변환하려고 하지 않습니다.

이 필드는 *ChannelType*이 MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_CLUSSDR 또는 MQCHT\_CLUSRCVR인 채널에만 관련됩니다. 다음 중 하나입니다.

#### **MQCDC\_SENDER\_CONVERSION**

송신자가 변환합니다.

#### **MQCDC\_NO\_SENDER\_CONVERSION**

송신자에 의한 변환 없음.

#### *DefReconnect (MQLONG)*

DefReconnect 채널 속성은 클라이언트 연결 채널에 대한 기본 다시 연결 속성 값을 설정합니다.

기본 자동 클라이언트 다시 연결 옵션입니다. 클라이언트 애플리케이션을 자동으로 다시 연결하도록 IBM MQ MQI client를 구성할 수 있습니다. 연결에 실패하면 IBM MQ MQI client가 큐 관리자에 다시 연결하려고 시도합니다. 이때 MQCONN 또는 MQCONNX MQI 호출을 발행하는 애플리케이션 클라이언트 없이 다시 연결하려고 시도합니다.

다시 연결은 MQCONNX 옵션입니다. DefReconnect 채널 속성을 사용하면 MQCONN을 사용하는 기존 애플리케이션에 다시 연결 동작을 추가할 수 있습니다. 또한 MQCONNX를 사용하는 애플리케이션의 다시 연결 동작을 변경할 수도 있습니다.

mqclient.ini 파일에서 DefRecon 값을 설정하여 다시 연결 동작을 설정 또는 수정할 수도 있습니다.

mqclient.ini 파일의 DefRecon 값이 DefReconnect 채널 속성보다 우선합니다.

## Syntax

**DefReconnect** (MQRCN\_NO (default) |MQRCN\_YES|MQRCN\_Q\_MGR|MQRCN\_DISABLED)

### 매개변수

#### **MQRCN\_NO**

MQRCN\_NO는 기본값입니다.

MQCONNX로 대체되지 않으면 클라이언트가 자동으로 다시 연결되지 않습니다.

#### **MQRCN\_YES**

MQCONNX로 대체되지 않으면 클라이언트가 자동으로 다시 연결됩니다.

#### **MQRCN\_Q\_MGR**

MQCONNX로 대체되지 않으면 클라이언트가 자동으로 다시 연결되지만 동일한 큐 관리자에만 연결됩니다.

QMGR 옵션은 MQCNO\_RECONNECT\_Q\_MGR와 동일한 효과를 갖습니다.

## MQRN\_DISABLED

MQCONNX MQI 호출을 사용하여 클라이언트 프로그램이 요청하는 경우에도 재연결이 사용 불가능합니다.

자동 클라이언트 재연결은 IBM MQ classes for Java에서 지원되지 않습니다.

표 822. 자동 재연결은 애플리케이션 및 채널 정의에 설정된 값에 따라 결정됩니다.				
DefReconnect	애플리케이션에 설정된 재연결 옵션			
	MQCNO_RECONNE CT	MQCNO_RECONNE CT_Q_MGR	MQCNO_RECONNE CT_AS_DEF	MQCNO_RECONNE CT_DISABLED
MQRN_NO	YES	QMGR	NO	NO
MQRN_YES	YES	QMGR	YES	NO
MQRN_Q_MGR	YES	QMGR	QMGR	NO
MQRN_DISABLED	NO	NO	NO	NO

### 관련 개념

자동 클라이언트 다시 연결

채널 및 클라이언트 다시 연결

클라이언트 구성 파일의 CHANNELS 스탠자

### 관련 참조

317 페이지의 『MQCNO에 대한 옵션 (MQLONG)』

MQCONNX의 조치를 제어하는 옵션입니다.

### Desc (MQCHAR64)

이 필드는 주석에 사용할 수 있습니다.

이 필드의 내용은 메시지 채널 에이전트에 중요하지 않습니다. 그러나, 표시될 수 있는 문자만 포함해야 합니다. 널(null) 문자는 포함할 수 없지만, 필요한 경우 오른쪽으로 공백을 채워넣을 수 있습니다. DBCS 설치시 필드는 DBCS 문자(최대 필드 길이 64비트에 따라)를 포함할 수 있습니다.

**참고:** 이 필드에 큐 관리자의 문자 세트(**CodedCharSetId** 큐 관리자 속성으로 정의)에 없는 문자가 있는 경우, 이 필드를 다른 큐 관리자로 송신하면 이러한 문자가 잘못 변환될 수 있습니다.

이 필드의 길이는 MQ\_CHANNEL\_DESC\_LENGTH에서 제공합니다.

### DiscInterval (MQLONG)

이 필드는 채널을 종료하기 전에 메시지가 전송 큐에 도착할 때까지 채널이 대기하는 최대 시간(초)을 지정합니다.

즉, 연결 끊기 간격을 지정합니다.

A 값이 0이면 MCA가 무제한 대기하게 됩니다.

TCP 프로토콜을 사용하는 서버 연결 채널의 경우, 간격은 클라이언트 비활동 연결 끊기 값(초)을 표시합니다. 서버 연결에 이 기간 동안 상위 클라이언트로부터 수신된 통신에 없으면 연결을 종료합니다. 서버 연결 비활동 간격은 클라이언트의 IBM MQ API 호출 간에만 적용되므로, 대기 호출을 포함한 MQGET을 장시간 실행하는 동안 클라이언트 연결이 끊어지지 않습니다.

이 속성은 TCP 이외의 프로토콜을 사용하는 서버 연결 채널에는 적용할 수 없습니다.

이 필드는 *ChannelType*이 MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_CLUSSDR, MQCHT\_CLUSRCVR 또는 MQCHT\_SVRCONN인 채널만 관련이 있습니다.

### ExitDataLength (MQLONG)

이 필드는 *MsgUserDataPtr*, *SendUserDataPtr* 및 *ReceiveUserDataPtr* 필드를 통해 처리되는 엑시트 사용자 데이터 항목 목록에서 각 사용자 데이터 항목의 길이(바이트)를 지정합니다.

이 길이가 반드시 MQ\_EXIT\_DATA\_LENGTH와 같을 필요는 없습니다.

엑시트의 입력 필드입니다. 이 필드는 *Version*이 MQCD\_VERSION\_4 미만일 경우에는 표시되지 않습니다.

#### *ExitNameLength (MQLONG)*

이 필드는 *MsgExitPtr*, *SendExitPtr* 및 *ReceiveExitPtr* 필드를 통해 처리되는 엑시트 이름 목록에서 각 이름의 길이(바이트)를 지정합니다.

이 길이가 반드시 *MQ\_EXIT\_NAME\_LENGTH*와 같을 필요는 없습니다.

엑시트의 입력 필드입니다. 이 필드는 *Version*이 *MQCD\_VERSION\_4* 미만일 경우에는 표시되지 않습니다.

#### *HdrCompList [2] (MQLONG)*

이 필드는 채널에 지원되는 헤더 데이터 압축 기술 목록을 지정합니다.

목록에는 다음 값 중 하나 이상이 있습니다.

##### **MQCOMPRESS\_NONE**

헤더 데이터 압축이 수행되지 않습니다.

##### **MQCOMPRESS\_SYSTEM**

헤더 데이터 압축이 수행됩니다.

##### **MQCOMPRESS\_NOT\_AVAILABLE**

목록에서 사용하지 않는 값은 이 값으로 설정됩니다.

엑시트의 입력 필드입니다. *Version*이 *MQCD\_VERSION\_8* 미만이면 이 필드가 존재하지 않습니다.

#### *HeartbeatInterval (MQLONG)*

이 필드는 하트비트 플로우 사이의 시간(초)을 지정합니다.

이 필드는 채널 유형에 따라 다음과 같이 해석됩니다.

- 채널 유형이 *MQCHT\_SENDER*, *MQCHT\_SERVER*, *MQCHT\_RECEIVER*, *MQCHT\_REQUESTER*, *MQCHT\_CLUSSDR* 또는 *MQCHT\_CLUSRCVR*인 경우 이 필드는 전송 큐에 메시지가 없을 때 송신 MCA에서 전달되는 하트비트 플로우 사이의 시간(초)입니다. 따라서 수신 MCA에 채널을 일시정지할 수 있는 기회가 제공됩니다. 유용하려면 *HeartbeatInterval*이(가) *DiscInterval*보다 작아야 합니다.
- *MQCD* 대화 공유 필드가 0으로 설정된 *MQCHT\_CLNTCONN* 또는 *MQCHT\_SVRCONN* 채널 유형의 경우, 이 필드는 MCA에서 클라이언트 애플리케이션을 대신하여 *MQGMO\_WAIT* 옵션으로 *MQGET* 호출을 발행할 때 서버 MCA로부터 전달되는 하트비트 플로우 사이의 시간(초 단위)입니다. 따라서 서버 MCA가 *MQGMO\_WAIT*가 지정된 *MQGET*을 실행하는 동안 클라이언트 연결이 실패하는 상황을 처리할 수 있습니다.
- *MQCD* 대화 공유 필드가 0이 아닌 값으로 설정되고 채널 유형이 *MQCHT\_CLNTCONN* 또는 *MQCHT\_SVRCONN*인 경우, 이 필드는 송신 또는 수신되는 데이터 플로우가 없을 때 하트비트 플로우 간의 시간(초)입니다. 이렇게 하면 채널을 효율적으로 일시정지할 수 있습니다.

값의 범위는 0 - 999 999입니다. 사용되는 값은 양쪽에 0 값을 지정하지 않는 한 송신 측과 수신 측에 지정된 값 중 더 큰 값입니다. 이 경우, 하트비트 교환이 발생하지 않습니다.

엑시트의 입력 필드입니다. 이 필드는 *Version*이 *MQCD\_VERSION\_4* 미만일 경우에는 표시되지 않습니다.

#### *KeepAliveInterval (MQLONG)*

이 필드는 채널의 활성 유지 시간에 대해 통신 스택에 전달되는 값을 지정합니다.

일부 구현에서 이 매개변수를 지원하지는 않더라도, TCP/IP 및 SPX 통신 프로토콜에 값을 적용할 수 있습니다.

값의 범위는 0 - 99 999이며, 단위는 초입니다. 값이 0이면 채널 활성 유지가 사용되지 않습니다. 단, (채널 활성 유지가 아닌) TCP/IP 활성 유지가 사용되면 여전히 활성 유지가 발생할 수 있습니다. 다음 특수 값도 유효합니다.

##### **MQKAI\_AUTO**

자동.

이 값은 다음과 같이 협상된 하트비트 간격에서 활성 유지 간격이 계산됨을 표시합니다.

- 협상된 하트비트 간격이 0보다 크면 사용되는 활성 유지 간격은 하트비트 간격에 60초를 더한 값입니다.
- 협상된 하트비트 간격이 0이면 사용되는 활성 유지 간격은 0입니다.
- z/OS에서 큐 관리자 오브젝트에 TCPKEEP(YES)을 지정하면 TCP/IP 활성 유지가 발생합니다.
- 다른 환경에서는 분산 큐잉 구성 파일의 TCP 스탠자에 **KEEPALIVE=YES** 매개변수를 지정하면 TCP/IP 활성 유지가 발생합니다.

이 필드는 *TransportType*이 MQXPT\_TCP 또는 MQXPT\_SPX인 채널만 관련이 있습니다.

엑시트의 입력 필드입니다. *Version*이 MQCD\_VERSION\_7 미만이면 이 필드가 존재하지 않습니다.

#### *LocalAddress (MQCHAR48)*

이 필드는 아웃바운드 통신 채널에 정의되는 로컬 TCP/IP 주소를 지정합니다.

아웃바운드 통신에 정의한 특정 주소가 없으면 이 필드가 공백입니다. 주소에는 선택적으로 포트 번호 또는 포트 번호 범위가 포함될 수 있습니다. 이 주소의 형식은 다음과 같습니다.

```
[ip-addr][(low-port[,high-port])]
```

여기서 대괄호([ ])는 선택적 정보를 지정하고, ip-addr은 IPv4 점분리 십진수, IPv6 16진 또는 영숫자 형식으로 지정되고, low-port 및 high-port는 괄호로 묶인 포트 번호입니다. 모두 선택사항입니다.

아웃바운드 통신의 특정 IP 주소, 포트 또는 포트 범위는 채널이 다른 TCP/IP 스택에서 재시작되는 복구 시나리오에 유용합니다.

*LocalAddress*는 양식이 *ConnectionName*과 유사하지만, 서로 혼동해서는 안 됩니다. *LocalAddress*는 로컬 통신의 특성을 지정하고, *ConnectionName*은 리모트 큐 관리자에 도달하는 방법을 지정합니다.

IBM MQ 9.3.0부터 채널 인스턴스가 작성되고 큐 관리자에 연결된 후 로컬 주소 필드가 MQCD 오브젝트에 설정 되도록 JMQUI (Java Message Queueing Interface) 가 업데이트되었습니다. 즉, Java에 작성된 채널 엑시트가 MQCD.get`LocalAddress`() 메소드를 호출하면 해당 메소드에서 채널 인스턴스가 사용 중인 로컬 주소를 리턴합니다. IBM MQ 9.3.0이전에는 채널 보안 엑시트가 채널 인스턴스에서 사용 중인 로컬 주소에 액세스할 수 없으며 MQCD.get`LocalAddress`() 메소드가 널을 리턴했습니다.

이 필드는 *TransportType*이 MQXPT\_TCP이고, *ChannelType*이 MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_REQUESTER, MQCHT\_CLNTCONN, MQCHT\_CLUSSDR 또는 MQCHT\_CLUSRCVR인 채널만 관련이 있습니다.

이 필드의 길이는 MQ\_LOCAL\_ADDRESS\_LENGTH에서 제공합니다. *Version*이 MQCD\_VERSION\_7 미만이면 이 필드가 표시되지 않습니다.

#### *LongMCAUserIdLength (MQLONG)*

이 필드는 *LongMCAUserIdPtr*이 가리키는 전체 MCA 사용자 ID의 길이(바이트)를 지정합니다.

이 필드는 *ChannelType*이 MQCHT\_CLNTCONN인 채널과는 관련이 없습니다.

이는 엑시트의 입출력(I/O) 필드입니다. *Version*이 MQCD\_VERSION\_6 미만이면 필드가 존재하지 않습니다.

#### *LongMCAUserIdPtr (MQPTR)*

이 필드는 긴 MCA 사용자 ID의 주소를 지정합니다.

*LongMCAUserIdLength*가 0보다 더 크면 이 필드는 전체 MCA 사용자 ID의 주소입니다. 전체 ID의 길이는 *LongMCAUserIdLength*에서 제공합니다. MCA 사용자 ID의 처음 12바이트도 *MCAUserIdentifier* 필드에 포함됩니다.

MCA 사용자 ID에 대한 자세한 정보는 *MCAUserIdentifier* 필드의 설명을 참조하십시오.

이 필드는 *ChannelType*이 MQCHT\_SDR, MQCHT\_SVR, MQCHT\_CLNTCONN 또는 MQCHT\_CLUSSDR인 채널과는 관련이 없습니다.

이는 엑시트의 입출력(I/O) 필드입니다. *Version*이 MQCD\_VERSION\_6 미만이면 필드가 존재하지 않습니다.

#### *LongRemoteUserIdLength (MQLONG)*

이 필드는 *LongRemoteUserIdPtr*이 가리키는 전체 리모트 사용자 ID의 길이(바이트)를 지정합니다.

이 필드는 *ChannelType*이 MQCHT\_CLNTCONN 또는 MQCHT\_SVRCONN인 채널에만 해당됩니다.

엑시트의 입력 필드입니다. *Version*이 MQCD\_VERSION\_6 미만이면 필드가 존재하지 않습니다.

#### *LongRemoteUserIdPtr (MQPTR)*

이 필드는 긴 리모트 사용자 ID의 주소를 지정합니다.

*LongRemoteUserIdLength*가 0보다 크면 이 플래그는 전체 리모트 사용자 ID의 주소입니다. 전체 ID의 길이는 *LongRemoteUserIdLength*에서 제공합니다. 리모트 사용자 ID의 처음 12바이트도 *RemoteUserIdentifier* 필드에 포함됩니다.

리모트 사용자 ID에 대한 자세한 정보는 *RemoteUserIdentifier* 필드의 설명을 참조하십시오.

이 필드는 *ChannelType*이 MQCHT\_CLNTCONN 또는 MQCHT\_SVRCONN인 채널에만 해당됩니다.

엑시트의 입력 필드입니다. *Version*이 MQCD\_VERSION\_6 미만이면 필드가 존재하지 않습니다.

#### *LongRetryCount (MQLONG)*

이 필드는 *ShortRetryCount*로 지정된 수에 도달한 후 사용되는 수를 지정합니다.

운영자에게 오류를 기록하기 전에 *LongRetryInterval*이 지정한 간격으로 리모트 시스템에 연결하려고 시도하는 최대 횟수를 지정합니다.

이 필드는 *ChannelType*이 MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_CLUSSDR 또는 MQCHT\_CLUSRCVR인 채널에만 관련됩니다.

#### *LongRetryInterval (MQLONG)*

이 필드는 리모트 시스템에 연결을 재시도할 때까지 대기하는 최대 시간(초)을 지정합니다.

활성 상태가 될 때까지 채널이 대기해야 하는 경우 재시도 간격이 늘어날 수 있습니다.

이 필드는 *ChannelType*이 MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_CLUSSDR 또는 MQCHT\_CLUSRCVR인 채널에만 관련됩니다.

#### *MaxInstances (MQLONG)*

이 필드는 시작할 수 있는 개별 서버 연결 채널의 최대 동시 인스턴스 수를 지정합니다.

이 필드는 서버 연결 채널에서만 사용됩니다.

필드 값의 범위는 0 - 999 999 999입니다. 값이 0이면 모든 클라이언트에 액세스할 수 없습니다.

이 필드의 기본값은 999 999 999입니다.

이 필드의 값을 현재 실행 중인 서버 연결 채널 인스턴스의 수보다 적은 숫자로 줄이는 경우, 실행 중인 인스턴스는 영향을 받지 않습니다. 현재 실행 중인 인스턴스 수가 필드의 값보다 적어질 만큼 충분한 수의 기존 인스턴스가 실행을 정지할 때까지 새 인스턴스를 시작할 수 없습니다.

#### *MaxInstancesPerClient (MQLONG)*

이 필드는 단일 클라이언트에서 시작할 수 있는 개별 서버 연결 채널의 최대 동시 인스턴스 수를 지정합니다.

이 컨텍스트에서는 동일한 리모트 네트워크 주소로부터의 연결이 동일한 클라이언트로부터의 연결로 간주됩니다.

이 필드는 서버 연결 채널에서만 사용됩니다.

필드 값의 범위는 0 - 999 999 999입니다. 값이 0이면 모든 클라이언트에 액세스할 수 없습니다.

이 필드의 기본값은 999 999 999입니다.

이 필드의 값을 현재 개별 클라이언트에서 실행 중인 서버 연결 채널 인스턴스의 수보다 적은 숫자로 줄이는 경우, 실행 중인 인스턴스는 영향을 받지 않습니다. 그러나 충분한 수의 기존 인스턴스가 실행을 정지하여 현재 실행 중인 인스턴스 수가 필드의 값보다 적어질 때까지 이러한 클라이언트 중 하나에서 새 인스턴스를 시작할 수 없습니다.

#### *MaxMsgLength(MQLONG)*

이 필드는 채널에서 전송될 수 있는 최대 메시지 길이를 지정합니다.

이는 리모트 채널 값과 비교되며 실제 최대값은 두 값 중 낮은 값입니다.



*MCAName (MQCHAR20)*

이 필드는 예약 필드입니다.

이 필드의 값은 공백입니다.

이 필드의 길이는 MQ\_MCA\_NAME\_LENGTH에서 제공합니다.

*MCASecurityId (MQBYTE40)*

이 필드는 MCA에 대한 보안 ID를 지정합니다.

이 필드는 *ChannelType*이 MQCHT\_CLNTCONN인 채널과는 관련이 없습니다.

다음 특수 값은 보안 ID가 없음을 나타냅니다.

#### **MQSID\_NONE**

보안 ID가 지정되지 않음.

이 값은 필드 길이 만큼의 2진 0입니다.

C 프로그래밍 언어의 경우, 상수 MQSID\_NONE\_ARRAY도 정의됩니다. 이 상수는 MQSID\_NONE과 동일한 값을 갖지만, 문자열이 아니라 문자 배열입니다.

이는 엑시트의 입출력(I/O) 필드입니다. 이 필드 길이는 MQ\_SECURITY\_ID\_LENGTH에서 제공합니다.

*Version*이 MQCD\_VERSION\_6 미만이면 이 필드가 표시되지 않습니다.

*MCAType (MQLONG)*

이 필드는 메시지 채널 에이전트 프로그램의 유형을 지정합니다.

이 필드는 *ChannelType*이 MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_REQUESTER, MQCHT\_CLUSSDR 또는 MQCHT\_CLUSRCVR인 채널만 관련이 있습니다.

값은 다음 중 하나입니다.

#### **MQMCAT\_PROCESS**

:NONE.

메시지 채널 에이전트가 개별 프로세스로 실행됩니다.

#### **MQMCAT\_THREAD**

스레드 (멀티플랫폼).

메시지 채널 에이전트는 개별 스레드로 실행됩니다.

*Version*이 MQCD\_VERSION\_2 미만이면 이 필드가 표시되지 않습니다.

*MCAUserIdentifier (MQCHAR12)*

이 필드는 메시지 채널 에이전트(MCA)의 사용자 ID를 지정합니다.

이 필드는 MCA 사용자 ID의 처음 12바이트를 사용하며 보안 에이전트에 의해 설정됩니다.

다음은 MCA 사용자 ID를 포함하는 두 개의 필드입니다.

- *MCAUserIdentifier*는 MCA 사용자 ID의 처음 12바이트를 포함하며, ID가 12바이트보다 짧으면 공백으로 채워집니다. *MCAUserIdentifier*는 공백일 수 있습니다.
- *LongMCAUserIdPtr*는 12바이트보다 길 수 있는 전체 MCA 사용자 ID를 가리킵니다. 길이는 *LongMCAUserIdLength*에서 제공합니다. 전체 ID는 후미 공백이 없으며 널(Null) 종료되지 않습니다. ID가 공백이면 *LongMCAUserIdLength*가 0이고 *LongMCAUserIdPtr*의 값이 정의되지 않습니다.

**참고:** *LongMCAUserIdPtr*은 *Version*이 MQCD\_VERSION\_6 미만이면 표시되지 않습니다.

MCA 사용자 ID가 공백이 아니면 IBM MQ 자원에 액세스하기 위한 권한 부여를 위해 메시지 채널 에이전트가 사용하는 사용자 ID를 지정합니다. 채널 유형 MQCHT\_REQUESTER, MQCHT\_RECEIVER 및 MQCHT\_CLUSRCVR의 경우, PutAuthority가 MQPA\_DEFAULT이면 목적지 큐에 넣기 조작에 대한 권한 검사에 사용되는 사용자 ID입니다.

MCA 사용자 ID가 공백이면 메시지 채널 에이전트는 기본 사용자 ID를 사용합니다.



메시지 채널 에이전트가 사용해야 하는 사용자 ID를 표시하도록 보안 엑시트가 MCA 사용자 ID를 설정할 수 있습니다. 엑시트는 *MCAUserIdentifier* 또는 *LongMCAUserIdPtr*이 가리키는 문자열을 변경할 수 있습니다. 둘 다 변경되지만 서로 다르게 변경되면, MCA는 *MCAUserIdentifier*보다 *LongMCAUserIdPtr*을 우선적으로 사용합니다. *LongMCAUserIdPtr*을 통해 처리된 문자열의 길이를 엑시트가 변경하는 경우, 그에 따라 *LongMCAUserIdLength*를 설정해야 합니다. 엑시트가 ID의 길이를 늘리면 엑시트는 필요한 길이의 스토리지를 할당하고, 해당 스토리지를 필수 ID에 설정하고, 스토리지의 주소를 *LongMCAUserIdPtr*에 배치해야 합니다. 엑시트가 MQXR\_TERM 이유와 함께 나중에 호출되는 경우 해당 스토리지 비우기를 담당합니다.

*ChannelType*이 MQCHT\_SVRCONN인 채널의 경우, 채널 정의에서 *MCAUserIdentifier*가 공백이면 클라이언트에서 전송되는 사용자 ID가 여기로 복사됩니다. 이 사용자 ID(서버에서 보안 엑시트가 수정한 후에)를 사용해 클라이언트 애플리케이션이 실행하게 됩니다.

MCA 사용자 ID는 *ChannelType*이 MQCHT\_SDR, MQCHT\_SVR, MQCHT\_CLNTCONN, MQCHT\_CLUSSDR인 채널과는 관련이 없습니다.

이는 엑시트의 입출력(I/O) 필드입니다. 이 필드의 길이는 MQ\_USER\_ID\_LENGTH에 지정되어 있습니다. *Version*이 MQCD\_VERSION\_2 미만이면 이 필드가 표시되지 않습니다.

#### *ModeName (MQCHAR8)*

이 필드는 LU 6.2 모드 이름을 지정합니다.

이 필드는 전송 프로토콜(*TransportType*)이 MQXPT\_LU62이고 *ChannelType*이 MQCHT\_SVRCONN 또는 MQCHT\_RECEIVER가 아닌 경우에만 관련이 있습니다.

이 필드는 항상 공백입니다. 정보는 대신 통신 부가 오브젝트에 있습니다.

이 필드의 길이는 MQ\_MODE\_NAME\_LENGTH에서 제공합니다.

#### *MsgCompList [16] (MQLONG)*

이 필드는 채널에 지원되는 메시지 데이터 압축 기술 목록을 지정합니다.

목록에는 다음 값 중 하나 이상이 있습니다.

#### **MQCOMPRESS\_NONE**

메시지 데이터 압축이 수행되지 않습니다.

#### **MQCOMPRESS\_RLE**

실행 길이 인코딩을 사용하여 메시지 데이터 압축이 수행됩니다.

#### **MQCOMPRESS\_ZLIBFAST**

메시지 데이터 압축은 zlib 압축 기술을 사용하여 수행합니다. 빠른 압축 시간을 선호합니다.

#### **MQCOMPRESS\_ZLIBHIGH**

메시지 데이터 압축은 zlib 압축 기술을 사용하여 수행합니다. 상위 레벨의 압축을 선호합니다.

#### **V 9.4.0 LZ4FAST**

메시지 데이터 압축은 속도 우선순위가 지정된 LZ4 인코딩을 사용하여 수행됩니다.

#### **V 9.4.0 LZ4HIGH**

메시지 데이터 압축은 압축 우선순위가 지정된 LZ4 인코딩을 사용하여 수행됩니다.

#### **MQCOMPRESS\_ANY**

큐 관리자가 지원하는 모든 압축 기술을 메시지 압축에 사용할 수 있습니다. MQCOMPRESS\_ANY는 수신자, 요청자 및 서버 연결 채널에만 유효합니다.

#### **MQCOMPRESS\_NOT\_AVAILABLE**

목록에서 사용하지 않는 값은 이 값으로 설정됩니다.

엑시트의 입력 필드입니다. *Version*이 MQCD\_VERSION\_8 미만이면 이 필드가 존재하지 않습니다.

#### *MsgExit (MQCHARn)*

이 필드는 채널 메시지 엑시트 이름을 지정합니다.

이름이 공백이 아닐 경우 엑시트는 다음 시기에 호출됩니다.

- 전송 큐(송신자 또는 서버)에서 메시지를 검색한 직후 또는 메시지를 목적지 큐(수신자 또는 요청자)에 넣기 직전

엑시트에는 수정을 위해 전체 애플리케이션 메시지 및 전송 큐 헤더가 주어집니다.

- 채널의 초기설정 및 종료 시점.

이 필드는 *ChannelType*이 MQCMT\_SVRCONN 또는 MQCMT\_CLNTCONN인 채널과는 관련이 없습니다. 이러한 채널에 대해서는 메시지 엑시트가 호출되지 않습니다.

다양한 환경에서 이 필드의 콘텐츠에 대한 설명은 1409 페이지의 『MQCD - 채널 정의』의 내용을 참조하십시오.

이 필드의 길이는 MQ\_EXIT\_NAME\_LENGTH에서 제공됩니다.

**참고:** 이 상수의 값은 환경에 따라 달라집니다.

#### *MsgExitPtr (MQPTR)*

이 필드는 첫 번째 *MsgExit* 필드의 주소를 지정합니다.

*MsgExitsDefined*가 0보다 크면 이 주소는 체인에 있는 각 채널 메시지 엑시트 이름 목록의 주소입니다.

*ExitNameLength* 길이의 필드에서 각 이름의 오른쪽은 공백으로 채워집니다. 각 엑시트에 하나씩, 차례로 인접한 *MsgExitsDefined* 필드가 있습니다.

엑시트에 의한 이름 변경사항은 메시지 채널 엑시트에서 명확하게 조치를 취하지 않더라도 보존됩니다. 호출되는 엑시트에는 변화가 없습니다.

*MsgExitsDefined*가 0이면 이 필드가 널(Null) 포인터입니다.

프로그래밍 언어가 포인터 데이터 유형을 지원하지 않는 플랫폼에서 이 필드는 적절한 길이의 바이트 문자열로 선언됩니다.

엑시트의 입력 필드입니다. 이 필드는 *Version*이 MQCD\_VERSION\_4 미만일 경우에는 표시되지 않습니다.

#### *MsgExitsDefined (MQLONG)*

이 필드는 체인에 정의된 채널 메시지 엑시트의 수를 지정합니다.

이는 0보다 크거나 같습니다.

엑시트의 입력 필드입니다. 이 필드는 *Version*이 MQCD\_VERSION\_4 미만일 경우에는 표시되지 않습니다.

#### *MsgRetryCount (MQLONG)*

이 필드는 첫 번째 시도에 실패한 후 MCA가 메시지를 넣으려고 시도하는 횟수를 지정합니다.

이 필드는 첫 번째 MQOPEN 또는 MQPUT이 완료 코드 MQCC\_FAILED와 함께 실패하면 MCA가 열기 또는 넣기 조작을 시도하는 횟수를 표시합니다. *MsgRetryExit*가 공백인지 아닌지에 따라 이 속성의 영향이 달라집니다.

- *MsgRetryExit*가 공백이면 **MsgRetryCount** 속성이 MCA가 재시도하는지 여부를 제어합니다. 속성 값이 0이면 재시도하지 않습니다. 속성 값이 0보다 크면 **MsgRetryInterval** 속성이 지정한 간격으로 재시도합니다.

다음 이유 코드의 경우에만 재시도합니다.

- MQRC\_PAGESET\_FULL
- MQRC\_PUT\_INHIBITED
- MQRC\_Q\_FULL

다른 이유 코드의 경우, MCA는 실패 메시지를 재시도하지 않고 즉시 정상 실패 처리를 진행합니다.

- *MsgRetryExit*가 공백이 아니면 **MsgRetryCount** 속성이 MCA에 영향을 주지 않습니다. 대신 메시지 재시도 엑시트에 따라 재시도 횟수 및 재시도 간격이 결정됩니다. **MsgRetryCount** 속성이 0이더라도 엑시트가 호출됩니다.

**MsgRetryCount** 속성은 MQCD 구조의 엑시트에서 사용할 수 있지만, 이를 적용하기 위해 엑시트가 필요하지는 않습니다. 엑시트가 MQCXP의 *ExitResponse* 필드에 MQXCC\_SUPPRESS\_FUNCTION을 리턴할 때까지 무제한으로 계속 재시도합니다.

이 필드는 *ChannelType*이 MQCMT\_REQUESTER, MQCMT\_RECEIVER 또는 MQCMT\_CLUSRCVR인 채널에만 관련됩니다.

*Version*이 MQCD\_VERSION\_3 미만이면 이 필드가 표시되지 않습니다.

#### *MsgRetryExit (MQCHARn)*

이 필드는 채널 메시지 재시도 엑시트 이름을 지정합니다.

메시지 재시도 엑시트는 MCA에 MQOPEN 또는 MQPUT 호출의 MQCC\_FAILED 완료 코드가 수신되면 MCA가 호출하는 엑시트입니다. 엑시트의 목적은 MQOPEN 또는 MQPUT 조작을 다시 시도하기 전에 MCA가 대기하는 시간 간격을 지정하는 것입니다. 또는, 조작을 다시 시도하지 않도록 엑시트를 설정할 수 있습니다.

완료 코드가 MQCC\_FAILED인 모든 이유 코드에 대해 엑시트가 호출됩니다. 엑시트 설정에 따라 MCA에서 다시 시도하도록 하는 이유 코드, 시도 횟수 및 시간 간격이 판별됩니다.

조작을 더 이상 시도하지 않으면 MCA에서 정상적인 실패 처리를 수행합니다. 이 처리에는 예외 보고 메시지를 생성하고(송신자가 설정한 경우), (송신자의 MQRO\_DEAD\_LETTER\_Q 또는 MQRO\_DISCARD\_MSG 지정 여부에 따라) 원래 메시지를 데드-레터 큐에 넣거나 메시지를 버리는 작업이 포함됩니다. 데드-레터 큐와 관련된 실패(예: 데드-레터 큐가 가득 참)로 인해 메시지 재시도 엑시트가 호출되지는 않습니다.

엑시트 이름이 공백이 아니면 다음 시점에 엑시트가 호출됩니다.

- 메시지를 다시 전달하기 전에 대기를 수행하기 직전
- 채널의 초기화 및 종료 시

다양한 환경에서 이 필드의 콘텐츠에 대한 설명은 1409 페이지의 『MQCD - 채널 정의』의 내용을 참조하십시오.

이 필드는 *ChannelType*이 MQCHT\_REQUESTER, MQCHT\_RECEIVER 또는 MQCHT\_CLUSRCVR인 채널에만 관련됩니다.

이 필드의 길이는 MQ\_EXIT\_NAME\_LENGTH에서 제공합니다.

**참고:** 이 상수의 값은 환경에 따라 달라집니다.

*Version*이 MQCD\_VERSION\_3 미만이면 이 필드가 표시되지 않습니다.

#### *MsgRetryInterval (MQLONG)*

이 필드는 열기 또는 넣기 조작을 재시도하는 최소 간격(밀리초)을 지정합니다.

*MsgRetryExit*가 공백인지 아닌지에 따라 이 속성의 영향이 달라집니다.

- *MsgRetryExit*가 공백이면 **MsgRetryInterval** 속성이 첫 번째 MQOPEN 또는 MQPUT이 완료 코드 MQCC\_FAILED와 함께 실패한 경우 메시지를 재시도하기까지 MCA가 대기하는 최소 시간을 지정합니다. 값이 0이면 이전 시도 후 최대한 빨리 재시도를 수행합니다. *MsgRetryCount*가 0보다 클 때에만 재시도가 수행됩니다.

또한 메시지 재시도 엑시트가 MQCXP의 *MsgRetryInterval* 필드에 유효하지 않은 값을 리턴하면 이 속성은 대기 시간으로 사용됩니다.

- *MsgRetryExit*가 공백이 아니면 **MsgRetryInterval** 속성이 MCA에 영향을 주지 않습니다. 대신, 메시지 재시도 엑시트가 MCA의 대기 시간을 판별합니다. **MsgRetryInterval** 속성은 MQCD 구조의 엑시트에서 사용할 수 있지만, 이를 적용하기 위해 엑시트가 필요하지는 않습니다.

값의 범위는 0 - 999 999 999입니다.

이 필드는 *ChannelType*이 MQCHT\_REQUESTER, MQCHT\_RECEIVER 또는 MQCHT\_CLUSRCVR인 채널에만 관련됩니다.

*Version*이 MQCD\_VERSION\_3 미만이면 이 필드가 표시되지 않습니다.

*Version*이 MQCD\_VERSION\_4 미만이면 이 구조의 다음 필드가 표시되지 않습니다.

#### *MsgRetryUserData (MQCHAR32)*

이 필드는 채널 메시지 재시도 엑시트 사용자 데이터를 지정합니다.

이 데이터는 **ChannelExitParms** 매개변수의 *ExitData* 필드에 있는 채널 메시지 재시도 엑시트에 전달됩니다(MQ\_CHANNEL\_EXIT 참조).

초기에 이 필드에는 채널 정의에 설정되어 있는 데이터가 포함됩니다. 그러나 MCA 인스턴스의 수명 중에 임의의 유형의 엑시트에 의해 이 필드의 콘텐츠에 대해 작성된 변경사항은 MCA에 의해 보존되며 이 MCA 인스턴스에 대한 엑시트(유형 무관)의 후속 호출에 표시됩니다. 해당 변경사항은 기타 MCA 인스턴스가 사용하는 채널 정의에 영향을 주지 않습니다. 임의의 문자(2진 데이터 포함)를 사용할 수 있습니다.

이 필드는 *ChannelType*이 MQCHT\_REQUESTER, MQCHT\_RECEIVER 또는 MQCHT\_CLUSRCVR인 채널에만 관련됩니다.

이 필드의 길이는 MQ\_EXIT\_DATA\_LENGTH에서 제공합니다. *Version*이 MQCD\_VERSION\_3 미만이면 이 필드가 표시되지 않습니다.

이 필드는 IBM MQ for IBM i에서 관련이 없습니다.

#### *MsgUserData (MQCHAR32)*

이 필드는 채널 메시지 엑시트 사용자 데이터를 지정합니다.

이 데이터는 **ChannelExitParms** 매개변수의 *ExitData* 필드에 있는 채널 메시지 엑시트에 전달됩니다 (MQ\_CHANNEL\_EXIT 참조).

초기에 이 필드에는 채널 정의에 설정되어 있는 데이터가 포함됩니다. 그러나 MCA 인스턴스의 수명 중에 임의의 유형의 엑시트에 의해 이 필드의 콘텐츠에 대해 작성된 변경사항은 MCA에 의해 보존되며 이 MCA 인스턴스에 대한 엑시트(유형 무관)의 후속 호출에 표시됩니다. 해당 변경사항은 기타 MCA 인스턴스가 사용하는 채널 정의에 영향을 주지 않습니다. 임의의 문자(2진 데이터 포함)를 사용할 수 있습니다.

이 필드의 길이는 MQ\_EXIT\_DATA\_LENGTH에서 제공합니다.

이 필드는 IBM MQ for IBM i에서 관련이 없습니다.

#### *MsgUserDataPtr (MQPTR)*

이 필드는 첫 번째 *MsgUserData* 필드의 주소를 지정합니다.

*MsgExitsDefined*가 0보다 크면 이 주소는 체인에 있는 각 채널 메시지 엑시트에 대한 사용자 데이터 항목 목록의 주소입니다.

*ExitDataLength* 길이 필드에서 각 사용자 데이터 항목의 오른쪽은 공백으로 채워집니다. 각 엑시트에 하나씩, 차례로 인접한 *MsgExitsDefined* 필드가 있습니다. 정의된 사용자 데이터 항목의 수가 엑시트 이름의 수보다 적은 경우, 정의되지 않은 사용자 데이터 항목은 공백으로 설정됩니다. 반대로, 정의된 사용자 데이터 항목의 수가 엑시트 이름의 수보다 많은 경우, 초과 사용자 데이터 항목은 무시되며 엑시트에 표시되지 않습니다.

엑시트에 의한 이들 값의 변경사항은 보존됩니다. 따라서 한 엑시트가 정보를 다른 엑시트로 전달할 수 있습니다. 변경사항에 대해 어떠한 확인 작업도 수행되지 않으므로 예를 들어, 필요한 경우 2진 데이터를 이러한 필드에 쓸 수 있습니다.

*MsgExitsDefined*가 0이면 이 필드가 널(Null) 포인터입니다.

프로그래밍 언어가 포인터 데이터 유형을 지원하지 않는 플랫폼에서 이 필드는 적절한 길이의 바이트 문자열로 선언됩니다.

엑시트의 입력 필드입니다. 이 필드는 *Version*이 MQCD\_VERSION\_4 미만일 경우에는 표시되지 않습니다.

#### *NetworkPriority (MQLONG)*

이 필드는 채널에 대한 네트워크 연결의 우선순위를 지정합니다.

특정 목적지에 대한 경로가 여러 개이면 우선순위가 가장 높은 경로가 선택됩니다. 값의 범위는 0 - 9이며, 0이 우선순위가 가장 낮습니다.

이 필드는 *ChannelType*이 MQCHT\_CLUSSDR 또는 MQCHT\_CLUSRCVR인 채널에만 해당됩니다.

엑시트의 입력 필드입니다. 이 필드는 *Version*이 MQCD\_VERSION\_5 미만인 경우에는 표시되지 않습니다.

*Version*이 MQCD\_VERSION\_6 미만이면 이 구조의 다음 필드가 표시되지 않습니다.

#### *NonPersistentMsgSpeed (MQLONG)*

이 필드는 비지속 메시지가 채널을 통과하는 속도를 지정합니다.

이 필드는 *ChannelType*이 MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_RECEIVER, MQCHT\_REQUESTER, MQCHT\_CLUSSDR 또는 MQCHT\_CLUSRCVR인 채널만 관련이 있습니다.

값은 다음 중 하나입니다.

#### **MQNPMS\_NORMAL**

정상 속도입니다.

채널이 MQNPMS\_NORMAL이 되도록 정의된 경우, 비지속 메시지가 정상 속도로 채널을 이동합니다. 채널 실패 발생 시 메시지가 손실되지 않는다는 이점이 있습니다. 또한 동일한 전송 큐에 있는 지속 메시지 및 비지속 메시지가 서로 상대적 순서를 유지합니다.

#### **MQNPMS\_FAST**

빠른 속도입니다.

채널이 MQNPMS\_FAST가 되도록 정의된 경우, 비지속 메시지가 빠른 속도로 채널을 이동합니다. 채널의 처리량이 향상되지만, 채널 실패 발생 시 비지속 메시지가 손실됩니다. 또한 비지속 메시지가 동일한 전송 큐에서 대기 중인 지속 메시지보다 먼저 이동할 수 있습니다. 즉, 지속 메시지 대비 비지속 메시지의 순서가 유지되지 않습니다. 그러나 비지속 메시지들은 서로 간에 순서가 유지됩니다. 마찬가지로, 지속 메시지들도 서로 간에 순서가 유지됩니다.

#### *Password (MQCHAR12)*

이 필드는 보안 SNA 세션을 리모트 메시지 채널 에이전트로 시작할 때 메시지 채널 에이전트에 사용되는 매개변수를 지정합니다.

이 필드는 AIX, Linux, and Windows에서만 공백이 아닐 수 있으며 *ChannelType*이 MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_REQUESTER 또는 MQCHT\_CLNTCONN인 채널에만 관련됩니다. z/OS에서는 이 필드가 관련이 없습니다.

이 필드의 길이는 MQ\_PASSWORD\_LENGTH에서 제공합니다. 그러나 처음 10자만 사용합니다.

*Version*이 MQCD\_VERSION\_2 미만이면 이 필드가 표시되지 않습니다.

#### *PropertyControl (MQLONG)*

이 필드는 메시지를 V6 또는 이전 큐 관리자(특성 디스크립터의 개념을 모르는 큐 관리자)에 전송하려고 할 때 메시지의 특성에 발생하는 항목을 지정합니다.

가능한 값은 다음 값 중 하나입니다.

#### **MQPROP\_COMPATIBILITY**

메시지에 **mcd.**, **jms.**, **usr.** 또는 **mqext.** 접두부의 특성이 포함되면 모든 메시지 특성이 MQRFH2 헤더에서 애플리케이션에 전달됩니다. 그렇지 않으면, 메시지 디스크립터(또는 확장자)에 포함된 특성을 제외한 메시지의 모든 특성이 제거되며 더 이상 애플리케이션에 액세스할 수 없습니다.

이 값이 기본값이며, JMS 관련 특성이 메시지 데이터의 MQRFH2 헤더에 있어야 하는 애플리케이션이 수정 없이 계속 작동할 수 있습니다.

#### **MQPROP\_NONE**

메시지 디스크립터(또는 확장자)의 특성을 제외한 메시지의 모든 특성은 메시지가 리모트 큐 관리자로 송신되기 전에 메시지에서 제거됩니다.

#### **MQPROP\_ALL**

메시지가 리모트 큐 관리자에게 송신될 때 메시지의 모든 특성이 메시지에 포함됩니다. 특성(메시지 디스크립터(또는 확장자)의 특성 제외)은 메시지 데이터에서 하나 이상의 MQRFH2 헤더에 배치됩니다.

이 속성은 송신자, 서버, 클러스터 송신자 및 클러스터 수신자 채널에 적용 가능합니다.

128 페이지의 『MQIA \*(정수 속성 선택자)』

168 페이지의 『MQPROP \*(큐 및 채널 특성 제어 값 및 최대 특성 길이)』

#### *PutAuthority (MQLONG)*

이 필드는 메시지와 연관된 컨텍스트 정보의 사용자 ID가 목적지 큐에 메시지를 넣을 수 있는 권한을 설정하는 데 사용되는지 여부를 지정합니다.

이 필드는 *ChannelType*이 MQCHT\_REQUESTER, MQCHT\_RECEIVER 또는 MQCHT\_CLUSRCVR인 채널에만 관련됩니다. 다음 중 하나입니다.

#### **MQPA\_DEFAULT**

기본 사용자 ID가 사용됩니다.

#### **MQPA\_CONTEXT**

컨텍스트 사용자 ID가 사용됩니다.

#### **z/OS MQPA\_ALTERNATE\_OR\_MCA**

메시지 디스크립터의 *UserIdentifier* 필드에 있는 사용자 ID가 사용됩니다. 네트워크에서 수신된 사용자 ID는 사용되지 않습니다. 이 값은 z/OS에서만 지원됩니다.

#### **z/OS MQPA\_ONLY\_MCA**

디폴트 사용자 ID가 사용됩니다. 네트워크에서 수신된 사용자 ID는 사용되지 않습니다. 이 값은 z/OS에서만 지원됩니다.

#### *QMGrName (MQCHAR48)*

이 필드는 엑시트가 연결할 수 있는 큐 관리자의 이름을 지정합니다.

*ChannelType*이 MQCHT\_CLNTCONN 이외인 채널의 경우, 이 필드는 엑시트가 연결할 수 있는 큐 관리자의 이름이며 AIX, Linux, and Windows에서는 항상 공백이 아닙니다.

이 필드의 길이는 MQ\_Q\_MGR\_NAME\_LENGTH로 제공됩니다.

#### *ReceiveExit (MQCHARn)*

이 필드는 채널 수신 엑시트 이름을 지정합니다.

이름이 공백이 아닐 경우 엑시트는 다음 시기에 호출됩니다.

- 수신된 네트워크 데이터가 처리되기 바로 직전.

엑시트에는 전체 전송 버퍼가 수신된 그대로 주어집니다. 필요에 따라 버퍼의 내용은 수정할 수 있습니다.

- 채널의 초기설정 및 종료 시점.

다양한 환경에서 이 필드의 콘텐츠에 대한 설명은 1409 페이지의 『MQCD - 채널 정의』의 내용을 참조하십시오.

이 필드의 길이는 MQ\_EXIT\_NAME\_LENGTH에서 제공합니다.

**참고:** 이 상수의 값은 환경에 따라 달라집니다.

#### *ReceiveExitPtr (MQPTR)*

이 필드는 첫 번째 *ReceiveExit* 필드의 주소를 지정합니다.

*ReceiveExitsDefined*가 0보다 크면 이 주소는 체인에 있는 각 채널 수신 엑시트 이름 목록의 주소입니다.

*ExitNameLength* 길이의 필드에서 각 이름의 오른쪽은 공백으로 채워집니다. 서로 인접한

*ReceiveExitsDefined* 필드가 각 엑시트에 하나씩 있습니다.

엑시트에 의한 이름 변경사항은 메시지 채널 엑시트에서 명확하게 조치를 취하지 않더라도 보존됩니다. 호출되는 엑시트에는 변화가 없습니다.

*ReceiveExitsDefined*가 0이면 이 필드는 널(null) 포인터입니다.

프로그래밍 언어가 포인터 데이터 유형을 지원하지 않는 플랫폼에서 이 필드는 적절한 길이의 바이트 문자열로 선언됩니다.

엑시트의 입력 필드입니다. 이 필드는 *Version*이 MQCD\_VERSION\_4 미만일 경우에는 표시되지 않습니다.

#### *ReceiveExitsDefined (MQLONG)*

이 필드는 체인에 정의된 채널 수신 엑시트의 수를 지정합니다.

이는 0보다 크거나 같습니다.

엑시트의 입력 필드입니다. 이 필드는 *Version*이 MQCD\_VERSION\_4 미만일 경우에는 표시되지 않습니다.



### *ReceiveUserData (MQCHAR32)*

이 채널은 채널 수신 엑시트 사용자 데이터를 지정합니다.

이 데이터는 **ChannelExitParms** 매개변수의 *ExitData* 필드에 있는 채널 수신 엑시트에 전달됩니다 (MQ\_CHANNEL\_EXIT 참조).

초기에 이 필드에는 채널 정의에 설정되어 있는 데이터가 포함됩니다. 그러나 MCA 인스턴스의 수명 중에 임의의 유형의 엑시트에 의해 이 필드의 콘텐츠에 대해 작성된 변경사항은 MCA에 의해 보존되며 이 MCA 인스턴스에 대한 엑시트(유형 무관)의 후속 호출에 표시됩니다. 이 항목이 다른 대화의 엑시트에 적용됩니다. 해당 변경사항은 기타 MCA 인스턴스가 사용하는 채널 정의에 영향을 주지 않습니다. 임의의 문자(2진 데이터 포함)를 사용할 수 있습니다.

이 필드의 길이는 MQ\_EXIT\_DATA\_LENGTH에서 제공합니다.

이 필드는 IBM MQ for IBM i에서 관련이 없습니다.

*Version*이 MQCD\_VERSION\_2 미만이면 이 구조의 다음 필드가 표시되지 않습니다.

### *ReceiveUserDataPtr (MQPTR)*

이 필드는 첫 번째 *ReceiveUserData* 필드의 주소를 지정합니다.

*ReceiveExitsDefined*가 0보다 크면 이 주소는 체인에 있는 각 채널 수신 엑시트에 대한 사용자 데이터 항목 목록의 주소입니다.

*ExitDataLength* 길이 필드에서 각 사용자 데이터 항목의 오른쪽은 공백으로 채워집니다. 서로 인접한 *ReceiveExitsDefined* 필드가 각 엑시트에 하나씩 있습니다. 정의된 사용자 데이터 항목의 수가 엑시트 이름의 수보다 적은 경우, 정의되지 않은 사용자 데이터 항목은 공백으로 설정됩니다. 반대로, 정의된 사용자 데이터 항목의 수가 엑시트 이름의 수보다 많은 경우, 초과 사용자 데이터 항목은 무시되며 엑시트에 표시되지 않습니다.

엑시트에 의한 이들 값의 변경사항은 보존됩니다. 따라서 한 엑시트가 정보를 다른 엑시트로 전달할 수 있습니다. 변경사항에 대해 어떠한 확인 작업도 수행되지 않으므로 예를 들어, 필요한 경우 2진 데이터를 이러한 필드에 쓸 수 있습니다.

*ReceiveExitsDefined*가 0이면 이 필드는 널(null) 포인터입니다.

프로그래밍 언어가 포인터 데이터 유형을 지원하지 않는 플랫폼에서 이 필드는 적절한 길이의 바이트 문자열로 선언됩니다.

엑시트의 입력 필드입니다. 이 필드는 *Version*이 MQCD\_VERSION\_4 미만일 경우에는 표시되지 않습니다.

*Version*이 MQCD\_VERSION\_5 미만이면 이 구조의 다음 필드가 표시되지 않습니다.

### *RemotePassword (MQCHAR12)*

이 필드는 파트너로부터의 비밀번호를 지정합니다.

*ChannelType*이 MQCHT\_CLNTCONN 또는 MQCHT\_SVRCONN인 경우에만 유효한 정보가 포함됩니다.

- MQCHT\_CLNTCONN 채널의 보안 엑시트인 경우, 이 비밀번호는 환경에서 가져온 비밀번호입니다. 엑시트는 서버의 보안 엑시트에 이를 송신하도록 선택할 수 있습니다.
- MQCHT\_SVRCONN 채널의 보안 엑시트인 경우, 이 필드에는 클라이언트 보안 엑스트가 없는 경우에 한해 클라이언트의 환경에서 가져온 비밀번호가 있을 수 있습니다. 엑시트는 이 비밀번호를 사용하여 *RemoteUserIdentifier*에서 사용자 ID의 유효성을 검증합니다.

클라이언트에 보안 엑시트가 없으면 클라이언트로부터의 보안 플로우에서 이 정보를 가져올 수 있습니다.

이 필드의 길이는 MQ\_PASSWORD\_LENGTH에서 제공합니다. *Version*이 MQCD\_VERSION\_2 미만이면 이 필드가 표시되지 않습니다.

### *RemoteSecurityId (MQBYTE40)*

이 필드는 리모트 사용자에게 대한 보안 ID를 지정합니다.

이 필드는 *ChannelType*이 MQCHT\_CLNTCONN 또는 MQCHT\_SVRCONN인 채널에만 해당됩니다.

다음 특수 값은 보안 ID가 없음을 나타냅니다.

## MQSID\_NONE

보안 ID가 지정되지 않음.

이 값은 필드 길이 만큼의 2진 0입니다.

C 프로그래밍 언어의 경우, 상수 MQSID\_NONE\_ARRAY도 정의됩니다. 이 상수는 MQSID\_NONE과 동일한 값을 갖지만, 문자열이 아니라 문자 배열입니다.

엑시트의 입력 필드입니다. 이 필드 길이는 MQ\_SECURITY\_ID\_LENGTH에서 제공합니다. *Version*이 MQCD\_VERSION\_6 미만이면 이 필드가 표시되지 않습니다.

*Version*이 MQCD\_VERSION\_7 미만이면 이 구조의 다음 필드가 표시되지 않습니다.

### *RemoteUserIdentifier* (MQCHAR12)

이 필드는 파트너로부터 사용자 ID의 처음 12바이트를 지정합니다.

다음은 리모트 사용자 ID를 포함하는 두 개의 필드입니다.

- *RemoteUserIdentifier*는 리모트 사용자 ID의 처음 12바이트를 포함하며, ID가 12바이트보다 짧으면 공백으로 채워집니다. *RemoteUserIdentifier*는 공백일 수 있습니다.
- *LongRemoteUserIdPtr*는 전체 리모트 사용자 ID를 가리키며, 12바이트보다 길 수 있습니다. 길이는 *LongRemoteUserIdLength*에서 제공합니다. 전체 ID는 후미 공백이 없으며 널(Null) 종료되지 않습니다. ID가 공백이면 *LongRemoteUserIdLength*가 0이고 *LongRemoteUserIdPtr*의 값이 정의되지 않습니다.

*LongRemoteUserIdPtr*은 *Version*이 MQCD\_VERSION\_6 미만이면 표시되지 않습니다.

리모트 사용자 ID는 *ChannelType*이 MQCHT\_CLNTCONN 또는 MQCHT\_SVRCONN인 채널만 관련이 있습니다.

- MQCHT\_CLNTCONN 채널의 보안 엑시트인 경우, 이 값은 환경에서 가져온 사용자 ID입니다. 엑시트는 서버의 보안 엑시트에 이를 송신하도록 선택할 수 있습니다.
- MQCHT\_SVRCONN 채널의 보안 엑시트인 경우, 클라이언트 보안 엑스트가 없으면 이 필드에 클라이언트의 환경에서 가져온 사용자 ID가 있을 수 있습니다. 엑시트는 이 사용자 ID(*RemotePassword*의 비밀번호 포함 가능)의 유효성을 검증하고 *MCAUserIdentifier*에서 값을 업데이트할 수 있습니다.

클라이언트에 보안 엑시트가 없으면 클라이언트로부터의 보안 플로우에서 이 정보를 가져올 수 있습니다.

이 필드의 길이는 MQ\_USER\_ID\_LENGTH에 지정되어 있습니다. *Version*이 MQCD\_VERSION\_2 미만이면 이 필드가 표시되지 않습니다.

### *SecurityExit* (MQCHARn)

이 필드는 채널 보안 엑시트 이름을 지정합니다.

이름이 공백이 아닐 경우 엑시트는 다음 시기에 호출됩니다.

- 채널을 설정한 직후.  
메시지가 전송되기 전에 연결 권한을 확인하기 위한 보안 플로우를 발생시킬 수 있는 기회가 엑시트에 주어집니다.
- 보안 메시지 플로우에 대한 응답을 받은 시점.  
리모트 시스템의 리모트 처리기에서 수신된 보안 메시지 플로우가 엑시트에 제공됩니다.
- 채널의 초기설정 및 종료 시점.

다양한 환경에서 이 필드의 콘텐츠에 대한 설명은 1409 페이지의 『MQCD - 채널 정의』의 내용을 참조하십시오.

이 필드의 길이는 MQ\_EXIT\_NAME\_LENGTH에서 제공합니다.

**참고:** 이 상수의 값은 환경에 따라 달라집니다.

### *SecurityUserData* (MQCHAR32)

이 채널은 채널 보안 엑시트 사용자 데이터를 지정합니다.



이 데이터는 **ChannelExitParms** 매개변수의 *ExitData* 필드에 있는 채널 보안 엑시트에 전달됩니다 (MQ\_CHANNEL\_EXIT 참조).

초기에 이 필드에는 채널 정의에 설정되어 있는 데이터가 포함됩니다. 그러나 MCA 인스턴스의 수명 중에 임의의 유형의 엑시트에 의해 이 필드의 콘텐츠에 대해 작성된 변경사항은 MCA에 의해 보존되며 이 MCA 인스턴스에 대한 엑시트(유형 무관)의 후속 호출에 표시됩니다. 이 항목이 다른 대화의 엑시트에 적용됩니다. 이러한 변경사항은 기타 MCA 인스턴스에 사용되는 채널 정의에 영향을 주지 않습니다. 임의의 문자(2진 데이터 포함)를 사용할 수 있습니다.

이 필드의 길이는 MQ\_EXIT\_DATA\_LENGTH에서 제공합니다.

이 필드는 IBM MQ for IBM i에서 관련이 없습니다.

#### *SendExit (MQCHARn)*

이 필드는 채널 송신 엑시트 이름을 지정합니다.

이름이 공백이 아닐 경우 엑시트는 다음 시기에 호출됩니다.

- 데이터가 네트워크상에서 송신된 직후.

데이터가 전송되기 전에 엑시트에 전체 전송 버퍼가 주어집니다. 필요에 따라 버퍼의 내용은 수정할 수 있습니다.

- 채널의 초기설정 및 종료 시점.

다양한 환경에서 이 필드의 콘텐츠에 대한 설명은 1409 페이지의 『MQCD - 채널 정의』의 내용을 참조하십시오.

이 필드의 길이는 MQ\_EXIT\_NAME\_LENGTH에서 제공합니다.

**참고:** 이 상수의 값은 환경에 따라 달라집니다.

#### *SendExitPtr (MQPTR)*

이 필드는 첫 번째 *SendExit* 필드의 주소를 지정합니다.

*SendExitsDefined*가 0보다 크면 이 주소는 체인에 있는 각 채널 송신 엑시트 이름 목록의 주소입니다.

*ExitNameLength* 길이의 필드에서 각 이름의 오른쪽은 공백으로 채워집니다. 서로 인접한 *SendExitsDefined* 필드가 각 엑시트에 하나씩 있습니다.

엑시트에 의한 이름 변경사항은 메시지 송신 엑시트에서 명확하게 조치를 취하지 않더라도 보존됩니다. 호출되는 엑시트에는 변화가 없습니다.

*SendExitsDefined*가 0이면 이 필드가 널(Null) 포인터입니다.

프로그래밍 언어가 포인터 데이터 유형을 지원하지 않는 플랫폼에서 이 필드는 적절한 길이의 바이트 문자열로 선언됩니다.

엑시트의 입력 필드입니다. 이 필드는 *Version*이 MQCD\_VERSION\_4 미만일 경우에는 표시되지 않습니다.

#### *SendExitsDefined (MQLONG)*

이 필드는 체인에 정의된 채널 송신 엑시트의 수를 지정합니다.

이는 0보다 크거나 같습니다.

엑시트의 입력 필드입니다. 이 필드는 *Version*이 MQCD\_VERSION\_4 미만일 경우에는 표시되지 않습니다.

#### *SendUserData (MQCHAR32)*

이 필드는 채널 송신 엑시트 사용자 데이터를 지정합니다.

이 데이터는 **ChannelExitParms** 매개변수의 *ExitData* 필드에 있는 채널 송신 엑시트에 전달됩니다 (MQ\_CHANNEL\_EXIT 참조).

초기에 이 필드에는 채널 정의에 설정되어 있는 데이터가 포함됩니다. 그러나 MCA 인스턴스의 수명 중에 임의의 유형의 엑시트에 의해 이 필드의 콘텐츠에 대해 작성된 변경사항은 MCA에 의해 보존되며 이 MCA 인스턴스에 대한 엑시트(유형 무관)의 후속 호출에 표시됩니다. 이 항목이 다른 대화의 엑시트에 적용됩니다. 해당 변경사항은 기타 MCA 인스턴스가 사용하는 채널 정의에 영향을 주지 않습니다. 임의의 문자(2진 데이터 포함)를 사용할 수 있습니다.

이 필드의 길이는 MQ\_EXIT\_DATA\_LENGTH에서 제공합니다.

이 필드는 IBM MQ for IBM i에서 관련이 없습니다.

#### *SendUserDataPtr (MQPTR)*

이 필드는 *SendUserData* 필드의 주소를 지정합니다.

*SendExitsDefined*가 0보다 크면 이 주소는 체인에 있는 각 채널 메시지 엑시트에 대한 사용자 데이터 항목 목록의 주소입니다.

*ExitDataLength* 길이 필드에서 각 사용자 데이터 항목의 오른쪽은 공백으로 채워집니다. 각 엑시트에 하나 씩, 차례로 인접한 *MsgExitsDefined* 필드가 있습니다. 정의된 사용자 데이터 항목의 수가 엑시트 이름의 수보다 적은 경우, 정의되지 않은 사용자 데이터 항목은 공백으로 설정됩니다. 반대로, 정의된 사용자 데이터 항목의 수가 엑시트 이름의 수보다 많은 경우, 초과 사용자 데이터 항목은 무시되며 엑시트에 표시되지 않습니다.

엑시트에 의한 이들 값의 변경사항은 보존됩니다. 따라서 한 엑시트가 정보를 다른 엑시트로 전달할 수 있습니다. 변경사항에 대해 어떠한 확인 작업도 수행되지 않으므로 예를 들어, 필요한 경우 2진 데이터를 이러한 필드에 쓸 수 있습니다.

*SendExitsDefined*가 0이면 이 필드가 널(Null) 포인터입니다.

프로그래밍 언어가 포인터 데이터 유형을 지원하지 않는 플랫폼에서 이 필드는 적절한 길이의 바이트 문자열로 선언됩니다.

엑시트의 입력 필드입니다. 이 필드는 *Version*이 MQCD\_VERSION\_4 미만일 경우에는 표시되지 않습니다.

#### *SeqNumberWrap (MQLONG)*

이 필드는 허용되는 가장 높은 메시지 순서 번호를 지정합니다.

이 값에 도달하면, 순서 번호가 1에서 다시 시작할 수 있도록 줄바꿈됩니다.

이 값은 협상할 수 없으며 로컬 및 리모트 채널 정의에서 일치해야 합니다.

이 필드는 *ChannelType*이 MQCHT\_SVRCONN 또는 MQCHT\_CLNTCONN인 채널과는 관련이 없습니다.

#### *SharingConversations (MQLONG)*

이 필드는 이 채널과 연관된 채널 인스턴스를 공유할 수 있는 최대 대화 수를 지정합니다.

이 필드는 클라이언트 연결 및 서버 연결 채널에서 사용됩니다.

값이 0이면 다음 속성과 관련하여 채널이 IBM WebSphere MQ 7.0 이전 버전에서와 똑같이 작동합니다.

- 대화 공유
- 미리 읽기
- STOP CHANNEL(*channelname*) MODE(QUIESCE)
- 하트비트
- 클라이언트 비동기 이용

값 1은 IBM MQ 작동을 위한 최소값입니다. 채널 인스턴스에서 하나의 대화만 허용되지만 미리 읽기, 비동기 이용 및 CLNTCONN-SVRCONN 하트비트 및 대기 채널 중지 동작을 사용할 수 있습니다.

엑시트의 입력 필드입니다. *Version*이 MQCD\_VERSION\_9 미만이면 표시되지 않습니다.

이 필드의 기본값은 10입니다.

**참고:** 채널에 적용된 *MaxInstances* 및 *MaxInstancesPerClient* 한계는 채널 인스턴스를 공유할 수 있는 대화 수가 아니라 채널 인스턴스 수를 제한합니다.

#### *ShortConnectionName (MQCHAR20)*

이 필드는 연결 이름의 첫 20바이트를 지정합니다.

*Version* 필드가 MQCD\_VERSION\_1이면 *ShortConnectionName*은 완전한 연결 이름을 포함합니다.

*Version* 필드가 MQCD\_VERSION\_2 이상이면 *ShortConnectionName*은 연결 이름의 첫 20자를 포함합니다. 완전한 연결 이름은 *ConnectionName* 필드에서 제공하며, *ShortConnectionName* 및 *ConnectionName*의 첫 20자는 동일합니다.

이 필드 내용에 대한 자세한 정보는 *ConnectionName*의 내용을 참조하십시오.

**참고:** 이 필드의 이름이 MQCD\_VERSION\_2와 후속 MQCD 버전에서 변경되었는데, 해당 필드를 이전에는 *ConnectionName*이라 했습니다.

이 필드의 길이는 MQ\_SHORT\_CONN\_NAME\_LENGTH에서 제공합니다.

#### *ShortRetryCount* (MQLONG)

이 필드는 리모트 시스템에 연결하려고 시도하는 최대 횟수를 지정합니다.

이 필드는 (대개 더 긴) *LongRetryCount* 및 *LongRetryInterval*을 사용하기 전에 *ShortRetryInterval*이 지정한 간격으로 리모트 시스템에 연결하기 위해 시도하는 최대 횟수입니다.

이 필드는 *ChannelType*이 MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_CLUSSDR 또는 MQCHT\_CLUSRCVR인 채널에만 관련됩니다.

#### *ShortRetryInterval* (MQLONG)

이 필드는 리모트 시스템에 연결을 재시도할 때까지 대기하는 최대 시간(초)을 지정합니다.

채널이 활성 상태가 되기를 대기해야 할 경우에는 재시도 사이의 간격이 확장될 수도 있습니다.

이 필드는 *ChannelType*이 MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_CLUSSDR 또는 MQCHT\_CLUSRCVR인 채널에만 관련됩니다.

#### *SPLProtection* (MQLONG)

This field specifies the value of the AMS security policy protection.

The value is one of the following:

##### **MQSPL\_PASSTHRU**

Pass through, unchanged, any messages sent or received by the MCA for this channel.

This value is relevant only for channels with a *ChannelType* of MQCHT\_SENDER, MQCHT\_SERVER, MQCHT\_RECEIVER, or MQCHT\_REQUESTER, and is the default value.

##### **MQSPL\_REMOVE**

Remove any AMS protection from messages retrieved from the transmission queue by the MCA, and send the messages to the partner.

This value is relevant only for channels with a *ChannelType* of MQCHT\_SENDER or MQCHT\_SERVER.

##### **MQSPL\_ ASPOLICY**

Based on the policy defined for the target queue, apply AMS protection to inbound messages prior to putting them on to the target queue.

This value is relevant only for channels with a *ChannelType* of MQCHT\_RECEIVER or MQCHT\_REQUESTER.


This is an input field to the exit. This field is not present if *Version* is less than MQCD\_VERSION\_12.

#### *SSLCipherSpec* (MQCHAR32)

이 필드는 TLS를 사용할 때 사용되는 암호 스펙을 지정합니다.

*SSLCipherSpec*이 공백이면 채널이 TLS를 사용하지 않습니다. 공백이 아니면 이 필드에는 사용 중인 CipherSpec을 지정하는 문자열이 있습니다.

이 매개변수는 모든 채널 유형에 대해 유효합니다. 이는 다음 플랫폼에서 지원됩니다.

-  AIX
-  IBM i

-  Linux
-  Windows
-  z/OS

채널 유형이 TCP 전송 유형(TRPTYPE)인 경우에만 유효합니다.

엑시트의 입력 필드입니다. 이 필드의 길이는 MQ\_SSL\_CIPHER\_SPEC\_LENGTH에서 제공합니다. *Version*이 MQCD\_VERSION\_7 미만이면 이 필드가 존재하지 않습니다.

#### *SSLClientAuth (MQLONG)*

이 필드는 TLS 클라이언트 인증의 필요 여부를 지정합니다.

이 필드는 SVRCONN 채널 정의만 관련이 있습니다.

다음 값 중 하나입니다.

#### **MQSCA\_REQUIRED**

클라이언트 인증은 필수입니다.

#### **MQSCA\_OPTIONAL**

클라이언트 인증은 선택사항입니다.

엑시트의 입력 필드입니다. *Version*이 MQCD\_VERSION\_7 미만이면 이 필드가 존재하지 않습니다.

#### *SSLPeerNameLength (MQLONG)*

이 필드는 *SSLPeerNamePtr*이 가리키는 TLS 피어 이름의 길이(바이트)를 지정합니다.

엑시트의 입력 필드입니다. *Version*이 MQCD\_VERSION\_7 미만이면 이 필드가 존재하지 않습니다.

#### *SSLPeerNamePtr (MQPTR)*

이 필드는 TLS 피어 이름의 주소를 지정합니다.

성공적인 TLS 데이터 교환 동안 인증서를 수신한 경우, 인증서를 수신한 채널 끝의 *SSLPeerNamePtr*에서 액세스하는 MQCD 필드에 인증서 제목의 식별 이름이 복사됩니다. 이렇게 하면 이 값이 로컬 사용자의 채널 정의에 있는 경우 채널의 *SSLPeerName* 값을 덮어쓰게 됩니다. 채널의 해당 끝에 보안 엑시트가 지정되면 MQCD의 피어 인증서로부터 식별 이름을 수신합니다.

엑시트의 입력 필드입니다. *Version*이 MQCD\_VERSION\_7 미만이면 이 필드가 존재하지 않습니다.

**참고:** IBM WebSphere MQ 7.1 릴리스 이전에 구성된 보안 엑시트 프로그램은 업데이트해야 할 수 있습니다. 자세한 정보는 [채널 보안 엑시트 프로그램](#)을 참조하십시오.

#### *StrucLength(MQLONG)*

이 필드는 MQCD 구조의 길이(바이트)를 지정합니다.

구조에 들어 있는 포인터 필드를 통해 처리된 문자열은 길이에 포함되지 않습니다. 값은 다음 중 하나입니다.

#### **MQCD\_LENGTH\_4**

버전-4 채널 정의 구조의 길이

#### **MQCD\_LENGTH\_5**

버전-5 채널 정의 구조의 길이

#### **MQCD\_LENGTH\_6**

버전-6 채널 정의 구조의 길이

#### **MQCD\_LENGTH\_7**

버전-7 채널 정의 구조의 길이

#### **MQCD\_LENGTH\_8**

버전-8 채널 정의 구조의 길이

#### **MQCD\_LENGTH\_9**

버전-9 채널 정의 구조의 길이

#### **MQCD\_LENGTH\_10**

버전-10 채널 정의 구조의 길이

## **MQCD\_LENGTH\_11**

버전-11 채널 정의 구조의 길이

## **z/OS MQCD\_LENGTH\_12**

버전-12 채널 정의 구조의 길이

다음 상수는 현재 버전의 길이를 지정합니다.

## **MQCD\_CURRENT\_LENGTH**

채널 정의 구조의 현재 버전 길이

**참고:** 이러한 상수의 값은 환경에 따라 고유합니다.

이 필드는 *Version*이 MQCD\_VERSION\_4 미만일 경우에는 표시되지 않습니다.

## *TpName (MQCHAR64)*

이 필드는 LU 6.2 트랜잭션 프로그램 이름을 지정합니다.

이 필드는 전송 프로토콜(*TransportType*)이 MQXPT\_LU62이고 *ChannelType*이 MQCHT\_SVRCONN 또는 MQCHT\_RECEIVER가 아닌 경우에만 관련이 있습니다.

이 필드는 대신 정보가 통신 부가 오브젝트에 포함된 플랫폼에서 항상 공백입니다.

이 필드의 길이는 MQ\_TP\_NAME\_LENGTH에서 제공합니다.

## *TransportType (MQLONG)*

이 필드는 사용할 전송 프로토콜을 지정합니다.

채널이 다른 끝에서 초기화된 경우 값을 검사하지 않습니다.

다음 값 중 하나입니다.

## **MQXPT\_LU62**

LU 6.2 전송 프로토콜.

## **MQXPT\_TCP**

TCP/IP 전송 프로토콜.

## **MQXPT\_NETBIOS**

NetBIOS 전송 프로토콜.

이 값은 Windows 환경에서 지원됩니다.

## **MQXPT\_SPX**

SPX 전송 프로토콜.

이 값은 Windows 및 이 시스템에 연결된 IBM MQ 클라이언트에서 지원됩니다.

## *UseDLQ (MQLONG)*

이 필드는 채널이 메시지를 전달할 수 없는 경우 데드-레터 큐(또는 미배달 메시지 큐)가 사용되는지 여부를 지정합니다.

값은 다음 중 하나입니다.

## **MQUSEDLQ\_NO**

채널이 전달할 수 없는 메시지는 실패로 처리됩니다. NPMSPEED 설정에 따라 채널에서 메시지가 제거되거나 채널이 종료됩니다.

## **MQUSEDLQ\_YES**

DEADQ 큐 관리자 속성이 데드-레터 큐의 이름을 제공하면 해당 큐가 사용되고, 그 밖의 경우에는 NO와 같이 작동합니다. 기본값은 YES입니다.

## *UserIdentifier (MQCHAR12)*

이 필드는 보안 SNA 세션을 리모트 메시지 채널 에이전트로 시작할 때 메시지 채널 에이전트에 사용되는 사용자 ID를 지정합니다.

이 필드는 AIX, Linux, and Windows에서만 공백이 아닐 수 있으며 *ChannelType*이 MQCMT\_SENDER, MQCMT\_SERVER, MQCMT\_REQUESTER 또는 MQCMT\_CLNTCONN인 채널에만 관련됩니다. z/OS에서는 이 필드가 관련이 없습니다.

이 필드의 길이는 MQ\_USER\_ID\_LENGTH에 지정되어 있습니다. 그러나 처음 10자만 사용합니다.

*Version*이 MQCD\_VERSION\_2 미만이면 이 필드가 표시되지 않습니다.

*Version (MQLONG)*

*Version* 필드는 구조에 설정할 수 있는 가장 높은 버전 번호를 지정합니다.

값은 환경에 따라 달라집니다.

#### **MQCD\_VERSION\_1**

버전 1 채널 정의 구조.

#### **MQCD\_VERSION\_2**

버전 2 채널 정의 구조.

#### **MQCD\_VERSION\_3**

버전 3 채널 정의 구조.

#### **MQCD\_VERSION\_4**

버전 4 채널 정의 구조.

#### **MQCD\_VERSION\_5**

버전 5 채널 정의 구조.

#### **MQCD\_VERSION\_6**

버전 6 채널 정의 구조.

#### **MQCD\_VERSION\_7**

버전 7 채널 정의 구조.

#### **MQCD\_VERSION\_8**

버전 8 채널 정의 구조.

#### **MQCD\_VERSION\_9**

버전 9 채널 정의 구조.

#### **MQCD\_VERSION\_10**

버전 10 채널 정의 구조.

#### **MQCD\_VERSION\_11**

버전 11 채널 정의 구조.

버전 11은 모든 플랫폼의 IBM MQ 8.0에서 필드에 설정할 수 있는 가장 높은 값입니다.

#### **z/OS MQCD\_VERSION\_12**

버전 12 채널 정의 구조.

버전 12는 IBM MQ 9.1.3에서 필드에 설정할 수 있는 가장 높은 값입니다.

구조의 최신 버전에만 있는 필드는 필드 설명에 그렇게 식별되어 있습니다. 다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### **MQCD\_CURRENT\_VERSION**

MQCD\_CURRENT\_VERSION에 설정된 값은 사용하는 채널 정의 구조의 현재 버전입니다.

MQCD\_CURRENT\_VERSION의 값은 환경에 따라 다릅니다. 여기에는 플랫폼에서 지원되는 가장 높은 값이 포함됩니다.

MQCD\_CURRENT\_VERSION은 다양한 프로그래밍 언어에 제공되는 헤더, 복사 및 포함 파일에서 기본 구조를 초기화하는 데 사용되지 않습니다. *Version*의 기본 초기화는 플랫폼과 릴리스에 따라 다릅니다.

헤더, 복사 및 포함 파일의 MQCD 선언은 MQCD\_VERSION\_6로 초기화됩니다. 추가 MQCD 필드를 사용하려면 애플리케이션이 버전 번호를 MQCD\_CURRENT\_VERSION으로 설정해야 합니다. 여러 환경 간에 이동 가능한 애플리케이션을 작성하는 경우, 모든 환경에서 지원되는 버전을 선택해야 합니다.

**팁:** MQCD 구조의 새 버전을 도입해도 기존 부분의 레이아웃은 변경되지 않습니다. 엑시트에서 버전 번호를 검사해야 합니다. 엑시트가 사용해야 하는 필드를 포함한 가장 낮은 버전보다 크거나 같아야 합니다.

### XmitQName (MQCHAR48)

이 필드는 메시지가 검색되는 전송 큐의 이름을 지정합니다.

이 필드는 *ChannelType*이 MQCHT\_SENDER 또는 MQCHT\_SERVER인 채널만 관련이 있습니다.

이 필드의 길이는 MQ\_Q\_NAME\_LENGTH로 제공됩니다.

## C 선언

이 선언은 MQCD 구조에 대한 C 선언입니다.

```
typedef struct tagMQCD MQCD;
typedef MQCD MQPOINTER PMQCD;
typedef PMQCD MQPOINTER PPMQCD;

struct tagMQCD {
    MQCHAR    ChannelName[20];           /* Channel definition name */
    MQLONG    Version;                  /* Structure version number */
    MQLONG    ChannelType;              /* Channel type */
    MQLONG    TransportType;            /* Transport type */
    MQCHAR    Desc[64];                 /* Channel description */
    MQCHAR    QMgrName[48];             /* Queue manager name */
    MQCHAR    XmitQName[48];           /* Transmission queue name */
    MQCHAR    ShortConnectionName[20]; /* First 20 bytes of */
                                           /* connection name */
    MQCHAR    MCAName[20];              /* Reserved */
    MQCHAR    ModeName[8];              /* LU 6.2 Mode name */
    MQCHAR    TpName[64];               /* LU 6.2 transaction program */
                                           /* name */
    MQLONG    BatchSize;                /* Batch size */
    MQLONG    DiscInterval;             /* Disconnect interval */
    MQLONG    ShortRetryCount;          /* Short retry count */
    MQLONG    ShortRetryInterval;       /* Short retry wait interval */
    MQLONG    LongRetryCount;           /* Long retry count */
    MQLONG    LongRetryInterval;        /* Long retry wait interval */
    MQCHAR    SecurityExit[128];        /* Channel security exit name */
    MQCHAR    MsgExit[128];            /* Channel message exit name */
    MQCHAR    SendExit[128];           /* Channel send exit name */
    MQCHAR    ReceiveExit[128];        /* Channel receive exit name */
    MQLONG    SeqNumberWrap;           /* Highest allowable message */
                                           /* sequence number */
    MQLONG    MaxMsgLength;             /* Maximum message length */
    MQLONG    PutAuthority;             /* Put authority */
    MQLONG    DataConversion;          /* Data conversion */
    MQCHAR    SecurityUserData[32];     /* Channel security exit user */
                                           /* data */
    MQCHAR    MsgUserData[32];          /* Channel message exit user */
                                           /* data */
    MQCHAR    SendUserData[32];        /* Channel send exit user */
                                           /* data */
    MQCHAR    ReceiveUserData[32];     /* Channel receive exit user */
                                           /* data */
    /* Ver:1 */
    MQCHAR    UserIdentifier[12];       /* User identifier */
    MQCHAR    Password[12];            /* Password */
    MQCHAR    MCAUserIdentifier[12];   /* First 12 bytes of MCA user */
                                           /* identifier */
    MQLONG    MCAType;                  /* Message channel agent type */
    MQCHAR    ConnectionName[264];     /* Connection name */
    MQCHAR    RemoteUserIdentifier[12]; /* First 12 bytes of user */
                                           /* identifier from partner */
    MQCHAR    RemotePassword[12];      /* Password from partner */
    /* Ver:2 */
    MQCHAR    MsgRetryExit[128];       /* Channel message retry exit */
                                           /* name */
    MQCHAR    MsgRetryUserData[32];    /* Channel message retry exit */
                                           /* user data */
    MQLONG    MsgRetryCount;           /* Number of times MCA will */
                                           /* try to put the message, */
                                           /* after first attempt has */
                                           /* failed */
    MQLONG    MsgRetryInterval;        /* Minimum interval in */
                                           /* milliseconds after which */
                                           /* the open or put operation */
                                           /* will be retried */
}
```

```

/* Ver:3 */
MQLONG    HeartbeatInterval;    /* Time in seconds between */
/* heartbeat flows */
MQLONG    BatchInterval;        /* Batch duration */
MQLONG    NonPersistentMsgSpeed; /* Speed at which */
/* nonpersistent messages are */
/* sent */
MQLONG    StrucLength;           /* Length of MQCD structure */
MQLONG    ExitNameLength;        /* Length of exit name */
MQLONG    ExitDataLength;        /* Length of exit user data */
MQLONG    MsgExitsDefined;       /* Number of message exits */
/* defined */
MQLONG    SendExitsDefined;      /* Number of send exits */
/* defined */
MQLONG    ReceiveExitsDefined;   /* Number of receive exits */
/* defined */
MQPTR     MsgExitPtr;            /* Address of first MsgExit */
/* field */
MQPTR     MsgUserDataPtr;        /* Address of first */
/* MsgUserData field */
MQPTR     SendExitPtr;           /* Address of first SendExit */
/* field */
MQPTR     SendUserDataPtr;       /* Address of first */
/* SendUserData field */
MQPTR     ReceiveExitPtr;        /* Address of first */
/* ReceiveExit field */
MQPTR     ReceiveUserDataPtr;    /* Address of first */
/* ReceiveUserData field */

/* Ver:4 */
MQPTR     ClusterPtr;            /* Address of a list of */
/* cluster names */
MQLONG    ClustersDefined;       /* Number of clusters to */
/* which the channel belongs */
/* Network priority */
MQLONG    NetworkPriority;
/* Ver:5 */
MQLONG    LongMCAUserIdLength;   /* Length of long MCA user */
/* identifier */
MQLONG    LongRemoteUserIdLength; /* Length of long remote user */
/* identifier */
MQPTR     LongMCAUserIdPtr;      /* Address of long MCA user */
/* identifier */
MQPTR     LongRemoteUserIdPtr;   /* Address of long remote */
/* user identifier */
MQBYTE40  MCASecurityId;         /* MCA security identifier */
MQBYTE40  RemoteSecurityId;      /* Remote security identifier */

/* Ver:6 */
MQCHAR    SSLCipherSpec[32];     /* TLS CipherSpec */
MQPTR     SSLPeerNamePtr;        /* Address of TLS peer name */
MQLONG    SSLPeerNameLength;     /* Length of TLS peer name */
MQLONG    SSLClientAuth;         /* Whether TLS client */
/* authentication is required */
MQLONG    KeepAliveInterval;     /* Keepalive interval */
MQCHAR    LocalAddress[48];      /* Local communications */
/* address */
MQLONG    BatchHeartbeat;        /* Batch heartbeat interval */

/* Ver:7 */
MQLONG    HdrCompList[2];        /* Header data compression */
/* list */
MQLONG    MsgCompList[16];       /* Message data compression */
/* list */
MQLONG    CLWLChannelRank;       /* Channel rank */
MQLONG    CLWLChannelPriority;    /* Channel priority */
MQLONG    CLWLChannelWeight;     /* Channel weight */
MQLONG    ChannelMonitoring;     /* Channel monitoring */
MQLONG    ChannelStatistics;     /* Channel statistics */

/* Ver:8 */
MQLONG    SharingConversations;  /* Limit on sharing */
/* conversations */
MQLONG    PropertyControl;       /* Message property control */
MQLONG    MaxInstances;          /* Limit on SVRCONN channel */
/* instances */
MQLONG    MaxInstancesPerClient; /* Limit on SVRCONN channel */
/* instances per client */
MQLONG    ClientChannelWeight;   /* Client channel weight */
MQLONG    ConnectionAffinity;    /* Connection affinity */

/* Ver:9 */
MQLONG    BatchDataLimit;        /* Batch data limit */
MQLONG    UseDLQ;                /* Use Dead Letter Queue */
MQLONG    DefReconnect;          /* Default client reconnect */
/* option */

/* Ver:10 */
MQCHAR64  CertificateLabel;      /* Certificate label */

```



```

/* Ver:11 */
MQLONG      SPLProtection      /* AMS Security policy protection */
/* Ver:12 */
};

```

## COBOL 선언

이 선언은 MQCD 구조에 대한 COBOL 선언입니다.

```

** MQCD structure
  10 MQCD.
    ** Channel definition name
    15 MQCD-CHANNELNAME PIC X(20).
    ** Structure version number
    15 MQCD-VERSION PIC S9(9) BINARY.
    ** Channel type
    15 MQCD-CHANNELTYPE PIC S9(9) BINARY.
    ** Transport type
    15 MQCD-TRANSPORTTYPE PIC S9(9) BINARY.
    ** Channel description
    15 MQCD-DESC PIC X(64).
    ** Queue manager name
    15 MQCD-QMGRNAME PIC X(48).
    ** Transmission queue name
    15 MQCD-XMITQNAME PIC X(48).
    ** First 20 bytes of connection name
    15 MQCD-SHORTCONNECTIONNAME PIC X(20).
    ** Reserved
    15 MQCD-MCANAME PIC X(20).
    ** LU 6.2 Mode name
    15 MQCD-MODENAME PIC X(8).
    ** LU 6.2 transaction program name
    15 MQCD-TPNAME PIC X(64).
    ** Batch size
    15 MQCD-BATCHSIZE PIC S9(9) BINARY.
    ** Disconnect interval
    15 MQCD-DISCINTERVAL PIC S9(9) BINARY.
    ** Short retry count
    15 MQCD-SHORTRETRYCOUNT PIC S9(9) BINARY.
    ** Short retry wait interval
    15 MQCD-SHORTRETRYINTERVAL PIC S9(9) BINARY.
    ** Long retry count
    15 MQCD-LONGRETRYCOUNT PIC S9(9) BINARY.
    ** Long retry wait interval
    15 MQCD-LONGRETRYINTERVAL PIC S9(9) BINARY.
    ** Channel security exit name
    15 MQCD-SECURITYEXIT PIC X(20).
    ** Channel message exit name
    15 MQCD-MSGEXIT PIC X(20).
    ** Channel send exit name
    15 MQCD-SENDEXIT PIC X(20).
    ** Channel receive exit name
    15 MQCD-RECEIVEEXIT PIC X(20).
    ** Highest allowable message sequence number
    15 MQCD-SEQNUMBERWRAP PIC S9(9) BINARY.
    ** Maximum message length
    15 MQCD-MAXMSGLLENGTH PIC S9(9) BINARY.
    ** Put authority
    15 MQCD-PUTAUTHORITY PIC S9(9) BINARY.
    ** Data conversion
    15 MQCD-DATACONVERSION PIC S9(9) BINARY.
    ** Channel security exit user data
    15 MQCD-SECURITYUSERDATA PIC X(32).
    ** Channel message exit user data
    15 MQCD-MSGUSERDATA PIC X(32).
    ** Channel send exit user data
    15 MQCD-SENDUSERDATA PIC X(32).
    ** Channel receive exit user data
    15 MQCD-RECEIVEUSERDATA PIC X(32).
    ** Ver:1 **
    ** User identifier
    15 MQCD-USERIDENTIFIER PIC X(12).
    ** Password
    15 MQCD-PASSWORD PIC X(12).
    ** First 12 bytes of MCA user identifier
    15 MQCD-MCAUSERIDENTIFIER PIC X(12).
    ** Message channel agent type
    15 MQCD-MCATYPE PIC S9(9) BINARY.
    ** Connection name

```

```

15 MQCD-CONNECTIONNAME PIC X(264).
** First 12 bytes of user identifier from partner
15 MQCD-REMOTEUSERIDENTIFIER PIC X(12).
** Password from partner
15 MQCD-REMOtepASSWORD PIC X(12).
** Ver:2 **
** Channel message retry exit name
15 MQCD-MSGRETRYEXIT PIC X(20).
** Channel message retry exit user data
15 MQCD-MSGRETRYUSERDATA PIC X(32).
** Number of times MCA will try to put the message, after first
** attempt has failed
15 MQCD-MSGRETRYCOUNT PIC S9(9) BINARY.
** Minimum interval in milliseconds after which the open or put
** operation will be retried
15 MQCD-MSGRETRYINTERVAL PIC S9(9) BINARY.
** Ver:3 **
** Time in seconds between heartbeat flows
15 MQCD-HEARTBEATINTERVAL PIC S9(9) BINARY.
** Batch duration
15 MQCD-BATCHINTERVAL PIC S9(9) BINARY.
** Speed at which nonpersistent messages are sent
15 MQCD-NONPERSISTENTMSGSPPEED PIC S9(9) BINARY.
** Length of MQCD structure
15 MQCD-STRUCLLENGTH PIC S9(9) BINARY.
** Length of exit name
15 MQCD-EXITNAMELENGTH PIC S9(9) BINARY.
** Length of exit user data
15 MQCD-EXITDATALENGTH PIC S9(9) BINARY.
** Number of message exits defined
15 MQCD-MSGEXITSDEFINED PIC S9(9) BINARY.
** Number of send exits defined
15 MQCD-SENDEXITSDEFINED PIC S9(9) BINARY.
** Number of receive exits defined
15 MQCD-RECEIVEEXITSDEFINED PIC S9(9) BINARY.
** Address of first MsgExit field
15 MQCD-MSGEXITPTR POINTER.
** Address of first MsgUserData field
15 MQCD-MSGUSERDATAPTR POINTER.
** Address of first SendExit field
15 MQCD-SENDEXITPTR POINTER.
** Address of first SendUserData field
15 MQCD-SENDUSERDATAPTR POINTER.
** Address of first ReceiveExit field
15 MQCD-RECEIVEEXITPTR POINTER.
** Address of first ReceiveUserData field
15 MQCD-RECEIVEUSERDATAPTR POINTER.
** Ver:4 **
** Address of a list of cluster names
15 MQCD-CLUSTERPTR POINTER.
** Number of clusters to which the channel belongs
15 MQCD-CLUSTERSDEFINED PIC S9(9) BINARY.
** Network priority
15 MQCD-NETWORKPRIORITY PIC S9(9) BINARY.
** Ver:5 **
** Length of long MCA user identifier
15 MQCD-LONGMCAUSERIDLENGTH PIC S9(9) BINARY.
** Length of long remote user identifier
15 MQCD-LONGREMOTEUSERIDLENGTH PIC S9(9) BINARY.
** Address of long MCA user identifier
15 MQCD-LONGMCAUSERIDPTR POINTER.
** Address of long remote user identifier
15 MQCD-LONGREMOTEUSERIDPTR POINTER.
** MCA security identifier
15 MQCD-MCASECURITYID PIC X(40).
** Remote security identifier
15 MQCD-REMOTESECURITYID PIC X(40).
** Ver:6 **
** TLS CipherSpec
15 MQCD-SSLCIPHERSPEC PIC X(32).
** Address of TLS peer name
15 MQCD-SSLPEERNAMEPTR POINTER.
** Length of TLS peer name
15 MQCD-SSLPEERNAMELENGTH PIC S9(9) BINARY.
** Whether TLS client authentication is required
15 MQCD-SSLCLIENTAUTH PIC S9(9) BINARY.
** Keepalive interval
15 MQCD-KEEPALIVEINTERVAL PIC S9(9) BINARY.
** Local communications address
15 MQCD-LOCALADDRESS PIC X(48).
** Batch heartbeat interval
15 MQCD-BATCHHEARTBEAT PIC S9(9) BINARY.

```

```

** Ver:7 **
** Header data compression list
  15 MQCD-HDRCOMPLIST PIC S9(9) BINARY.
** Message data compression list
  15 MQCD-MSGCOMPLIST PIC S9(9) BINARY.
** Channel rank
  15 MQCD-CLWLCHANNELRANK PIC S9(9) BINARY.
** Channel priority
  15 MQCD-CLWLCHANNELPRIORITY PIC S9(9) BINARY.
** Channel weight
  15 MQCD-CLWLCHANNELWEIGHT PIC S9(9) BINARY.
** Channel monitoring
  15 MQCD-CHANNELMONITORING PIC S9(9) BINARY.
** Channel statistics
  15 MQCD-CHANNELSTATISTICS PIC S9(9) BINARY.
** Ver:8 **
** Limit on sharing conversations
  15 MQCD-SHARINGCONVERSATIONS PIC S9(9) BINARY.
** Message property control
  15 MQCD-PROPERTYCONTROL PIC S9(9) BINARY.
** Limit on SVRCONN channel instances
  15 MQCD-MAXINSTANCES PIC S9(9) BINARY.
** Limit on SVRCONN channel instances per client
  15 MQCD-MAXINSTANCESPERCLIENT PIC S9(9) BINARY.
** Client channel weight
  15 MQCD-CLIENTCHANNELWEIGHT PIC S9(9) BINARY.
** Connection affinity
  15 MQCD-CONNECTIONAFFINITY PIC S9(9) BINARY.
** Ver:9 **
** Batch data limit
  15 MQCD-BATCHDATALIMIT PIC S9(9) BINARY.
** Use Dead Letter Queue
  15 MQCD-USEDLQ PIC S9(9) BINARY.
** Default client reconnect option
  15 MQCD-DEFRECONNECT PIC S9(9) BINARY.
** Ver:10 **
** Certificate Label
  15 MQCD-CERTLABL PIC X (64)
** Ver:11 **
** AMS Security policy protection
  15 MQCD-SPLPROTECTION PIC S9(9) BINARY
** Ver:12 **

```

## RPG 선언(ILE)

이 선언은 MQCD 구조에 대한 RPG 선언입니다.

```

D* MQCD Structure
D*
D* Channel definition name
D CDCHN          1      20
D* Structure version number
D CDVER          21     24I 0
D* Channel type
D CDCHT          25     28I 0
D* Transport type
D CDTRT          29     32I 0
D* Channel description
D CDDDES         33     96
D* Queue manager name
D CDQM           97    144
D* Transmission queue name
D CDXQ          145    192
D* First 20 bytes of connection name
D CDSCN         193    212
D* Reserved
D CDMCA         213    232
D* LU 6.2 Mode name
D CDMOD         233    240
D* LU 6.2 transaction program name
D CDTP          241    304
D* Batch size
D CDBS          305    308I 0
D* Disconnect interval
D CDDI          309    312I 0
D* Short retry count
D CDSRC         313    316I 0
D* Short retry wait interval
D CDSRI         317    320I 0

```

```

D* Long retry count
D CDLRC          321    324I 0
D* Long retry wait interval
D CDLRI          325    328I 0
D* Channel security exit name
D CDSCX          329    348
D* Channel message exit name
D CDMSX          349    368
D* Channel send exit name
D CDSNX          369    388
D* Channel receive exit name
D CDRCX          389    408
D* Highest allowable message sequence number
D CDSNW          409    412I 0
D* Maximum message length
D CDMML          413    416I 0
D* Put authority
D CDPA           417    420I 0
D* Data conversion
D CDDC           421    424I 0
D* Channel security exit user data
D CDSCD          425    456
D* Channel message exit user data
D CDMSD          457    488
D* Channel send exit user data
D CDSND          489    520
D* Channel receive exit user data
D CDRCU          521    552
D* Ver:1 **
D* User identifier
D CDUID          553    564
D* Password
D CDPW           565    576
D* First 12 bytes of MCA user identifier
D CDAUI          577    588
D* Message channel agent type
D CDCAT          589    592I 0
D* Connection name
D CDCON          593    848
D CDCN2          849    856
D* First 12 bytes of user identifier from partner
D CDRUI          857    868
D* Password from partner
D CDRPW          869    880
D* Ver:2 **
D* Channel message retry exit name
D CDMRX          881    900
D* Channel message retry exit user data
D CDMRD          901    932
D* Number of times MCA will try to put the message, after first
D* attempt has failed
D CDMRC          933    936I 0
D* Minimum interval in milliseconds after which the open or put
D* operation will be retried
D CDMRI          937    940I 0
D* Ver:3 **
D* Time in seconds between heartbeat flows
D CDHBI          941    944I 0
D* Batch duration
D CDBI           945    948I 0
D* Speed at which nonpersistent messages are sent
D CDNPM          949    952I 0
D* Length of MQCD structure
D CDLEN          953    956I 0
D* Length of exit name
D CDXNL          957    960I 0
D* Length of exit user data
D CDXDL          961    964I 0
D* Number of message exits defined
D CDMXD          965    968I 0
D* Number of send exits defined
D CDSXD          969    972I 0
D* Number of receive exits defined
D CDRXD          973    976I 0
D* Address of first MsgExit field
D CDMXP          977    992*
D* Address of first MsgUserData field
D CDMUP          993   1008*
D* Address of first SendExit field
D CDSXP         1009   1024*
D* Address of first SendUserData field
D CDSUP         1025   1040*

```

```

D* Address of first ReceiveExit field
D CDRXP          1041  1056*
D* Address of first ReceiveUserData field
D CDRUP          1057  1072*
D* Ver:4 **
D* Address of a list of cluster names
D CDCLP          1073  1088*
D* Number of clusters to which the channel belongs
D CDCLD          1089  1092I 0
D* Network priority
D CDNP           1093  1096I 0
D* Ver:5 **
D* Length of long MCA user identifier
D CDLML          1097  1100I 0
D* Length of long remote user identifier
D CDRL           1101  1104I 0
D* Address of long MCA user identifier
D CDLMP          1105  1120*
D* Address of long remote user identifier
D CDLRP          1121  1136*
D* MCA security identifier
D CDMSI          1137  1176
D* Remote security identifier
D CDRSI          1177  1216
D* Ver:6 **
D* TLS CipherSpec
D CDSCS          1217  1248
D* Address of TLS peer name
D CDSPN          1249  1264*
D* Length of TLS peer name
D CDSPL          1265  1268I 0
D* Whether TLS client authentication is required
D CDSCA          1269  1272I 0
D* Keepalive interval
D CDKAI          1273  1276I 0
D* Local communications address
D CDLOA          1277  1324
D* Batch heartbeat interval
D CDBHB          1325  1328I 0
D* Ver:7 **
D* Header data compression list
D CDHCL0
D CDHCL1          1329  1332I 0
D CDHCL2          1333  1336I 0
D CDHCL          10I 0 DIM(2) OVERLAY(CDHCL0)
D* Message data compression list
D CDMCL0
D CDMCL1          1337  1340I 0
D CDMCL2          1341  1344I 0
D CDMCL3          1345  1348I 0
D CDMCL4          1349  1352I 0
D CDMCL5          1353  1356I 0
D CDMCL6          1357  1360I 0
D CDMCL7          1361  1364I 0
D CDMCL8          1365  1368I 0
D CDMCL9          1369  1372I 0
D CDMCL10         1373  1376I 0
D CDMCL11         1377  1380I 0
D CDMCL12         1381  1384I 0
D CDMCL13         1385  1388I 0
D CDMCL14         1389  1392I 0
D CDMCL15         1393  1396I 0
D CDMCL16         1397  1400I 0
D CDMCL          10I 0 DIM(16) OVERLAY(CDMCL0)
D* Channel rank
D CDCWCR         1401  1404I 0
D* Channel priority
D CDCWCP         1405  1408I 0
D* Channel weight
D CDCWCW         1409  1412I 0
D* Channel monitoring
D CDCHLMON       1413  1416I 0
D* Channel statistics
D CDCHLST        1417  1420I 0
D* Ver:8 **
D* Limit on sharing conversations
D CDSHC          1421  1424I 0
D* Message property control
D CDPRC          1425  1428I 0
D* Limit on SVRCONN channel instances
D CDMXIN         1429  1432I 0
D* Limit on SVRCONN channel instances per client

```

```

D CDMXIC 1433 1436I 0
D* Client channel weight
D CDCLNCHLW 1437 1440I 0
D* Connection affinity
D CDCONNAFF 1441 1444I 0
D* Ver:9 **
D* Batch data limit
D CDBDL 1445 1448I 0
D* Use Dead Letter Queue
D CDUDLQ 1449 1452I 0
D* Default client reconnect option
D CDDRCN 1453 1456I 0
D* Ver:10 **

```

## System/390 어셈블러 선언

이 선언은 MQCD 구조에 대한 System/390 어셈블러 선언입니다.

```

MQCD DSECT
MQCD_CHANNELNAME DS CL20 Channel definition name
MQCD_VERSION DS F Structure version number
MQCD_CHANNELTYPE DS F Channel type
MQCD_TRANSPORTTYPE DS F Transport type
MQCD_DESC DS CL64 Channel description
MQCD_QMGRNAME DS CL48 Queue manager name
MQCD_XMITQNAME DS CL48 Transmission queue name
MQCD_SHORTCONNECTIONNAME DS CL20 First 20 bytes of connection
* name
MQCD_MCANAME DS CL20 Reserved
MQCD_MODENAME DS CL8 LU 6.2 Mode name
MQCD_TPNAME DS CL64 LU 6.2 transaction program name
MQCD_BATCHSIZE DS F Batch size
MQCD_DISCINTERVAL DS F Disconnect interval
MQCD_SHORTRETRYCOUNT DS F Short retry count
MQCD_SHORTRETRYINTERVAL DS F Short retry wait interval
MQCD_LONGRETRYCOUNT DS F Long retry count
MQCD_LONGRETRYINTERVAL DS F Long retry wait interval
MQCD_SECURITYEXIT DS CLn Channel security exit name
MQCD_MSGEXIT DS CLn Channel message exit name
MQCD_SENDEXIT DS CLn Channel send exit name
MQCD_RECEIVEEXIT DS CLn Channel receive exit name
MQCD_SEQNUMBERWRAP DS F Highest allowable message
* sequence number
MQCD_MAXMSGLLENGTH DS F Maximum message length
MQCD_PUTAUTHORITY DS F Put authority
MQCD_DATACONVERSION DS F Data conversion
MQCD_SECURITYUSERDATA DS CL32 Channel security exit user data
MQCD_MSGUSERDATA DS CL32 Channel message exit user data
MQCD_SENDUSERDATA DS CL32 Channel send exit user data
MQCD_RECEIVEUSERDATA DS CL32 Channel receive exit user data
MQCD_USERIDENTIFIER DS CL12 User identifier
MQCD_PASSWORD DS CL12 Password
MQCD_MCAUSERIDENTIFIER DS CL12 First 12 bytes of MCA user
* identifier
MQCD_MCATYPE DS F Message channel agent type
MQCD_CONNECTIONNAME DS CL264 Connection name
MQCD_REMOTEUSERIDENTIFIER DS CL12 First 12 bytes of user
* identifier from partner
MQCD_REMOTEPASSWORD DS CL12 Password from partner
MQCD_MSGRETRYEXIT DS CLn Channel message retry exit name
MQCD_MSGRETRYUSERDATA DS CL32 Channel message retry exit user
* data
MQCD_MSGRETRYCOUNT DS F Number of times MCA will try to
* put the message, after the
* first attempt has failed
MQCD_MSGRETRYINTERVAL DS F Minimum interval in
* milliseconds after which the
* open or put operation will be
* retried
MQCD_HEARTBEATINTERVAL DS F Time in seconds between
* heartbeat flows
MQCD_BATCHINTERVAL DS F Batch duration
MQCD_NONPERSISTENTMSGSPPEED DS F Speed at which nonpersistent
* messages are sent
MQCD_STRUCLLENGTH DS F Length of MQCD structure
MQCD_EXITNAMELENGTH DS F Length of exit name
MQCD_EXITDATALENGTH DS F Length of exit user data
MQCD_MSGEXITDEFINED DS F Number of message exits defined
MQCD_SENDEXITDEFINED DS F Number of send exits defined

```

MQCD_RECEIVEEXITSDEFINED	DS	F	Number of receive exits defined
MQCD_MSGEXITPTR	DS	F	Address of first MSGEXIT field
MQCD_MSGUSERDATAPTR	DS	F	Address of first MSGUSERDATA field
*			
MQCD_SENDEXITPTR	DS	F	Address of first SENDEXIT field
MQCD_SENDUSERDATAPTR	DS	F	Address of first SENDUSERDATA field
*			
MQCD_RECEIVEEXITPTR	DS	F	Address of first RECEIVEEXIT field
*			
MQCD_RECEIVEUSERDATAPTR	DS	F	Address of first RECEIVEUSERDATA field
*			
MQCD_CLUSTERPTR	DS	F	Address of a list of cluster names
*			
MQCD_CLUSTERSDEFINED	DS	F	Number of clusters to which the channel belongs
*			
MQCD_NETWORKPRIORITY	DS	F	Network priority
MQCD_LONGMCAUSERIDLENGTH	DS	F	Length of long MCA user identifier
*			
MQCD_LONGREMOTEUSERIDLENGTH	DS	F	Length of long remote user identifier
*			
MQCD_LONGMCAUSERIDPTR	DS	F	Address of long MCA user identifier
*			
MQCD_LONGREMOTEUSERIDPTR	DS	F	Address of long remote user identifier
*			
MQCD_MCASECURITYID	DS	XL40	MCA security identifier
MQCD_REMOTESECURITYID	DS	XL40	Remote security identifier
MQCD_SSLCIPHERSPEC	DS	CL32	TLS CipherSpec
MQCD_SSLPEERNAMEPTR	DS	F	Address of TLS peer name
MQCD_SSLPEERNAMELENGTH	DS	F	Length of TLS peer name
MQCD_SSLCLIENTAUTH	DS	F	Whether TLS client authentication is required
*			
MQCD_KEEPLIVEINTERVAL	DS	F	Keepalive interval
MQCD_LOCALADDRESS	DS	CL48	Local communications address
MQCD_BATCHHEARTBEAT	DS	F	Batch heartbeat interval
MQCD_HDRCOMPLIST	DS	CL2	Header data compression list
MQCD_MSGCOMPLIST	DS	CL16	Message data compression list
MQCD_CLWLCHANNELRANK	DS	F	Channel rank
MQCD_CLWLCHANNELPRIORITY	DS	F	Channel priority
MQCD_CLWLCHANNELWEIGHT	DS	F	Channel weight
MQCD_CHANNELMONITORING	DS	F	Channel monitoring
MQCD_CHANNELSTATISTICS	DS	F	Channel statistics
MQCD_SHARINGCONVERSATIONS	DS	F	Limit on sharing conversations
*			
MQCD_PROPERTYCONTROL	DS	F	Message property control
*			
MQCD_SHARINGCONVERSATIONS	DS	F	Limit on sharing conversations
MQCD_PROPERTYCONTROL	DS	F	Message property control
MQCD_MAXINSTANCES	DS	F	Limit on SVRCONN chl instances
MQCD_MAXINSTANCESPERCLIENT	DS	F	Limit on SVRCONN chl instances per client
MQCD_CLIENTCHANNELWEIGHT	DS	F	Channel weight
MQCD_CONNECTIONAFFINITY	DS	F	Connection Affinity
MQCD_BATCHDATALIMIT	DS	F	Batch data limit
MQCD_USEDLO	DS	F	Use dead-letter queue
MQCD_DEFRECONNECT	DS	F	Default client reconnect option
MQCD_CERTLABL	DS	F	Certificate label
MQCD_SPLPROTECTION	DS	F	AMS Security policy protection
MQCD_LENGTH	EQU	*-MQCD	
	ORG	MQCD	
MQCD_AREA	DS	CL(MQCD_LENGTH)	

## Visual Basic 선언

이 선언은 MQCD 구조의 Visual Basic 선언입니다.

Visual Basic에서는 MQCONNX 호출에서 MQCD 구조와 MQCNO 구조를 함께 사용할 수 있습니다.

Type MQCD		
ChannelName	As String*20	'Channel definition name'
Version	As Long	'Structure version number'
ChannelType	As Long	'Channel type'
TransportType	As Long	'Transport type'
Desc	As String*64	'Channel description'
QMgrName	As String*48	'Queue manager name'
XmitQName	As String*48	'Transmission queue name'
ShortConnectionName	As String*20	'First 20 bytes of connection name'
MCAName	As String*20	'Reserved'
ModeName	As String*8	'LU 6.2 Mode name'

TpName	As String*64	'LU 6.2 transaction program name'
BatchSize	As Long	'Batch size'
DiscInterval	As Long	'Disconnect interval'
ShortRetryCount	As Long	'Short retry count'
ShortRetryInterval	As Long	'Short retry wait interval'
LongRetryCount	As Long	'Long retry count'
LongRetryInterval	As Long	'Long retry wait interval'
SecurityExit	As String*128	'Channel security exit name'
MsgExit	As String*128	'Channel message exit name'
SendExit	As String*128	'Channel send exit name'
ReceiveExit	As String*128	'Channel receive exit name'
SeqNumberWrap	As Long	'Highest allowable message' 'sequence number'
MaxMsgLength	As Long	'Maximum message length'
PutAuthority	As Long	'Put authority'
DataConversion	As Long	'Data conversion'
SecurityUserData	As String*32	'Channel security exit user data'
MsgUserData	As String*32	'Channel message exit user data'
SendUserData	As String*32	'Channel send exit user data'
ReceiveUserData	As String*32	'Channel receive exit user data'
UserIdentifier	As String*12	'User identifier'
Password	As String*12	'Password'
MCAUserIdentifier	As String*12	'First 12 bytes of MCA user' 'identifier'
MCAType	As Long	'Message channel agent type'
ConnectionName	As String*264	'Connection name'
RemoteUserIdentifier	As String*12	'First 12 bytes of user' 'identifier from partner'
RemotePassword	As String*12	'Password from partner'
MsgRetryExit	As String*128	'Channel message retry exit name'
MsgRetryUserData	As String*32	'Channel message retry exit user' 'data'
MsgRetryCount	As Long	'Number of times MCA will try to' 'put the message, after the' 'first attempt has failed'
MsgRetryInterval	As Long	'Minimum interval in' 'milliseconds after which the' 'open or put operation will be' 'retried'
HeartbeatInterval	As Long	'Time in seconds between' 'heartbeat flows'
BatchInterval	As Long	'Batch duration'
NonPersistentMsgSpeed	As Long	'Speed at which nonpersistent' 'messages are sent'
StrucLength	As Long	'Length of MQCD structure'
ExitNameLength	As Long	'Length of exit name'
ExitDataLength	As Long	'Length of exit user data'
MsgExitsDefined	As Long	'Number of message exits defined'
SendExitsDefined	As Long	'Number of send exits defined'
ReceiveExitsDefined	As Long	'Number of receive exits defined'
MsgExitPtr	As MQPTR	'Address of first MsgExit field'
MsgUserDataPtr	As MQPTR	'Address of first MsgUserData' 'field'
SendExitPtr	As MQPTR	'Address of first SendExit field'
SendUserDataPtr	As MQPTR	'Address of first SendUserData' 'field'
ReceiveExitPtr	As MQPTR	'Address of first ReceiveExit' 'field'
ReceiveUserDataPtr	As MQPTR	'Address of first' 'ReceiveUserData field'
ClusterPtr	As MQPTR	'Address of a list of cluster' 'names'
ClustersDefined	As Long	'Number of clusters to which the' 'channel belongs'
NetworkPriority	As Long	'Network priority'
LongMCAUserIdLength	As Long	'Length of long MCA user' 'identifier'
LongRemoteUserIdLength	As Long	'Length of long remote user' 'identifier'
LongMCAUserIdPtr	As MQPTR	'Address of long MCA user' 'identifier'
LongRemoteUserIdPtr	As MQPTR	'Address of long remote user' 'identifier'
MCASecurityId	As MQBYTE40	'MCA security identifier'
RemoteSecurityId	As MQBYTE40	'Remote security identifier'
SSLCipherSpec	As String*32	'TLS CipherSpec'
SSLPeerNamePtr	As MQPTR	'Address of TLS peer name'
SSLPeerNameLength	As Long	'Length of TLS peer name'
SSLClientAuth	As Long	'Whether TLS client' 'authentication is required'
KeepAliveInterval	As Long	'Keepalive interval'
LocalAddress	As String*48	'Local communications address'



BatchHeartbeat	As Long	'Batch heartbeat interval'
HdrCompList(0 to 1)	As Long2	'Header data compression list'
MsgCompList(0 To 15)	As Long16	'Message data compression list'
CLWLChannelRank	As Long	'Channel Rank'
CLWLChannelPriority	As Long	'Channel priority'
CLWLChannelWeight	As Long	'Channel Weight'
ChannelMonitoring	As Long	'Channel Monitoring control'
ChannelStatistics	As Long	'Channel Statistics'
End Type		

### 채널 엑시트에서 MQCD 필드 변경

채널 엑시트는 MQCD의 필드를 변경할 수 있습니다. 그러나, 나열된 상황을 제외하고는 일반적으로 이러한 변경 사항이 적용되지 않습니다.

채널 엑시트 프로그램이 MQCD 데이터 구조에서 필드를 변경하면 새 값은 일반적으로 IBM MQ 채널 프로세스에서 무시됩니다. 그러나 새 값은 MQCD에 남아 있으며, 엑시트 체인에 남아 있는 모든 엑시트와 채널 인스턴스를 공유하는 모든 대화에 전달됩니다.

SharingConversations가 MQCXP 구조의 FALSE로 설정되면 특정 필드의 변경사항은 엑시트 프로그램의 유형, 채널의 유형 및 엑시트 이유 코드에 따라 적용될 수 있습니다. 다음 표는 변경할 수 있는 필드, 채널 동작에 영향을 미치는 필드 및 해당 상황을 보여줍니다. 엑시트 프로그램이 그 외 환경에서 이러한 필드 중 하나를 변경하거나 나열되지 않는 필드를 변경하는 경우, 새 값이 채널 프로세스에서 무시됩니다. 새 값이 MQCD에 남아 있으며 엑시트 체인의 나머지 엑시트 및 채널 인스턴스를 공유하는 대화로 전달됩니다.

MQCXP SharingConversations를 FALSE로 설정한 경우, 초기화를 위해 호출될 때(MQXR\_INIT) 어떤 유형의 엑시트 프로그램이든 모든 유형의 채널에서 ChannelName 필드가 변경될 수 있습니다. 보안 엑시트만 MQCXP SharingConversations 값에 상관없이 MCAUserIdentifier 필드를 변경할 수 있습니다.

표 823. 변경할 수 있는 필드 및 채널 동작에 대한 영향			
필드	엑시트 이유 코드	엑시트 유형	채널 유형
ChannelName	MQXR_INIT	모두	모두
TransportType	MQXR_INIT	모두	모두
XmitQName	MQXR_INIT	모두	SDR, RCVR
ModeName	MQXR_INIT	모두	모두
TpName	MQXR_INIT	모두	모두
BatchSize	MQXR_INIT	모두	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
DiscInterval	MQXR_INIT	모두	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
ShortRetryCount	MQXR_INIT	모두	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
ShortRetryInterval	MQXR_INIT	모두	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR

표 823. 변경할 수 있는 필드 및 채널 동작에 대한 영향 (계속)			
필드	엑시트 이유 코드	엑시트 유형	채널 유형
LongRetryCount	MQXR_INIT	모두	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
LongRetryInterval	MQXR_INIT	모두	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
SeqNumberWrap	MQXR_INIT	모두	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
MaxMsgLength	MQXR_INIT	모두	모두
PutAuthority	MQXR_INIT	모두	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
DataConversion	MQXR_INIT	모두	모두
MCAUserIdentifier	MQXR_INIT, MQXR_INIT_SEC, MQXR_SEC_MSG, MQXR_SEC_PARMS	보안	RCVR, RQSTR, SVRCONN, CLUSRCVR
ConnectionName	MQXR_INIT	모두	SDR, SVR, RQSTR, CLNTCONN, CLUSSDR, CLUSRCVR
MsgRetryUserData	MQXR_INIT	모두	RCVR, RQSTR, CLUSRCVR
MsgRetryCount	MQXR_INIT	모두	RCVR, RQSTR, CLUSRCVR
MsgRetryInterval	MQXR_INIT	모두	RCVR, RQSTR, CLUSRCVR
HeartbeatInterval	MQXR_INIT	모두	모두
BatchInterval	MQXR_INIT	모두	SDR, SVR, CLUSSDR, CLUSRCVR

표 823. 변경할 수 있는 필드 및 채널 동작에 대한 영향 (계속)			
필드	엑시트 이유 코드	엑시트 유형	채널 유형
NonPersistentMsgSpeed	MQXR_INIT	모두	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
MCASecurityId	MQXR_INIT, MQXR_INIT_SEC, MQXR_SEC_MSG, MQXR_SEC_PARMS	보안	SDR, SVR, RCVR, RQSTR, SVRCONN, CLUSSDR, CLUSRCVR
SSLCipherSpec	MQXR_INIT	모두	모두
SSLPeerNamePtr	MQXR_INIT	모두	모두
SSLPeerNameLength	MQXR_INIT	모두	모두
SSLClientAuth	MQXR_INIT	모두	SVR, RCVR, RQSTR, SVRCONN, CLUSRCVR
KeepAliveInterval	MQXR_INIT	모두	모두
LocalAddress	MQXR_INIT	모두	SDR, SVR, RQSTR, CLNTCONN, CLUSSDR, CLUSRCVR
BatchHeartbeat	MQXR_INIT	모두	SDR, SVR, CLUSSDR, CLUSRCVR
HdrCompList	MQXR_INIT	모두	모두
MsgCompList	MQXR_INIT	모두	모두
ChannelMonitoring	MQXR_INIT	모두	SDR, SVR, RCVR, RQSTR, SVRCONN, CLUSSDR, CLUSRCVR
ChannelStatistics	MQXR_INIT	모두	SDR, SVR, RCVR, RQSTR, CLUSSDR, CLUSRCVR
SharingConversations	MQXR_INIT	모두	SVRCONN, CLNTCONN
PropertyControl	MQXR_INIT	모두	SDR, SVR, CLUSSDR, CLUSRCVR

## MQCXP - 채널 엑시트 매개변수

MQCXP 구조는 메시지 채널 에이전트(MCA), 클라이언트 연결 채널 또는 서버 연결 채널이 호출하는 각 엑시트 유형으로 전달됩니다.

MQ\_CHANNEL\_EXIT를 참조하십시오.

뒤이어 나오는 설명에서 "엑시트에 대한 입력"으로 표시된 필드는 엑시트가 제어를 채널로 리턴하면 채널에서 무시됩니다. 채널 엑시트 매개변수 블록에서 엑시트가 변경하는 모든 입력 필드는 다음 호출을 위해 보존되지 않습니다. 입출력(I/O) 필드(예: *ExitUserArea* 필드)에 적용된 변경사항은 해당 엑시트 인스턴스의 호출에 대해서만 보존됩니다. 이러한 변경사항은 동일한 채널에 정의된 서로 다른 엑시트 사이 또는 서로 다른 채널에 정의된 동일한 엑시트 사이에 데이터를 전달하는 데 사용할 수 있습니다.

### 관련 참조

[1448 페이지의 『필드』](#)

이 주제는 MQCXP 구조의 모든 필드를 나열하고 각 필드를 설명합니다.

[1458 페이지의 『C 선언』](#)

이 선언은 MQCXP 구조에 대한 C 선언입니다.

[1459 페이지의 『COBOL 선언』](#)

이 선언은 MQCXP 구조에 대한 COBOL 선언입니다.

[1460 페이지의 『RPG 선언\(ILE\)』](#)

이 선언은 MQCXP 구조에 대한 RPG 선언입니다.

[1461 페이지의 『System/390 어셈블러 선언』](#)

이 선언은 MQCXP 구조에 대한 System/390 어셈블러 선언입니다.

### 필드

이 주제는 MQCXP 구조의 모든 필드를 나열하고 각 필드를 설명합니다.

*StrucId (MQCHAR4)*

이 필드는 구조 ID를 지정합니다.

값은 다음과 같아야 합니다.

#### MQCXP\_STRUC\_ID

채널 엑시트 매개변수 구조의 ID.

C 프로그래밍 언어의 경우, 상수 MQCXP\_STRUC\_ID\_ARRAY도 정의됩니다. 이 상수는 MQCXP\_STRUC\_ID와 동일한 값을 갖지만, 문자열이 아니라 문자 배열입니다.

엑시트의 입력 필드입니다.

*Version(MQLONG)*

이 필드는 구조 버전 번호를 지정합니다.

값은 환경에 따라 달라집니다.

#### MQCXP\_VERSION\_1


버전-1 채널 엑시트 매개변수 구조.

#### MQCXP\_VERSION\_2

버전-2 채널 엑시트 매개변수 구조.

#### MQCXP\_VERSION\_3

버전-3 채널 엑시트 매개변수 구조.

 AIX and Linux 시스템에서 필드는 이 값을 갖습니다(다른 곳에 나열되지 않음).

#### MQCXP\_VERSION\_4

버전-4 채널 엑시트 매개변수 구조.

#### MQCXP\_VERSION\_5


버전-5 채널 엑시트 매개변수 구조.

### **MQCXP\_VERSION\_6**

버전-6 채널 엑시트 매개변수 구조.

### **MQCXP\_VERSION\_8**

버전-8 채널 엑시트 매개변수 구조.

 z/OS에서 필드는 이 값을 갖습니다.

### **MQCXP\_VERSION\_9**

버전-9 채널 엑시트 매개변수 구조.

다음 환경에서 필드는 이 값을 갖습니다.

-  AIX
-  IBM i
-  Linux
-  Windows
-  z/OS

구조의 최신 버전에만 있는 필드는 필드 설명에 그렇게 식별되어 있습니다. 다음 상수는 현재 버전의 버전 번호를 지정합니다.

### **MQCXP\_CURRENT\_VERSION**

채널 엑시트 매개변수 구조의 현재 버전.

값은 환경에 따라 다릅니다.

**참고:** MQCXP 구조의 새 버전을 도입해도 기존 부분의 레이아웃은 변경되지 않습니다. 따라서 엑시트는 버전 번호가 해당 엑시트가 사용해야 하는 필드를 포함한 가장 낮은 버전보다 크거나 같은지 확인해야 합니다.

엑시트의 입력 필드입니다.

#### *ExitId (MQLONG)*

이 필드는 호출되는 엑시트의 유형을 지정하고 엑시트 루틴에 대한 입력에서 설정됩니다.

가능한 값은 다음과 같습니다.

#### **MQXT\_CHANNEL\_SEC\_EXIT**

채널 보안 엑시트.

#### **MQXT\_CHANNEL\_MSG\_EXIT**

채널 메시지 엑시트.

#### **MQXT\_CHANNEL\_SEND\_EXIT**

채널 송신 엑시트.

#### **MQXT\_CHANNEL\_RCV\_EXIT**

채널 수신 엑시트.

#### **MQXT\_CHANNEL\_MSG\_RETRY\_EXIT**

채널 메시지 재시도 엑시트.

#### **MQXT\_CHANNEL\_AUTO\_DEF\_EXIT**

채널 자동 정의 엑시트.

z/OS에서 이 유형의 엑시트는 MQCHT\_CLUSSDR 및 MQCHT\_CLUSRCVR 유형의 채널에만 지원됩니다.

엑시트의 입력 필드입니다.

#### *ExitReason (MQLONG)*

이 필드는 엑시트 호출 이유를 지정하고 엑시트 루틴에 대한 입력에서 설정됩니다.

자동 정의 엑시트에는 사용되지 않습니다. 가능한 값은 다음과 같습니다.

#### **MQXR\_INIT**

엑시트 초기화.

이 값은 엑시트를 처음 호출하고 있음을 표시합니다. 엑시트가 필요한 자원(예: 메모리)을 확보하고 초기화할 수 있습니다.

#### **MQXR\_TERM**

엑시트 종료.

이 값은 엑시트가 곧 종료됨을 표시합니다. 엑시트는 초기화된 이후 확보한 자원(예: 메모리)을 비워야 합니다.

#### **MQXR\_MSG**

메시지 처리.

이 값은 메시지 처리를 위해 엑시트를 호출하고 있음을 표시합니다. 이 값은 채널 메시지 엑시트에 대해서만 발생합니다.

#### **MQXR\_XMIT**

종료 처리.

이 값은 채널 송신 및 수신 엑시트에 대해서만 발생합니다.

#### **MQXR\_SEC\_MSG**

보안 메시지 수신.

이 값은 채널 보안 엑시트에 대해서만 발생합니다.

#### **MQXR\_INIT\_SEC**

보안 교환 시작.

이 값은 채널 보안 엑시트에 대해서만 발생합니다.

보안 교환을 시작하는 기회를 제공하기 위해, MQXR\_INIT로 호출한 직후 수신자의 보안 엑시트는 항상 이 이 유와 함께 호출됩니다. 이러한 기회를 거부하면(MQXCC\_SEND\_SEC\_MSG 또는 MQXCC\_SEND\_AND\_REQUEST\_SEC\_MSG 대신 MQXCC\_OK 리턴) 송신자의 보안 엑시트가 MQXR\_INIT\_SEC와 함께 호출됩니다.

수신자의 보안 엑시트가 보안 교환을 시작하는 경우(MQXCC\_SEND\_SEC\_MSG 또는 MQXCC\_SEND\_AND\_REQUEST\_SEC\_MSG 리턴) 송신자의 보안 엑시트는 MQXR\_INIT\_SEC와 함께 호출되지 않고, 대신 수신자의 메시지를 처리하도록 MQXR\_SEC\_MSG와 함께 호출됩니다. (어느 경우이든 먼저 MQXR\_INIT와 함께 호출됩니다.)

보안 엑시트 중 하나가 채널의 종료를 요구하지 않는 한(*ExitResponse*를 MQXCC\_SUPPRESS\_FUNCTION 또는 MQXCC\_CLOSE\_CHANNEL로 설정), 보안 교환은 교환을 시작한 측에서 완료되어야 합니다. 따라서 보안 엑시트가 MQXR\_INIT\_SEC와 함께 호출되어 교환을 시작한 경우, 다음 번에 엑시트를 호출할 때 MQXR\_SEC\_MSG와 함께 호출됩니다. 이 동작은 엑시트 처리를 위한 보안 메시지의 유무에 상관없이 발생합니다. 파트너가 MQXCC\_SEND\_SEC\_MSG 또는 MQXCC\_SEND\_AND\_REQUEST\_SEC\_MSG를 리턴하는 경우에는 보안 메시지가 있지만, 파트너가 MQXCC\_OK를 리턴하거나 파트너에 보안 엑시트가 없는 경우에는 없습니다. 처리할 보안 메시지가 없는 경우, 시작 측의 보안 엑시트는 값이 0인 *DataLength*와 함께 다시 호출됩니다.

#### **MQXR\_RETRY**

메시지 재시도.

이 값은 메시지 재시도 엑시트에 대해서만 발생합니다.

#### **MQXR\_AUTO\_CLUSSDR**

클러스터 송신자 채널의 자동 정의.

이 값은 채널 자동 정의 엑시트에 대해서만 발생합니다.

#### **MQXR\_AUTO\_RECEIVER**

수신자 채널의 자동 정의.

이 값은 채널 자동 정의 엑시트에 대해서만 발생합니다.

#### **MQXR\_AUTO\_SVRCONN**

서버 연결 채널의 자동 정의.

이 값은 채널 자동 정의 엑시트에 대해서만 발생합니다.

## MQXR\_AUTO\_CLUSRCVR

클러스터 수신자 채널의 자동 정의.

이 값은 채널 자동 정의 엑시트에 대해서만 발생합니다.

## MQXR\_SEC\_PARMS

보안 매개변수

이 값은 보안 엑시트에만 적용되고 MQCSP 구조가 엑시트에 전달되고 있음을 표시합니다. 자세한 정보는 329 페이지의 『MQCSP - 보안 매개변수』의 내용을 참조하십시오.

### 참고:

1. 채널에 둘 이상의 엑시트가 정의된 경우, MCA 시작 시 각각 MQXR\_INIT와 함께 호출됩니다. 또한 MCA 종료 시에는 MQXR\_TERM과 함께 호출됩니다.
2. 채널 자동 정의 엑시트의 경우, *Version*이 MQCXP\_VERSION\_4 미만이면 *ExitReason*이 설정되지 않습니다. 이 경우, MQXR\_AUTO\_SVRCONN 값이 포함됩니다.

엑시트의 입력 필드입니다.

### ExitResponse (MQLONG)

이 필드는 엑시트의 응답을 지정합니다.

이 필드는 MCA와 통신하기 위해 엑시트가 설정합니다. 다음 값 중 하나여야 합니다.

## MQXCC\_OK

엑시트가 성공적으로 완료되었습니다.

- 채널 보안 엑시트의 경우, 이 값은 메시지 전송을 정상적으로 진행할 수 있음을 표시합니다.
- 채널 메시지 재시도 엑시트의 경우, 이 값은 MCA가 MQCXP의 *MsgRetryInterval* 필드에 있는 엑시트가 리턴한 시간 간격 동안 대기했다가 메시지를 다시 시도해야 함을 표시합니다.

*ExitResponse2* 필드에 추가 정보가 있을 수 있습니다.

## MQXCC\_SUPPRESS\_FUNCTION

함수 차단.

- 채널 보안 엑시트의 경우, 이 값은 채널을 종료해야 함을 표시합니다.
- 채널 메시지 엑시트의 경우, 이 값은 메시지가 목적지로 더 이상 진행되지 않음을 표시합니다. 대신, MCA가 예외 보고 메시지를 생성하고(원래 메시지의 송신자가 요청한 경우) 데드-레터 큐의 원래 버퍼에 포함된 메시지를 배치하거나(송신자가 MQRO\_DEAD\_LETTER\_Q를 지정한 경우), 이를 제거합니다(송신자가 MQRO\_DISCARD\_MSG를 지정한 경우).

지속 메시지의 경우, 송신자가 MQRO\_DEAD\_LETTER\_Q를 지정했지만 데드-레터 큐에 넣을 수 없거나 데드-레터 큐가 없으면 원래 메시지는 전송 큐에 남아 있고 보고 메시지가 생성되지 않습니다. 또한 보고 메시지를 생성할 수 없으면 원래 메시지가 전송 큐에 그대로 남아 있습니다.

데드-레터 큐의 메시지 시작 부분에 있는 MQDLH 구조의 *Feedback* 필드는 메시지를 데드-레터 큐에 넣은 이유를 표시합니다. 이 피드백 코드는 예외 보고 메시지의 메시지 디스크립터 필드에도 사용됩니다(송신자가 요청한 경우).

- 채널 메시지 재시도 엑시트의 경우, 이 값은 MCA가 대기했다가 메시지를 다시 시도하지 않음을 표시합니다. 대신, MCA는 일반 실패 처리를 즉시 진행합니다(메시지 송신자가 지정하는 대로 메시지를 데드-레터 큐에 넣거나 제거함).
- 채널 자동 정의 엑시트의 경우, MQXCC\_OK 또는 MQXCC\_SUPPRESS\_FUNCTION을 지정해야 합니다. 두 값 모두 지정되지 않으면 기본적으로 MQXCC\_SUPPRESS\_FUNCTION이 사용되고 자동 정의가 중단됩니다.

이 응답은 채널 송신 및 수신 엑시트에 대해 지원되지 않습니다.

## MQXCC\_SEND\_SEC\_MSG

송신 보안 메시지.

이 값은 채널 보안 엑시트만 설정할 수 있습니다. 엑시트가 파트너로 전송되어야 하는 보안 메시지를 제공했음을 표시합니다.

## MQXCC\_SEND\_AND\_REQUEST\_SEC\_MSG

응답을 요구하는 보안 메시지 송신.

이 값은 채널 보안 엑시트만 설정할 수 있습니다. 다음을 표시합니다.

- 엑시트가 파트너로 전송되어야 하는 보안 메시지를 제공했습니다. 그리고
- 엑시트가 파트너의 응답을 요청합니다. 응답이 수신되지 않으면, 엑시트가 통신의 처리 여부를 아직 결정하지 않았기 때문에 채널을 종료해야 합니다.

## MQXCC\_SUPPRESS\_EXIT

엑시트 차단.

- 이 값은 보안 엑시트 또는 자동 정의 엑시트 이외의 모든 유형의 채널 엑시트에서 설정할 수 있습니다. 엑시트가 MQXR\_TERM의 *ExitReason*과 함께 다시 호출되는 경우 채널이 중단될 때까지 (채널 정의에서 해당 이름이 공백인 것처럼) 해당 엑시트의 추가 호출을 차단합니다.
- 메시지 재시도 엑시트가 이 값을 리턴하는 경우, 후속 메시지에 대한 메시지 재시도는 *MsgRetryCount* 및 *MsgRetryInterval* 채널 속성을 통해 평소대로 제어합니다. 현재 메시지의 경우, MCA는 이유 코드가 MCA가 일반적으로 재시도하는 내용인 경우에 한해 *MsgRetryInterval* 채널 속성이 지정한 간격에 따라 미해결 재시도 횟수를 수행합니다(1409 페이지의 『MQCD - 채널 정의』에 설명된 *MsgRetryCount* 필드 참조). 미해결 재시도 수는 **MsgRetryCount** 속성의 값이며 엑시트가 현재 메시지에 MQXCC\_OK를 리턴한 횟수보다 적습니다. 이 수가 음수이면 MCA는 현재 메시지에 대해 더 이상 재시도를 수행하지 않습니다.

## MQXCC\_CLOSE\_CHANNEL

채널 닫기.

이 값은 자동 정의 엑시트를 제외한 모든 유형의 채널 엑시트가 설정할 수 있습니다.

대화 공유를 사용할 수 없는 경우, 이 값은 채널을 닫습니다.

대화 공유를 사용할 수 있는 경우, 이 값은 대화를 종료합니다. 이 대화가 채널의 유일한 대화이면 채널도 닫힙니다.

이 필드는 엑시트의 입출력(I/O) 필드입니다.

### ExitResponse2 (MQLONG)

이 필드는 엑시트로부터의 2차 응답을 지정합니다.

이 필드는 엑시트 루틴에 대한 입력에서 0으로 설정됩니다. 엑시트가 IBM MQ 채널 함수에 추가 정보를 제공하기 위해 설정할 수 있습니다. 자동 정의 엑시트에는 사용되지 않습니다.

엑시트는 다음 값 중 하나 이상을 설정할 수 있습니다. 둘 이상의 값이 필요하면 값이 추가됩니다. 유효하지 않은 조합이 명시되어 있으며, 그 외의 조합은 허용됩니다.

## MQXR2\_PUT\_WITH\_DEF\_ACTION

기본 조치로 넣기.

이 값은 수신자의 채널 메시지 엑시트가 설정합니다. MCA의 기본 조치로 메시지를 넣게 되고 이는 MCA의 기본 사용자 ID 또는 메시지의 MQMD(메시지 디스크립터)의 컨텍스트 *UserIdentifier*임을 표시합니다.

값은 엑시트 호출 시 설정된 초기값인 0입니다. 이 상수는 설명 목적으로 제공됩니다.

## MQXR2\_PUT\_WITH\_DEF\_USERID

기본 사용자 ID로 넣기.

이 값은 수신자의 채널 메시지 엑시트만 설정할 수 있습니다. MCA의 기본 사용자 ID로 메시지를 넣게 됨을 표시합니다.

## MQXR2\_PUT\_WITH\_MSG\_USERID

메시지 사용자 ID로 넣기.

이 값은 수신자의 채널 메시지 엑시트만 설정할 수 있습니다. 메시지의 MQMD(메시지 디스크립터)의 컨텍스트 *UserIdentifier*를 사용하여 메시지를 넣게 됨을 표시합니다(엑시트에 의해 수정될 수 있음).

MQXR2\_PUT\_WITH\_DEF\_ACTION, MQXR2\_PUT\_WITH\_DEF\_USERID, MQXR2\_PUT\_WITH\_MSG\_USERID 중 하나만 설정해야 합니다.



## **MQXR2\_USE\_AGENT\_BUFFER**

에이전트 버퍼 사용.

이 값은 전달되는 데이터가 *ExitBufferAddr*이 아닌 *AgentBuffer*에 있음을 표시합니다.

값은 엑시트 호출 시 설정된 초기값인 0입니다. 이 상수는 설명 목적으로 제공됩니다.

## **MQXR2\_USE\_EXIT\_BUFFER**

엑시트 버퍼 사용.

이 값은 전달되는 데이터가 *AgentBuffer*이 아닌 *ExitBufferAddr*에 있음을 표시합니다.

MQXR2\_USE\_AGENT\_BUFFER와 MQXR2\_USE\_EXIT\_BUFFER 중 하나만 설정해야 합니다.

## **MQXR2\_DEFAULT\_CONTINUATION**

기본 연속.

체인의 다음 엑시트로 연속되는지 여부는 마지막으로 호출된 엑시트의 응답에 따라 결정됩니다.

- MQXCC\_SUPPRESS\_FUNCTION이나 MQXCC\_CLOSE\_CHANNEL이 리턴되는 경우, 체인의 다른 엑시트가 호출되지 않습니다.
- 그렇지 않으면, 체인의 다음 엑시트가 호출됩니다.

## **MQXR2\_CONTINUE\_CHAIN**

다음 엑시트 계속.

## **MQXR2\_SUPPRESS\_CHAIN**

체인의 나머지 엑시트 건너뛰기.

이는 엑시트의 입출력(I/O) 필드입니다.

### *Feedback(MQLONG)*

이 필드는 피드백 코드를 지정합니다.

이 필드는 엑시트 루틴에 대한 입력에서 MQFB\_NONE으로 설정됩니다.

채널 메시지 엑시트가 *ExitResponse* 필드를 MQXCC\_SUPPRESS\_FUNCTION으로 설정하는 경우, *Feedback* 필드는 데드 레터(미배달 메시지) 큐에 메시지를 넣은 이유를 식별하고, 요청에 따라 예외 보고를 송신해 주는 피드백 코드를 지정합니다. 이 경우, *Feedback* 필드가 MQFB\_NONE이면 다음 피드백 코드가 사용됩니다.

## **MQFB\_STOPPED\_BY\_MSG\_EXIT**

채널 메시지 엑시트가 중지한 메시지.

채널 보안, 송신, 수신 및 메시지 재시도에 의해 이 필드에 리턴된 값이 MCA에 사용되지 않습니다.

자동 정의 엑시트에 의해 이 필드에 리턴된 값은 *ExitResponse*가 MQXCC\_OK인 경우에는 사용되지 않고, 그렇지 않은 경우에는 이벤트 메시지의 *AuxErrorDataInt1* 매개변수에 사용됩니다.

엑시트의 입출력(I/O) 필드입니다.

### *MaxSegmentLength (MQLONG)*

이 필드는 단일 전송에서 송신할 수 있는 최대 바이트 길이를 지정합니다.

자동 정의 엑시트에는 사용되지 않습니다. 전송 세그먼트의 크기가 *MaxSegmentLength*보다 큰 값으로 늘어나지 않도록 이 엑시트가 확인하기 때문에 채널 송신 엑시트에 필요합니다. 이 길이에는 엑시트가 변경하지 않아야 하는 초기 8바이트가 포함됩니다. 해당 값은 채널 시작 시 IBM MQ 채널 함수 간에 협상됩니다. 세그먼트 길이에 대한 자세한 정보는 채널 엑시트 프로그램 작성의 내용을 참조하십시오.

*ExitReason*이 MQXR\_INIT이면 이 필드의 값은 의미가 없습니다.

엑시트의 입력 필드입니다.

### *ExitUserArea (MQBYTE16)*

이 필드는 엑시트가 사용할 수 있는 필드인 엑시트 사용자 영역을 지정합니다.

엑시트(*ExitReason*이 MQXR\_INIT로 설정됨)의 첫 호출 전에 2진 0으로 초기화되므로 엑시트가 이 필드에 수행한 모든 변경사항은 엑시트 호출에서 보존됩니다.

다음 값이 정의됩니다.

#### **MQXUA\_NONE**

사용자 정보가 없습니다.

이 값은 필드 길이 만큼의 2진 0입니다.

C 프로그래밍 언어의 경우, 상수 MQXUA\_NONE\_ARRAY도 정의됩니다. 이 상수는 MQXUA\_NONE과 동일한 값을 갖지만, 문자열이 아니라 문자 배열입니다.

이 필드의 길이는 MQ\_EXIT\_USER\_AREA\_LENGTH에서 제공합니다. 이는 엑시트의 입출력(I/O) 필드입니다.

#### *ExitData (MQCHAR32)*

이 필드는 엑시트 데이터를 지정합니다.

이 필드는 엑시트 루틴에 대한 입력에서 IBM MQ 채널 함수가 채널 정의에서 가져온 정보로 설정됩니다. 이러한 정보를 사용할 수 없으면 이 필드는 모두 공백입니다.

이 필드의 길이는 MQ\_EXIT\_DATA\_LENGTH에서 제공합니다.

엑시트의 입력 필드입니다.

*Version*이 MQCXP\_VERSION\_2 미만이면 이 구조의 다음 필드가 표시되지 않습니다.

#### *MsgRetryCount (MQLONG)*

이 필드는 메시지의 재시도 횟수를 지정합니다.

특정 메시지에 대해 엑시트가 처음으로 호출될 때는 이 필드의 값이 0(아직 재시도 안 함)입니다. 해당 메시지에 대해 이후 호출할 때마다 MCA에서 값이 1씩 증가합니다.

엑시트의 입력 필드입니다. *ExitReason*이 MQXR\_INIT이면 이 필드의 값은 의미가 없습니다. 이 필드는 *Version*이 MQCXP\_VERSION\_2 미만인 경우에는 표시되지 않습니다.

#### *MsgRetryInterval (MQLONG)*

이 필드는 Put 조작을 재시도하는 최소 간격(밀리초)을 지정합니다.

특정 메시지에 대해 엑시트를 처음 호출하면 이 필드에 *MsgRetryInterval* 채널 속성 값이 있습니다. 이 엑시트는 값을 그대로 두거나, 수정하여 다른 시간 간격(밀리초)을 지정할 수 있습니다. 엑시트가 *ExitResponse*에 MQXCC\_OK를 리턴하면 MCA는 최소한 이 시간 간격 동안 대기했다가 MQOPEN 또는 MQPUT 조작을 재시도합니다. 지정된 시간 간격은 0 이상이어야 합니다.

해당 메시지에 대해 엑시트를 두 번째 호출하거나 이후 호출할 때 이 필드에 이전 엑시트 호출에서 리턴된 값이 들어 있습니다.

*MsgRetryInterval* 필드에 리턴된 값이 0보다 작거나 999,999,999보다 크고 *ExitResponse*가 MQXCC\_OK인 경우, MCA에서 MQCXP의 *MsgRetryInterval* 필드를 무시하고 *MsgRetryInterval* 채널 속성이 지정한 간격 동안 대기합니다.

이는 엑시트의 입출력(I/O) 필드입니다. *ExitReason*이 MQXR\_INIT이면 이 필드의 값은 의미가 없습니다. 이 필드는 *Version*이 MQCXP\_VERSION\_2 미만인 경우에는 표시되지 않습니다.

#### *MsgRetryReason (MQLONG)*

이 필드는 이전 메시지 넣기 시도의 이유 코드를 지정합니다.

이 필드는 이전 메시지 넣기 시도의 이유 코드로, MQRC\_\* 값 중 하나입니다.

엑시트의 입력 필드입니다. *ExitReason*이 MQXR\_INIT이면 이 필드의 값은 의미가 없습니다. 이 필드는 *Version*이 MQCXP\_VERSION\_2 미만인 경우에는 표시되지 않습니다.

*Version*이 MQCXP\_VERSION\_3 미만이면 이 구조의 다음 필드가 표시되지 않습니다.

#### *HeaderLength (MQLONG)*

이 필드는 헤더 정보의 길이를 지정합니다.

이 필드는 메시지 엑시트 및 메시지 재시도 엑시트에 대해서만 관련이 있습니다. 이 값은 메시지 데이터의 시작 부분에 있는 라우팅 헤더 구조의 길이입니다. MQXQH 구조, MQMDE(메시지 설명 확장 헤더), (분배 목록 메시지의 경우) MQDH 구조와 MQXQH 구조를 따르는 MQOR 및 MQPMR 레코드의 배열입니다.

메시지 엑시트는 헤더 정보를 조사하고 필요에 따라 수정하지만 엑시트가 리턴한 데이터가 여전히 올바른 형식이어야 합니다. 예를 들어, 수신 측 메시지 엑시트가 보완적으로 변경하더라도 송신 측에서 헤더 데이터를 암호화하거나 압축하지 않아야 합니다.

메시지 엑시트가 길이를 변경하듯 헤더 정보를 수정하는 경우(예: 분배 목록 메시지에 다른 목적지 추가), 리턴 전에 그에 따라 *HeaderLength*의 값을 변경해야 합니다.

이는 엑시트의 입출력(I/O) 필드입니다. *ExitReason*이 MQXR\_INIT이면 이 필드의 값은 의미가 없습니다. *Version*이 MQCXP\_VERSION\_3 미만이면 이 필드가 존재하지 않습니다.

#### *PartnerName (MQCHAR48)*

이 필드는 파트너의 이름을 지정합니다.

파트너의 이름은 다음과 같습니다.

- SVRCONN 채널의 경우, 클라이언트에서 로그인한 사용자 ID입니다.
- 다른 모든 채널 유형의 경우, 파트너의 큐 관리자 이름입니다.

엑시트를 초기화할 때, 초기 협상 전까지 큐 관리자가 파트너 이름을 알지 못하기 때문에 이 필드는 공백입니다.

엑시트의 입력 필드입니다. *Version*이 MQCXP\_VERSION\_3 미만이면 이 필드가 존재하지 않습니다.

#### *FAPLevel (MQLONG)*

협상된 형식 및 프로토콜 레벨입니다.

엑시트의 입력 필드입니다. 이 필드는 IBM 서비스의 지시에 따라서만 변경되어야 합니다. *Version*이 MQCXP\_VERSION\_3 미만이면 이 필드가 존재하지 않습니다.

#### *CapabilityFlags (MQLONG)*

MQCF\_NONE 또는 MQCF\_DIST\_LISTS로 용량 플래그를 설정할 수 있습니다.

다음 용량 플래그를 설정할 수 있습니다.

#### **MQCF\_NONE**

플래그가 없습니다.

#### **MQCF\_DIST\_LISTS**

분배 목록이 지원됩니다.

엑시트의 입력 필드입니다. *Version*이 MQCXP\_VERSION\_3 미만이면 이 필드가 존재하지 않습니다.

#### *ExitNumber (MQLONG)*

이 필드는 엑시트의 서수를 지정합니다.

*ExitId*에 정의된 유형 내에서 엑시트의 서수입니다. 예를 들어, 호출되는 엑시트가 정의된 세 번째 메시지 엑시트인 경우 이 필드에는 값 3이 포함됩니다. 엑시트 유형이 엑시트 목록을 정의할 수 없는 유형(예: 보안 엑시트)인 경우 이 필드의 값은 1입니다.

엑시트의 입력 필드입니다. *Version*이 MQCXP\_VERSION\_3 미만이면 이 필드가 존재하지 않습니다.

*Version*이 MQCXP\_VERSION\_5 미만이면 이 구조의 다음 필드가 표시되지 않습니다.

#### *ExitSpace (MQLONG)*

이 필드는 엑시트가 사용하도록 예약된 전송 버퍼의 바이트 수를 지정합니다.

이 필드는 송신 엑시트에 대해서만 관련이 있습니다. IBM MQ 채널 함수가 전송 큐에서 엑시트가 사용하도록 예약하는 공간(바이트)을 지정합니다. 이 필드는 엑시트가 다른 끝의 보안 수신 엑시트에 사용하도록 소량의 데이터(대개, 수백 바이트를 초과하지 않음)를 전송 큐에 추가할 수 있도록 합니다. 송신 엑시트에서 추가한 데이터는 수신 엑시트에서 제거해야 합니다.

z/OS에서는 값이 항상 0입니다.

**참고:** 이 기능은 성능 저하를 유발하거나 채널 작동을 중단시킬 수 있으므로 대량의 데이터를 송신하는 데 사용하지 않아야 합니다.

*ExitSpace*를 설정하면 엑시트가 사용할 수 있도록 항상 전송 큐에 최소한 해당 수의 바이트가 있음이 보장됩니다. 그러나, 엑시트가 예약된 크기보다 더 적게 사용하거나 전송 버퍼에 여유 공간이 있어 예약된 크기보다 더 많이 사용할 수도 있습니다. 버퍼의 엑시트 공간은 기존 데이터 다음에 제공됩니다.

엑시트는 *ExitReason*의 값이 MQXR\_INIT인 경우에만 *ExitSpace*를 설정할 수 있습니다. 그 밖의 모든 경우에는 엑시트가 리턴한 값이 무시됩니다. 엑시트에 대한 입력에서 MQXR\_INIT 호출에 대한 *ExitSpace*는 0이고, 다른 경우에는 MQXR\_INIT 호출에서 리턴된 값입니다.

MQXR\_INIT 호출이 리턴한 값이 음수이거나 체인의 모든 송신 엑시트에 대해 요청된 엑시트 공간을 예약한 후 전송 버퍼에 메시지 데이터용으로 1024바이트 미만이 있는 경우, MCA는 오류 메시지를 출력하고 채널을 닫습니다. 마찬가지로, 데이터 전송 중에 송신 엑시트 체인의 엑시트가 예약된 것보다 더 많은 사용자 공간을 할당하여 전송 버퍼에서 메시지 데이터에 대해 1024바이트보다 적은 공간이 남으면, MCA가 오류 메시지를 출력하고 채널을 닫습니다. 한계를 1024로 설정하면 플로우를 분할할 필요 없이 채널의 제어 및 관리 플로우를 송신 엑시트 체인에서 처리할 수 있습니다.

이는 *ExitReason*이 MQXR\_INIT이면 엑시트에 대한 입출력(I/O) 필드이고, 그 밖의 모든 경우에는 입력 필드입니다. *Version*이 MQCXP\_VERSION\_5 미만이면 필드가 표시되지 않습니다.

#### *SSLCertUserId* (MQCHAR12)

이 필드는 리모트 인증서와 연관된 *UserId*를 지정합니다.

z/OS를 제외한 모든 플랫폼에서 공백임

엑시트의 입력 필드입니다. *Version*이 MQCXP\_VERSION\_6 미만이면 필드가 표시되지 않습니다.

#### *SSLRemCertIssNameLength* (MQLONG)

이 필드는 *SSLCertRemoteIssuerNamePtr*이 가리키는 리모트 인증서 발행자의 전체 식별 이름의 길이(바이트)를 지정합니다.

엑시트의 입력 필드입니다. *Version*이 MQCXP\_VERSION\_6 미만이면 필드가 표시되지 않습니다. TLS 채널이 아니면 값이 0입니다.

#### *SSLRemCertIssNamePtr* (PMQVOID)

이 필드는 리모트 인증서 발행자의 전체 식별 이름의 주소를 지정합니다.

TLS 채널이 아니면 값이 널 포인터입니다.

엑시트의 입력 필드입니다. *Version*이 MQCXP\_VERSION\_6 미만이면 필드가 표시되지 않습니다.

**참고:** IBM WebSphere MQ 7.1부터 주제 식별 이름 및 송신자 식별 이름을 판별할 때 채널 보안 엑시트의 동작이 변경됩니다. 자세한 정보는 [채널 보안 엑시트 프로그램](#)을 참조하십시오.

#### *SecurityParms* (PMQCSP)

이 필드는 인증 신임 정보를 지정하는 데 사용되는 MQCSP 구조의 주소를 지정합니다.

이 필드의 초기값은 널 포인터입니다.

이는 엑시트의 입출력(I/O) 필드입니다. *Version*이 MQCXP\_VERSION\_6 미만이면 필드가 표시되지 않습니다.

이 필드에서 엑시트가 리턴하는 값은 MQXR\_TERM까지 IBM MQ에 의해 사용 가능해야 합니다.

#### *CurHdrCompression* (MQLONG)

이 필드는 현재 헤더 데이터를 압축하는 데 사용되는 기술을 지정합니다.

이 필드는 다음 중 하나로 설정됩니다.

#### **MQCOMPRESS\_NONE**

헤더 데이터 압축이 수행되지 않습니다.

#### **MQCOMPRESS\_SYSTEM**

헤더 데이터 압축이 수행됩니다.

이 값은 송신 채널의 메시지 엑시트가 MQCD의 HdrCompList 필드에서 액세스하고 협상 및 지원되는 값 중 하나로 대체할 수 있습니다. 이렇게 하면 헤더 데이터 압축에 사용된 기술이 메시지 콘텐츠를 기반으로 하는 각 메시지에 선택됩니다. 대체된 값은 현재 메시지에만 사용됩니다. 속성이 지원되지 않는 값으로 대체되면 채널이 종료됩니다. 송신 채널의 메시지 엑시트 외부에서 대체된 경우 값이 무시됩니다.

이는 엑시트의 입출력(I/O) 필드입니다. *Version*이 MQCXP\_VERSION\_6 미만이면 필드가 표시되지 않습니다.

#### *CurMsgCompression (MQLONG)*

이 필드는 현재 메시지 데이터를 압축하는 데 사용되는 기술을 지정합니다.

이 필드는 다음 중 하나로 설정됩니다.

#### **MQCOMPRESS\_NONE**

헤더 데이터 압축이 수행되지 않습니다.

#### **MQCOMPRESS\_RLE**

실행 길이 인코딩을 사용하여 메시지 데이터 압축이 수행됩니다.

#### **MQCOMPRESS\_ZLIBFAST**

메시지 데이터 압축은 zlib 압축 기술을 사용하여 수행합니다. 빠른 압축 시간을 선호합니다.

#### **MQCOMPRESS\_ZLIBHIGH**

메시지 데이터 압축은 zlib 압축 기술을 사용하여 수행합니다. 상위 레벨의 압축을 선호합니다.

이 값은 송신 채널의 메시지 엑시트가 MQCD의 MsgCompList 필드에서 액세스하고 협상 및 지원되는 값 중 하나로 대체할 수 있습니다. 이렇게 하면 메시지 데이터 압축에 사용된 기술이 메시지 콘텐츠를 기반으로 하는 각 메시지에 결정됩니다. 대체된 값은 현재 메시지에만 사용됩니다. 속성이 지원되지 않는 값으로 대체되면 채널이 종료됩니다. 송신 채널의 메시지 엑시트 외부에서 대체된 경우 값이 무시됩니다.

이는 엑시트의 입출력(I/O) 필드입니다. *Version*이 MQCXP\_VERSION\_6 미만이면 필드가 표시되지 않습니다.

#### *Hconn (MQHCONN)*

이 필드는 엑시트 내에서 MQI 호출을 작성해야 하는 경우 엑시트가 사용하는 연결 핸들을 지정합니다.

이 필드는 클라이언트 연결 채널에서 실행되는 엑시트와 관련이 없으며, 이 경우 값이 MQHC\_UNUSABLE\_HCONN (-1)입니다.

엑시트의 입력 필드입니다. *Version*이 MQCXP\_VERSION\_7 미만이면 필드가 표시되지 않습니다.

#### *SharingConversations (MQBOOL)*

이 필드는 대화가 현재 이 채널 인스턴스에서 실행할 수 있는 유일한 대화인지, 둘 이상의 대화를 현재 이 채널 인스턴스에서 실행할 수 있는지 여부를 지정합니다.

엑시트 프로그램이 동시에 실행되는 다른 엑시트 프로그램에서 MQCD를 대체하는 위험에 노출되는지 여부도 표시합니다.

이 필드는 클라이언트 연결 또는 서버 연결 채널에서 실행 중인 엑시트 프로그램만 관련이 있습니다.

이 필드는 다음 중 하나로 설정됩니다.

#### **FALSE**

엑시트 인스턴스는 이 채널 인스턴스에서 현재 실행되는 유일한 엑시트 인스턴스입니다. 따라서 엑시트가 다른 채널 인스턴스에서 실행 중인 기타 엑시트와 경합하지 않고 MQCD 필드를 안전하게 업데이트할 수 있습니다. MQCD 필드의 변경사항이 채널에 적용되는지 여부는 [1445 페이지의 『채널 엑시트에서 MQCD 필드 변경』](#)의 MQCD 필드 테이블에 정의되어 있습니다.

#### **TRUE**

엑시트 인스턴스는 이 채널 인스턴스에서 현재 실행되는 유일한 엑시트 인스턴스가 아닙니다. MQXR\_INIT 이외의 엑시트 이유에 대해 [1445 페이지의 『채널 엑시트에서 MQCD 필드 변경』](#)의 MQCD 필드 테이블에 나열된 변경사항을 제외하고, MQCD의 변경사항이 채널에 적용되지 않습니다. 이 엑시트가 MQCD 필드를 업데이트하는 경우, 이 채널 인스턴스에서 실행되는 엑시트에 직렬화를 제공하여 다른 대화에서 동시에 실행되는 다른 엑시트와 경합하는 일이 없도록 하십시오.

엑시트의 입력 필드입니다. *Version*이 MQCXP\_VERSION\_7 미만이면 필드가 표시되지 않습니다.

#### *MCAUserSource (MQLONG)*

이 필드는 제공된 MCA 사용자 ID의 소스를 지정합니다.

값은 다음 중 하나입니다.

### **MQUSRC\_MAP**

사용자 ID를 MCAUSER 속성에 지정합니다.

### **MQUSRC\_CHANNEL**

사용자 ID를 인바운드 파트너에서 가져오거나 채널 오브젝트에 정의된 MCAUSER 필드에 지정합니다.

엑시트의 입력 필드입니다. Version이 MQCXP\_VERSION\_8 미만이면 필드가 표시되지 않습니다.

#### *pEntryPoints (PMQIEP)*

이 필드는 MQI 또는 DCI 호출의 인터페이스 시작점 주소를 지정합니다.

Version이 MQCXP\_VERSION\_8 미만이면 필드가 표시되지 않습니다.

#### *RemoteProduct (MQCHAR4)*

이 필드는 리모트 제품 이름을 지정합니다.

이 필드는 DISPLAY CHSTATUS의 **RPRODUCT** 필드에 표시된 대로 클라이언트의 원격 제품 (예: C 또는 Java) 을 식별합니다.

Version이 MQCXP\_VERSION\_9 미만이면 필드가 표시되지 않습니다.

#### *RemoteVersion (MQCHAR8)*

이 필드는 리모트 버전의 이름을 지정합니다.

이 필드는 DISPLAY CHSTATUS의 **RVERSION** 필드에 표시된 대로 클라이언트 라이브러리의 버전을 식별합니다.

Version이 MQCXP\_VERSION\_9 미만이면 필드가 표시되지 않습니다.

## **C 선언**

이 선언은 MQCXP 구조에 대한 C 선언입니다.

```
typedef struct tagMQCXP MQCXP;
struct tagMQCXP {
    MQCHAR4    StructId;           /* 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 */
    MQLONG    MaxSegmentLength;  /* Maximum segment length */
    MQBYTE16  ExitUserArea;      /* Exit user area */
    MQCHAR32  ExitData;          /* Exit data */
    MQLONG    MsgRetryCount;     /* Number of times the message has been
    retried */
    MQLONG    MsgRetryInterval;  /* Minimum interval in milliseconds after
    which the put operation should be
    retried */
    MQLONG    MsgRetryReason;    /* Reason code from previous attempt to
    put the message */
    MQLONG    HeaderLength;      /* Length of header information */
    MQCHAR48  PartnerName;      /* Partner Name */
    MQLONG    FAPLevel;         /* Negotiated Formats and Protocols
    level */
    MQLONG    CapabilityFlags;   /* Capability flags */
    MQLONG    ExitNumber;       /* Exit number */
    /* Ver:3 */
    /* Ver:4 */
    MQLONG    ExitSpace;        /* Number of bytes in transmission buffer
    reserved for exit to use */
    /* Ver:5 */
    MQCHAR12  SSLCertUserid;     /* User identifier associated
    with remote TLS certificate */
    MQLONG    SSLRemCertIssNameLength; /* Length of
    distinguished name of issuer
    of remote TLS certificate */
    MQPTR     SSLRemCertIssNamePtr; /* Address of
    distinguished name of issuer
    of remote TLS certificate */
    PMQVOID   SecurityParms;     /* Security parameters */
    MQLONG    CurHdrCompression; /* Header data compression
```

```

MQLONG    CurMsgCompression; /* Message data compression
                                used for current message */
/* Ver:6 */
MQHCONN   Hconn; /* Connection handle */
MQBOOL    SharingConversations; /* Multiple conversations
                                possible on channel inst? */
/* Ver:7 */
MQLONG    MCAUserSource; /* Source of the provided MCA user ID */
PMQIEP    pEntryPoints; /* Address of the MQIEP structure */
/* Ver:8 */
MQCHAR4   RemoteProduct; /* The identifier for the remote product */
MQCHAR8   RemoteVersion; /* The version of the remote product */
/* Ver:9 */
};

```

## COBOL 선언

이 선언은 MQCXP 구조에 대한 COBOL 선언입니다.

```

** MQCXP structure
10 MQCXP.
** Structure identifier
15 MQCXP-STRUCID PIC X(4).
** Structure version number
15 MQCXP-VERSION PIC S9(9) BINARY.
** Type of exit
15 MQCXP-EXITID PIC S9(9) BINARY.
** Reason for invoking exit
15 MQCXP-EXITREASON PIC S9(9) BINARY.
** Response from exit
15 MQCXP-EXITRESPONSE PIC S9(9) BINARY.
** Secondary response from exit
15 MQCXP-EXITRESPONSE2 PIC S9(9) BINARY.
** Feedback code
15 MQCXP-FEEDBACK PIC S9(9) BINARY.
** Maximum segment length
15 MQCXP-MAXSEGMENTLENGTH PIC S9(9) BINARY.
** Exit user area
15 MQCXP-EXITUSERAREA PIC X(16).
** Exit data
15 MQCXP-EXITDATA PIC X(32).
** Number of times the message has been retried
15 MQCXP-MSGRETRYCOUNT PIC S9(9) BINARY.
** Minimum interval in milliseconds after which the put operation
** should be retried
15 MQCXP-MSGRETRYINTERVAL PIC S9(9) BINARY.
** Reason code from previous attempt to put the message
15 MQCXP-MSGRETRYREASON PIC S9(9) BINARY.
** Length of header information
15 MQCXP-HEADERLENGTH PIC S9(9) BINARY.
** Partner Name
15 MQCXP-PARTNERNAME PIC X(48).
** Negotiated Formats and Protocols level
15 MQCXP-FAPLEVEL PIC S9(9) BINARY.
** Capability flags
15 MQCXP-CAPABILITYFLAGS PIC S9(9) BINARY.
** Exit number
15 MQCXP-EXITNUMBER PIC S9(9) BINARY.
** Number of bytes in transmission buffer reserved for exit to use
15 MQCXP-EXITSPACE PIC S9(9) BINARY.
** User Id associated with remote certificate
15 MQCXP-SSLCERTUSERID PIC X(12).
** Length of distinguished name of issuer of remote TLS
** certificate
15 MQCXP-SSLREMCERTISSNAMELENGTH PIC S9(9) BINARY.
** Address of distinguished name of issuer of remote TLS
** certificate
15 MQCXP-SSLREMCERTISSNAMEPTR POINTER.
** Security parameters
15 MQCXP-SECURITYPARMS PIC S9(18) BINARY.
** Header data compression used for current message
15 MQCXP-CURHDRCOMPRESSSION PIC S9(9) BINARY.
** Message data compression used for current message
15 MQCXP-CURMSGCOMPRESSSION PIC S9(9) BINARY.
** Connection handle
15 MQCXP-HCONN PIC S9(9) BINARY.
** Multiple conversations possible on channel instance?
15 MQCXP-SHARINGCONVERSATIONS PIC S9(9) BINARY.

```

```

** Source of the provided MCA user ID
15 MQCXP-MCAUSERSOURCE PIC S9(9) BINARY.
** Identifier of the remote product
15 MQCXP-RPRODUCT PIC X(4).
** Identifier of the remote version
15 MQCXP-RVERSION PIC X(8).

```

## RPG 선언(ILE)

이 선언은 MQCXP 구조에 대한 RPG 선언입니다.

```

D*.1....:....2....:....3....:....4....:....5....:....6....:....7..
D* MQCXP Structure
D*
D* Structure identifier
D CXSID 1 4
D* Structure version number
D CXVER 5 8I 0
D* Type of exit
D CXXID 9 12I 0
D* Reason for invoking exit
D CXREA 13 16I 0
D* Response from exit
D CXRES 17 20I 0
D* Secondary response from exit
D CXRE2 21 24I 0
D* Feedback code
D CXFB 25 28I 0
D* Maximum segment length
D CXMSL 29 32I 0
D* Exit user area
D CXUA 33 48
D* Exit data
D CXDAT 49 80
D* Number of times the message has been retried
D CXMRC 81 84I 0
D* Minimum interval in milliseconds after which the put operation
D* should be retried
D CXMRI 85 88I 0
D* Reason code from previous attempt to put the message
D CXMRR 89 92I 0
D* Length of header information
D CXHDL 93 96I 0
D* Partner Name
D CXPNM 97 144
D* Negotiated Formats and Protocols level
D CXFAP 145 148I 0
D* Capability flags
D CXCAP 149 152I 0
D* Exit number
D CXEXN 153 156I 0
D* Number of bytes in transmission buffer reserved for exit to use
D CXHDL 157 160I 0
D* User identifier associated with remote TLS certificate
D CXSSLCU 161 172
D* Length of distinguished name of issuer of remote TLS certificate
D CXSRCINL 173 176I 0
D* Address of distinguished name of issuer of remote TLS certificate
D CXSRCINP 177 192*
D* Security parameters
D CXSECP 193 208*
D* Header data compression used for current message
D CXCHC 209 212I 0
D* Message data compression used for current message
D CXCMC 213 216I 0
D* Connection handle
D CXHCONN 217 220I 0
D* Multiple conversations possible on channel instance?
D CXSHARECONV 221 224I 0
D* Source of the provided MCA user ID
D MCAUSERSOURCE 225 228I 0
D* Identifier of the remote product
D CXRPRO 229 232I 0
D* Identifier of the remote version
D CXRVER 233 240I 0

```



## System/390 어셈블러 선언

이 선언은 MQCXP 구조에 대한 System/390 어셈블러 선언입니다.

```
MQCXP          DSECT
MQCXP_STRUCID  DS    CL4  Structure identifier
MQCXP_VERSION  DS    F    Structure version number
MQCXP_EXITID   DS    F    Type of exit
MQCXP_EXITREASON DS    F    Reason for invoking exit
MQCXP_EXITRESPONSE DS    F    Response from exit
MQCXP_EXITRESPONSE2 DS    F    Secondary response from exit
MQCXP_FEEDBACK DS    F    Feedback code
MQCXP_MAXSEGMENTLENGTH DS    F    Maximum segment length
MQCXP_EXITUSERAREA DS    XL16 Exit user area
MQCXP_EXITDATA DS    CL32 Exit data
MQCXP_MSGRETRYCOUNT DS    F    Number of times the message has been
*          retried
MQCXP_MSGRETRYINTERVAL DS    F    Minimum interval in milliseconds
*          after which the put operation should
*          be retried
MQCXP_MSGRETRYREASON DS    F    Reason code from previous attempt to
*          put the message
MQCXP_HEADERLENGTH DS    F    Length of header information
MQCXP_PARTNERNAME DS    CL48 Partner Name
MQCXP_FAPLEVEL  DS    F    Negotiated Formats and Protocols
*          level
MQCXP_CAPABILITYFLAGS DS    F    Capability flags
MQCXP_EXITNUMBER DS    F    Exit number
MQCXP_EXITSPEACE DS    F    Number of bytes in transmission
*          buffer reserved for exit to use
MQCXP_SSLCERTUSERID DS    CL12 User identifier associated with
*          remote TLS certificate
MQCXP_SSLREMCERTISSNAMELENGTH DS    F    Length of distinguished name
*          of issuer of remote TLS certificate
MQCXP_SSLREMCERTISSNAMEPTR DS    F    Address of distinguished name
*          of issuer of remote TLS certificate
MQCXP_SECURITYPARMS DS    F    Address of security parameters
MQCXP_CURHDRCOMPRESSION DS    F    Header data compression used for
*          current message
MQCXP_CURMSGCOMPRESSION DS    F    Message data compression used for
*          current message
MQCXP_HCONN    DS    F    Connection handle
MQCXP_SHARINGCONVERSATIONS DS    F    Multiple conversations possible on
*          channel inst?
MQCXP_MCAUSERSOURCE DS    F    Source of the provided MCA user ID
MQCXP_RPRODUCT  DS    CL4  Identifier of the remote product
MQCXP_RVERSION  DS    CL8  Identifier of the remote version

MQCXP_LENGTH   EQU    *-MQCXP
                ORG    MQCXP
MQCXP_AREA     DS    CL(MQCXP_LENGTH)
```

## MQXWD - 엑시트 대기 디스크립터

MQXWD 구조는 MQXWAIT 호출의 입출력(I/O) 매개변수입니다.

이 구조는 z/OS에서만 지원됩니다.

### 관련 참조

[1461 페이지의 『필드』](#)

이 주제는 MQXWD 구조의 모든 필드를 나열하고 각 필드를 설명합니다.

[1462 페이지의 『C 선언』](#)

이 선언은 MQXWD 구조에 대한 C 선언입니다.

[1462 페이지의 『System/390 어셈블러 선언』](#)

이 선언은 MQXWD 구조에 대한 System/390 어셈블러 선언입니다.

### 필드

이 주제는 MQXWD 구조의 모든 필드를 나열하고 각 필드를 설명합니다.

*StrucId (MQCHAR4)*

이 필드는 구조 ID를 지정합니다.

값은 다음과 같아야 합니다.

### **MQXWD\_STRUC\_ID**

엑시트 대기 디스크립터 구조의 ID

C 프로그래밍 언어의 경우, 상수 MQXWD\_STRUC\_ID\_ARRAY도 정의됩니다. 이 상수는 MQXWD\_STRUC\_ID와 동일한 값을 갖지만, 문자열이 아니라 문자 배열입니다.

이 필드의 초기값은 MQXWD\_STRUC\_ID입니다.

### *Version(MQLONG)*

이 필드는 구조 버전 번호를 지정합니다.

값은 다음과 같아야 합니다.

### **MQXWD\_VERSION\_1**

엑시트 대기 디스크립터 구조의 버전 번호.

이 필드의 초기값은 MQXWD\_STRUC\_1입니다.

### *Reserved1 (MQLONG)*

이 필드는 예약됩니다. 값이 0이어야 합니다.

입력 필드입니다.

### *Reserved2 (MQLONG)*

이 필드는 예약됩니다. 값이 0이어야 합니다.

입력 필드입니다.

### *Reserved3 (MQLONG)*

이 필드는 예약됩니다. 값이 0이어야 합니다.

입력 필드입니다.

### *ECB (MQLONG)*

이 필드는 대기하는 이벤트 제어 블록을 지정합니다.

이 필드는 대기하는 이벤트 제어 블록(ECB)입니다. MQXWAIT 호출 발행 전에는 0으로 설정해야 하고, 성공적으로 완료되면 게시 코드가 들어 있습니다.

이 필드는 입출력(I/O) 필드입니다.

## **C 선언**

이 선언은 MQXWD 구조에 대한 C 선언입니다.

```
typedef struct tagMQXWD MQXWD;
struct tagMQXWD {
    MQLONG  StrucId;      /* Structure identifier */
    MQLONG  Version;     /* Structure version number */
    MQLONG  Reserved1;   /* Reserved */
    MQLONG  Reserved2;   /* Reserved */
    MQLONG  Reserved3;   /* Reserved */
    MQLONG  ECB;         /* Event control block to wait on */
};
```

## **System/390 어셈블러 선언**

이 선언은 MQXWD 구조에 대한 System/390 어셈블러 선언입니다.

```
MQXWD          DSECT
MQXWD_STRUCID  DS    CL4  Structure identifier
MQXWD_VERSION  DS    F    Structure version number
MQXWD_RESERVED1 DS    F    Reserved
MQXWD_RESERVED2 DS    F    Reserved
MQXWD_RESERVED3 DS    F    Reserved
MQXWD_ECB      DS    F    Event control block to wait on
*
```

MQXWD_LENGTH	EQU	*-MQXWD
	ORG	MQXWD
MQXWD_AREA	DS	CL(MQXWD_LENGTH)

## 클러스터 워크로드 엑시트 호출 및 데이터 구조

이 절에서는 클러스터 워크로드 엑시트 및 데이터 구조에 대한 참조 정보를 제공합니다. 이는 일반 사용 프로그래밍 인터페이스 정보입니다.


다음 프로그래밍 언어로 클러스터 워크로드 엑시트를 작성할 수 있습니다.

- C
- System/390 어셈블러(IBM MQ for z/OS)

호출은 다음에 설명되어 있습니다.

- [1463 페이지의 『MQ\\_CLUSTER\\_WORKLOAD\\_EXIT - 호출 설명』](#)

엑시트에서 사용하는 구조 데이터 유형은 다음에 설명되어 있습니다.

- [1465 페이지의 『MQXCLWLN - 클러스터 워크로드 레코드 탐색』](#)
- [1469 페이지의 『MQWXP - 클러스터 워크로드 엑시트 매개변수 구조』](#)
- [1476 페이지의 『MQWDR - 클러스터 워크로드 목적지 레코드 구조』](#)
- [1481 페이지의 『MQWQR - 클러스터 워크로드 큐 레코드 구조』](#)
- [1486 페이지의 『MQWCR - 클러스터 워크로드 클러스터 레코드 구조』](#)
-  [z/OS에서 CLUSTER 명령의 비동기 동작](#)

이 절 전체에서는 큐 관리자 속성 및 큐 속성을 모두 보여줍니다. MQSC 명령에서 사용되는 동등한 이름은 아래에 표시되어 있습니다. MQSC 명령에 대한 세부사항은 [MQSC 명령](#)을 참조하십시오.

표 824. 큐 관리자 속성	
전체 이름	MQSC에서 사용된 이름
<i>ClusterWorkloadData</i>	CLWLDATA
<i>ClusterWorkloadExit</i>	CLWLEXIT
<i>ClusterWorkloadLength</i>	CLWLLEN

표 825. 큐 속성	
전체 이름	MQSC에서 사용된 이름
<i>DefBind</i>	DEFBIND
<i>DefPersistence</i>	DEFPSIST
<i>DefPriority</i>	DEFPRTY
<i>InhibitPut</i>	PUT
<i>QDesc</i>	DESCR

### 관련 태스크

[클러스터 워크로드 엑시트 작성 및 컴파일](#)

## MQ\_CLUSTER\_WORKLOAD\_EXIT - 호출 설명

클러스터 워크로드 엑시트는 큐 관리자가 메시지를 사용 가능한 큐 관리자로 라우트하기 위해 호출합니다.

**참고:** 큐 관리자가 MQ\_CLUSTER\_WORKLOAD\_EXIT(이)라는 시작점을 제공하지 않습니다. 대신, 클러스터 워크로드 엑시트의 이름은 ClusterWorkloadExit 큐 관리자 속성에 의해 정의됩니다.

MQ\_CLUSTER\_WORKLOAD\_EXIT 엑시트는 모든 플랫폼에서 지원됩니다.

## 구문

MQ\_CLUSTER\_WORKLOAD\_EXIT (*ExitParms*)

### 관련 참조

[MQXCLWLN](#) - 클러스터 워크로드 레코드 탐색

MQXCLWLN 호출은 클러스터 캐시에 저장된 MQWDR, MQWQR 및 MQWCR 레코드의 체인을 통해 탐색하는 데 사용됩니다.

[MQWXP](#) - 클러스터 워크로드 엑시트 매개변수 구조

다음 표에는 MQWXP(클러스터 워크로드 엑시트 매개변수 구조)의 필드가 요약되어 설명되어 있습니다.

[MQWDR](#) - 클러스터 워크로드 목적지 레코드 구조

다음 표에는 MQWDR(클러스터 워크로드 목적지 레코드 구조)의 필드가 요약되어 설명되어 있습니다.

[MQWQR](#) - 클러스터 워크로드 큐 레코드 구조

다음 표에는 MQWQR(클러스터 워크로드 큐 레코드 구조)의 필드가 요약되어 설명되어 있습니다.

[MQWCR](#) - 클러스터 워크로드 클러스터 레코드 구조

다음 표에는 MQWCR 클러스터 워크로드 레코드 구조의 필드가 요약되어 설명되어 있습니다.

### **MQ\_CLUSTER\_WORKLOAD\_EXIT의 매개변수**

MQ\_CLUSTER\_WORKLOAD\_EXIT 호출에서 매개변수의 설명입니다.

#### **ExitParms (MQWXP) - 입출력(I/O)**

엑시트 매개변수 블록.

- 엑시트는 워크로드를 관리하는 방법을 표시하기 위해 MQWXP에 정보를 설정합니다.

### 관련 참조

[사용법 참고](#)

클러스터 워크로드 엑시트에 의해 수행되는 함수는 엑시트의 제공자에 의해 정의됩니다. 그러나 엑시트는 연관된 제어 블록 MQWXP에 정의되어 있는 규칙에 따라야 합니다.

[MQ\\_CLUSTER\\_WORKLOAD\\_EXIT의 언어 호출](#)

MQ\_CLUSTER\_WORKLOAD\_EXIT에서는 두 개의 언어(C 및 상위 레벨 어셈블러)를 지원합니다.

### **사용법 참고**

클러스터 워크로드 엑시트에 의해 수행되는 함수는 엑시트의 제공자에 의해 정의됩니다. 그러나 엑시트는 연관된 제어 블록 MQWXP에 정의되어 있는 규칙에 따라야 합니다.

큐 관리자가 MQ\_CLUSTER\_WORKLOAD\_EXIT(이)라는 시작점을 제공하지 않습니다. 그러나 C 프로그래밍 언어의 MQ\_CLUSTER\_WORKLOAD\_EXIT 이름에는 typedef가 제공됩니다. 매개변수가 유효할 수 있도록 사용자 작성 엑시트를 선언하려면 typedef를 사용하십시오.

### 관련 참조

[MQ\\_CLUSTER\\_WORKLOAD\\_EXIT의 매개변수](#)

MQ\_CLUSTER\_WORKLOAD\_EXIT 호출에서 매개변수의 설명입니다.

[MQ\\_CLUSTER\\_WORKLOAD\\_EXIT의 언어 호출](#)

MQ\_CLUSTER\_WORKLOAD\_EXIT에서는 두 개의 언어(C 및 상위 레벨 어셈블러)를 지원합니다.

### **MQ\_CLUSTER\_WORKLOAD\_EXIT의 언어 호출**

MQ\_CLUSTER\_WORKLOAD\_EXIT에서는 두 개의 언어(C 및 상위 레벨 어셈블러)를 지원합니다.

## C 호출

```
MQ_CLUSTER_WORKLOAD_EXIT (&ExitParms);
```

MQ\_CLUSTER\_WORKLOAD\_EXIT을(를) 클러스터 워크로드 엑시트 함수의 이름으로 바꾸십시오.

다음과 같이 **MQ\_CLUSTER\_WORKLOAD\_EXIT** 매개변수를 선언하십시오.

```
MQWXP ExitParms; /* Exit parameter block */
```

## 상위 레벨 어셈블러 호출

```
CALL EXITNAME, (EXITPARMS)
```

매개변수를 다음과 같이 선언합니다.

```
EXITPARMS      CMQWXA      Exit parameter block
```

### 관련 참조

[MQ\\_CLUSTER\\_WORKLOAD\\_EXIT](#)의 매개변수

MQ\_CLUSTER\_WORKLOAD\_EXIT 호출에서 매개변수의 설명입니다.

### 사용법 참고

클러스터 워크로드 엑시트에 의해 수행되는 함수는 엑시트의 제공자에 의해 정의됩니다. 그러나 엑시트는 연관된 제어 블록 MQWXP에 정의되어 있는 규칙에 따라야 합니다.

## MQXCLWLN - 클러스터 워크로드 레코드 탐색

MQXCLWLN 호출은 클러스터 캐시에 저장된 MQWDR, MQWQR 및 MQWCR 레코드의 체인을 통해 탐색하는 데 사용됩니다.

클러스터 캐시는 클러스터 관련 정보를 저장하는 데 사용되는 주 기억장치의 영역입니다.

클러스터 캐시가 정적이면 이는 고정된 크기를 갖습니다. 이를 동적으로 설정하면 클러스터 캐시가 필요에 따라 확장될 수 있습니다.

시스템 매개변수 또는 매크로를 사용하여 클러스터 캐시의 유형을 STATIC 또는 DYNAMIC으로 설정하십시오.

- ▶ **Multi** 멀티플랫폼에서 시스템 매개변수 ClusterCacheType 을 사용하십시오.
- ▶ **z/OS** z/OS의 CSQ6SYSP 매크로에서 CLCACHE 매개변수를 사용하십시오.

## 구문

```
MQXCLWLN (ExitParms, CurrentRecord, NextOffset, NextRecord, Compcode, Reason)
```

### 관련 참조

[MQ\\_CLUSTER\\_WORKLOAD\\_EXIT](#) - 호출 설명

클러스터 워크로드 엑시트는 큐 관리자가 메시지를 사용 가능한 큐 관리자로 라우트하기 위해 호출합니다.

[MQWXP](#) - 클러스터 워크로드 엑시트 매개변수 구조

다음 표에는 MQWXP(클러스터 워크로드 엑시트 매개변수 구조)의 필드가 요약되어 설명되어 있습니다.

[MQWDR](#) - 클러스터 워크로드 목적지 레코드 구조

다음 표에는 MQWDR(클러스터 워크로드 목적지 레코드 구조)의 필드가 요약되어 설명되어 있습니다.

[MQWQR](#) - 클러스터 워크로드 큐 레코드 구조

다음 표에는 MQWQR(클러스터 워크로드 큐 레코드 구조)의 필드가 요약되어 설명되어 있습니다.

## MQWCR - 클러스터 워크로드 클러스터 레코드 구조

다음 표에는 MQWCR 클러스터 워크로드 레코드 구조의 필드가 요약되어 설명되어 있습니다.

## MQXCLWLN의 매개변수 - 클러스터 워크로드 레코드 탐색

MQXCLWLN 호출에서 매개변수의 설명입니다.

### ExitParms (MQWXP) - 입출력(I/O)

엑시트 매개변수 블록.

이 구조는 엑시트 호출과 관련된 정보를 포함합니다. 엑시트는 워크로드를 관리하는 방법을 표시하기 위해 이 구조에서 정보를 설정합니다.

### CurrentRecord (MQPTR) - 입력

현재 레코드의 주소.

이 구조는 현재 엑시트에 의해 검사되는 레코드의 주소와 관련된 정보가 포함되어 있습니다. 레코드는 다음 유형 중 하나여야 합니다.

- 클러스터 워크로드 목적지 레코드(MQWDR)
- 클러스터 워크로드 큐 레코드(MQWQR)
- 클러스터 워크로드 클러스터 레코드(MQWCR)

### NextOffset (MQLONG) - 입력

다음 레코드의 오프셋.

이 구조에는 다음 레코드 또는 구조의 오프셋과 관련된 정보가 포함되어 있습니다. *NextOffset*은(는) 현재 레코드에 있는 적절한 오프셋 필드의 값이며 다음 필드 중 하나여야 합니다.

- MQWDR의 ChannelDefOffset 필드
- MQWDR의 ClusterRecOffset 필드
- MQWQR의 ClusterRecOffset 필드
- MQWCR의 ClusterRecOffset 필드

### NextRecord (MQPTR) - 출력

다음 레코드 또는 구조의 주소.

이 구조에는 다음 레코드 또는 구조의 주소와 관련된 정보가 포함되어 있습니다. *CurrentRecord*이(가) MQWDR의 주소이고 *NextOffset*이(가) ChannelDefOffset 필드의 값이면, *NextRecord*은(는) 채널 정의 구조(MQCD)의 주소입니다.

다음 레코드 또는 구조가 없는 경우, 큐 관리자는 *NextRecord*을(를) 널 포인터로 설정하고 호출은 완료 코드 MQCC\_WARNING 및 이유 코드 MQRC\_NO\_RECORD\_AVAILABLE을(를) 리턴합니다.

### CompCode(MQLONG) - 출력

완료 코드.

완료 코드의 값은 다음 값 중 하나입니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_WARNING

경고(일부 완료).

#### MQCC\_FAILED

호출에 실패했습니다.

### Reason (MQLONG) - 출력

CompCode를 규정하는 이유 코드

CompCode가 MQCC\_OK인 경우:

**MQRC\_NONE****(0, X'0000')**

보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_WARNING인 경우:

**MQRC\_NO\_RECORD\_AVAILABLE****(2359, X'0937')**

사용 가능한 레코드가 없습니다. 체인의 다음 레코드의 주소를 확보하기 위해 MQXCLWLN 호출이 클러스터 워크로드 엑시트에서 발행되었습니다. 현재 레코드는 체인의 마지막 레코드입니다. 정정 조치: 없음.

*CompCode*이(가) MQCC\_FAILED인 경우:

**MQRC\_CURRENT\_RECORD\_ERROR****(2357, X'0935')**

**CurrentRecord** 매개변수가 유효하지 않음. 체인의 다음 레코드의 주소를 확보하기 위해 MQXCLWLN 호출이 클러스터 워크로드 엑시트에서 발행되었습니다. **CurrentRecord** 매개변수에 지정된 주소가 유효한 레코드의 주소가 아닙니다.

**CurrentRecord**은(는) 목적지 레코드, MQWDR, 큐 레코드(MQWQR) 또는 클러스터 캐시 내에 상주하는 클러스터 레코드(MQWCR)의 주소여야 합니다. 정정 조치: 클러스터 워크로드 엑시트가 클러스터 캐시에 상주하는 올바른 레코드의 주소를 전달하는지 확인하십시오.

**MQRC\_ENVIRONMENT\_ERROR****(2012, X'07DC')**

호출이 환경에서 올바르지 않습니다. MQXCLWLN 호출이 실행되었지만, 클러스터 워크로드 엑시트의 호출이 아닙니다.

**MQRC\_NEXT\_OFFSET\_ERROR****(2358, X'0936')**

**NextOffset** 매개변수가 유효하지 않음. 체인의 다음 레코드의 주소를 확보하기 위해 MQXCLWLN 호출이 클러스터 워크로드 엑시트에서 발행되었습니다. **NextOffset** 매개변수로 지정된 오프셋이 유효하지 않습니다. **NextOffset**은(는) 다음 필드 중 하나의 값이어야 합니다.

- MQWDR의 ChannelDefOffset 필드
- MQWDR의 ClusterRecOffset 필드
- MQWQR의 ClusterRecOffset 필드
- MQWCR의 ClusterRecOffset 필드

정정 조치: **NextOffset** 매개변수에 지정된 값이 이전에 나열된 필드 중 하나의 값인지 확인하십시오.

**MQRC\_NEXT\_RECORD\_ERROR****(2361, X'0939')**

**NextRecord** 매개변수가 유효하지 않음.

**MQRC\_WXP\_ERROR****(2356, X'0934')**

워크로드 엑시트 매개변수 구조가 올바르지 않습니다. 체인의 다음 레코드의 주소를 확보하기 위해 MQXCLWLN 호출이 클러스터 워크로드 엑시트에서 발행되었습니다. 다음 이유 중 하나로 인해 워크로드 엑시트 매개변수 구조 **ExitParms**이(가) 유효하지 않습니다.

- 매개변수 포인터가 올바르지 않습니다. 항상 올바르지 않은 매개변수 포인터를 발견할 수 있는 것은 아닙니다. 발견되지 않으면 예측할 수 없는 결과가 발생합니다.
- StrucId 필드가 MQWXP\_STRUC\_ID이(가) 아닙니다.
- Version 필드가 MQWXP\_VERSION\_2이(가) 아닙니다.
- Context 필드에 큐 관리자가 엑시트에 전달한 값이 포함되지 않습니다.

정정 조치: **ExitParms**에 지정된 매개변수가 엑시트가 호출될 때 엑시트에 전달된 MQWXP 구조인지 확인하십시오.

## 관련 참조

[MQXCLWLN의 사용법 참고 - 클러스터 워크로드 레코드 탐색](#)

캐시가 정적인 경우에도 MQXCLWLN을 사용하여 클러스터 레코드를 탐색합니다.

[MQXCLWLN의 언어 호출](#)

MQXCLWLN에서는 두 개의 언어(C 및 상위 레벨 어셈블러)를 지원합니다.

## MQXCLWLN의 사용법 참고 - 클러스터 워크로드 레코드 탐색

캐시가 정적인 경우에도 MQXCLWLN을 사용하여 클러스터 레코드를 탐색합니다.

클러스터 캐시가 동적인 경우에는 MQXCLWLN 호출을 사용하여 레코드를 탐색해야 합니다. 단순 포인터 및 오프셋 산술을 사용하여 레코드를 탐색하는 경우, 엑시트가 비정상적으로 종료됩니다.

클러스터 캐시가 정적인 경우에는 MQXCLWLN을 사용하여 레코드를 탐색할 필요가 없습니다. 일반적으로는 캐시가 정적인 경우에도 MQXCLWLN을 사용합니다. 그리고 워크로드 엑시트를 변경하지 않고도 클러스터 캐시를 동적으로 변경할 수 있습니다.

## 관련 참조

[MQXCLWLN의 매개변수 - 클러스터 워크로드 레코드 탐색](#)

MQXCLWLN 호출에서 매개변수의 설명입니다.

[MQXCLWLN의 언어 호출](#)

MQXCLWLN에서는 두 개의 언어(C 및 상위 레벨 어셈블러)를 지원합니다.

## MQXCLWLN의 언어 호출

MQXCLWLN에서는 두 개의 언어(C 및 상위 레벨 어셈블러)를 지원합니다.

## C 호출

```
MQXCLWLN (&ExitParms, CurrentRecord, NextOffset, &NextRecord, &CompCode, &Reason) ;
```

매개변수를 다음과 같이 선언합니다.

```
Typedef struct tagMQXCLWLN {
MQWXP ExitParms; /* Exit parameter block */
MQPTR CurrentRecord; /* Address of current record*/
MQLONG NextOffset; /* Offset of next record */
MQPTR NextRecord; /* Address of next record or structure */
MQLONG CompCode; /* Completion code */
MQLONG Reason; /* Reason code qualifying CompCode */
}
```

## 상위 레벨 어셈블러 호출

```
CALL MQXCLWLN, (CLWLEXITPARMS, CURRENTRECORD, NEXTOFFSET, NEXTRECORD, COMPCODE, REASON)
```

매개변수를 다음과 같이 선언합니다.

```
CLWLEXITPARMS CMQWXP, Cluster workload exit parameter block
CURRENTRECORD CMQWDRA, Current record
NEXTOFFSET DS F Next offset
NEXTRECORD DS F Next record
COMPCODE DS F Completion code
REASON DS F Reason code qualifying COMPCODE
```

## 관련 참조

[MQXCLWLN의 매개변수 - 클러스터 워크로드 레코드 탐색](#)

MQXCLWLN 호출에서 매개변수의 설명입니다.

[MQXCLWLN의 사용법 참고 - 클러스터 워크로드 레코드 탐색](#)

캐시가 정적인 경우에도 MQXCLWLN을 사용하여 클러스터 레코드를 탐색합니다.



## MQWXP - 클러스터 워크로드 엑시트 매개변수 구조

다음 표에는 MQWXP(클러스터 워크로드 엑시트 매개변수 구조)의 필드가 요약되어 설명되어 있습니다.

표 826. MQWXP의 필드		
필드	설명	페이지
<i>StrucId</i>	구조 ID	<a href="#">StrucId</a>
<i>Version</i>	구조 버전 번호	<a href="#">Version</a>
<i>ExitId</i>	엑시트 유형	<a href="#">ExitId</a>
<i>ExitReason</i>	엑시트 호출의 이유	<a href="#">ExitReason</a>
<i>ExitResponse</i>	엑시트로부터의 응답	<a href="#">ExitResponse</a>
<i>ExitResponse2</i>	엑시트로부터의 2차 응답	<a href="#">ExitResponse2</a>
<i>Feedback</i>	피드백 코드	<a href="#">피드백</a>
<i>Flags</i>	플래그 값. 이 비트 플래그는 넣기가 실행되는 메시지에 대한 정보를 표시하기 위해 사용됩니다.	<a href="#">Flags</a>
<i>ExitUserArea</i>	엑시트 사용자 영역	<a href="#">ExitUserArea</a>
<i>ExitData</i>	엑시트 데이터	<a href="#">ExitData</a>
<i>MsgDescPtr</i>	메시지 디스크립터(MQMD)의 주소	<a href="#">MsgDescPtr</a>
<i>MsgBufferPtr</i>	일부 또는 모든 메시지 데이터를 포함하는 버퍼의 주소	<a href="#">MsgBufferPtr</a>
<i>MsgBufferLength</i>	메시지 데이터가 포함된 버퍼의 길이	<a href="#">MsgBufferLength</a>
<i>MsgLength</i>	전체 메시지의 길이	<a href="#">MsgLength</a>
<i>QName</i>	큐 이름	<a href="#">QName</a>
<i>QMgrName</i>	로컬 큐 관리자의 이름	<a href="#">QMgrName</a>
<i>DestinationCount</i>	가능한 목적지의 수	<a href="#">DestinationCount</a>
<i>DestinationChosen</i>	선택된 목적지	<a href="#">DestinationChosen</a>
<i>DestinationArrayPtr</i>	목적지 레코드(MQWDR)에 대한 포인터의 배열의 주소	<a href="#">DestinationArrayPtr</a>
<i>QArrayPtr</i>	큐 레코드(MQWQR)에 대한 포인터의 배열의 주소	<a href="#">QArrayPtr</a>
<b>참고:</b> Version이 MQWXP_VERSION_2 미만인 경우 나머지 필드는 무시됩니다.		
<i>CacheContext</i>	컨텍스트 정보	<a href="#">CacheContext</a>
<i>CacheType</i>	클러스터 캐시의 유형	<a href="#">CacheType</a>
<b>참고:</b> Version이 MQWXP_VERSION_3 미만인 경우 나머지 필드는 무시됩니다.		
<i>CLWLMRUChannels</i>	허용된 활성 아웃바운드 클러스터 채널의 최대 수	<a href="#">CLWLMRUChannels</a>
<b>참고:</b> Version이 MQWXP_VERSION_4 미만인 경우 나머지 필드는 무시됩니다.		
<i>pEntryPoints</i>	MQI 및 DCI 호출의 작성을 허용하기 위한 MQIEP 구조의 주소	<a href="#">pEntryPoints</a>

클러스터 워크로드 엑시트 매개변수 구조는 클러스터 워크로드 엑시트에 전달되는 정보를 설명합니다.

클러스터 워크로드 엑시트 매개변수 구조는 모든 플랫폼에서 지원됩니다.

또한 MQWXP1, MQWXP2 및 MQWXP3 구조는 역호환성을 위해 사용 가능합니다.

## 관련 참조

### MQ\_CLUSTER\_WORKLOAD\_EXIT - 호출 설명

클러스터 워크로드 엑시트는 큐 관리자가 메시지를 사용 가능한 큐 관리자로 라우트하기 위해 호출합니다.

### MQXCLWLN - 클러스터 워크로드 레코드 탐색

MQXCLWLN 호출은 클러스터 캐시에 저장된 MQWDR, MQWQR 및 MQWCR 레코드의 체인을 통해 탐색하는 데 사용됩니다.

### MQWDR - 클러스터 워크로드 목적지 레코드 구조

다음 표에는 MQWDR(클러스터 워크로드 목적지 레코드 구조)의 필드가 요약되어 설명되어 있습니다.

### MQWQR - 클러스터 워크로드 큐 레코드 구조

다음 표에는 MQWQR(클러스터 워크로드 큐 레코드 구조)의 필드가 요약되어 설명되어 있습니다.

### MQWCR - 클러스터 워크로드 클러스터 레코드 구조

다음 표에는 MQWCR 클러스터 워크로드 레코드 구조의 필드가 요약되어 설명되어 있습니다.

## **MQWXP의 필드 - 클러스터 워크로드 엑시트 매개변수 구조**

MQWXP(클러스터 워크로드 엑시트 매개변수 구조)의 필드 설명

### **StrucId (MQCHAR4) - 입력**

클러스터 워크로드 엑시트 매개변수 구조의 구조 ID.

- StrucId 값은 MQWXP\_STRUC\_ID입니다.
- C 프로그래밍 언어의 경우 상수 MQWXP\_STRUC\_ID\_ARRAY도 정의됩니다. 이 값은 MQWXP\_STRUC\_ID과 (와) 동일합니다. 이는 문자열 대신 문자의 배열입니다.

### **Version (MQLONG) - 입력**

구조 버전 번호를 표시합니다. 버전은 다음 값 중 하나를 갖습니다.

#### **MQWXP\_VERSION\_1**

버전-1 클러스터 워크로드 엑시트 매개변수 구조.

MQWXP\_VERSION\_1은(는) 모든 환경에서 지원됩니다.

#### **MQWXP\_VERSION\_2**

버전-2 클러스터 워크로드 엑시트 매개변수 구조.

MQWXP\_VERSION\_2은(는) 다음 환경에서 지원됩니다.

-  AIX
-  IBM i
-  Linux
-  Windows

#### **MQWXP\_VERSION\_3**

버전-3 클러스터 워크로드 엑시트 매개변수 구조.

MQWXP\_VERSION\_3은(는) 다음 환경에서 지원됩니다.

-  AIX
-  IBM i
-  Linux
-  Windows

#### **MQWXP\_VERSION\_4**

버전-4 클러스터 워크로드 엑시트 매개변수 구조.

MQWXP\_VERSION\_4은(는) 다음 환경에서 지원됩니다.

- **AIX** AIX
- **IBM i** IBM i
- **Linux** Linux
- **Windows** Windows

**MQWXP\_CURRENT\_VERSION**

클러스터 워크로드 엑시트 매개변수 구조의 현재 버전.

**ExitId (MQLONG) - 입력**

호출되는 엑시트의 유형을 표시합니다. 클러스터 워크로드 엑시트는 유일하게 지원되는 엑시트입니다.

- ExitId 값은 MQXT\_CLUSTER\_WORKLOAD\_EXIT여야 합니다.

**ExitReason (MQLONG) - 입력**

클러스터 워크로드 엑시트를 호출하는 이유를 표시합니다. ExitReason은 다음 값 중 하나를 취합니다.

**MQXR\_INIT**

엑시트를 처음 호출 중임을 표시합니다.

엑시트에서 필요할 수 있는 자원(예: 주 기억장치)을 가져와서 초기화하십시오.

**MQXR\_TERM**

엑시트가 종료될 예정임을 표시합니다.

초기화된 이후 엑시트가 가져왔을 수 있는 자원(예: 주 기억장치)을 해제하십시오.

**MQXR\_CLWL\_OPEN**

MQOPEN에 의해 호출됩니다.

**MQXR\_CLWL\_PUT**

MQPUT 또는 MQPUT1에 의해 호출됩니다.

**MQXR\_CLWL\_MOVE**

채널 상태가 변경된 경우 MCA에 의해 호출됩니다.

**MQXR\_CLWL\_REPOS**

저장소 관리자 PCF 메시지에 대해 MQPUT 또는 MQPUT1에 의해 호출됩니다.

**MQXR\_CLWL\_REPOS\_MOVE**

채널 상태가 변경된 경우 저장소 관리자 PCF 메시지에 대해 MCA에 의해 호출됩니다.

**ExitResponse (MQLONG) - 출력**

ExitResponse를 설정하여 메시지 처리가 계속되는지 여부를 표시합니다. 다음 값 중 하나여야 합니다.

**MQXCC\_OK**

메시지 처리를 정상적으로 계속합니다.

- DestinationChosen은 메시지를 송신해야 할 목적지를 식별합니다.

**MQXCC\_SUPPRESS\_FUNCTION**

메시지 처리를 중단합니다.

- 큐 관리자에서 취하는 조치는 엑시트가 호출된 이유에 따라 다릅니다.

표 827. 큐 관리자에서 취하는 조치	
ExitReason	수행 조치
<ul style="list-style-type: none"> <li>- MQXR_CLWL_OPEN</li> <li>- MQXR_CLWL_REPOS</li> <li>- MQXR_CLWL_PUT</li> </ul>	MQOPEN, MQPUT 또는 MQPUT1 호출이 완료 코드 MQCC_FAILED 및 이유 코드 MQRC_STOPPED_BY_CLUSTER_EXIT로 실패합니다.
<ul style="list-style-type: none"> <li>- MQXR_CLWL_MOVE</li> <li>- MQXR_CLWL_REPOS_MOVE</li> </ul>	메시지가 데드-레터 큐에 놓입니다.

## **MQXCC\_SUPPRESS\_EXIT**

현재 메시지 처리를 정상적으로 계속합니다. 큐 관리자가 종료될 때까지 엑시트를 다시 호출하지 마십시오.

ClusterWorkloadExit 큐 관리자 속성이 공백인 것처럼 큐 관리자가 후속 메시지를 처리합니다. DestinationChosen은 현재 메시지가 송신되는 목적지를 식별합니다.

### **기타 값**

MQXCC\_SUPPRESS\_FUNCTION이(가) 지정된 것처럼 메시지를 처리합니다.

## **ExitResponse2 ( MQLONG ) - 입출력(I/O)**

ExitResponse2를 설정하여 큐 관리자에 자세한 정보를 제공할 수 있습니다.

- MQXR2\_STATIC\_CACHE은(는) 기본값이며 엑시트에 들어갈 때 설정됩니다.
- ExitReason 의 값이 MQXR\_INIT인 경우, 엑시트는 ExitResponse2에 다음 값 중 하나를 설정할 수 있습니다.

### **MQXR2\_STATIC\_CACHE**

엑시트에서 정적 클러스터 캐시가 필요합니다.

- 클러스터 캐시가 정적인 경우, 엑시트가 MQXCLWLN 호출을 사용하여 클러스터 캐시에서 레코드 체인을 탐색할 필요가 없습니다.
- 클러스터 캐시가 동적인 경우, 엑시트는 캐시의 레코드를 정확하게 탐색할 수 없습니다.

**참고:** 큐 관리자는 엑시트가 ExitResponse 필드에 MQXCC\_SUPPRESS\_EXIT을(를) 리턴한 것처럼 MQXR\_INIT 호출에서 리턴을 처리합니다.

### **MQXR2\_DYNAMIC\_CACHE**

엑시트가 정적 또는 동적 캐시에서 작동될 수 있습니다.

- 엑시트가 이 값을 리턴하면, 엑시트는 MQXCLWLN 호출을 사용하여 클러스터 캐시에서 레코드 체인을 탐색해야 합니다.

## **Feedback ( MQLONG ) - 입력**

예약된 필드. 값은 0입니다.

## **Flags ( MQLONG ) - 입력**

넣는 메시지에 대한 정보를 표시합니다.

- Flags의 값은 MQWXP\_PUT\_BY\_CLUSTER\_CHL입니다. 메시지는 로컬이 아닌 클러스터 채널에서 또는 비클러스터 채널에서 시작됩니다. 즉, 메시지를 다른 클러스터 큐 관리자에서 가져왔습니다.

## **Reserved ( MQLONG ) - 입력**

예약된 필드. 값은 0입니다.

## **ExitUserArea ( MQBYTE16 ) - 입출력(I/O)**

ExitUserArea를 설정하여 엑시트에 대한 호출 간에 통신할 수 있습니다.

- ExitUserArea는 엑시트의 최초 호출 전에 2진수 0으로 초기화됩니다. 엑시트가 이 필드에서 수행한 모든 변경사항은 MQCONN 호출과 이와 일치되는 MQDISC 호출 사이에서 발생한 엑시트의 호출에서 유지됩니다. MQDISC 호출이 발생하면 필드가 2진수 0으로 재설정됩니다.
- 엑시트의 첫 번째 호출은 MQXR\_INIT 값이 있는 ExitReason 필드로 표시됩니다.
- 다음 상수가 정의됩니다.

### **MQXUA\_NONE - 문자열**

### **MQXUA\_NONE\_ARRAY - 문자 배열**

사용자 정보가 없습니다. 두 상수 모두 필드 길이에 대해 2진수 0입니다.

### **MQ\_EXIT\_USER\_AREA\_LENGTH**

ExitUserArea의 길이입니다.

## **ExitData ( MQCHAR32 ) - 입력**

ClusterWorkloadData 큐 관리자 속성 값. 해당 속성에 대해 정의된 값이 없는 경우 이 필드가 모두 공백입니다.

- ExitData의 길이는 MQ\_EXIT\_DATA\_LENGTH에 의해 제공됩니다.

#### **MsgDescPtr ( PMQMD ) - 입력**

처리되는 메시지에 대한 메시지 디스크립터(MQMD)의 사본의 주소.

- 엑시트에서 수행된 메시지 디스크립터에 대한 모든 변경사항은 큐 관리자에 의해 무시됩니다.
- ExitReason이 다음 값 중 하나를 갖는 경우, MsgDescPtr이 널 포인터로 설정되며 메시지 디스크립터가 엑시트로 전달되지 않습니다.
  - MQXR\_INIT
  - MQXR\_TERM
  - MQXR\_CLWL\_OPEN

#### **MsgBufferPtr ( PMQVOID ) - 입력**

메시지 데이터의 첫 번째 MsgBufferLength 바이트 사본을 포함하는 버퍼의 주소.

- 엑시트에서 수행한 메시지 데이터에 대한 모든 변경사항은 큐 관리자에 의해 무시됩니다.
  - 다음의 경우 메시지 데이터가 엑시트에 전달되지 않습니다.
    - MsgDescPtr이 널 포인터입니다.
    - 메시지에 데이터가 없습니다.
    - ClusterWorkloadLength 큐 관리자 속성이 0입니다.
- 이러한 경우에 MsgBufferPtr은 널 포인터입니다.

#### **MsgBufferLength ( MQLONG ) - 입력**

엑시트로 전달된 메시지 데이터를 포함하는 버퍼의 길이.

- 길이는 ClusterWorkloadLength 큐 관리자 속성에 의해 제어됩니다.
- 길이는 전체 메시지의 길이 미만일 수 있습니다(MsgLength 참조).

#### **MsgLength ( MQLONG ) - 입력**

엑시트에 전달된 전체 메시지의 길이.

- MsgBufferLength는 전체 메시지의 길이보다 짧을 수 있습니다.
- ExitReason이 MQXR\_INIT, MQXR\_TERM 또는 MQXR\_CLWL\_OPEN이면 MsgLength는 0입니다.

#### **QName ( MQCHAR48 ) - 입력**

목적지 큐의 이름. 큐는 클러스터 큐입니다.

- QName의 길이는 MQ\_Q\_NAME\_LENGTH입니다.

#### **QMgrName ( MQCHAR48 ) - 입력**

클러스터 워크로드 엑시트를 호출한 로컬 큐 관리자의 이름.

- QMgrName의 길이는 MQ\_Q\_MGR\_NAME\_LENGTH입니다.

#### **DestinationCount ( MQLONG ) - 입력**

가능한 목적지의 수. 목적지는 목적지 큐의 인스턴스이며, 목적지 레코드에서 설명합니다.

- 목적지 레코드는 MQWDR 구조입니다. 큐의 각 인스턴스에 대해 가능한 개별 라우트 구조는 하나입니다.
- MQWDR 구조는 포인터 배열에 주소 지정되어 있습니다(DestinationArrayPtr 참조).

#### **DestinationChosen ( MQLONG ) - 입출력(I/O)**

선택된 목적지.

- 메시지가 송신되는 라우트 및 큐 인스턴스를 식별하는 MQWDR 구조의 번호.
- 값은 1 - DestinationCount 범위입니다.
- 엑시트에 대한 입력에서 DestinationChosen이 큐 관리자가 선택한 라우트 및 큐 인스턴스를 표시합니다. 엑시트는 이 선택사항을 채택하거나 다른 라우트 및 큐 인스턴스를 선택할 수 있습니다.

- 엑시트에서 설정한 값은 1 - DestinationCount의 범위여야 합니다. 다른 값이 리턴되면, 큐 관리자는 엑시트에 대한 입력에서 DestinationChosen 값을 사용합니다.

**DestinationArrayPtr ( PPMQWDR ) - 입력**

목적지 레코드(MQWDR)에 대한 포인터의 배열의 주소.

- DestinationCount개의 목적지 레코드가 있습니다.

**QArrayPtr ( PPMQWQR ) - 입력**

큐 레코드(MQWQR)에 대한 포인터의 배열의 주소.

- 큐 레코드가 사용 가능한 경우 이 중에서 DestinationCount개가 있습니다.
- 큐 레코드가 사용 불가능한 경우 QArrayPtr은 널 포인터입니다.

참고: DestinationCount가 0보다 큰 경우에도 QArrayPtr은 널 포인터일 수 있습니다.

**CacheContext ( MQPTR ) : 버전 2 - 입력**

CacheContext 필드는 큐 관리자에서 사용하기 위해 예약됩니다. 엑시트는 이 필드 값을 대체하지 않아야 합니다.

**CacheType ( MQLONG ) : 버전 2 - 입력**

클러스터 캐시에는 다음 유형 중 하나가 있습니다.

**MQCLCT\_STATIC**

캐시는 정적입니다.

- 캐시 크기는 고정되어 있으며, 큐 관리자의 작동에 따라 증가될 수 없습니다.
- 이 캐시 유형의 레코드를 탐색하기 위해 MQXCLWLN 호출을 사용할 필요는 없습니다.

**MQCLCT\_DYNAMIC**

캐시는 동적입니다.

- 변화되는 클러스터 정보를 수용하기 위해 캐시의 크기가 증가될 수 있습니다.
- 이 캐시 유형의 레코드를 탐색하려면 MQXCLWLN 호출을 사용해야 합니다.

**CLWLMRUChannels ( MQLONG ) : 버전 3 - 입력**

클러스터 워크로드 선택 알고리즘이 사용하기 위해 고려해야 할 최대 활성 아웃바운드 클러스터 채널 수를 표시합니다.

- CLWLMRUChannels의 값은 1 - 999 999 999입니다.

**pEntryPoints ( PMQIEP ) : 버전 4**

MQI 및 DCI 호출을 작성할 수 있는 MQIEP 구조의 주소입니다.

**관련 참조**

MQWXP의 초기값 및 언어 선언

MQWXP(클러스터 워크로드 엑시트 매개변수 구조)의 C 및 상위 레벨 어셈블러 언어 선언의 초기값.

**MQWXP의 초기값 및 언어 선언**

MQWXP(클러스터 워크로드 엑시트 매개변수 구조)의 C 및 상위 레벨 어셈블러 언어 선언의 초기값.

표 828. MQWXP의 필드		
필드 이름	상수의 이름	상수의 값
StrucId	MQWXP_STRUC_ID	'WXP_'
Version	MQWXP_VERSION_2	2
ExitId	없음	0
ExitReason	MQXCC_OK	0
ExitResponse	없음	0

표 828. MQWXP의 필드 (계속)

필드 이름	상수의 이름	상수의 값
<i>ExitResponse2</i>	없음	0
<i>Flags</i>	없음	0
<i>ExitUserArea</i>	{MQXUA_NONE_ARRAY}	0
<i>ExitData</i>	없음	" "
<i>MsgDescPtr</i>	없음	NULL
<i>MsgBufferPtr</i>	없음	NULL
<i>MsgBufferLength</i>	없음	0
<i>MsgBufferPtr</i>	없음	0
<i>QName</i>	없음	" "
<i>QMgrName</i>	없음	" "
<i>DestinationCount</i>	없음	0
<i>DestinationChosen</i>	없음	0
<i>DestinationArrayPtr</i>	없음	NULL
<i>QArrayPtr</i>	없음	NULL
<i>CacheContext</i>	없음	NULL
<i>CacheType</i>	MQCLCT_DYNAMIC	1
<i>CLWLMRUChannels</i>	없음	0
<i>pEntryPoints</i>	없음	NULL

**참고:**

- ~ 기호는 단일 공백 문자를 나타냅니다.
- C 프로그래밍 언어에서는 매크로 변수 MQWXP\_DEFAULT에 기본값이 포함됩니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오.

```
MQWDR MyWXP = {MQWXP_DEFAULT};
```

## C 선언

```
typedef struct tagMQWXP {
    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;    /* Reserved */
    MQLONG     Feedback;        /* Reserved */
    MQLONG     Flags;            /* Flags */
    MQBYTE16   ExitUserArea;     /* Exit user area */
    MQCHAR32   ExitData;         /* Exit data */
    PMQMD      MsgDescPtr;       /* Address of message descriptor */
    PMQVOID    MsgBufferPtr;     /* Address of buffer containing some
    or all of the message data */
    MQLONG     MsgBufferLength;  /* Length of buffer containing message
    data */
    MQLONG     MsgLength;        /* Length of complete message */
};
```

```

MQCHAR48 QName; /* Queue name */
MQCHAR48 QMgrName; /* Name of local queue manager */
MQLONG DestinationCount; /* Number of possible destinations */
MQLONG DestinationChosen; /* Destination chosen */
PPMQWDR DestinationArrayPtr; /* Address of an array of pointers to
destination records */

PPMQWQR QArrayPtr; /* Address of an array of pointers to
queue records */

/* version 1 */
MQPTR CacheContext; /* Context information */
MQLONG CacheType; /* Type of cluster cache */
/* version 2 */
MQLONG CLWLMRUChannels; /* Maximum number of most recently
used cluster channels */

/* version 3 */
PMQIEP pEntryPoints; /* Address of the MQIEP structure */
/* version 4 */
};

```

## High Level Assembler

```

MQWXP          DSECT
MQWXP_STRUCID  DS    CL4      Structure identifier
MQWXP_VERSION  DS    F        Structure version number
MQWXP_EXITID   DS    F        Type of exit
MQWXP_EXITREASON DS    F      Reason for invoking exit
MQWXP_EXITRESPONSE DS    F    Response from exit
MQWXP_EXITRESPONSE2 DS    F    Reserved
MQWXP_FEEDBACK DS    F        Reserved
MQWXP_RESERVED DS    F        Reserved
MQWXP_EXITUSERAREA DS    XL16  Exit user area
MQWXP_EXITDATA  DS    CL32    Exit data
MQWXP_MSGDESCPTR DS    F      Address of message
* descriptor
MQWXP_MSGBUFFERPTR DS    F    Address of buffer containing
* some or all of the message
* data
MQWXP_MSGBUFFERLENGTH DS    F  Length of buffer containing
* message data
MQWXP_MSGLENGTH DS    F      Length of complete message
MQWXP_QNAME     DS    CL48    Queue name
MQWXP_QMGRNAME DS    CL48    Name of local queue manager
MQWXP_DESTINATIONCOUNT DS    F  Number of possible
* destinations
MQWXP_DESTINATIONCHOSEN DS    F  Destination chosen
MQWXP_DESTINATIONARRAYPTR DS    F  Address of an array of
* pointers to destination
* records
MQWXP_QARRAYPTR DS    F      Address of an array of
* pointers to queue records
MQWXP_CACHECONTEXT DS    F    Context information
MQWXP_CACHETYPE   DS    F      Type of cluster cache
MQWXP_CLWLMRUCHANNELS DS    F  Number of most recently used
* channels for workload balancing

MQWXP_LENGTH    EQU    *-MQWXP  Length of structure
                ORG    MQWXP
MQWXP_AREA      DS    CL(MQWXP_LENGTH)

```

### 관련 참조

[MQWXP의 필드 - 클러스터 워크로드 엑시트 매개변수 구조](#)  
[MQWXP\(클러스터 워크로드 엑시트 매개변수 구조\)의 필드 설명](#)

## MQWDR - 클러스터 워크로드 목적지 레코드 구조

다음 표에는 MQWDR(클러스터 워크로드 목적지 레코드 구조)의 필드가 요약되어 설명되어 있습니다.

표 829. MQWDR의 필드		
필드	설명	페이지
<i>StrucId</i>	구조 ID	<a href="#">StrucId</a>



표 829. MQWDR의 필드 (계속)

필드	설명	페이지
<i>Version</i>	구조 버전 번호	<a href="#">Version</a>
<i>StrucLength</i>	MQWDR 구조의 길이	<a href="#">StrucLength</a>
<i>QMGrFlags</i>	큐 관리자 플래그	<a href="#">QMGrFlags</a>
<i>QMGrIdentifier</i>	큐 관리자 ID	<a href="#">QMGrIdentifier</a>
<i>QMGrName</i>	큐 관리자 이름	<a href="#">QMGrName</a>
<i>ClusterRecOffset</i>	첫 번째 클러스터 레코드(MQWCR)의 논리 오프셋	<a href="#">ClusterRecOffset</a>
<i>ChannelState</i>	채널 상태	<a href="#">ChannelState</a>
<i>ChannelDefOffset</i>	채널 정의 구조(MQCD)의 논리 오프셋	<a href="#">ChannelDefOffset</a>
참고: Version이 MQWDR_VERSION_2 미만인 경우 나머지 필드는 무시됩니다.		
<i>DestSeqNumber</i>	채널 목적지 순서 번호	<a href="#">DestSeqNumber</a>
<i>DestSeqFactor</i>	가중치에 대한 채널 목적지 시퀀스 요인	<a href="#">DestSeqFactor</a>

클러스터 워크로드 목적지 레코드 구조에는 메시지에 대한 가능한 목적지 중 하나와 관련된 정보가 포함되어 있습니다. 목적지 큐의 각 인스턴스마다 하나의 클러스터 워크로드 목적지 레코드 구조가 있습니다.

클러스터 워크로드 목적지 레코드 구조는 모든 환경에서 지원됩니다.

또는 MQWDR1 및 MQWDR2 구조는 역호환성을 위해 사용 가능합니다.

#### 관련 참조

##### [MQ\\_CLUSTER\\_WORKLOAD\\_EXIT - 호출 설명](#)

클러스터 워크로드 엑시트는 큐 관리자가 메시지를 사용 가능한 큐 관리자로 라우트하기 위해 호출합니다.

##### [MQXCLWLN - 클러스터 워크로드 레코드 탐색](#)

MQXCLWLN 호출은 클러스터 캐시에 저장된 MQWDR, MQWQR 및 MQWCR 레코드의 체인을 통해 탐색하는 데 사용됩니다.

##### [MQWXP - 클러스터 워크로드 엑시트 매개변수 구조](#)

다음 표에는 MQWXP(클러스터 워크로드 엑시트 매개변수 구조)의 필드가 요약되어 설명되어 있습니다.

##### [MQWQR - 클러스터 워크로드 큐 레코드 구조](#)

다음 표에는 MQWQR(클러스터 워크로드 큐 레코드 구조)의 필드가 요약되어 설명되어 있습니다.

##### [MQWCR - 클러스터 워크로드 클러스터 레코드 구조](#)

다음 표에는 MQWCR 클러스터 워크로드 레코드 구조의 필드가 요약되어 설명되어 있습니다.

### MQWDR의 필드 - 클러스터 워크로드 목적지 레코드 구조

MQWDR(클러스터 워크로드 목적지 레코드 구조)에서 매개변수의 설명입니다.

#### StrucId (MQCHAR4) - 입력

클러스터 워크로드 목적지 레코드 구조의 구조 ID.

- StrucId 값은 MQWDR\_STRUC\_ID입니다.
- C 프로그래밍 언어의 경우 상수 MQWDR\_STRUC\_ID\_ARRAY도 정의됩니다. 이 값은 MQWDR\_STRUC\_ID과 (와) 동일합니다. 이는 문자열 대신 문자의 배열입니다.

#### Version (MQLONG) - 입력

구조 버전 번호. 버전은 다음 값 중 하나를 갖습니다.

##### MQWDR\_VERSION\_1

버전-1 클러스터 워크로드 목적지 레코드.

**MQWDR\_VERSION\_2**

버전-2 클러스터 워크로드 목적지 레코드.

**MQWDR\_CURRENT\_VERSION**

클러스터 워크로드 목적지 레코드의 현재 버전.

**StrucLength (MQLONG) - 입력**

MQWDR 구조의 길이입니다. StrucLength는 다음 값 중 하나를 갖습니다.

**MQWDR\_LENGTH\_1**

버전-1 클러스터 워크로드 목적지 레코드의 길이.

**MQWDR\_LENGTH\_2**

버전-2 클러스터 워크로드 목적지 레코드의 길이.

**MQWDR\_CURRENT\_LENGTH**

클러스터 워크로드 목적지 레코드의 현재 버전의 길이.

**QMGrFlags (MQLONG) - 입력**

MQWDR 구조에서 설명하는 목적지 큐의 인스턴스를 호스팅하는 큐 관리자의 특성을 표시하는 큐 관리자 플래그. 다음의 플래그가 정의됩니다.

**MQQMF\_REPOSITORY\_Q\_MGR**

목적지는 전체 저장소 큐 관리자입니다.

**MQQMF\_CLUSSDR\_USER\_DEFINED**

클러스터-송신자 채널이 수동으로 정의되었습니다.

**MQQMF\_CLUSSDR\_AUTO\_DEFINED**

클러스터-송신자 채널이 자동으로 정의되었습니다.

**MQQMF\_AVAILABLE**

목적지 큐 관리자가 메시지 수신에 사용 가능합니다.

**기타 값**

필드의 기타 플래그는 내부 용도로 큐 관리자에 의해 설정될 수 있습니다.

**QMGrIdentifier (MQCHAR48) - 입력**

큐 관리자 ID는 MQWDR 구조에서 설명하는 목적지 큐의 인스턴스를 호스팅하는 큐 관리자의 고유 ID입니다.

- ID는 큐 관리자에 의해 생성됩니다.
- QMGrIdentifier의 길이는 MQ\_Q\_MGR\_IDENTIFIER\_LENGTH입니다.

**QMGrName (MQCHAR48) - 입력**

MQWDR 구조에서 설명하는 목적지 큐의 인스턴스를 호스팅하는 큐 관리자의 이름.

- QMGrName은 로컬 큐 관리자는 물론 클러스터의 다른 큐 관리자의 이름일 수 있습니다.
- QMGrName의 길이는 MQ\_Q\_MGR\_NAME\_LENGTH입니다.

**ClusterRecOffset (MQLONG) - 입력**

MQWDR 구조에 속하는 첫 번째 MQWCR 구조의 논리 오프셋.

- 정적 캐시의 경우, ClusterRecOffset은 MQWDR 구조에 속하는 첫 번째 MQWCR 구조의 오프셋입니다.
- 오프셋은 MQWDR 구조의 시작에서 바이트 단위로 측정됩니다.
- 동적 캐시가 있는 포인터 산술에 논리 오프셋을 사용하지 마십시오. 다음 레코드의 주소를 확보하려면, MQXCLWLN 호출을 사용해야 합니다.

**ChannelState (MQLONG) - 입력**

MQWDR 구조에서 식별하는 큐 관리자에 로컬 큐 관리자를 링크하는 채널의 상태. 가능한 값은 다음과 같습니다.

**MQCHS\_BINDING**

파트너와 채널을 협상 중입니다.

**MQCHS\_INACTIVE**

채널이 활성 상태가 아닙니다.

**MQCHS\_INITIALIZING**

채널이 초기화 중입니다.

**MQCHS\_PAUSED**

채널이 일시정지되었습니다.

**MQCHS\_REQUESTING**

요청자 채널이 연결을 요청하고 있습니다.

**MQCHS\_RETRYING**

채널이 연결 설정을 재시도 중입니다.

**MQCHS\_RUNNING**

채널이 메시지를 전송 중이거나 대기 중입니다.

**MQCHS\_STARTING**

채널이 활성화 대기 중입니다.

**MQCHS\_STOPPING**

채널이 중지 중입니다.

**MQCHS\_STOPPED**

채널이 중지되었습니다.

**ChannelDefOffset (MQLONG) - 입력**

MQWDR 구조에서 식별하는 큐 관리자에 로컬 큐 관리자를 링크하는 채널에 대한 채널 정의(MQCD)의 논리 오프셋.

- ChannelDefOffset은 ClusterRecOffset과 유사함
- 논리 오프셋은 포인터 산술에서 사용될 수 없습니다. 다음 레코드의 주소를 확보하려면, MQXCLWLN 호출을 사용해야 합니다.

**DestSeqFactor (MQLONG) - 입력**

가중치를 기반으로 채널의 선택을 허용하는 목적지 시퀀스 요인.

- DestSeqFactor는 큐 관리자가 이를 변경하기 전에 사용됩니다.
- 워크로드 관리자는 메시지가 가중치에 따라 채널에 분배되도록 보장하는 방식으로 DestSeqFactor를 늘립니다.

**DestSeqNumber (MQLONG) - 입력**

큐 관리자가 변경하기 전의 클러스터 채널 목적지 값.

- 워크로드 관리자는 메시지가 해당 채널에 놓일 때마다 DestSeqNumber를 늘립니다.
- 워크로드 엑시트는 DestSeqNumber를 사용하여 메시지가 놓일 채널을 결정할 수 있습니다.

**관련 참조**

MQWDR의 초기값 및 언어 선언

MQWDR(클러스터 워크로드 목적지 레코드)의 C 및 상위 레벨 어셈블러 언어 선언의 초기값.

**MQWDR의 초기값 및 언어 선언**

MQWDR(클러스터 워크로드 목적지 레코드)의 C 및 상위 레벨 어셈블러 언어 선언의 초기값.

표 830. MQWDR의 필드		
필드 이름	상수의 이름	상수의 값
<i>StrucId</i>	MQWDR_STRUC_ID	'WDR~'
<i>Version</i>	MQWDR_VERSION_1	1
<i>StrucLength</i>	MQWDR_CURRENT_LENGTH <sup>3</sup>	136
<i>QMgrFlags</i>	MQWDR_NONE	0
<i>QMgrIdentifier</i>	없음	""

표 830. MQWDR의 필드 (계속)

필드 이름	상수의 이름	상수의 값
<i>QMgrName</i>	없음	""
<i>ClusterRecOffset</i>	없음	0
<i>ChannelState</i>	없음	0
<i>ChannelDefOffset</i>	없음	0
<i>DestSeqNumber</i>	없음	0
<i>DestSeqFactor</i>	없음	0

**참고:**

1. ~ 기호는 단일 공백 문자를 나타냅니다.
2. C 프로그래밍 언어에서는 매크로 변수 MQWDR\_DEFAULT에 기본값이 포함됩니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오.

```
MQWDR MyWDR = {MQWDR_DEFAULT};
```

3. 초기값은 구조의 길이를 의도적으로 버전 1의 구조가 아닌 현재 버전의 길이로 설정합니다.

### High Level Assembler

```
MQWDR          DSECT
MQWDR_STRUCID DS CL4      Structure identifier
MQWDR_VERSION DS F        Structure version number
MQWDR_STRUCLNGTH DS F      Length of MQWDR structure
MQWDR_QMGRFLAGS DS F      Queue manager flags
MQWDR_QMGRIDENTIFIER DS CL48 Queue manager identifier
MQWDR_QMGRNAME DS CL48   Queue manager name
MQWDR_CLUSTERRECOFFSET DS F Offset of first cluster
*              record
MQWDR_CHANNELSTATE DS F    Channel state
MQWDR_CHANNELDEFOFFSET DS F Offset of channel definition
*              structure
MQWDR_LENGTH EQU *-MQWDR Length of structure
MQWDR_AREA     DS CL(MQWDR_LENGTH)
```

### C 선언

```
typedef struct tagMQWDR {
    MQCHAR4  StrucId;          /* Structure identifier */
    MQLONG   Version;         /* Structure version number */
    MQLONG   StrucLength;     /* Length of MQWDR structure */
    MQLONG   QMgrFlags;       /* Queue manager flags */
    MQCHAR48 QMgrIdentifier;  /* Queue manager identifier */
    MQCHAR48 QMgrName;        /* Queue manager name */
    MQLONG   ClusterRecOffset; /* Offset of first cluster record */
    MQLONG   ChannelState;    /* Channel state */
    MQLONG   ChannelDefOffset; /* Offset of channel definition structure */
    /* Ver:1 */
    MQLONG   DestSeqNumber;    /* Cluster channel destination sequence number */
    MQINT64  DestSeqFactor;    /* Cluster channel factor sequence number */
    /* Ver:2 */
};
```

**관련 참조**

[MQWDR의 필드 - 클러스터 워크로드 목적지 레코드 구조](#)

MQWDR(클러스터 워크로드 목적지 레코드 구조)에서 매개변수의 설명입니다.

## MQWQR - 클러스터 워크로드 큐 레코드 구조

다음 표에는 MQWQR(클러스터 워크로드 큐 레코드 구조)의 필드가 요약되어 설명되어 있습니다.

표 831. MQWQR의 필드		
필드	설명	페이지
<i>StrucId</i>	구조 ID	<a href="#">StrucId</a>
<i>Version</i>	구조 버전 번호	<a href="#">Version</a>
<i>StrucLength</i>	MQWQR 구조의 길이	<a href="#">StrucLength</a>
<i>QFlags</i>	큐 플래그	<a href="#">QFlags</a>
<i>QName</i>	큐 이름	<a href="#">QName</a>
<i>QMgrIdentifier</i>	큐 관리자 ID	<a href="#">QMgrIdentifier</a>
<i>ClusterRecOffset</i>	첫 번째 클러스터 레코드(MQWCR)의 오프셋	<a href="#">ClusterRecOffset</a>
<i>QType</i>	큐 유형	<a href="#">QType</a>
<i>QDesc</i>	큐 설명	<a href="#">QDesc</a>
<i>DefBind</i>	기본 바인딩	<a href="#">DefBind</a>
<i>DefPersistence</i>	기본 메시지 지속성	<a href="#">DefPersistence</a>
<i>DefPriority</i>	기본 메시지 우선순위	<a href="#">DefPriority</a>
<i>InhibitPut</i>	큐에서 넣기 조작이 허용되는지 여부	<a href="#">InhibitPut</a>
참고: <i>Version</i> 이 MQWQR_VERSION_2 미만인 경우 나머지 필드는 무시됩니다.		
<i>CWLQueuePriority</i>	큐의 우선순위를 표시하는 0 - 9 값	<a href="#">CWLQueuePriority</a>
<i>CLWLQueueRank</i>	큐의 순위를 표시하는 0 - 9 값	<a href="#">CLWLQueueRank</a>
참고: <i>Version</i> 이 MQWQR_VERSION_3 미만인 경우 나머지 필드는 무시됩니다.		
<i>DefPutResponse</i>	기본 넣기 응답	<a href="#">DefPutResponse</a>

클러스터 워크로드 큐 레코드 구조에는 메시지에 대한 가능한 목적지 중 하나와 관련된 정보가 포함되어 있습니다. 목적지 큐의 각 인스턴스마다 하나의 클러스터 워크로드 큐 레코드 구조가 있습니다.

클러스터 워크로드 큐 레코드 구조는 모든 환경에서 지원됩니다.

또한 MQWQR1 및 MQWQR2 구조는 역호환성을 위해 사용 가능합니다.

### 관련 참조

#### [MQ CLUSTER WORKLOAD EXIT - 호출 설명](#)

클러스터 워크로드 엑시트는 큐 관리자가 메시지를 사용 가능한 큐 관리자로 라우트하기 위해 호출합니다.

#### [MQXCLWLN - 클러스터 워크로드 레코드 탐색](#)

MQXCLWLN 호출은 클러스터 캐시에 저장된 MQWDR, MQWQR 및 MQWCR 레코드의 체인을 통해 탐색하는 데 사용됩니다.

#### [MQWXP - 클러스터 워크로드 엑시트 매개변수 구조](#)

다음 표에는 MQWXP(클러스터 워크로드 엑시트 매개변수 구조)의 필드가 요약되어 설명되어 있습니다.

#### [MQWDR - 클러스터 워크로드 목적지 레코드 구조](#)

다음 표에는 MQWDR(클러스터 워크로드 목적지 레코드 구조)의 필드가 요약되어 설명되어 있습니다.

#### [MQWCR - 클러스터 워크로드 클러스터 레코드 구조](#)

다음 표에는 MQWCR 클러스터 워크로드 레코드 구조의 필드가 요약되어 설명되어 있습니다.

## **MQWQR의 필드 - 클러스터 워크로드 큐 레코드 구조**

MQWQR(클러스터 워크로드 큐 레코드 구조)의 필드의 설명입니다.

### **StrucId (MQCHAR4) - 입력**

클러스터 워크로드 큐 레코드 구조의 구조 ID.

- StrucId 값은 MQWQR\_STRUC\_ID입니다.
- C 프로그래밍 언어의 경우 상수 MQWQR\_STRUC\_ID\_ARRAY도 정의됩니다. 이 값은 MQWQR\_STRUC\_ID과 (와) 동일합니다. 이는 문자열 대신 문자의 배열입니다.

### **Version (MQLONG) - 입력**

구조 버전 번호. 버전은 다음 값 중 하나를 갖습니다.

#### **MQWQR\_VERSION\_1**

버전-1 클러스터 워크로드 큐 레코드.

#### **MQWQR\_VERSION\_2**

버전-2 클러스터 워크로드 큐 레코드.

#### **MQWQR\_VERSION\_3**

버전-3 클러스터 워크로드 큐 레코드.

#### **MQWQR\_CURRENT\_VERSION**

클러스터 워크로드 큐 레코드의 현재 버전.

### **StrucLength (MQLONG) - 입력**

MQWQR 구조의 길이. StrucLength는 다음 값 중 하나를 갖습니다.

#### **MQWQR\_LENGTH\_1**

버전-1 클러스터 워크로드 큐 레코드의 길이.

#### **MQWQR\_LENGTH\_2**

버전-2 클러스터 워크로드 큐 레코드의 길이.

#### **MQWQR\_LENGTH\_3**

버전-3 클러스터 워크로드 큐 레코드의 길이.

#### **MQWQR\_CURRENT\_LENGTH**

클러스터 워크로드 큐 레코드의 현재 버전의 길이.

### **QFlags (MQLONG) - 입력**

큐 플래그는 큐의 특성을 표시합니다. 다음의 플래그가 정의됩니다.

#### **MQQF\_LOCAL\_Q**

목적지는 로컬 큐입니다.

#### **MQQF\_CLWL\_USEQ\_ANY**

넣기에서 로컬 및 리모트 큐의 사용을 허용합니다.

#### **MQQF\_CLWL\_USEQ\_LOCAL**

로컬 큐 넣기만 허용합니다.

#### **기타 값**

필드의 기타 플래그는 내부 용도로 큐 관리자에 의해 설정될 수 있습니다.

### **QName (MQCHAR48) - 입력**

메시지의 가능한 목적지 중 하나인 큐의 이름.

- QName의 길이는 MQ\_Q\_NAME\_LENGTH입니다.

### **QMgrIdentifier (MQCHAR48) - 입력**

큐 관리자는 MQWQR 구조에서 설명하는 큐의 인스턴스를 호스팅하는 큐 관리자의 고유 ID입니다.

- ID는 큐 관리자에 의해 생성됩니다.
- QMgrIdentifier의 길이는 MQ\_Q\_MGR\_IDENTIFIER\_LENGTH입니다.

### **ClusterRecOffset (MQLONG) - 입력**

MQWQR 구조에 속하는 첫 번째 MQWCR 구조의 논리 오프셋.

- 정적 캐시의 경우, ClusterRecOffset은 MQWQR 구조에 속하는 첫 번째 MQWCR 구조의 오프셋입니다.
- 오프셋은 MQWQR 구조의 시작에서 바이트 단위로 측정됩니다.
- 동적 캐시가 있는 포인터 산술에 논리 오프셋을 사용하지 마십시오. 다음 레코드의 주소를 확보하려면, MQXCLWLN 호출을 사용해야 합니다.

#### **QType (MQLONG) - 입력**

목적지 큐의 큐 유형. 가능한 값은 다음과 같습니다.

##### **MQCQT\_LOCAL\_Q**

로컬 큐.

##### **MQCQT\_ALIAS\_Q**

알리어스 큐입니다.

##### **MQCQT\_REMOTE\_Q**

리모트 큐.

##### **MQCQT\_Q\_MGR\_ALIAS**

큐 관리자 알리어스.

#### **QDesc (MQCHAR64) - 입력**

MQWQR 구조에서 설명하는 목적지 큐의 인스턴스를 호스팅하는 큐 관리자에 정의된 큐 설명 큐 속성.

- QDesc의 길이는 MQ\_Q\_DESC\_LENGTH입니다.

#### **DefBind (MQLONG) - 입력**

MQWQR 구조에서 설명하는 목적지 큐의 인스턴스를 호스팅하는 큐 관리자에 정의된 기본 바인딩 큐 속성. 어느 하나 MQBND\_BIND\_ON\_OPEN 또는 MQBND\_BIND\_ON\_GROUP 클러스터가 있는 그룹을 사용할 때 지정해야 합니다. 다음 값이 가능합니다.

##### **MQBND\_BIND\_ON\_OPEN**

MQOPEN 호출에 의해 고정된 바인딩.

##### **MQBND\_BIND\_NOT\_FIXED**

바인딩이 고정되지 않습니다.

##### **MQBND\_BIND\_ON\_GROUP**

애플리케이션을 통해 메시지 그룹이 모두 동일한 목적지 인스턴스에 할당되도록 요청할 수 있습니다.

#### **DefPersistence (MQLONG) - 입력**

MQWQR 구조에서 설명하는 목적지 큐의 인스턴스를 호스팅하는 큐 관리자에 정의된 기본 메시지 지속성 큐 속성. 가능한 값은 다음과 같습니다.

##### **MQPER\_PERSISTENT**

메시지가 지속됩니다.

##### **MQPER\_NOT\_PERSISTENT**

메시지가 지속되지 않습니다.

#### **DefPriority (MQLONG) - 입력**

MQWQR 구조에서 설명하는 목적지 큐의 인스턴스를 호스팅하는 큐 관리자에 정의된 기본 메시지 우선순위 큐 속성. 우선순위 범위는 0 - MaxPriority입니다.

- 0은 최하위 우선순위입니다.
- MaxPriority는 목적지 큐의 이 인스턴스를 호스팅하는 큐 관리자의 큐 관리자 속성입니다.

#### **InhibitPut (MQLONG) - 입력**

MQWQR 구조에서 설명하는 목적지 큐의 인스턴스를 호스팅하는 큐 관리자에 정의된 넣기 금지된 큐 속성. 가능한 값은 다음과 같습니다.

##### **MQQA\_PUT\_INHIBITED**

Put 조작이 금지됩니다.

##### **MQQA\_PUT\_ALLOWED**

Put 조작이 허용됩니다.

**CLWLQueuePriority (MQLONG) - 입력**

MQWQR 구조에서 설명하는 목적지 큐의 인스턴스를 호스팅하는 큐 관리자에 정의된 클러스터 워크로드 큐 우선순위 속성.

**CLWLQueueRank (MQLONG) - 입력**

MQWQR 구조에서 설명하는 목적지 큐의 인스턴스를 호스팅하는 큐 관리자에 정의된 클러스터 워크로드 큐 순위.

**DefPutResponse (MQLONG) - 입력**

MQWQR 구조에서 설명하는 목적지 큐의 인스턴스를 호스팅하는 큐 관리자에 정의된 기본 넣기 응답 큐 속성. 가능한 값은 다음과 같습니다.

**MQPRT\_SYNC\_RESPONSE**

MQPUT 또는 MQPUT1 호출에 대한 동기적 응답.

**MQPRT\_ASYNC\_RESPONSE**

MQPUT 또는 MQPUT1 호출에 대한 비동기 응답.

**관련 참조**

MQWQR의 초기값 및 언어 선언

MQWQR(클러스터 워크로드 큐 레코드)의 C 및 상위 레벨 어셈블러 언어 선언의 초기값.

**MQWQR의 초기값 및 언어 선언**

MQWQR(클러스터 워크로드 큐 레코드)의 C 및 상위 레벨 어셈블러 언어 선언의 초기값.

표 832. MQWQR의 필드		
필드 이름	상수의 이름	상수의 값
<i>StrucId</i>	MQWQR_STRUC_ID_ARRAY	'WQR'
<i>Version</i>	MQWQR_VERSION_1	1
<i>StrucLength</i>	MQWQR_CURRENT_LENGTH <sup>3</sup>	212
<i>QFlags</i>	없음	0
<i>QName</i>	없음	""
<i>QMgrIdentifier</i>	없음	""
<i>ClusterRecOffset</i>	없음	0
<i>QType</i>	없음	0
<i>QDesc</i>	없음	""
<i>DefBind</i>	없음	0
<i>DefPersistence</i>	없음	0
<i>DefPriority</i>	없음	0
<i>InhibitPut</i>	없음	0
<i>CLWLQueuePriority</i>	없음	0
<i>CLWLQueueRank</i>	없음	0
<i>DefPutResponse</i>	없음	1



표 832. MQWQR의 필드 (계속)

필드 이름	상수의 이름	상수의 값
<b>참고:</b>		
1. ~ 기호는 단일 공백 문자를 나타냅니다.		
2. C 프로그래밍 언어에서는 매크로 변수 MQWQR_DEFAULT에 기본값이 포함됩니다. 구조의 필드에 초기값을 제공하기 위해 다음 방법으로 사용하십시오.		
<pre>MQWQR MyWQR = {MQWQR_DEFAULT};</pre>		
3. 초기값은 구조의 길이를 의도적으로 버전 1의 구조가 아닌 현재 버전의 길이로 설정합니다.		

## C 선언

```
typedef struct tagMQWQR {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    StrucLength;      /* Length of MQWQR structure */
    MQLONG    QFlags;          /* Queue flags */
    MQCHAR48  QName;           /* Queue name */
    MQCHAR48  QMgrIdentifier;   /* Queue manager identifier */
    MQLONG    ClusterRecOffset; /* Offset of first cluster record */
    MQLONG    QType;           /* Queue type */
    MQCHAR64  QDesc;           /* Queue description */
    MQLONG    DefBind;         /* Default binding */
    MQLONG    DefPersistence;  /* Default message persistence */
    MQLONG    DefPriority;     /* Default message priority */
    MQLONG    InhibitPut;     /* Whether put operations on the queue
                               are allowed */

    /* version 2 */
    MQLONG    CLWLQueuePriority; /* Queue priority */
    MQLONG    CLWLQueueRank;    /* Queue rank */
    /* version 3 */
    MQLONG    DefPutResponse;   /* Default put response */
};
```

## High Level Assembler

MQWQR	DSECT		
MQWQR_STRUCID	DS	CL4	Structure identifier
MQWQR_VERSION	DS	F	Structure version number
MQWQR_STRUCLength	DS	F	Length of MQWQR structure
MQWQR_QFLAGS	DS	F	Queue flags
MQWQR_QNAME	DS	CL48	Queue name
MQWQR_QMGRIDENTIFIER	DS	CL48	Queue manager identifier
MQWQR_CLUSTERRECOFFSET	DS	F	Offset of first cluster record
*			
MQWQR_QTYPE	DS	F	Queue type
MQWQR_QDESC	DS	CL64	Queue description
MQWQR_DEFBIND	DS	F	Default binding
MQWQR_DEFPERSISTENCE	DS	F	Default message persistence
MQWQR_DEFPRIOITY	DS	F	Default message priority
MQWQR_INHIBITPUT	DS	F	Whether put operations on the queue are allowed
*			
MQWQR_DEFPUTRESPONSE	DS	F	Default put response
MQWQR_LENGTH	EQU	*-MQWQR	Length of structure
	ORG	MQWQR	
MQWQR_AREA	DS	CL(MQWQR_LENGTH)	

## 관련 참조

[MQWQR의 필드 - 클러스터 워크로드 큐 레코드 구조](#)

MQWQR(클러스터 워크로드 큐 레코드 구조)의 필드의 설명입니다.

## MQWCR - 클러스터 워크로드 클러스터 레코드 구조

다음 표에는 MQWCR 클러스터 워크로드 레코드 구조의 필드가 요약되어 설명되어 있습니다.

표 833. MQWCR의 필드		
필드	설명	페이지
<i>ClusterName</i>	클러스터의 이름	<a href="#">ClusterName</a>
<i>ClusterRecOffset</i>	다음 클러스터 레코드(MQWCR)의 오프셋	<a href="#">ClusterRecOffset</a>
<i>ClusterFlags</i>	클러스터 플래그	<a href="#">ClusterFlags</a>

클러스터 워크로드 클러스터 레코드 구조에는 클러스터에 대한 정보가 포함되어 있습니다. 목적지 큐가 속하는 각 클러스터마다, 하나의 클러스터 워크로드 클러스터 레코드 구조가 있습니다.

클러스터 워크로드 클러스터 레코드 구조는 모든 환경에서 지원됩니다.

### 관련 참조

#### [MQ CLUSTER WORKLOAD EXIT - 호출 설명](#)

클러스터 워크로드 엑시트는 큐 관리자가 메시지를 사용 가능한 큐 관리자로 라우트하기 위해 호출합니다.

#### [MQXCLWLN - 클러스터 워크로드 레코드 탐색](#)

MQXCLWLN 호출은 클러스터 캐시에 저장된 MQWDR, MQWQR 및 MQWCR 레코드의 체인을 통해 탐색하는 데 사용됩니다.

#### [MQWXP - 클러스터 워크로드 엑시트 매개변수 구조](#)

다음 표에는 MQWXP(클러스터 워크로드 엑시트 매개변수 구조)의 필드가 요약되어 설명되어 있습니다.

#### [MQWDR - 클러스터 워크로드 목적지 레코드 구조](#)

다음 표에는 MQWDR(클러스터 워크로드 목적지 레코드 구조)의 필드가 요약되어 설명되어 있습니다.

#### [MQWQR - 클러스터 워크로드 큐 레코드 구조](#)

다음 표에는 MQWQR(클러스터 워크로드 큐 레코드 구조)의 필드가 요약되어 설명되어 있습니다.

### MQWCR의 필드 - 클러스터 워크로드 클러스터 레코드 구조.

MQWCR(클러스터 워크로드 클러스터 레코드 구조)의 필드의 설명입니다.

#### **ClusterName (MQCHAR48) - 입력**

MQWCR 구조를 소유하는 목적지 큐의 인스턴스가 속하는 클러스터의 이름입니다. 목적지 큐 인스턴스는 MQWDR 구조에 의해 설명됩니다.

- ClusterName의 길이는 MQ\_CLUSTER\_NAME\_LENGTH입니다.

#### **ClusterRecOffset (MQLONG) - 입력**

다음 MQWCR 구조의 논리 오프셋.

- 추가 MQWCR 구조가 없으면 ClusterRecOffset이 0입니다.
- 오프셋은 MQWCR 구조의 시작에서 바이트 단위로 측정됩니다.

#### **ClusterFlags (MQLONG) - 입력**

클러스터 플래그는 MQWCR 구조에서 식별하는 큐 관리자의 특성을 표시합니다. 다음의 플래그가 정의됩니다.

##### **MQQMF\_REPOSITORY\_Q\_MGR**

목적지는 전체 저장소 큐 관리자입니다.

##### **MQQMF\_CLUSSDR\_USER\_DEFINED**

클러스터-송신자 채널이 수동으로 정의되었습니다.

##### **MQQMF\_CLUSSDR\_AUTO\_DEFINED**

클러스터-송신자 채널이 자동으로 정의되었습니다.

##### **MQQMF\_AVAILABLE**

목적지 큐 관리자가 메시지 수신에 사용 가능합니다.

## 기타 값

필드의 기타 플래그는 내부 용도로 큐 관리자에 의해 설정될 수 있습니다.

## 관련 참조

[MQWCR의 초기값 및 언어 선언](#)

MQWCR(클러스터 워크로드 클러스터 레코드 구조)의 C 및 상위 레벨 어셈블러 언어 선언의 초기값.

## MQWCR의 초기값 및 언어 선언

MQWCR(클러스터 워크로드 클러스터 레코드 구조)의 C 및 상위 레벨 어셈블러 언어 선언의 초기값.

표 834. MQWCR의 필드		
필드 이름	상수의 이름	상수의 값
<i>ClusterName</i>	없음	" "
<i>ClusterRecOffset</i>	없음	0
<i>ClusterFlags</i>	없음	0

## C 선언

```
typedef struct tagMQWCR {
    MQCHAR48 ClusterName; /* Cluster name */
    MQLONG ClusterRecOffset; /* Offset of next cluster record */
    MQLONG ClusterFlags; /* Cluster flags */
};
```

## High Level Assembler

```
MQWCR DSECT
MQWCR_CLUSTERNAME DS CL48 Cluster name
MQWCR_CLUSTERRECOFFSET DS F Offset of next cluster
* record
MQWCR_CLUSTERFLAGS DS F Cluster flags
MQWCR_LENGTH EQU *-MQWCR Length of structure
ORG MQWCR
MQWCR_AREA DS CL(MQWCR_LENGTH)
```

## 관련 참조

[MQWCR의 필드 - 클러스터 워크로드 클러스터 레코드 구조.](#)

MQWCR(클러스터 워크로드 클러스터 레코드 구조)의 필드의 설명입니다.

## API 엑시트 참조

이 절에서는 주로 API 엑시트를 작성하는 프로그래머에게 도움이 되는 참조 정보를 제공합니다.

## 일반 사용 참고사항

### 참고:

1. 모든 엑시트 함수는 MQXEP 호출을 발행할 수 있습니다. 이 호출은 특히 API 엑시트 함수에서 사용하기 위한 것입니다.
2. MQ\_INIT\_EXIT 함수는 MQXEP 외의 다른 MQ 호출은 발행할 수 없습니다.
3. 현재 연결에 대해 MQDISC 호출을 발행할 수 없습니다.
4. 엑시트 함수가 MQCNO\_HANDLE\_SHARE\_NONE 옵션으로 MQCONN 호출 또는 MQCONNX 호출을 발행한 경우, 호출은 MQRC\_ALREADY\_CONNECTED 이유 코드와 함께 완료되며 리턴된 핸들은 매개변수로 엑시트에 전달된 핸들과 동일합니다.
5. 일반적으로 API 엑시트 함수가 MQI 호출을 발행하는 경우 API 엑시트가 반복적으로 호출되지 않습니다. 하지만 엑시트 함수가 MQCNO\_HANDLE\_SHARE\_BLOCK 또는 MQCNO\_HANDLE\_SHARE\_NO\_BLOCK 옵션

으로 MQCONNX 호출을 발행하는 경우, 호출은 새 공유 핸들을 리턴합니다. 이 경우 엑시트 스위트에 고유 연결 핸들을 제공하므로, 애플리케이션의 작업 단위와 관계 없는 작업 단위가 제공됩니다. 엑시트 스위트는 이 핸들을 사용하여 고유 작업 단위 내에서 메시지를 넣고 가져올 수 있으며 해당 작업 단위를 커밋하거나 백아웃할 수 있으므로, 어떤 식으로든 애플리케이션의 작업 단위에 영향을 미치지 않고 이를 모두 수행할 수 있습니다.

엑시트 함수는 애플리케이션에서 사용 중인 핸들과는 다른 연결 핸들을 사용하고 있기 때문에 엑시트 함수에 의해 발행된 MQ 호출의 결과로 관련 API 엑시트 함수가 호출됩니다. 따라서 엑시트 함수를 반복적으로 호출할 수 있습니다. MQAXP의 *ExitUserArea* 필드와 엑시트 체인 영역에 모두 연결 핸들 범위가 있다는 점에 유의하십시오. 결과적으로 엑시트 함수는 이러한 영역을 사용하여 이미 활성 상태이고 반복적으로 호출되는 엑시트 함수의 다른 인스턴스로 신호를 보낼 수 없습니다.

- 또한 엑시트 함수는 애플리케이션의 작업 단위 내에서 메시지를 넣고 가져올 수 있습니다. 애플리케이션이 작업 단위를 커밋하거나 백아웃하면 작업 단위(애플리케이션 또는 엑시트 함수)에서 메시지를 넣는 사용자와 상관없이 작업 단위 내의 모든 메시지도 함께 커밋되거나 백아웃됩니다. 그러나 엑시트로 인해 애플리케이션이 평소보다 빨리 시스템 한계를 초과할 수 있습니다(예: 작업 단위의 커밋되지 않은 메시지의 최대 수를 초과함).

엑시트 함수는 이 방법으로 애플리케이션의 작업 단위를 사용하는 경우 애플리케이션의 작업 단위를 커밋하여 애플리케이션의 올바른 작동을 방해할 수 있으므로 일반적으로 MQCMIT 호출을 발행하지 않도록 해야 합니다. 하지만 엑시트 함수는 작업 단위의 커밋을 방지하는 심각한 오류를 발견한 경우(예: 애플리케이션 작업 단위의 일부로 메시지를 넣는 중 오류 발생) MQBACK 호출을 발행해야 할 때도 있습니다. MQBACK이 호출되는 경우, 애플리케이션 작업 단위 경계가 변경되지 않도록 주의하십시오. 이 상황에서는 작업 단위가 백아웃된 사실을 애플리케이션이 감지할 수 있도록 하기 위해 엑시트 함수를 적절한 값으로 설정하여 완료 코드 MQCC\_WARNING 및 이유 코드 MQRC\_BACKED\_OUT이 애플리케이션으로 리턴되는지 확인해야 합니다.

엑시트 함수가 애플리케이션의 연결 핸들을 사용하여 MQ 호출을 발행하는 경우, 해당 호출의 결과로 API 엑시트 함수가 추가로 호출되지는 않습니다.

- MQXR\_BEFORE 엑시트 함수가 비정상적으로 종료된 경우, 큐 관리자가 장애를 복구할 수 있습니다. 큐 관리자가 장애를 복구하는 경우, 큐 관리자는 엑시트 함수가 MQXCC\_FAILED를 리턴한 것처럼 처리를 계속합니다. 큐 관리자가 복구할 수 없는 경우에는 애플리케이션이 종료됩니다.
- MQXR\_AFTER 엑시트 함수가 비정상적으로 종료된 경우, 큐 관리자가 장애를 복구할 수 있습니다. 큐 관리자가 장애를 복구하는 경우, 큐 관리자는 엑시트 함수가 MQXCC\_FAILED를 리턴한 것처럼 처리를 계속합니다. 큐 관리자가 복구할 수 없는 경우에는 애플리케이션이 종료됩니다. 후자의 경우, 작업 단위 외부에서 검색된 메시지는 유실된다는 사실을 알아야 합니다. 이는 큐에서 메시지를 제거한 직후 애플리케이션이 실패하는 것과 동일한 상황입니다.
- MCA 프로세스는 두 단계 커밋을 수행합니다.

API 엑시트가 준비된 MCA 프로세스에서 MQCMIT를 인터셉트하고 작업 단위 내에서 조치를 수행하려고 하면, 이유 코드 MQRC\_UOW\_NOT\_AVAILABLE이 표시되면서 조치가 실패합니다.

- IBM MQ 다중 설치가 있는 경우, 이후 버전에 추가된 새 기능이 이전 버전에서 작동하지 않을 수 있으므로 이전 버전의 IBM MQ에 대해 작성된 엑시트를 사용하십시오. 릴리스 간의 변경사항에 대한 자세한 정보는 [IBM MQ 8.0의 변경사항](#)을 참조하십시오.

## IBM MQ API 엑시트 매개변수 구조(MQAXP)

외부 제어 블록인 MQAXP 구조는 API 엑시트에 대한 입력 또는 출력 매개변수로 사용됩니다. 이 주제에서는 큐 관리자가 엑시트 함수를 처리하는 방법에 대한 정보도 제공합니다.

MQAXP는 다음 C 선언을 포함합니다.

```
typedef struct tagMQAXP {
    MQCHAR4   StructId;           /* Structure identifier */
    MQLONG    Version;           /* Structure version number */
    MQLONG    ExitId;           /* Exit Identifier */
    MQLONG    ExitReason;       /* Exit invocation reason */
    MQLONG    ExitResponse;     /* Response code from exit */
    MQLONG    ExitResponse2;   /* Secondary response code from exit */
    MQLONG    Feedback;        /* Feedback code from exit */
    MQLONG    APICallerType;    /* MQSeries API caller type */
    MQBYTE16  ExitUserArea;    /* User area for use by exit */
}
```

```

MQCHAR32 ExitData;          /* Exit data area */
MQCHAR48 ExitInfoName;     /* Exit information name */
MQBYTE48 ExitPDArea;      /* Problem determination area */
MQCHAR48 QMgrName;        /* Name of local queue manager */
PMQACH ExitChainAreaPtr;  /* Inter exit communication area */
MQHCONFIG Hconfig;        /* Configuration handle */
MQLONG Function;          /* Function Identifier */
/* Ver:1 */
MQHMSG ExitMsgHandle      /* Exit message handle
/* Ver:2 */
};

```

API 엑시트의 함수를 호출하면 다음 매개변수 목록이 전달됩니다.

#### StrucId (MQCHAR4) - 입력

엑시트 매개변수 구조 ID이며, 값은 다음과 같습니다.

```
MQAXP_STRUC_ID.
```

엑시트 핸들러가 입력 항목에서 이 필드를 각 엑시트 함수로 설정합니다.

#### Version (MQLONG) - 입력

구조 버전 번호이며, 값은 다음과 같습니다.

##### MQAXP\_VERSION\_1

버전 1 API 엑시트 매개변수 구조.

##### MQAXP\_VERSION\_2

버전 2 API 엑시트 매개변수 구조.

##### MQAXP\_CURRENT\_VERSION

API 엑시트 매개변수 구조에 대한 현재 버전 번호.

엑시트 핸들러가 입력 항목에서 이 필드를 각 엑시트 함수로 설정합니다.

#### ExitId (MQLONG) - 입력

엑시트 유형을 나타내는 입력 ID이며, 엑시트 루틴에 대한 입력에 설정됩니다.

##### MQXT\_API\_EXIT

API 엑시트.

#### ExitReason (MQLONG) - 입력

엑시트를 호출하는 이유로, 각 엑시트 함수에 대한 입력에 설정됩니다.

##### MQXR\_CONNECTION

엑시트가 MQCONN 또는 MQCONNX 호출 이전에 자체 초기화하거나 MQDISC 호출 이후 자체 종료하기 위해 호출됩니다.

##### MQXR\_BEFORE

엑시트가 API 호출을 실행하기 전에 또는 MQGET에서 데이터를 변환하기 전에 호출됩니다.

##### MQXR\_AFTER

엑시트가 API 호출을 실행한 후에 호출됩니다.

#### ExitResponse (MQLONG) - 출력

엑시트의 응답이며, 각 엑시트 함수에 대한 입력에서 다음으로 초기화됩니다.

##### MQXCC\_OK

정상적으로 계속됩니다.

이 필드는 엑시트 함수 실행 결과를 큐 관리자에게 전달하도록 엑시트 함수를 통해 설정되어야 합니다. 값은 다음 중 하나여야 합니다.

##### MQXCC\_OK

엑시트 함수가 성공적으로 완료되었습니다. 정상적으로 계속됩니다.

이 값은 모든 MQXR\_\* 엑시트 함수로 설정할 수 있습니다. 체인의 이후 엑시트 함수를 호출할지 결정하는 데 ExitResponse2가 사용됩니다.

##### MQXCC\_FAILED

오류로 인해 엑시트 함수가 실패했습니다.

이 값은 모든 MQXR\_\* 엑시트 함수로 설정할 수 있습니다. 큐 관리자가 CompCode를 MQCC\_FAILED로, Reason을 다음으로 설정합니다.

- 함수가 MQ\_INIT\_EXIT인 경우 MQRC\_API\_EXIT\_INIT\_ERROR
- 함수가 MQ\_TERM\_EXIT인 경우 MQRC\_API\_EXIT\_TERM\_ERROR
- 그 외 모든 엑시트 함수의 경우 MQRC\_API\_EXIT\_ERROR

설정된 값은 체인의 이후 엑시트 함수를 통해 대체할 수 있습니다.

ExitResponse2는 무시됩니다. 큐 관리자는 MQXR2\_SUPPRESS\_CHAIN이 리턴된 것처럼 처리를 계속합니다.

### **MQXCC\_SUPPRESS\_FUNCTION**

IBM MQ API 함수를 차단합니다.

이 값은 MQXR\_BEFORE 엑시트 함수로만 설정할 수 있습니다. API 호출은 무시합니다. MQ\_DATA\_CONV\_ON\_GET\_EXIT에 의해 리턴되면 데이터 변환이 무시됩니다. 큐 관리자가 CompCode를 MQCC\_FAILED로, Reason을 MQRC\_SUPPRESSED\_BY\_EXIT로 설정하지만, 설정된 값은 체인의 이후 엑시트 함수를 통해 대체할 수 있습니다. 호출에 대한 다른 매개변수는 엑시트가 해당 항목에서 벗어나도 그대로 유지됩니다. 체인의 이후 엑시트 함수를 호출할지 결정하는 데 ExitResponse2가 사용됩니다.

이 값이 MQXR\_AFTER 또는 MQXR\_CONNECTION 엑시트 함수에 의해 설정된 경우, 큐 관리자는 MQXCC\_FAILED가 리턴된 것처럼 처리를 계속합니다.

### **MQXCC\_SKIP\_FUNCTION**

IBM MQ API 함수를 건너뛸니다.

이 값은 MQXR\_BEFORE 엑시트 함수로만 설정할 수 있습니다. API 호출은 무시합니다. MQ\_DATA\_CONV\_ON\_GET\_EXIT에 의해 리턴되면 데이터 변환이 무시됩니다. 엑시트 함수가 CompCode 및 Reason을 애플리케이션에 리턴된 값으로 설정해야 하지만, 설정된 값은 체인의 이후 엑시트 함수를 통해 대체할 수 있습니다. 호출에 대한 다른 매개변수는 엑시트가 해당 항목에서 벗어나도 그대로 유지됩니다. 체인의 이후 엑시트 함수를 호출할지 결정하는 데 ExitResponse2가 사용됩니다.

이 값이 MQXR\_AFTER 또는 MQXR\_CONNECTION 엑시트 함수에 의해 설정된 경우, 큐 관리자는 MQXCC\_FAILED가 리턴된 것처럼 처리를 계속합니다.

### **MQXCC\_SUPPRESS\_EXIT**

엑시트 세트에 속한 모든 엑시트 함수를 차단합니다.

이 값은 MQXR\_BEFORE 및 MQXR\_AFTER 엑시트 함수로만 설정할 수 있습니다. 이 논리 연결의 경우, 이 엑시트 세트에 속한 엑시트 함수의 모든 후속 호출을 무시합니다. MQ\_TERM\_EXIT 함수가 MQXR\_CONNECTION의 ExitReason과 함께 호출되는 경우 논리 연결 끊기가 발생할 때까지 이러한 무시 동작이 계속됩니다.

엑시트 함수가 CompCode 및 Reason을 애플리케이션에 리턴된 값으로 설정해야 하지만, 설정된 값은 체인의 이후 엑시트 함수를 통해 대체할 수 있습니다. 호출에 대한 다른 매개변수는 엑시트가 해당 항목에서 벗어나도 그대로 유지됩니다. ExitResponse2는 무시됩니다.

이 값이 MQXR\_CONNECTION 엑시트 함수에 의해 설정된 경우, 큐 관리자는 MQXCC\_FAILED가 리턴된 것처럼 처리를 계속합니다.

ExitResponse 및 ExitResponse2 간의 상호작용과 엑시트 처리에 미치는 영향에 대한 자세한 정보는 [1492 페이지의 『큐 관리자가 엑시트 함수를 처리하는 방법』](#)의 내용을 참조하십시오.

### **ExitResponse2 (MQLONG) - 출력**

MQXR\_BEFORE 엑시트 함수에 대한 1차 엑시트 응답 코드를 규정하는 2차 엑시트 응답 코드입니다. 다음으로 초기화됩니다.

```
MQXR2_DEFAULT_CONTINUATION
```

IBM MQ API 호출 엑시트 함수에 대한 입력에서, 다음 값 중 하나로 설정할 수 있습니다.

### **MQXR2\_DEFAULT\_CONTINUATION**

ExitResponse 값에 따라 체인에서 다음 엑시트를 계속하는지 여부.



ExitResponse가 MQXCC\_SUPPRESS\_FUNCTION 또는 MQXCC\_SKIP\_FUNCTION이면 MQXR\_BEFORE 체인의 이후 엑시트 함수 그리고 MQXR\_AFTER 체인에서 일치하는 엑시트 함수를 무시합니다. MQXR\_BEFORE 체인 앞쪽에 있는 엑시트 함수와 일치하는 엑시트 함수를 MQXR\_AFTER 체인에서 호출하십시오.

그렇지 않은 경우에는 체인의 다음 엑시트를 호출합니다.

### **MQXR2\_SUPPRESS\_CHAIN**

체인을 차단합니다.

이 API 호출 환경에 대해 MQXR\_BEFORE 체인의 이후 엑시트 함수 그리고 MQXR\_AFTER 체인에서 일치하는 엑시트 함수를 무시합니다. MQXR\_BEFORE 체인 앞쪽에 있는 엑시트 함수와 일치하는 엑시트 함수를 MQXR\_AFTER 체인에서 호출하십시오.

### **MQXR2\_CONTINUE\_CHAIN**

체인의 다음 엑시트를 계속합니다.

ExitResponse 및 ExitResponse2 간의 상호작용과 엑시트 처리에 미치는 영향에 대한 자세한 정보는 [1492 페이지의 『큐 관리자가 엑시트 함수를 처리하는 방법』](#)의 내용을 참조하십시오.

### **Feedback (MQLONG) - 입출력(I/O)**

엑시트 함수 호출 간에 피드백 코드를 전달합니다. 체인의 첫 번째 엑시트의 첫 번째 함수를 호출하기 전에

```
MQFB_NONE (0)
```

으로 초기화됩니다.

엑시트는 이 필드를 올바른 MQFB\_\* 또는 MQRC\_\* 값을 포함한 모든 값으로 설정할 수 있습니다. 엑시트는 이 필드를 MQFB\_APPL\_FIRST ~ MQFB\_APPL\_LAST 범위에서 사용자 정의 피드백 값으로 설정할 수도 있습니다.

### **APICallerType (MQLONG) - 입력**

IBM MQ API 호출자가 큐 관리자에 대해 외부 또는 내부인지 여부를 나타내는 API 호출자 유형입니다 (MQXACT\_EXTERNAL 또는 MQXACT\_INTERNAL).

### **ExitUserArea (MQBYTE16) - 입출력(I/O)**

특정 ExitInfoObject와 연관된 모든 엑시트에 사용할 수 있는 사용자 영역입니다. hconn에 대한 첫 번째 엑시트 함수(MQ\_INIT\_EXIT)를 호출하기 전에 MQXUA\_NONE (ExitUserArea 길이의 경우 2진 0)으로 초기화됩니다. 그 후부터 엑시트 함수에 의한 이 필드의 변경사항이 동일한 엑시트 함수를 호출할 때 보존됩니다.

이 필드의 값은 4 MQLONG의 배수로 지정됩니다.

엑시트는 이 영역에서 할당하는 스토리지를 고정할 수도 있습니다.

각 hconn마다 엑시트 체인의 엑시트는 각기 다른 ExitUserArea를 갖습니다. ExitUserArea를 체인의 엑시트가 공유할 수 없으며, 한 엑시트의 ExitUserArea 콘텐츠를 체인의 다른 엑시트에 사용할 수 없습니다.

C 프로그램의 경우, MQXUA\_NONE\_ARRAY 상수는 MQXUA\_NONE과 동일한 값으로 정의되지만, 문자열이 아닌 문자의 배열입니다.

이 필드의 길이는 MQ\_EXIT\_USER\_AREA\_LENGTH에서 제공합니다.

### **ExitData (MQCHAR32) - 입력**

엑시트 데이터이며, 각 엑시트 함수에 대한 입력에서 엑시트에 제공된 32자의 엑시트 고유 데이터로 설정됩니다. 엑시트에서 값을 정의하지 않으면 이 필드는 모두 공백입니다.

이 필드의 길이는 MQ\_EXIT\_DATA\_LENGTH에서 제공합니다.

### **ExitInfoName (MQCHAR48) - 입력**

엑시트 정보 이름이며, 각 엑시트 함수에 대한 입력에서 스탠자의 엑시트 정의에 지정된 ApiExit\_name으로 설정됩니다.

### **ExitPDArea (MQBYTE48) - 입출력(I/O)**

문제점 판별 영역이며, 각 엑시트 함수의 호출에 대해 MQXPDA\_NONE(필드 길이의 경우 2진 0)으로 초기화됩니다.

C 프로그램의 경우, MQXPDA\_NONE\_ARRAY 상수는 MQXPDA\_NONE과 동일한 값으로 정의되지만, 문자열이 아닌 문자의 배열입니다.

엑시트 핸들러는 함수가 성공하더라도 항상 엑시트 끝의 IBM MQ 추적에 이 영역을 작성합니다.

이 필드의 길이는 MQ\_EXIT\_PD\_AREA\_LENGTH에서 제공합니다.

#### QMGrName(MQCHAR48) - 입력

애플리케이션이 연결되고, IBM MQ API 호출을 처리한 결과로 엑시트를 호출한 큐 관리자의 이름입니다.

MQCONN 또는 MQCONNX 호출에서 제공한 큐 관리자의 이름이 공백이면 애플리케이션이 서버이든 클라이언트이든 이 필드는 애플리케이션이 연결된 큐 관리자의 이름으로 계속 설정됩니다.

엑시트 핸들러가 입력 항목에서 이 필드를 각 엑시트 함수로 설정합니다.

이 필드의 길이는 MQ\_Q\_MGR\_NAME\_LENGTH로 제공됩니다.

#### ExitChainAreaPtr (PMQACH) - 입출력(I/O)

체인의 다른 엑시트를 호출할 때 데이터를 전달하는 데 사용됩니다. 엑시트 체인의 첫 번째 엑시트의 첫 번째 함수(ExitReason MQXR\_CONNECTION 포함 MQ\_INIT\_EXIT)를 호출하기 전에 NULL 포인터로 설정됩니다. 한 호출에서 엑시트가 리턴한 값이 다음 호출로 전달됩니다.

엑시트 체인 영역의 사용 방법에 대한 자세한 정보는 [1496 페이지의 『엑시트 체인 영역 및 엑시트 체인 영역 헤더\(MQACH\)』](#)의 내용을 참조하십시오.

#### Hconfig(MQHCONFIG) - 입력

초기화되는 함수 세트를 나타내는 구성 핸들입니다. 이 값은 MQ\_INIT\_EXIT 함수에서 큐 관리자에 의해 생성되고, 나중에 API 엑시트 함수로 전달됩니다. 각 엑시트 함수에 대한 입력에서 설정됩니다.

MQIEP 구조에 대한 포인터로 Hconfig를 사용하여 MQI 및 DCI 호출을 작성할 수 있습니다. HConfig 매개변수를 MQIEP 구조에 대한 포인터로 사용하기 전에 HConfig의 처음 4바이트가 MQIEP 구조의 StrucId와 일치하는지 확인해야 합니다.

#### Function (MQLONG) - 입력

함수 ID이고, 올바른 값은 [1497 페이지의 『외부 상수』](#)에 설명된 MQXF\_\* 상수입니다.

엑시트 호출로 이어진 IBM MQ API 호출에 따라 각 엑시트 함수에 대한 입력에서 엑시트 핸들러가 이 필드를 올바른 값으로 설정합니다.

#### ExitMsgHandle (MQHMSG) - 입출력(I/O)

Function이 MQXF\_GET이고 ExitReason이 MQXR\_AFTER이면 올바른 메시지 핸들이 이 필드에 리턴되어 메시지 디스크립터 필드, 그리고 API 엑시트를 등록할 때 MQXEPO 구조에 지정된 ExitProperties 문자열과 일치하는 다른 모든 특성에 API 엑시트가 액세스할 수 있습니다.

ExitMsgHandle에 리턴된 메시지 디스크립터 이외의 특성은 (지정되었더라도) MQGMO 구조의 MsgHandle에서 또는 메시지 데이터에서 사용할 수 없습니다.

Function이 MQXF\_GET이고 ExitReason이 MQXR\_BEFORE일 때 엑시트 프로그램이 이 필드를 MQHM\_NONE으로 설정하면 ExitMsgHandle 특성 채우기가 차단됩니다.

이 필드는 Version이 MQAXP\_VERSION\_2 미만인 경우 설정되지 않습니다.

### 큐 관리자가 엑시트 함수를 처리하는 방법

엑시트 함수의 리턴에서 큐 관리자가 수행하는 처리는 ExitResponse 및 ExitResponse2에 따라 다릅니다.

[1493 페이지의 표 835](#)에 사용 가능한 조합과 MQXR\_BEFORE 엑시트 함수에 대한 영향이 요약되어 있으며, 다음 정보도 제공합니다.

- API 호출의 CompCode 및 Reason 매개변수를 설정하는 사람
- MQXR\_BEFORE 체인의 나머지 엑시트 함수와 MQXR\_AFTER 체인의 일치하는 엑시트 함수의 호출 여부
- API 호출의 호출 여부

MQXR\_AFTER 엑시트 함수의 경우:

- CompCode 및 Reason은 MQXR\_BEFORE와 같은 방법으로 설정합니다.
- ExitResponse2가 무시됩니다(MQXR\_AFTER 체인의 남은 엑시트 함수는 항상 호출됨).



- MQXCC\_SUPPRESS\_FUNCTION 및 MQXCC\_SKIP\_FUNCTION은 올바르지 않습니다.

MQXR\_CONNECTION 엑시트 함수의 경우:

- CompCode 및 Reason은 MQXR\_BEFORE와 같은 방법으로 설정합니다.
- ExitResponse2는 무시됩니다.
- MQXCC\_SUPPRESS\_FUNCTION, MQXCC\_SKIP\_FUNCTION, MQXCC\_SUPPRESS\_EXIT는 올바르지 않습니다.

엑시트 또는 큐 관리자가 CompCode 및 Reason을 설정하는 모든 경우에 설정한 값은 나중에 호출된 엑시트 또는 API 호출(API 호출이 나중에 호출된 경우)에 의해 변경될 수 있습니다.

표 835. MQXR_BEFORE 엑시트 처리			
ExitResponse의 값	CompCode 및 Reason을 설정하는 항목	ExitResponse2(기본 연속) 체인의 값	ExitResponse2(기본 연속) API의 값
MQXCC_OK	exit	Y	Y
MQXCC_SUPPRESS_EXIT	exit	Y	Y
MQXCC_SUPPRESS_FUNCTION	큐 관리자	N	N
MQXCC_SKIP FUNCTION	exit	N	N
MQXCC_FAILED	큐 관리자	N	N

## 클라이언트가 엑시트 함수를 처리하는 방법

일반적으로 클라이언트를 서버 애플리케이션과 동일한 방식으로 엑시트 함수를 처리하며, 함수가 서버에 있던 클라이언트에 있던 이 구조의 *QMGrName* 속성이 적용됩니다.

그러나, 클라이언트에 *mqs.ini* 파일의 개념이 없으면 *ApiExitCommon* 및 *APIExitTemplate* 스탠자가 적용되지 않습니다. *ApiExitLocal* 스탠자만 적용되며, 이 스탠자는 *mqclient.ini* 파일에 구성됩니다.

## IBM MQ API 엑시트 컨텍스트 구조(MQAXC)

외부 제어 블록인 MQAXC 구조는 API 엑시트에 대한 입력 매개변수로 사용됩니다.

MQAXC는 다음 C 선언을 포함합니다.

```
typedef struct tagMQAXC {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    Environment;      /* Environment */
    MQCHAR12  UserId;           /* UserId associated with appl */
    MQBYTE40  SecurityId        /* Extension to UserId running appl */
    MQCHAR264 ConnectionName;   /* Connection name */
    MQLONG    LongMCAUserIdLength; /* long MCA user identifier length */
    MQLONG    LongRemoteUserIdLength; /* long remote user identifier length */
    MQPTR     LongMCAUserIdPtr; /* long MCA user identifier address */
    MQPTR     LongRemoteUserIdPtr; /* long remote user identifier address */
    MQCHAR28  ApplName;         /* Application name */
    MQLONG    ApplType;         /* Application type */
    MQPID     ProcessId;        /* Process identifier */
    MQTID     ThreadId;         /* Thread identifier */

    /* Ver:1 */
    MQCHAR    ChannelName[20]   /* Channel Name */
    MQBYTE4   Reserved1;        /* Reserved */
    PMQCD     pChannelDefinition; /* Channel Definition pointer */
};
```

MQAXC의 매개변수는 다음과 같습니다.

**StrucId (MQCHAR4) - 입력**

엑시트 컨텍스트 ID이며, MQXC\_STRUC\_ID 값을 갖습니다. C 프로그램의 경우, MQXC\_STRUC\_ID\_ARRAY 상수는 MQXC\_STRUC\_ID와 동일한 값으로 정의되지만, 문자열이 아닌 문자의 배열입니다.

엑시트 핸들러가 입력 항목에서 이 필드를 각 엑시트 함수로 설정합니다.

**Version (MQLONG) - 입력**

구조 버전 번호이며, 값은 다음과 같습니다.

**MQXC\_VERSION\_2**

엑시트 컨텍스트 구조의 버전 번호.

**MQXC\_CURRENT\_VERSION**

엑시트 컨텍스트 구조에 대한 현재 버전 번호.

엑시트 핸들러가 입력 항목에서 이 필드를 각 엑시트 함수로 설정합니다.

**Environment (MQLONG) - 입력**

IBM MQ API 호출이 발행되고 엑시트 함수가 구동된 환경. 이 필드의 올바른 값은 다음과 같습니다.

**MQXE\_OTHER**

이 값은 API 엑시트가 서버 애플리케이션에서 호출되는 경우 해당 엑시트 호출에서 일관됩니다. 이는 API 엑시트가 클라이언트에서 변화없이 실행되고 달라진 점이 없음을 의미합니다.

엑시트가 클라이언트에서 실행되는지 여부를 반드시 판별해야 하는 경우, 엑시트는 *ChannelName* 및 *ChannelDefinition* 필드를 확인하여 이 작업을 수행합니다.

**MQXE\_MCA**

메시지 채널 에이전트

**MQXE\_MCA\_SVRCONN**

클라이언트 역할을 하는 메시지 채널 에이전트

**MQXE\_COMMAND\_SERVER**

명령 서버

**MQXE\_MQSC**

runmqsc 명령 해석기

엑시트 핸들러가 입력 항목에서 이 필드를 각 엑시트 함수로 설정합니다.

**UserId (MQCHAR12) - 입력**

애플리케이션과 연관된 사용자 ID입니다. 특히, 클라이언트 연결의 경우, 이 필드에는 채널 코드가 실행되는 사용자 ID가 아닌 채택된 사용자의 사용자 ID가 표시됩니다. 클라이언트에서 공백 사용자 ID가 제공되면 이미 사용 중인 사용자 ID가 변경되지 않습니다. 즉, 새로운 사용자 ID가 채택되지 않습니다.

엑시트 핸들러가 입력 항목에서 이 필드를 각 엑시트 함수로 설정합니다. 이 필드의 길이는 MQ\_USER\_ID\_LENGTH에 지정되어 있습니다.

클라이언트의 경우, 클라이언트에서 서버로 송신한 사용자 ID입니다. 사용자 ID를 변경하는 MCAUser 또는 CHLAUTH 구성이 있을 수 있으므로 큐 관리자에서 클라이언트가 실행되고 있는 유효한 사용자 ID가 아닐 수 있음에 유의하십시오.

**SecurityId (MQBYTE40) - 입력**

애플리케이션을 실행하는 사용자 ID의 확장입니다. 해당 길이는 MQ\_SECURITY\_ID\_LENGTH에서 제공됩니다.

클라이언트의 경우, 클라이언트에서 서버로 송신한 사용자 ID입니다. 사용자 ID를 변경하는 MCAUser 또는 CHLAUTH 구성이 있을 수 있으므로 큐 관리자에서 클라이언트가 실행되고 있는 유효한 사용자 ID가 아닐 수 있음에 유의하십시오.

**ConnectionName (MQCHAR264) - 입력**

연결 이름 필드이며, 클라이언트의 주소로 설정됩니다. 예를 들어, TCP/IP의 경우 클라이언트 IP 주소가 됩니다.

이 필드의 길이는 MQ\_CONN\_NAME\_LENGTH에서 제공합니다.

클라이언트의 경우, 큐 관리자의 파트너 주소입니다.

**LongMCAUserIdLength (MQLONG) - 입력**

긴 MCA 사용자 ID의 길이.

MCA가 큐 관리자에 연결되면 이 필드는 긴 MCA 사용자 ID의 길이(또는 해당 ID가 없는 경우 0)로 설정됩니다.

클라이언트의 경우, 클라이언트 긴 사용자 ID입니다.

**LongRemoteUserIdLength (MQLONG) - 입력**

긴 리모트 사용자 ID의 길이.

MCA가 큐 관리자에 연결되면 이 필드는 긴 리모트 사용자 ID의 길이로 설정됩니다. 그렇지 않으면, 이 필드는 0으로 설정됩니다.

클라이언트의 경우, 이 필드를 0으로 설정하십시오.

**LongMCAUserIdPtr (MQPTR) - 입력**

긴 MCA 사용자 ID의 주소.

MCA가 큐 관리자에 연결되면 이 필드는 긴 MCA 사용자 ID의 주소(또는 해당 ID가 없는 경우 널 포인터)로 설정됩니다.

클라이언트의 경우, 클라이언트 긴 사용자 ID입니다.

**LongRemoteUserIdPtr (MQPTR) - 입력**

긴 리모트 사용자 ID의 주소.

MCA가 큐 관리자에 연결되면 이 필드는 긴 리모트 사용자 ID의 주소(또는 해당 ID가 없는 경우 널 포인터)로 설정됩니다.

클라이언트의 경우, 이 필드를 0으로 설정하십시오.

**ApplName (MQCHAR28) - 입력**

IBM MQ API 호출을 발행한 애플리케이션 또는 컴포넌트의 이름.

ApplName 생성 규칙과 MQPUT의 기본 이름 생성 규칙이 같습니다.

운영 체제에서 프로그램 이름을 조회하면 이 필드의 값을 찾을 수 있습니다. 해당 길이는 MQ\_APPL\_NAME\_LENGTH에서 제공됩니다.

**ApplType (MQLONG) - 입력**

IBM MQ API 호출을 발행한 애플리케이션 또는 컴포넌트의 유형.

애플리케이션이 컴파일되는 플랫폼에서는 값이 MQAT\_DEFAULT이거나 정의된 MQAT\_\* 값 중 하나입니다.

엑시트 핸들러가 입력 항목에서 이 필드를 각 엑시트 함수로 설정합니다.

**ProcessId (MQPID) - 입력**

운영 체제 프로세스 ID.

해당하는 경우, 엑시트 핸들러는 각 엑시트 함수에 대한 입력에서 이 필드를 설정합니다.

**ThreadId (MQTID) - 입력**

MQ 스레드 ID. 이 ID는 MQ 추적과 FFST 덤프에 사용된 ID와 동일하지만, 운영 체제 스레드 ID와 다를 수 있습니다.

해당하는 경우, 엑시트 핸들러는 각 엑시트 함수에 대한 입력에서 이 필드를 설정합니다.

**ChannelName (MQCHAR) - 입력**

알려진 해당 경우, 채널의 이름이며 공백으로 채워집니다.

해당 사항이 없으면 이 필드는 널 문자로 설정됩니다.

**Reserved1 (MQBYTE4) - 입력**

이 필드는 예약됩니다.

**ChannelDefinition (PMQCD) - 입력**

알려진 해당 경우, 사용되는 채널 정의에 대한 포인터입니다.

해당 사항이 없으면 이 필드는 널 문자로 설정됩니다.

IBM MQ 채널 대신 연결이 처리되는 경우에만 포인터가 완료되고 채널 정의를 읽었음에 유의하십시오.

특히, 채널의 첫 번째 MQCONN 호출이 작성될 때 채널 정의가 서버에 제공되지 않습니다. 또한, 포인터를 채우면 이 포인터가 가리키는 구조(및 하위 구조)가 읽기 전용으로 처리되어야 합니다. 구조 업데이트는 예측 불가능한 결과를 일으킬 수 있으므로 지원되지 않습니다.

클라이언트의 경우, 클라이언트에 지정된 값이 있는 필드 이외의 필드에 클라이언트 애플리케이션에 적절한 값이 표시됩니다.

## 엑시트 체인 영역 및 엑시트 체인 영역 헤더(MQACH)

필요하면 엑시트 함수는 엑시트 체인 영역의 스토리지를 확보하고 MQAXP의 ExitChainAreaPtr이 이 스토리지를 가리키도록 설정할 수 있습니다.

엑시트(동일하거나 서로 다른 엑시트 함수)는 여러 엑시트 체인 영역을 확보하고 이를 연결할 수 있습니다. 엑시트 체인 영역은 엑시트 핸들러에서 호출되는 경우에만 이 목록에 추가되거나 제거되어야 합니다. 이렇게 하면 다른 스레드가 동시에 목록에 영역을 추가하거나 제거함으로써 발생하는 직렬화 문제가 없습니다.

엑시트 체인 영역은 MQACH 헤더 구조로 시작해야 하며, 이에 대한 C 선언은 다음과 같습니다.

```
typedef struct tagMQACH {
    MQCHAR4   StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQLONG    StrucLength;      /* Length of the MQACH structure */
    MQLONG    ChainAreaLength; /* Exit chain area length */
    MQCHAR48  ExitInfoName     /* Exit information name */
    PMQACH    NextChainAreaPtr; /* Pointer to next exit chain area */
};
```

엑시트 체인 영역 헤더의 필드는 다음과 같습니다.

### StrucId (MQCHAR4) - 입력

엑시트 체인 영역 구조 ID이며, 초기 값은 MQACH\_STRUC\_ID의 MQACH\_DEFAULT에 의해 정의됩니다.

C 프로그램의 경우, MQACH\_STRUC\_ID\_ARRAY 상수는 MQACH\_STRUC\_ID와 동일한 값으로 정의되지만, 문자열이 아닌 문자의 배열입니다.

### Version (MQLONG) - 입력

구조 버전 번호이며, 다음과 같습니다.

#### MQACH\_VERSION\_1

엑시트 매개변수 구조에 대한 버전 번호.

#### MQACH\_CURRENT\_VERSION

엑시트 컨텍스트 구조에 대한 현재 버전 번호.

MQACH\_DEFAULT에 의해 정의되는 이 필드의 초기 값은 MQACH\_CURRENT\_VERSION입니다.

**참고:** 이 구조의 새 버전을 도입해도 기존 부분의 레이아웃은 변경되지 않습니다. 엑시트 함수는 버전 번호가 해당 엑시트 함수가 사용해야 하는 필드를 포함한 가장 낮은 버전보다 크거나 같은지 확인해야 합니다.

### StrucLength (MQLONG) - 입력

MQACH 구조의 길이. 엑시트는 이 필드를 통해 엑시트 데이터의 시작을 판별하고, 이를 엑시트가 작성한 구조의 길이로 설정합니다.

MQACH\_DEFAULT에 의해 정의되는 이 필드의 초기 값은 MQACH\_CURRENT\_LENGTH입니다.

### ChainAreaLength (MQLONG) - 입력

엑시트 체인 영역 길이이며, MQACH 헤더를 포함하는 현재 엑시트 체인 영역의 전체 길이로 설정됩니다.

MQACH\_DEFAULT에 의해 정의되는 이 필드의 초기 값은 0입니다.

### ExitInfoName (MQCHAR48) - 입력

엑시트 정보 이름.

엑시트가 MQACH 구조를 작성하는 경우, 고유의 ExitInfoName으로 이 필드를 초기화해야 합니다. 그래야 나중에 이 엑시트의 다른 인스턴스 또는 보완 엑시트에서 이 MQACH 구조를 찾을 수 있습니다.

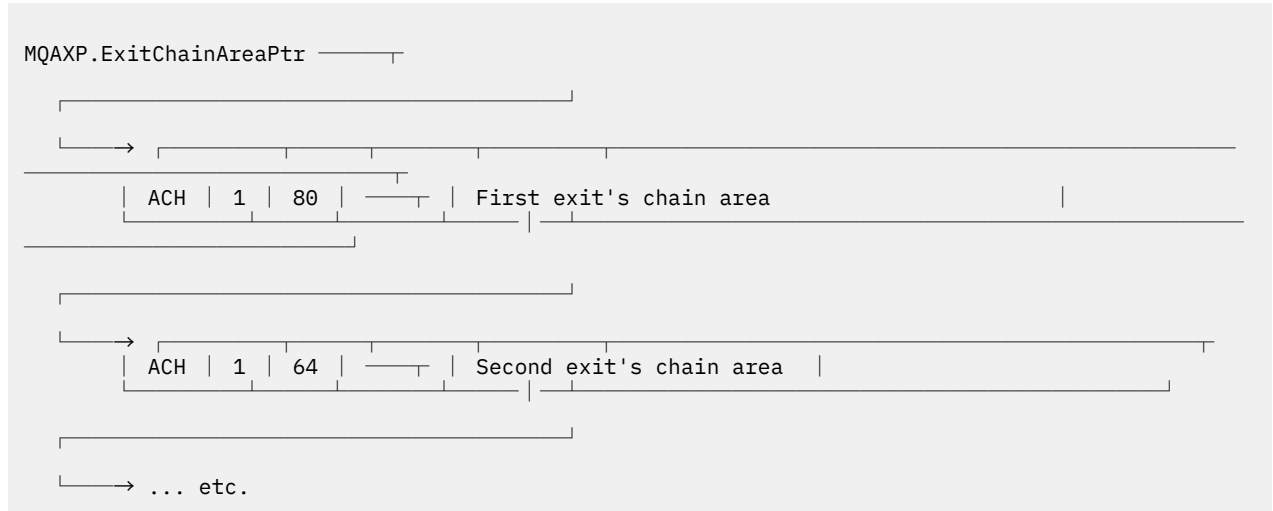
MQACH\_DEFAULT에 의해 정의되는 이 필드의 초기 값은 길이가 0인 문자열(“”)입니다.

## NextChainAreaPtr (PMQACH) - 입력

다음 엑시트 체인 영역에 대한 포인터이며, MQACH\_DEFAULT에 의해 정의되는 초기값은 널 포인터(NULL)입니다.

엑시트 함수는 확보한 엑시트 체인 영역의 스토리지를 릴리스하고, 목록에서 엑시트 체인 영역을 제거하도록 체인 포인터를 조작해야 합니다.

엑시트 체인 영역은 다음과 같이 구성할 수 있습니다.



## 외부 상수

이 주제를 API 엑시트에 사용할 수 있는 외부 상수에 관한 참조 정보로 사용하십시오.

API 엑시트에 사용할 수 있는 외부 상수는 다음과 같습니다.

### MQXF\_\* (엑시트 함수 ID)

MQXF_INIT	1	X'00000001'
MQXF_TERM	2	X'00000002'
MQXF_CONN	3	X'00000003'
MQXF_CONNX	4	X'00000004'
MQXF_DISC	5	X'00000005'
MQXF_OPEN	6	X'00000006'
MQXF_CLOSE	7	X'00000007'
MQXF_PUT1	8	X'00000008'
MQXF_PUT	9	X'00000009'
MQXF_GET	10	X'0000000A'
MQXF_DATA_CONV_ON_GET	11	X'0000000B'
MQXF_INQ	12	X'0000000C'
MQXF_SET	13	X'0000000D'
MQXF_BEGIN	14	X'0000000E'
MQXF_CMIT	15	X'0000000F'
MQXF_BACK	16	X'00000010'
MQXF_STAT	18	X'00000012'
MQXF_CB	19	X'00000013'
MQXF_CTL	20	X'00000014'
MQXF_CALLBACK	21	X'00000015'
MQXF_SUB	22	X'00000016'
MQXF_SUBRQ	23	X'00000017'
MQXF_XACLOSE	24	X'00000018'
MQXF_XACOMMIT	25	X'00000019'
MQXF_XACOMplete	26	X'0000001A'
MQXF_XAEND	27	X'0000001B'
MQXF_XAFORGET	28	X'0000001C'
MQXF_XAOPEN	29	X'0000001D'
MQXF_XAPREPRE	30	X'0000001E'
MQXF_XARECOVER	31	X'0000001F'
MQXF_XAROLLBACK	32	X'00000020'
MQXF_XASTART	33	X'00000021'
MQXF_AXREG	34	X'00000022'
MQXF_AXUNREG	35	X'00000023'

## MQXR\_\* (엑시트 이유)

MQXR_BEFORE	1	X'00000001'
MQXR_AFTER	2	X'00000002'
MQXR_CONNECTION	3	X'00000003'

## MQXE\_\* (환경)

MQXE_OTHER	0	X'00000000'
MQXE_MCA	1	X'00000001'
MQXE_MCA_SVRCONN	2	X'00000002'
MQXE_COMMAND_SERVER	3	X'00000003'
MQXE_MQSC	4	X'00000004'

## MQ\*\_\* (추가 상수)

MQAXP_VERSION_1	1	
MQAXP_VERSION_2	2	
MQAXC_VERSION_1	1	
MQACH_VERSION_1	1	
MQAXP_CURRENT_VERSION	1	
MQAXC_CURRENT_VERSION	1	
MQACH_CURRENT_VERSION	1	
MQXACT_EXTERNAL	1	
MQXACT_INTERNAL	2	
MQXT_API_EXIT	2	
MQACH_LENGTH_1	68 (32-bit platforms) 72 (64-bit platforms) 80 (128-bit platforms)	
MQACH_CURRENT_LENGTH	68 (32-bit platforms) 72 (64-bit platforms) 80 (128-bit platforms)	

## MQ\*\_\* (널 상수)

MQXPDA_NONE	X'00...00' (48 nulls)
MQXPDA_NONE_ARRAY	'\0','\0',...,'\0','\0'

## MQXCC\_\* (완료 코드)

MQXCC_FAILED	-8
--------------	----

## MQRC\_\* (이유 코드)

### MQRC\_API\_EXIT\_ERROR 2374 X'00000946'

엑시트 함수 호출이 올바르지 않은 응답 코드를 리턴했거나 어떤 식으로든 실패했으며, 큐 관리자가 수행할 다음 조치를 판별할 수 없습니다.

MQAXP의 ExitResponse 및 ExitResponse2 필드를 조사하여 잘못된 응답 코드를 판별하고, 올바른 응답 코드를 리턴하도록 엑시트를 변경하십시오.

### MQRC\_API\_EXIT\_INIT\_ERROR 2375 X'00000947'

큐 관리자가 API 엑시트 함수에 대한 실행 환경을 초기화하는 동안 오류가 발생했습니다.

### MQRC\_API\_EXIT\_TERM\_ERROR 2376 X'00000948'

큐 관리자가 API 엑시트 함수에 대한 실행 환경을 닫는 동안 오류가 발생했습니다.

### MQRC\_EXIT\_REASON\_ERROR 2377 X'00000949'

엑시트 시작점 등록 호출(MQXEP) 호출에 제공되는 ExitReason 필드의 값에 오류가 있습니다.

ExitReason 필드의 값을 조사하여 잘못된 엑시트 이유 값을 판별하고 정정하십시오.

## MQRC\_RESERVED\_VALUE\_ERROR 2378 X'0000094A'

Reserved 필드의 값에 오류가 있습니다.

Reserved 필드의 값을 조사하여 Reserved 값을 판별하고 정정하십시오.

## C 언어 typedefs

이 주제에서는 C 언어에서 사용할 수 있는 API 엑시트와 연관된 typedefs에 대한 정보를 제공합니다.

다음은 API 엑시트와 연관된 C 언어 typedefs입니다.

```
typedef PMLONG MQPOINTER PPMQLONG;
typedef PMQBYTE MQPOINTER PPMQBYTE;
typedef PMQHOBX MQPOINTER PPMQHOBX;
typedef PMQOD MQPOINTER PPMQOD;
typedef PMQMD MQPOINTER PPMQMD;
typedef PMQPMO MQPOINTER PPMQPMO;
typedef PMQGMO MQPOINTER PPMQGMO;
typedef PMQCNO MQPOINTER PPMQCNO;
typedef PMQBO MQPOINTER PPMQBO;

typedef MQAXP MQPOINTER PMQAXP;
typedef MQACH MQPOINTER PMQACH;
typedef MQAXC MQPOINTER PMQAXC;

typedef MQCHAR MQCHAR16[16];
typedef MQCHAR16 MQPOINTER PMQCHAR16;

typedef MQLONG MQPID;
typedef MQLONG MQTID;
```

## 엑시트 시작점 등록 호출(MQXEP)

MQXEP, MQXEP C 언어 호출 및 MQXEP C 함수 프로토타입에 대해 알아보려면 이 정보를 사용하십시오.

MQXEP 호출 사용 목적:

1. 엑시트 함수를 호출하는 IBM MQ API 엑시트 호출 시점의 전후 등록
2. 엑시트 함수 시작점 지정
3. 엑시트 함수 시작점 등록 취소

일반적으로 MQ\_INIT\_EXIT 엑시트 함수에서 MQXEP 호출을 코딩하지만, 후속 엑시트 함수에 지정할 수도 있습니다.

MQXEP 호출을 사용하여 이미 등록된 엑시트 함수를 등록하는 경우, 두 번째 MQXEP 호출이 성공적으로 완료되고 등록된 엑시트 함수를 대체합니다.

MQXEP 호출을 사용하여 NULL 엑시트 함수를 등록하는 경우, MQXEP 호출이 성공적으로 완료되고 엑시트 함수가 등록 취소됩니다.

MQXEP 호출을 사용하여 연결 요청 수명 동안 특정 엑시트 함수를 등록, 등록 취소 및 재등록하는 경우, 이전에 등록된 엑시트 함수가 다시 활성화됩니다. 이 엑시트 함수와 연관된 스토리지가 여전히 할당되어 있으면 엑시트 함수에서 이 스토리지를 사용할 수 있습니다. (이 스토리지는 일반적으로 종료 엑시트 함수를 호출하는 동안 릴리스됩니다.)

MQXEP에 대한 인터페이스는 다음과 같습니다.

```
MQXEP (Hconfig, ExitReason, Function, EntryPoint, &ExitOpts, &CompCode, &Reason)
```

설명:

### Hconfig(MQHCONFIG) - 입력

초기화되는 함수 세트를 포함한 API 엑시트를 나타내는 구성 핸들입니다. 이 값은 MQ\_INIT\_EXIT 함수 호출 직전에 큐 관리자에 의해 생성되고, MQAXP에서 각 API 엑시트 함수로 전달됩니다.

### ExitReason (MQLONG) - 입력

시작점이 등록되는 이유로, 다음과 같습니다.

- 연결 레벨 초기화 또는 종료(MQXR\_CONNECTION)
- IBM MQ API 호출(MQXR\_BEFORE) 이전
- IBM MQ API 호출(MQXR\_AFTER) 이후

**Function (MQLONG) - 입력**

함수 ID이며, 올바른 값은 MQXF\_\* 상수입니다(1497 페이지의 『외부 상수』 참조).

**EntryPoint (PMQFUNC) - 입력**

엑시트 함수가 등록되는 시작점의 주소. 값이 널이면 엑시트 함수가 제공되지 않았거나 엑시트 함수의 이전 등록을 등록 취소하고 있음을 표시합니다.

**ExitOpts(MQXEPO)**

API 엑시트는 API 엑시트의 등록 방법을 제어하는 옵션을 지정할 수 있습니다. 이 필드에 널 포인터가 지정된 경우, MQXEPO 구조의 기본값이 사용됩니다.

**CompCode(MQLONG) - 출력**

완료 코드이며, 올바른 값은 다음과 같습니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_FAILED**

호출에 실패했습니다.

**Reason(MQLONG) - 출력**

완료 코드를 규정하는 이유 코드입니다.

완료 코드가 MQCC\_OK인 경우:

**MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

완료 코드가 MQCC\_FAILED인 경우

**MQRC\_HCONFIG\_ERROR**

(2280, X'8E8') 제공된 구성 핸들이 올바르지 않습니다. MQAXP의 구성 핸들을 사용하십시오.

**MQRC\_EXIT\_REASON\_ERROR**

(2377, X'949') 제공된 엑시트 함수 호출 이유가 올바르지 않거나 제공된 엑시트 함수 ID에 대해 올바르지 않습니다.

올바른 엑시트 함수 호출 이유(MQXR\_\* 값) 중 하나를 사용하거나, 올바른 함수 ID 및 엑시트 이유 조합을 사용하십시오. (1500 페이지의 표 836 참조).

**MQRC\_FUNCTION\_ERROR**

(2281, X'8E9') 제공된 함수 ID가 API 엑시트 이유에 올바르지 않습니다. 다음 표는 함수 ID 및 ExitReasons의 올바른 조합을 보여줍니다.

표 836. 함수 ID 및 ExitReasons의 올바른 조합	
Function	ExitReason
MQXF_INIT MQXF_TERM	MQXR_CONNECTION



표 836. 함수 ID 및 ExitReasons의 올바른 조합 (계속)

Function	ExitReason
MQXF_CONN MQXF_CONNX MQXF_DISC MQXF_OPEN MQXF_CLOSE MQXF_PUT1 MQXF_PUT MQXF_GET MQXF_INQ MQXF_SET MQXF_BEGIN MQXF_CMIT MQXF_BACK MQXF_STAT MQXF_CB MQXF_CTL MQXF_CALLBACK MQXF_SUB MQXF_SUBRQ	MQXR_BEFORE MQXR_AFTER
MQXF_DATA_CONV_ON_GET	MQXR_BEFORE

**MQRC\_RESOURCE\_PROBLEM**

(2102, X'836') 엑시트 함수를 등록하거나 등록 취소하려고 했으나 자원 문제점 때문에 실패했습니다.

**MQRC\_UNEXPECTED\_ERROR**

(2195, X'893') 엑시트 함수를 등록하거나 등록 취소하려고 했으나 예기치 않게 실패했습니다.

**MQRC\_PROPERTY\_NAME\_ERROR**

(2442, X'098A') 올바르지 않은 ExitProperties 이름입니다.

**MQRC\_XEPO\_ERROR**

(2507, X'09CB') 엑시트 옵션 구조가 올바르지 않습니다.

**MQXEP C 언어 호출**

```
MQXEP (Hconfig, ExitReason, Function, EntryPoint, &ExitOpts, &CompCode, &Reason);
```

매개변수 선언 목록:

```
MQHCONFIG    Hconfig;        /* Configuration handle */
MQLONG       ExitReason;   /* Exit reason */
MQLONG       Function;     /* Function identifier */
PMQFUNC      EntryPoint;   /* Function entry point */
MQXEPO       ExitOpts;     /* Options that control the action of MQXEP */
MQLONG       CompCode;     /* Completion code */
MQLONG       Reason;       /* Reason code qualifying completion
                             code */
```

**MQXEP C 함수 프로토타입**

```
void MQXEP (
MQHCONFIG    Hconfig,        /* Configuration handle */
MQLONG       ExitReason,    /* Exit reason */
MQLONG       Function,      /* Function identifier */
PMQFUNC      EntryPoint,    /* Function entry point */
```

```

PMQXEPO      pExitOpts;      /* Options that control the action of MQXEP */
PMQLONG      pCompCode,    /* Address of completion code */
PMQLONG      pReason);    /* Address of reason code qualifying completion
                           code */

```

## 엑시트 함수

이 절에서는 함수 호출 사용 시 유용한 일부 일반 정보를 제공하고 개별 엑시트 함수를 호출하는 방법에 대해 설명합니다.

API 엑시트 루틴의 일반 규칙과 엑시트 실행 환경을 설정 및 정리하는 작업에 대해 알아보려면 이 정보를 사용하십시오.

## API 엑시트 루틴의 일반 규칙

API 엑시트 루틴 호출 시 다음과 같은 일반 규칙이 적용됩니다.

- 모든 경우에서 API 호출 매개변수의 유효성을 검증하기 이전 그리고 보안 검사 이전에 API 엑시트 함수가 구동됩니다(MQCONN, MQCONNX 또는 MQOPEN의 경우).
- 엑시트 루틴에 입력되고 출력되는 필드의 값은 다음과 같습니다.
  - 사전 IBM MQ API 엑시트 함수에 대한 입력에서는 필드의 값이 애플리케이션 프로그램 또는 이전 엑시트 함수 호출에 의해 설정될 수 있습니다.
  - 사전 IBM MQ API 엑시트 함수의 출력에서는 필드 값이 바뀌지 않고 그대로 유지되거나 엑시트 함수에 의해 몇 가지 다른 값으로 설정될 수 있습니다.
  - 사후 IBM MQ API 엑시트 함수에 대한 입력에서는 필드의 값이 IBM MQ API 호출을 처리한 후 큐 관리자가 설정한 값이거나, 엑시트 함수 체인의 이전 엑시트 함수 호출에 의해 값으로 설정될 수 있습니다.
  - 사후 IBM MQ API 호출 엑시트 함수의 출력에서는 필드의 값이 바뀌지 않고 그대로 유지되거나 엑시트 함수에 의해 몇 가지 다른 값으로 설정될 수 있습니다.
- 엑시트 함수는 ExitResponse 및 ExitResponse2 필드를 사용하여 큐 관리자와 통신해야 합니다.
- CompCode 및 Reason 코드 필드가 애플리케이션에 다시 전달됩니다. 큐 관리자 및 엑시트 함수가 CompCode 및 Reason 코드 필드를 설정할 수 있습니다.
- MQXEP 호출은 MQXEP를 호출하는 엑시트 함수에 새 이유 코드를 리턴합니다. 그러나 엑시트 함수가 이러한 새 이유 코드를 기존 및 새로운 애플리케이션이 이해할 수 있는 기존의 이유 코드로 변환할 수 있습니다.
- 각 엑시트 함수 프로토타입은 CompCode 및 Reason을 제외하고 API 함수와 비슷한 매개변수를 사용하지만 더 높은 수준의 간접 기능을 지원합니다.
- API 엑시트는 MQI 호출(MQDISC 제외)을 발행할 수 있지만, 이 MQI 호출이 API 엑시트를 자체 호출하지는 않습니다.

애플리케이션이 서버에 있든 클라이언트에 있든 API 엑시트 호출의 순서 지정을 예측할 수 없습니다. API 엑시트 BEFORE 호출 뒤에 AFTER 호출이 바로 이어지지 않을 수 있습니다.

BEFORE 호출 뒤에 다른 BEFORE 호출이 이어질 수 있습니다. 예를 들면, 다음과 같습니다.

```

BEFORE MQCTL
BEFORE 콜백
BEFORE MQPUT
AFTER MQPUT
AFTER 콜백
AFTER MQCTL

```

또는

```

BEFORE XAOPEN
BEFORE MQCONNX
AFTER MQCONNX
AFTER XAOPEN

```

클라이언트에서 MQCONN 또는 MQCONNX 호출의 동작을 수정할 수 있는 엑시트가 있으며, 그 이름은 PreConnect 엑시트입니다. PreConnect 엑시트는 MQCONN 또는 MQCONNX 호출에서 큐 관리자 이름을 포함하여 매개변수를 수정할 수 있습니다. 클라이언트는 이 엑시트를 먼저 호출한 다음 MQCONN 또는 MQCONNX 호출을 호출합니다. 초기 MQCONN 또는 MQCONNX 호출만 API 엑시트를 호출하며 이후의 다시 연결 호출은 아무런 영향이 없습니다.

## 실행 환경

일반적으로 MQAXP의 ExitResponse 및 ExitResponse2 필드를 사용하여 엑시트 함수의 모든 오류가 엑시트 핸들러로 전달됩니다.

그러면 이 오류는 MQCC\_\* 및 MQRC\_\* 값으로 변환되고, CompCode 및 Reason 필드에서 애플리케이션에 다시 전달됩니다. 그러나, 엑시트 핸들러 로직에 발생한 오류는 CompCode 및 Reason 필드에서 MQCC\_\* 및 MQRC\_\* 값으로 애플리케이션에 전달됩니다.

MQ\_TERM\_EXIT 함수가 오류를 리턴할 경우:

- MQDISC 호출이 이미 발생함
- 사후 MQ\_TERM\_EXIT 엑시트 함수를 구동(그리고 엑시트 실행 환경 정리를 수행)하는 다른 기회가 없음
- 엑시트 실행 환경 정리가 수행되지 않음

엑시트가 여전히 사용 중이므로 로드 해제할 수 없습니다. 또한, 사전 엑시트가 성공한 엑시트 체인의 다른 등록된 엑시트가 반대 순서대로 구동됩니다.

## 엑시트 실행 환경 설정

명시적 MQCONN 또는 MQCONNX 호출을 처리하는 동안 엑시트 핸들링 로직이 엑시트 초기화 함수(MQ\_INIT\_EXIT)를 호출하기 전에 엑시트 실행 환경을 설정합니다. 엑시트 실행 환경 설정에는 엑시트 로딩, 스토리지 확보, 엑시트 매개변수 구조 초기화가 포함됩니다. 엑시트 구성 핸들러도 할당됩니다.

이 단계에서 오류가 발생하면 CompCode MQCC\_FAILED 및 다음 이유 코드 중 하나가 표시되면서 MQCONN 또는 MQCONNX 호출이 실패합니다.

### MQRC\_API\_EXIT\_LOAD\_ERROR

API 엑시트 모듈을 로드하려고 했으나 실패했습니다.

### MQRC\_API\_EXIT\_NOT\_FOUND

API 엑시트 모듈에서 API 엑시트 함수를 찾을 수 없습니다.

### MQRC\_STORAGE\_NOT\_AVAILABLE

API 엑시트 함수의 실행 환경을 초기화하려고 했으나 사용 가능한 스토리지가 충분하지 않아 실패했습니다.

### MQRC\_API\_EXIT\_INIT\_ERROR

API 엑시트 함수에 대한 실행 환경을 초기화하는 동안 오류가 발생했습니다.

## 엑시트 실행 환경 정리

명시적 MQDISC 호출 또는 애플리케이션 종료의 결과로 암시적 다시 연결 요청을 처리하는 동안 엑시트 핸들링 로직이 엑시트 종료 함수(MQ\_TERM\_EXIT)를 호출한 후 엑시트 실행 환경을 정리해야 할 수 있습니다(등록된 경우).

엑시트 실행 환경 정리에는 엑시트 매개변수 구조의 스토리지 해제, 이전에 메모리에 로드한 모듈 삭제가 포함됩니다.

이 단계에서 오류가 발생하면 CompCode MQCC\_FAILED 및 다음 이유 코드가 표시되면서 명시적 MQDISC 호출이 실패합니다(암시적 연결 끊기 요청에서 오류가 강조 표시되지 않음).

### MQRC\_API\_EXIT\_TERM\_ERROR

API 엑시트 함수에 대한 실행 환경을 닫는 동안 오류가 발생했습니다. 엑시트가 MQ\_TERM\* API 엑시트 함수 호출 전후에 MQDISC에서 실패를 리턴하지 않아야 합니다.

## 클라이언트의 API 엑시트

클라이언트는 PreConnect 엑시트를 사용하여 MQCONN 및 MQCONNX 호출의 동작을 수정하고 API 엑시트 특성을 지원하지 않습니다.

## PreConnect 엑시트

클라이언트에서 PreConnect 엑시트는 LDAP 서버와 같은 중앙 저장소에서 채널 정의를 찾아보는 데 사용할 수 있습니다.

PreConnect 엑시트는 MQCONN 또는 MQCONNX 호출의 임의 매개변수 또는 모든 매개변수를 수정할 수도 있습니다(예: 큐 관리자 이름).

클라이언트 애플리케이션의 경우, MQCONN 또는 MQCONNX API 엑시트가 큐 관리자의 이름이 알려진 후에만 호출되고 PreConnect 엑시트가 이 이름을 변경할 수 있기 때문에 PreConnect 엑시트가 API 엑시트 이전에 호출되어야 합니다.

초기 MQCONN 또는 MQCONNX 호출만 엑시트를 호출할 수 있습니다.

## API 엑시트 특성

서버에서 API 엑시트는 초기화 시간에 MQXEPO 구조를 등록할 수 있습니다. MQXEPO 구조에는 엑시트와 관련 있는 특성 그룹을 자세히 설명하는 ExitProperties 필드가 있습니다. 이 필드는 애플리케이션 메시지 특성 핸들과 별도로 엑시트가 조작할 수 있는 별도의 메시지 특성 핸들을 생성하는 효과가 있습니다.

클라이언트에서 API 엑시트 특성이 지원되지 않습니다. 클라이언트에 특성 그룹 이름을 등록하려고 하는 경우, 이유 코드 MQRC\_EXIT\_PROPS\_NOT\_SUPPORTED가 표시되면서 함수가 실패합니다.

## 백아웃 - MQ\_BACK\_EXIT

MQ\_BACK\_EXIT는 사전 및 사후 백아웃 처리를 수행하는 백아웃 엑시트 함수를 제공합니다. 사전 및 사후 백아웃 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_BACK을 사용하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
MQ_BACK_EXIT (&ExitParms, &ExitContext, &Hconn, &CompCode, &Reason)
```

여기서 매개변수는 다음과 같습니다.

### ExitParms (MQAXP) - 입출력(I/O)

엑시트 매개변수 구조입니다.

### ExitContext (MQAXC) - 입출력(I/O)

엑시트 컨텍스트 구조입니다.

### Hconn (MQHCONN) - 입력

연결 핸들입니다.

### CompCode (MQLONG) - 입출력(I/O)

완료 코드의 올바른 값은 다음과 같습니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_WARNING

부분 완료입니다.

#### MQCC\_FAILED

호출에 실패했습니다.

### Reason (MQLONG) - 입출력(I/O)

완료 코드를 규정하는 이유 코드입니다.

완료 코드가 MQCC\_OK인 경우 유일하게 올바른 값은 다음과 같습니다.

#### MQRC\_NONE

(0, x'000') 보고할 이유가 없습니다.

완료 코드가 MQCC\_FAILED 또는 MQCC\_WARNING인 경우 엑시트 함수는 이유 코드 필드를 올바른 MQRC\_\* 값으로 설정할 수 있습니다.

## C 언어 호출

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP    ExitParms;      /* Exit parameter structure */
MQAXC    ExitContext;    /* Exit context structure */
MQHCONN  Hconn;          /* Connection handle */
MQLONG   CompCode;       /* Completion code */
MQLONG   Reason;         /* Reason code qualifying completion code */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
MQ_BACK_EXIT (&ExitParms, &ExitContext, &Hconn, &CompCode, &Reason);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
void MQENTRY MQ_BACK_EXIT (
PMQAXP    pExitParms,    /* Address of exit parameter structure */
PMQAXC    pExitContext,  /* Address of exit context structure */
PMQHCONN  pHconn,        /* Address of connection handle */
PMQLONG   pCompCode,     /* Address of completion code */
PMQLONG   pReason);      /* Address of reason code qualifying completion
                           code */
```

### 시작 - MQ\_BEGIN\_EXIT

MQ\_BEGIN\_EXIT는 사전 및 사후 MQBEGIN 호출 처리를 수행하는 시작 엑시트 함수를 제공합니다. 엑시트 이 유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_BEGIN을 사용하여 사전 및 사후 MQBEGIN 호출 엑시트 함수를 등록하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
MQ_BEGIN_EXIT (&ExitParms, &ExitContext, &Hconn, &pBeginOptions, &CompCode,
               &Reason)
```

여기서 매개변수는 다음과 같습니다.

#### ExitParms (MQAXP) - 입출력(I/O)

엑시트 매개변수 구조입니다.

#### ExitContext (MQAXC) - 입출력(I/O)

엑시트 컨텍스트 구조입니다.

#### Hconn (MQHCONN) - 입력

연결 핸들입니다.

#### pBeginOptions (PMQBO)- 입출력(I/O)

시작 옵션에 대한 포인터.

#### CompCode (MQLONG) - 입출력(I/O)

완료 코드의 올바른 값은 다음과 같습니다.

##### MQCC\_OK

정상적으로 완료되었습니다.

##### MQCC\_WARNING

부분 완료입니다.

##### MQCC\_FAILED

호출에 실패했습니다.

#### Reason (MQLONG) - 입출력(I/O)

완료 코드를 규정하는 이유 코드입니다.

완료 코드가 MQCC\_OK인 경우 유일하게 올바른 값은 다음과 같습니다.

## MQRC\_NONE

(0, x'000') 보고할 이유가 없습니다.

완료 코드가 MQCC\_FAILED 또는 MQCC\_WARNING인 경우 엑시트 함수는 이유 코드 필드를 올바른 MQRC\_\* 값으로 설정할 수 있습니다.

## C 언어 호출

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP    ExitParms;      /* Exit parameter structure */
MQAXC    ExitContext;    /* Exit context structure */
MQHCONN  Hconn;          /* Connection handle */
PMQBO    pBeginOptions;  /* Ptr to begin options */
MQLONG   CompCode;       /* Completion code */
MQLONG   Reason;         /* Reason code qualifying completion code */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
MQ_BEGIN_EXIT (&ExitParms, &ExitContext, &Hconn, &pBeginOptions, &CompCode,
               &Reason);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
void MQENTRY MQ_BEGIN_EXIT (
PMQAXP    pExitParms,      /* Address of exit parameter structure */
PMQAXC    pExitContext,    /* Address of exit context structure */
PMQHCONN  pHconn,          /* Address of connection handle */
PPMQBO    ppBeginOptions,  /* Address of ptr to begin options */
PMQLONG   pCompCode,       /* Address of completion code */
PMQLONG   pReason);        /* Address of reason code qualifying completion
                             code */
```

## 콜백 - MQ\_CALLBACK\_EXIT

MQ\_CALLBACK\_EXIT는 사전 및 사후 콜백 처리를 수행하는 엑시트 함수를 제공합니다. 사전 및 사후 콜백 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_CALLBACK을 사용하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
MQ_CALLBACK_EXIT (&ExitParms, &ExitContext, &Hconn, &pMsgDesc, &pGetMsgOpts,
                  &pBuffer, &pMQCBCContext)
```

여기서 매개변수는 다음과 같습니다.

### ExitParms (MQAXP) - 입출력(I/O)

엑시트 매개변수 구조

### ExitContext (MQAXC) - 입출력(I/O)

엑시트 컨텍스트 구조

### Hconn(MQHCONN) - 입출력(I/O)

연결 핸들

### pMsgDesc

메시지 디스크립터

### pGetMsgOpts

MQGET의 조치를 제어하는 옵션

### pBuffer

메시지 데이터를 포함하는 영역

### pMQCBCContext

콜백에 대한 컨텍스트 데이터

## C 언어 호출

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP      ExitParms;      /* Exit parameter structure */
MQAXC      ExitContext;    /* Exit context structure */
MQHCONN    Hconn;         /* Connection handle */
PMQMD      pMsgDesc;      /* Message descriptor */
PMQGMO     pGetMsgOpts;    /* Options that define the operation of the consumer */
PMQVOID    pBuffer;      /* Area to contain the message data */
PMQCBC     pContext;      /* Context data for the callback */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
MQ_SUBRQ_EXIT (&ExitParms, &ExitContext, &Hconn, &pMsgDesc, &pGetMsgOpts, &pBuffer,
               &pContext);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
void MQENTRY MQ_CALLBACK_EXIT (
PMQAXP      pExitParms;    /* Exit parameter structure */
PMQAXC      pExitContext;  /* Exit context structure */
PMQHCONN    pHconn;       /* Connection handle */
PPMMD       ppMsgDesc;    /* Message descriptor */
PPMQGMO     ppGetMsgOpts; /* Options that define the operation of the consumer */
PPMVOID     ppBuffer;     /* Area to contain the message data */
PPMQCBC     ppContext;)   /* Context data for the callback */
```

## 사용법 참고

- 콜백 엑시트는 이용자가 호출되기 전 그리고 이용자의 이용자 함수가 완료된 후에 호출됩니다. MQMD 및 MQGMO 구조를 대체할 수 있더라도 메시지가 이용자 함수에 전달하기 위해 큐에서 이미 제거되었으므로 사전 엑시트에서 값을 변경해도 큐에서 메시지 검색을 다시 구동하지 않습니다.

### 콜백 함수 관리 - MQ\_CB\_EXIT

MQ\_CB\_EXIT는 사전 및 사후 MQCB 호출을 수행하는 엑시트 함수를 제공합니다. 사전 및 사후 MQCB 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_CB를 사용하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
MQ_CB_EXIT (&ExitParms, &ExitContext, &Hconn, &Operation, &pCallbackDesc,
            &Hobj, &pMsgDesc, &pGetMsgOpts, &CompCode, &Reason)
```

여기서 매개변수는 다음과 같습니다.

#### ExitParms (MQAXP) - 입출력(I/O)

엑시트 매개변수 구조

#### ExitContext (MQAXC) - 입출력(I/O)

엑시트 컨텍스트 구조

#### Hconn(MQHCONN) - 입출력(I/O)

연결 핸들

#### Operation (MQLONG) - 입출력(I/O)

조작 값

#### pCallbackDesc (PMQCBD) - 입출력(I/O)

콜백 디스크립터

#### Hobj (MQHOBJ) - 입출력(I/O)

오브젝트 핸들

#### pMsgDesc (PMQMD) - 입출력(I/O)

메시지 디스크립터

## **pGetMsgOpts (PMQGMO) - 입출력(I/O)**

MQCB의 조치를 제어하는 옵션

## **CompCode (MQLONG) - 입출력(I/O)**

완료 코드

## **Reason (MQLONG) - 입출력(I/O)**

CompCode를 규정하는 이유 코드

## **C 언어 호출**

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP    ExitParms;      /* Exit parameter structure */
MQAXC    ExitContext;    /* Exit context structure */
MQHCONN  Hconn;         /* Connection handle */
MQLONG   Operation;     /* Operation value. */
MQCBD    pMsgDesc;      /* Callback descriptor. */
MQHOBJ   Hobj;         /* Object handle. */
PMQMD    pMsgDesc;      /* Message descriptor */
PMQGMO   pGetMsgOpts;   /* Options that define the operation of the consumer */
MQLONG   CompCode;     /* Completion code. */
MQLONG   Reason;       /* Reason code qualifying CompCode. */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
MQ_CB_EXIT (&ExitParms, &ExitContext, &Hconn, &Operation, &Hobj, &pMsgDesc,
            &pGetMsgOpts, &CompCode, &Reason);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
void MQENTRY MQ_CB_EXIT (
PMQAXP    pExitParms;    /* Exit parameter structure */
PMQAXC    pExitContext; /* Exit context structure */
PMQHCONN  pHconn;      /* Connection handle */
PMQLONG   pOperation;   /* Callback operation */
PMQHOBJ   pHobj;       /* Object handle */
PPMQMD    ppMsgDesc;    /* Message descriptor */
PPMQGMO   ppGetMsgOpts; /* Options that control the action of MQCB */
PMQLONG   pCompCode;    /* Completion code */
PMQLONG   pReason;     /* Reason code qualifying CompCode */
```

## **닫기 - MQ\_CLOSE\_EXIT**

MQ\_CLOSE\_EXIT는 사전 및 사후 MQCLOSE 호출 처리를 수행하는 닫기 엑시트 함수를 제공합니다. 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_CLOSE를 사용하여 사전 및 사후 MQCLOSE 호출 엑시트 함수를 등록하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
MQ_CLOSE_EXIT (&ExitParms, &ExitContext, &Hconn, &pHobj,
               &Options, &CompCode, &Reason)
```

여기서 매개변수는 다음과 같습니다.

### **ExitParms (MQAXP) - 입출력(I/O)**

엑시트 매개변수 구조입니다.

### **ExitContext (MQAXC) - 입출력(I/O)**

엑시트 컨텍스트 구조입니다.

### **Hconn (MQHCONN) - 입력**

연결 핸들입니다.

### **pHobj (PMQHOBj) - 입력**

오브젝트 핸들에 대한 포인터.



## Options (MQLONG)- 입출력(I/O)

닫기 옵션.

## CompCode (MQLONG) - 입출력(I/O)

완료 코드의 올바른 값은 다음과 같습니다.

### **MQCC\_OK**

정상적으로 완료되었습니다.

### **MQCC\_FAILED**

호출에 실패했습니다.

## Reason (MQLONG) - 입출력(I/O)

완료 코드를 규정하는 이유 코드입니다.

완료 코드가 MQCC\_OK인 경우 유일하게 올바른 값은 다음과 같습니다.

### **MQRC\_NONE**

(0, x'000') 보고할 이유가 없습니다.

완료 코드가 MQCC\_FAILED이면 엑시트 함수는 이유 코드 필드를 올바른 MQRC\_\* 값으로 설정할 수 있습니다.

## C 언어 호출

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP      ExitParms;      /* Exit parameter structure */
MQAXC      ExitContext;    /* Exit context structure */
MQHCONN    Hconn;         /* Connection handle */
PMQHOBJS   pHobj;         /* Ptr to object handle */
MQLONG     Options;       /* Close options */
MQLONG     CompCode;      /* Completion code */
MQLONG     Reason;        /* Reason code */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
MQ_CLOSE_EXIT (&ExitParms, &ExitContext,&Hconn, &pHobj, &Options,
               &CompCode, &Reason);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
void MQENTRY MQ_CLOSE_EXIT (
PMQAXP      pExitParms,    /* Address of exit parameter structure */
PMQAXC      pExitContext,  /* Address of exit context structure */
PMQHCONN    pHconn,       /* Address of connection handle */
PPMHOBJS    ppHobj,       /* Address of ptr to object handle */
PMQLONG     pOptions,      /* Address of close options */
PMQLONG     pCompCode,     /* Address of completion code */
PMQLONG     pReason);     /* Address of reason code qualifying
                           completion code */
```

## 커밋 - MQ\_CMIT\_EXIT

MQ\_CMIT\_EXIT는 사전 및 사후 커밋 처리를 수행하는 커밋 엑시트 함수를 제공합니다. 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_CMIT를 사용하여 사전 및 사후 약속 호출 엑시트 함수를 등록하십시오.

커밋 조작이 실패하고 트랜잭션이 백아웃되면 MQCC\_WARNING 및 MQRC\_BACKED\_OUT이 표시되면서 MQCMIT 호출이 실패합니다. 이 리턴 코드와 이유 코드가 사후 MQCMIT 엑시트 함수에 전달되어 엑시트 함수에 작업 단위가 백아웃되었음을 알려줍니다.

이 함수의 인터페이스는 다음과 같습니다.

```
MQ_CMIT_EXIT (&ExitParms, &ExitContext, &Hconn, &CompCode, &Reason)
```

여기서 매개변수는 다음과 같습니다.

#### **ExitParms (MQAXP) - 입출력(I/O)**

엑시트 매개변수 구조입니다.

#### **ExitContext (MQAXC) - 입출력(I/O)**

엑시트 컨텍스트 구조입니다.

#### **Hconn (MQHCONN) - 입력**

연결 핸들입니다.

#### **CompCode (MQLONG) - 입출력(I/O)**

완료 코드의 올바른 값은 다음과 같습니다.

##### **MQCC\_OK**

정상적으로 완료되었습니다.

##### **MQCC\_WARNING**

부분 완료입니다.

##### **MQCC\_FAILED**

호출에 실패했습니다.

#### **Reason (MQLONG) - 입출력(I/O)**

완료 코드를 규정하는 이유 코드입니다.

완료 코드가 MQCC\_OK인 경우 유일하게 올바른 값은 다음과 같습니다.

##### **MQRC\_NONE**

(0, x'000') 보고할 이유가 없습니다.

완료 코드가 MQCC\_FAILED 또는 MQCC\_WARNING인 경우 엑시트 함수는 이유 코드 필드를 올바른 MQRC\_\* 값으로 설정할 수 있습니다.

## **C 언어 호출**

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP    ExitParms;      /* Exit parameter structure */
MQAXC    ExitContext;    /* Exit context structure */
MQHCONN  Hconn;          /* Connection handle */
MQLONG   CompCode;       /* Completion code */
MQLONG   Reason;         /* Reason code qualifying completion code */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
MQ_CMIT_EXIT (&ExitParms, &ExitContext, &Hconn, &CompCode, &Reason);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
void MQENTRY MQ_CMIT_EXIT (
PMQAXP    pExitParms,      /* Address of exit parameter structure */
PMQAXC    pExitContext,    /* Address of exit context structure */
PMQHCONN  pHconn,          /* Address of connection handle */
PMQLONG   pCompCode,       /* Address of completion code */
PMQLONG   pReason);        /* Address of reason code qualifying completion
                             code */
```

## **사용법 참고**

1. 여기에 설명된 MQ\_GET\_EXIT 함수 인터페이스는 MQXF\_GET 엑시트 함수 및 [1516 페이지의 『MQXF DATA CONV ON GET』](#) 엑시트 함수에 사용됩니다.

두 엑시트 함수에 별도의 시작점이 정의되므로 둘 다 인터셉트하려면 MQXEP 호출을 두 번 사용해야 합니다. 이 호출에 대해 함수 ID MQXF\_GET을 사용하십시오.

MQ\_GET\_EXIT 인터페이스는 MQXF\_GET 및 MQXF\_DATA\_CONV\_ON\_GET에 대해 동일하기 때문에 단일 엑시트 함수를 두 가지 모두에 사용할 수 있습니다. MQAXP 구조의 *Function* 필드는 호출된 엑시트 함수를 표시합니다. 또는 MQXEP 호출을 사용하여 두 경우에 대해 다른 엑시트 함수를 등록할 수 있습니다.

## 연결 및 연결 확장 - MQ\_CONN\_EXIT

MQ\_CONN\_EXIT는 MQCONN 처리 전 및 후에 수행할 연결 엑시트 함수 및 MQCONNX 처리 전 및 후에 수행할 확장 엑시트 함수를 제공합니다.

여기에 설명된 동일한 인터페이스가 MQCONN 및 MQCONNX 호출 엑시트 함수에 대해 호출됩니다.

메시지 채널 에이전트(MCA)가 인바운드 클라이언트 연결에 응답하는 경우, 클라이언트 상태를 완전히 알기 전에 MCA는 연결하고 다수의 IBM MQ API 호출을 수행할 수 있습니다. 이 API 호출은 MCA 프로그램 기반의 MQAXC와 함께 API 엑시트 함수를 호출합니다(예: MQAXC의 UerId 및 ConnectionName 필드).

MCA가 후속 인바운드 클라이언트 API 호출에 응답할 때 MQAXC 구조는 인바운드 클라이언트를 기반으로 하고 UserId 및 ConnectionName 필드를 적절하게 설정합니다.

MQCONN 또는 MQCONNX 호출에서 애플리케이션이 설정한 큐 관리자 이름이 기본 연결 호출에 전달됩니다. 사전 MQ\_CONN\_EXIT에서 큐 관리자의 이름을 변경하려고 하지만 아무런 영향이 없습니다.

사전 및 사후 MQCONN 및 MQCONNX 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_CONN 및 MQXF\_CONNXX를 사용하십시오.

현재 올바른 환경이 설정되지 않았으므로 이유 MQXR\_BEFORE에 대해 호출된 MQ\_CONN\_EXIT 엑시트는 IBM MQ API 호출을 발행하지 않아야 합니다.

MQ\_CONN\_EXIT는 호출하는 연결에 대해 API 엑시트 호출에서 MQDISC를 호출할 수 없습니다. 이 제한이 클라이언트 및 서버 API 엑시트 모두에 적용됩니다.

MQCONN 및 MQCONNX의 인터페이스는 동일합니다.

```
MQ_CONN_EXIT (&ExitParms, &ExitContext, &pQMgrName, &pConnectOpts,  
&pHconn, &CompCode, &Reason);
```

여기서 매개변수는 다음과 같습니다.

### ExitParms (MQAXP) - 입출력(I/O)

엑시트 매개변수 구조입니다.

### ExitContext (MQAXC) - 입출력(I/O)

엑시트 컨텍스트 구조입니다.

### pQMgrName (PMQCHAR) - 입력

MQCONNX 호출에 제공된 큐 관리자 이름에 대한 포인터. 엑시트가 MQCONN 또는 MQCONNX 호출에서 이 이름을 변경하지 않아야 합니다.

### pConnectOpts (PMQCNO) - 입출력(I/O)

MQCONNX 호출의 조치를 제어하는 옵션에 대한 포인터.

자세한 내용은 312 페이지의 『MQCNO - 연결 옵션』의 내용을 참조하십시오.

엑시트 함수 MQXF\_CONN의 경우, pConnectOpts는 기본 연결 옵션 구조(MQCNO\_DEFAULT)를 가리킵니다.

### pHconn (PMQHCONN) - 입력

연결 핸들에 대한 포인터.

### CompCode (MQLONG) - 입출력(I/O)

완료 코드의 올바른 값은 다음과 같습니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_WARNING

경고(일부 완료)

#### MQCC\_FAILED

호출에 실패했습니다.

## Reason (MQLONG) - 입출력(I/O)

완료 코드를 규정하는 이유 코드입니다.

완료 코드가 MQCC\_OK인 경우 유일하게 올바른 값은 다음과 같습니다.

### MQRC\_NONE

(0, x'000') 보고할 이유가 없습니다.

완료 코드가 MQCC\_FAILED 또는 MQCC\_WARNING인 경우 엑시트 함수는 이유 코드 필드를 올바른 MQRC\_\* 값으로 설정할 수 있습니다.

## C 언어 호출

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP      ExitParms;      /* Exit parameter structure */
MQAXC      ExitContext;    /* Exit context structure */
PMQCHAR    pQMgrName;     /* Ptr to Queue manager name */
PMQCN0     pConnectOpts;  /* Ptr to Connection options */
PMQHCONN   pHconn;       /* Ptr to Connection handle */
MQLONG     CompCode;      /* Completion code */
MQLONG     Reason;        /* Reason code */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
MQ_CONNX_EXIT (&ExitParms, &ExitContext, &pQMgrName, &pConnectOpts,
               &pHconn, &CompCode, &Reason);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
void MQENTRY MQ_CONNX_EXIT (
PMQAXP      pExitParms,    /* Address of exit parameter structure */
PMQAXC      pExitContext,  /* Address of exit context structure */
PPMQCHAR    ppQMgrName,   /* Address of ptr to queue manager name */
PPMQCN0     ppConnectOpts, /* Address of ptr to connection options */
PPMQHCONN   ppHconn,      /* Address of ptr to connection handle */
PMQLONG     pCompCode,    /* Address of completion code */
PMQLONG     pReason);     /* Address of reason code qualifying
                           completion code */
```

## 사용법 참고

- 여기에 설명된 MQ\_CONNX\_EXIT 함수 인터페이스는 MQCONN 호출 및 MQCONNX 호출에 사용됩니다. 그러나 이들 두 호출에는 별도의 시작점이 정의됩니다. 두 호출을 인터셉트하려면 함수 ID MQXF\_CONN 및 MQXF\_CONN에 각각 한 번씩 MQXEP 호출을 두 번 이상 사용해야 합니다.  
MQ\_CONNX\_EXIT 인터페이스가 MQCONN 및 MQCONNX에 동일하기 때문에 두 호출에 하나의 엑시트 함수를 사용할 수 있습니다. MQAXP 구조의 *Function* 필드는 진행 중인 호출을 표시합니다. 또는 MQXEP 호출을 사용하여 두 호출에 대해 다른 엑시트 함수를 등록할 수 있습니다.
- 메시지 채널 에이전트(MCA)가 인바운드 클라이언트 연결에 응답하는 경우, 클라이언트 상태를 완전히 알기 전에 MCA는 다수의 MQ 호출을 발행할 수 있습니다. 이러한 MQ 호출에서는 API 엑시트 함수가 클라이언트 관련 데이터(예: 사용자 ID 또는 연결 이름)가 아닌 MCA 관련 데이터를 포함하는 MQAXC 구조로 호출되게 됩니다. 그러나, 클라이언트 상태가 완전히 알려지고 나면 후속 MQ 호출에서 API 엑시트 함수가 MQAXC 구조의 적절한 클라이언트 데이터와 함께 호출되게 됩니다.
- 모든 MQXR\_BEFORE 엑시트 함수는 큐 관리자가 매개변수 유효성 검증을 수행하기 전에 호출됩니다. 그러므로 매개변수가 올바르지 않을 수 있습니다(올바르지 않은 매개변수 주소의 포인터 포함).  
큐 관리자가 권한 검사를 수행하기 전에 MQ\_CONNX\_EXIT 함수가 호출됩니다.
- 엑시트 함수가 MQCONN 또는 MQCONNX 호출에 지정된 큐 관리자의 이름을 변경하지 않아야 합니다. 엑시트 함수에 의해 이름이 변경되면 결과가 정의되지 않습니다.
- MQ\_CONNX\_EXIT의 MQXR\_BEFORE 엑시트 함수는 MQXEP 이외의 MQ 호출을 발행할 수 없습니다.

## 콜백 제어 - MQ\_CTL\_EXIT

MQ\_CTL\_EXIT는 사전 및 사후 콜백 처리 제어를 수행하는 구독 요청 엑시트 함수를 제공합니다. 사전 및 사후 콜백 제어 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_CTL을 사용하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
MQ_CTL_EXIT (&Hconn, &Operation, &ControlOpts, &CompCode, &Reason)
```

여기서 매개변수는 다음과 같습니다.

### Hconn(MQHCONN) - 입출력(I/O)

연결 핸들입니다.

### Operation (MQLONG) 입출력(I/O)

지정된 오브젝트 핸들에 정의된 콜백에서 처리 중인 조작

### ControlOpts (MQCTLO) 입출력(I/O)

MQCTL의 조치를 제어하는 옵션

### CompCode (MQLONG) - 입출력(I/O)

완료 코드의 올바른 값은 다음과 같습니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_WARNING

부분 완료입니다.

#### MQCC\_FAILED

호출에 실패했습니다.

### Reason (MQLONG) - 입출력(I/O)

완료 코드를 규정하는 이유 코드입니다.

완료 코드가 MQCC\_OK인 경우 유일하게 올바른 값은 다음과 같습니다.

#### MQRC\_NONE

(0, x'000') 보고할 이유가 없습니다.

완료 코드가 MQCC\_FAILED 또는 MQCC\_WARNING인 경우 엑시트 함수는 이유 코드 필드를 올바른 MQRC\_\* 값으로 설정할 수 있습니다.

## C 언어 호출

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQHCONN  Hconn;          /* Connection handle */
MQLONG   Operation;     /* Operation being processed */
MQCTLO   ControlOpts;  /* Options that control the action of MQCTL */
MQLONG   CompCode;     /* Completion code */
MQLONG   Reason;       /* Reason code qualifying completion code */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
MQ_CTL_EXIT (&Hconn, &Operation, &ControlOpts, &CompCode, &Reason);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
void MQENTRY MQ_CTL_EXIT (
PMQHCONN  pHconn;      /* Address of connection handle */
PMQLONG   pOperation;  /* Address of operation being processed */
PMQCTLO   pControlOpts; /* Address of options that control the action of MQCTL */
PMQLONG   pCompCode;   /* Address of completion code */
PMQLONG   pReason;     /* Address of reason code qualifying completion code */
```

## 연결 끊기 - MQ\_DISC\_EXIT

MQ\_DISC\_EXIT는 사전 및 사후 MQDISC 엑시트 처리를 수행하는 연결 끊기 엑시트 함수를 제공합니다. 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_DISC를 사용하여 사전 및 사후 MQDISC 호출 엑시트 함수를 등록하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
MQ_DISC_EXIT (&ExitParms, &ExitContext, &pHconn,  
&CompCode, &Reason);
```

여기서 매개변수는 다음과 같습니다.

### ExitParms (MQAXP) - 입출력(I/O)

엑시트 매개변수 구조입니다.

### ExitContext (MQAXC) - 입출력(I/O)

엑시트 컨텍스트 구조입니다.

### pHconn (PMQHCONN) - 입력

연결 핸들에 대한 포인터.

사전 MQDISC 호출의 경우, 이 필드의 값은 다음 중 하나입니다.

- MQCONN 또는 MQCONNX 호출에 리턴된 연결 핸들
- 환경 고유 어댑터가 큐 관리자에 연결된 환경의 경우, 0
- 이전 엑시트 함수 호출이 설정한 값

사후 MQDISC 호출의 경우, 이 필드의 값은 0 또는 이전 엑시트 함수 호출이 설정한 값입니다.

### CompCode (MQLONG) - 입출력(I/O)

완료 코드의 올바른 값은 다음과 같습니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_WARNING

부분 완료

#### MQCC\_FAILED

호출에 실패했습니다.

### Reason (MQLONG) - 입출력(I/O)

완료 코드를 규정하는 이유 코드입니다.

완료 코드가 MQCC\_OK인 경우 유일하게 올바른 값은 다음과 같습니다.

#### MQRC\_NONE

(0, x'000') 보고할 이유가 없습니다.

완료 코드가 MQCC\_FAILED 또는 MQCC\_WARNING인 경우 엑시트 함수는 이유 코드 필드를 올바른 MQRC\_\* 값으로 설정할 수 있습니다.

## C 언어 호출

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP      ExitParms;      /* Exit parameter structure */  
MQAXC      ExitContext;    /* Exit context structure */  
PMQHCONN   pHconn;        /* Ptr to Connection handle */  
MQLONG     CompCode;      /* Completion code */  
MQLONG     Reason;        /* Reason code */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
MQ_DISC_EXIT (&ExitParms, &ExitContext, &pHconn,
              &CompCode, &Reason);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
void MQENTRY MQ_DISC_EXIT (
  PMQAXP      pExitParms,      /* Address of exit parameter structure */
  PMQAXC      pExitContext,    /* Address of exit context structure */
  PPMQHCONN   ppHconn,        /* Address of ptr to connection handle */
  PMQLONG     pCompCode,      /* Address of completion code */
  PMQLONG     pReason);       /* Address of reason code qualifying
                               completion code */
```

## 가져오기 - MQ\_GET\_EXIT

MQ\_GET\_EXIT는 사전 및 사후 MQGET 호출 처리를 수행하기 위한 get exit 함수를 제공합니다.

다음과 같은 두 가지 함수 ID가 있습니다.

1. 사전 및 사후 MQGET 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 MQXF\_GET을 사용하십시오.
2. MQXF\_DATA\_CONV\_ON\_GET 함수 ID에 대한 자세한 정보는 [1516 페이지의 『MQXF DATA CONV ON GET』](#)의 내용을 참조하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
MQ_GET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
             &pGetMsgOpts, &BufferLength, &pBuffer, &pDataLength,
             &CompCode, &Reason)
```

여기서 매개변수는 다음과 같습니다.

### ExitParms (MQAXP) - 입출력(I/O)

엑시트 매개변수 구조입니다.

### ExitContext (MQAXC) - 입출력(I/O)

엑시트 컨텍스트 구조입니다.

### Hconn (MQHCONN) - 입력

연결 핸들입니다.

### Hobj (MQHOBJ) - 입출력(I/O)

오브젝트 핸들.

### pMsgDesc (PMQMD) - 입출력(I/O)

메시지 디스크립터에 대한 포인터.

### pGetMsgOpts (PMQMO) - 입출력(I/O)

메시지 가져오기 옵션에 대한 포인터.

### BufferLength (MQLONG) - 입출력(I/O)

메시지 버퍼 길이.

### pBuffer (PMQBYTE) - 입출력(I/O)

메시지 버퍼에 대한 포인터.

### pDataLength (PMQLONG) - 입출력(I/O)

데이터 길이 필드에 대한 포인터.

### CompCode (MQLONG) - 입출력(I/O)

완료 코드의 올바른 값은 다음과 같습니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_WARNING

부분 완료입니다.

## MQCC\_FAILED

호출에 실패했습니다.

### Reason (MQLONG) - 입출력(I/O)

완료 코드를 규정하는 이유 코드입니다.

완료 코드가 MQCC\_OK인 경우 유일하게 올바른 값은 다음과 같습니다.

## MQRC\_NONE

(0, x'000') 보고할 이유가 없습니다.

완료 코드가 MQCC\_FAILED 또는 MQCC\_WARNING인 경우 엑시트 함수는 이유 코드 필드를 올바른 MQRC\_\* 값으로 설정할 수 있습니다.

## C 언어 호출

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP      ExitParms;      /* Exit parameter structure */
MQAXC      ExitContext;    /* Exit context structure */
MQHCONN    Hconn;         /* Connection handle */
MQHOBJ     Hobj;          /* Object handle */
PMQMD      pMsgDesc;      /* Ptr to message descriptor */
PMQPMO     pGetMsgOpts;   /* Ptr to get message options */
MQLONG     BufferLength;   /* Message buffer length */
PMQBYTE    pBuffer;      /* Ptr to message buffer */
PMQLONG    pDataLength;   /* Ptr to data length field */
MQLONG     CompCode;     /* Completion code */
MQLONG     Reason;       /* Reason code */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
MQ_GET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
             &pGetMsgOpts, &BufferLength, &pBuffer, &pDataLength,
             &CompCode, &Reason)
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
void MQENTRY MQ_GET_EXIT (
PMQAXP      pExitParms,    /* Address of exit parameter structure */
PMQAXC      pExitContext,  /* Address of exit context structure */
PMQHCONN    pHconn,       /* Address of connection handle */
PMQHOBJ     pHobj,        /* Address of object handle */
PPMQMD      ppMsgDesc,    /* Address of ptr to message descriptor */
PPMQGMO     ppGetMsgOpts, /* Address of ptr to get message options */
PMQLONG     pBufferLength, /* Address of message buffer length */
PPMQBYTE    ppBuffer,     /* Address of ptr to message buffer */
PMQLONG     ppDataLength, /* Address of ptr to data length field */
PMQLONG     pCompCode,    /* Address of completion code */
PMQLONG     pReason);    /* Address of reason code qualifying
                           completion code */
```

## 사용법 참고

- 여기에 설명된 MQ\_GET\_EXIT 함수 인터페이스는 MQXF\_GET 엑시트 함수 및 [1516 페이지의 『MQXF DATA CONV ON GET』](#) 엑시트 함수에 사용됩니다.

두 엑시트 함수에 별도의 시작점이 정의되므로 둘 다 인터셉트하려면 MQXEP 호출을 두 번 사용해야 합니다. 이 호출에 대해 함수 ID MQXF\_GET을 사용하십시오.

MQ\_GET\_EXIT 인터페이스는 MQXF\_GET 및 MQXF\_DATA\_CONV\_ON\_GET에 대해 동일하기 때문에 단일 엑시트 함수를 두 가지 모두에 사용할 수 있습니다. MQAXP 구조의 *Function* 필드는 호출된 엑시트 함수를 표시합니다. 또는 MQXEP 호출을 사용하여 두 경우에 대해 다른 엑시트 함수를 등록할 수 있습니다.

## MQXF\_DATA\_CONV\_ON\_GET

MQXF\_DATA\_CONV\_ON\_GET 함수 ID는 MQ\_GET\_EXIT와 함께 사용됩니다.



이 호출에 대한 인터페이스 정보와 샘플 C 언어 선언은 MQ\_GET\_EXIT의 내용을 참조하십시오.

## 사용법 참고

등록된 경우, 메시지가 애플리케이션에 도착할 때 데이터 변환이 발생하기 전에 이 시작점이 호출됩니다. 이 기능은 메시지가 데이터 변환에 전달되기 전에 API 엑시트가 복호화나 압축 해제와 같은 처리를 수행해야 하는 경우 유용할 수 있습니다. 필요하다면, 엑시트는 MQXCC\_SUPPRESS\_FUNCTION을 리턴하여 데이터 변환이 무시되도록 합니다. 자세한 정보는 MQAXP 구조를 참조하십시오.

클라이언트에서 이 시작점에 등록하면 클라이언트 시스템의 로컬에서 데이터 변환이 수행되는 효과가 있습니다. 따라서 올바른 조작을 위해 애플리케이션 변환 엑시트를 클라이언트에 설치하는 것이 필요할 수 있습니다. MQXF\_DATA\_CONV\_ON\_GET은 비동기 이용에도 사용됩니다.

MQ\_GET\_EXIT 호출을 사용할 때 엑시트 이유 MQXR\_BEFORE와 함께 MQXF\_DATA\_CONV\_ON\_GET을 사용하여 사전 MQGET 데이터 변환 엑시트 함수를 등록하십시오.

MQXF\_DATA\_CONV\_ON\_GET에 대한 MQXR\_AFTER 엑시트 함수가 없습니다. MQXF\_GET의 MQXR\_AFTER 엑시트 함수는 데이터 변환 후 엑시트 처리에 필요한 기능을 제공합니다.

MQ\_GET\_EXIT 호출에 별도의 시작점이 정의되므로 두 엑시트 함수를 인터셉트하려면 MQXEP 호출을 두 번 사용해야 합니다. 이 호출에 대해 함수 ID MQXF\_DATA\_CONV\_ON\_GET을 사용하십시오.

MQ\_GET\_EXIT 인터페이스는 MQXF\_GET 및 MQXF\_DATA\_CONV\_ON\_GET에 대해 동일하기 때문에 단일 엑시트 함수를 두 가지 모두에 사용할 수 있습니다. MQAXP 구조의 *Function* 필드는 호출된 엑시트 함수를 표시합니다. 또는 MQXEP 호출을 사용하여 두 경우에 대해 다른 엑시트 함수를 등록할 수 있습니다.

## 초기화 - MQ\_INIT\_EXIT

MQ\_INIT\_EXIT에서는 MQAXP에서 ExitReason을 MQXR\_CONNECTION으로 설정하여 표시되는 연결 레벨 초기화를 제공합니다.

초기화하는 동안 다음에 유의하십시오.

- MQ\_INIT\_EXIT 함수가 MQXEP를 호출하여 IBM MQ API verb와 관심 있는 ENTRY 및 EXIT 지점을 등록합니다.
- 엑시트가 모든 IBM MQ API verb를 인터셉트할 필요는 없습니다. 엑시트 함수는 관심 대상으로 등록된 경우에만 호출됩니다.
- 엑시트에 사용되는 스토리지는 초기화하는 동안 확보할 수 있습니다.
- 이 함수에 대한 호출에 실패하면 MQAXP의 ExitResponse 필드 값에 따라 결정되는 CompCode 및 이유가 표시되면서 이를 호출한 MQCONN 또는 MQCONNX 호출이 실패합니다.
- 현재 올바른 환경이 설정되지 않았으므로 MQ\_INIT\_EXIT 엑시트가 IBM MQ API 호출을 발행하지 않아야 합니다.
- MQ\_INIT\_EXIT가 MQXCC\_FAILED와 함께 실패하면 MQCC\_FAILED 및 MQRC\_API\_EXIT\_ERROR가 표시되면서 큐 관리자가 이를 호출한 MQCONN 또는 MQCONNX 호출에서 리턴됩니다.
- 첫 번째 MQ\_INIT\_EXIT를 호출하기 전에 API 엑시트 함수 실행 환경을 초기화하는 동안 큐 관리자에 오류가 발생하면 MQCC\_FAILED 및 MQRC\_API\_EXIT\_INIT\_ERROR가 표시되면서 큐 관리자가 MQ\_INIT\_EXIT를 호출한 MQCONN 또는 MQCONNX 호출에서 리턴됩니다.

MQ\_INIT\_EXIT의 인터페이스는 다음과 같습니다.

```
MQ_INIT_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

여기서 매개변수는 다음과 같습니다.

### ExitParms (MQAXP) - 입출력(I/O)

엑시트 매개변수 구조입니다.

### ExitContext (MQAXC) - 입출력(I/O)

엑시트 컨텍스트 구조입니다.

### CompCode (MQLONG) - 입출력(I/O)

완료 코드에 대한 포인터이며, 올바른 값은 다음과 같습니다.

### **MQCC\_OK**

정상적으로 완료되었습니다.

### **MQCC\_WARNING**

부분 완료입니다.

### **MQCC\_FAILED**

호출에 실패했습니다.

### **Reason (MQLONG) - 입출력(I/O)**

완료 코드를 규정하는 이유 코드에 대한 포인터.

완료 코드가 MQCC\_OK인 경우 유일하게 올바른 값은 다음과 같습니다.

### **MQRC\_NONE**

(0, x'000') 보고할 이유가 없습니다.

완료 코드가 MQCC\_FAILED 또는 MQCC\_WARNING인 경우 엑시트 함수는 이유 코드 필드를 올바른 MQRC\_\* 값으로 설정할 수 있습니다.

애플리케이션에 리턴되는 CompCode 및 Reason은 MQAXP의 ExitResponse 필드 값에 따라 다릅니다.

## **C 언어 호출**

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```

MQAXP      ExitParms;      /* Exit parameter structure */
MQAXC      ExitContext;    /* Exit context structure */
MQLONG     CompCode;       /* Completion code */
MQLONG     Reason;         /* Reason code */

```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
MQ_INIT_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```

void MQENTRY MQ_INIT_EXIT (
PMQAXP      pExitParms,    /* Address of exit parameter structure */
PMQAXC      pExitContext,  /* Address of exit context structure */
PMQLONG     pCompCode,     /* Address of completion code */
PMQLONG     pReason);      /* Address of reason code qualifying
                             completion code */

```

## **사용법 참고**

1. MQ\_INIT\_EXIT 함수는 MQXEP 호출을 발행하여 인터셉트할 특정 MQ 호출에 대해 엑시트 함수의 주소를 등록할 수 있습니다. 모든 MQ 호출을 인터셉트하거나 MQXR\_BEFORE 및 MQXR\_AFTER 호출을 둘 다 인터셉트할 필요는 없습니다. 예를 들어, 엑시트 스위트는 MQPUT의 MQXR\_BEFORE 호출만 인터셉트하도록 선택할 수 있습니다.
2. 엑시트 스위트의 엑시트 함수에 사용되는 스토리지는 MQ\_INIT\_EXIT 함수를 통해 확보할 수 있습니다. 또는, 필요할 때 엑시트 함수가 호출되면 엑시트 함수가 스토리지를 확보할 수 있습니다. 단, 엑시트 스위트가 종료되기 전에 모든 스토리지를 비워야 합니다. MQ\_TERM\_EXIT 함수가 스토리지 또는 이전에 호출된 엑시트 함수를 비울 수 있습니다.
3. MQ\_INIT\_EXIT가 MQAXP의 ExitResponse 필드에 MQXCC\_FAILED를 리턴하거나 어떤 식으로든 실패하면 MQ\_INIT\_EXIT 호출을 유발한 MQCONN 또는 MQCONNx 호출도 실패하고 **CompCode** 및 **Reason** 매개 변수가 적절한 값으로 설정됩니다.
4. MQ\_INIT\_EXIT 함수는 MQXEP 이외의 MQ 호출을 발행할 수 없습니다.

## 조회 - MQ\_INQ\_EXIT

MQ\_INQ\_EXIT는 사전 및 사후 MQINQ 호출 처리를 수행하는 조회 엑시트 함수를 제공합니다. 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_INQ를 사용하여 사전 및 사후 MQINQ 호출 엑시트 함수를 등록하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
MQ_INQ_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,  
            &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,  
            &pCharAttrs, &CompCode, &Reason)
```

여기서 매개변수는 다음과 같습니다.

### ExitParms (MQAXP) - 입출력(I/O)

엑시트 매개변수 구조입니다.

### ExitContext (MQAXC) - 입출력(I/O)

엑시트 컨텍스트 구조입니다.

### Hconn (MQHCONN) - 입력

연결 핸들입니다.

### Hobj (MQHOBJ) - 입력

오브젝트 핸들.

### SelectorCount (MQLONG) - 입력

선택자 수

### pSelectors (PMQLONG) - 입출력(I/O)

선택자 값 배열에 대한 포인터.

### IntAttrCount (MQLONG) - 입력

정수 속성의 수.

### pIntAttrs (PMQLONG) - 입출력(I/O)

정수 속성 값 배열에 대한 포인터.

### CharAttrLength (MQLONG) - 입출력(I/O)

문자 속성 배열 길이.

### pCharAttrs (PMQCHAR) - 입출력(I/O)

문자 속성 배열에 대한 포인터.

### CompCode (MQLONG) - 입출력(I/O)

완료 코드의 올바른 값은 다음과 같습니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_WARNING

부분 완료입니다.

#### MQCC\_FAILED

호출에 실패했습니다.

### Reason (MQLONG) - 입출력(I/O)

완료 코드를 규정하는 이유 코드입니다.

완료 코드가 MQCC\_OK인 경우 유일하게 올바른 값은 다음과 같습니다.

#### MQRC\_NONE

(0, x'000') 보고할 이유가 없습니다.

완료 코드가 MQCC\_FAILED 또는 MQCC\_WARNING인 경우 엑시트 함수는 이유 코드 필드를 올바른 MQRC\_\* 값으로 설정할 수 있습니다.

## C 언어 호출

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```

MQAXP    ExitParms;        /* Exit parameter structure */
MQAXC    ExitContext;     /* Exit context structure */
MQHCONN  Hconn;          /* Connection handle */
MQHOBJ   Hobj;           /* Object handle */
MQLONG   SelectorCount;  /* Count of selectors */
PMQLONG  pSelectors;     /* Ptr to array of attribute selectors */
MQLONG   IntAttrCount;   /* Count of integer attributes */
PMQLONG  pIntAttrs;     /* Ptr to array of integer attributes */
MQLONG   CharAttrLength; /* Length of char attributes array */
PMQCHAR  pCharAttrs;    /* Ptr to character attributes */
MQLONG   CompCode;      /* Completion code */
MQLONG   Reason;        /* Reason code qualifying completion code */

```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```

MQ_INQ_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
             &pCharAttrs, &CompCode, &Reason)

```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```

void MQENTRY MQ_INQ_EXIT (
PMQAXP    pExitParms,      /* Address of exit parameter structure */
PMQAXC    pExitContext,    /* Address of exit context structure */
PMQHCONN  pHconn,         /* Address of connection handle */
PMQHOBJ   pHobj,          /* Address of object handle */
PMQLONG   pSelectorCount, /* Address of selector count */
PPMQLONG  ppSelectors,    /* Address of ptr to array of selectors */
PMQLONG   pIntAttrCount;  /* Address of count of integer attributes */
PPMQLONG  ppIntAttrs,     /* Address of ptr to array of integer attributes */
PMQLONG   pCharAttrLength, /* Address of character attribute length */
PPMQCHAR  ppCharAttrs,    /* Address of ptr to character attributes array */
PMQLONG   pCompCode,      /* Address of completion code */
PMQLONG   pReason);       /* Address of reason code qualifying completion
                           code */

```

## 열기 - MQ\_OPEN\_EXIT

MQ\_OPEN\_EXIT는 사전 및 사후 MQOPEN 호출 처리를 수행하는 열기 엑시트 함수를 제공합니다. 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_OPEN을 사용하여 사전 및 사후 MQOPEN 호출 엑시트 함수를 등록하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```

MQ_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &Options,
              &pHobj, &CompCode, &Reason)

```

여기서 매개변수는 다음과 같습니다.

### ExitParms (MQAXP) - 입출력(I/O)

엑시트 매개변수 구조입니다.

### ExitContext (MQAXC) - 입출력(I/O)

엑시트 컨텍스트 구조입니다.

### Hconn (MQHCONN) - 입력

연결 핸들입니다.

### pObjDesc (PMQOD) - 입출력(I/O)

오브젝트 디스크립터에 대한 포인터.

### Options (MQLONG)- 입출력(I/O)

열기 옵션.

### pHobj (PMQHOBJ) - 입력

오브젝트 핸들에 대한 포인터.

### CompCode (MQLONG) - 입출력(I/O)

완료 코드의 올바른 값은 다음과 같습니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_WARNING**

부분 완료

**MQCC\_FAILED**

호출에 실패했습니다.

**Reason (MQLONG) - 입출력(I/O)**

완료 코드를 규정하는 이유 코드입니다.

완료 코드가 MQCC\_OK인 경우 유일하게 올바른 값은 다음과 같습니다.

**MQRC\_NONE**

(0, x'000') 보고할 이유가 없습니다.

완료 코드가 MQCC\_FAILED 또는 MQCC\_WARNING인 경우 엑시트 함수는 이유 코드 필드를 올바른 MQRC\_\* 값으로 설정할 수 있습니다.

**C 언어 호출**

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP      ExitParms;      /* Exit parameter structure */
MQAXC      ExitContext;    /* Exit context structure */
MQHCONN    Hconn;         /* Connection handle */
PMQOD      pObjDesc;      /* Ptr to object descriptor */
MQLONG     Options;       /* Open options */
PMQHOBJS   pHobj;        /* Ptr to object handle */
MQLONG     CompCode;      /* Completion code */
MQLONG     Reason;       /* Reason code */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
MQ_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &Options,
              &pHobj, &CompCode, &Reason);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
void MQENTRY MQ_OPEN_EXIT (
PMQAXP      pExitParms,    /* Address of exit parameter structure */
PMQAXC      pExitContext,  /* Address of exit context structure */
PMQHCONN    pHconn,       /* Address of connection handle */
PPMQOD      ppObjDesc,    /* Address of ptr to object descriptor */
PMQLONG     pOptions,     /* Address of open options */
PPMQHOBJS   ppHobj,       /* Address of ptr to object handle */
PMQLONG     pCompCode,    /* Address of completion code */
PMQLONG     pReason);     /* Address of reason code qualifying
                             completion code */
```

**넣기 - MQ\_PUT\_EXIT**

MQ\_PUT\_EXIT는 사전 및 사후 MQPUT 호출 처리를 수행하는 넣기 엑시트 함수를 제공합니다. MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_PUT를 사용하여 사전 및 사후 MQPUT 호출 엑시트 함수를 등록하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
MQ_PUT_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
            &pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)
```

여기서 매개변수는 다음과 같습니다.

**ExitParms (MQAXP) - 입출력(I/O)**

엑시트 매개변수 구조입니다.

**ExitContext (MQAXC) - 입출력(I/O)**

엑시트 컨텍스트 구조입니다.

**Hconn (MQHCONN) - 입력**

연결 핸들입니다.

**Hobj (MQHOBJ) - 입출력(I/O)**

오브젝트 핸들.

**pMsgDesc (PMQMD) - 입출력(I/O)**

메시지 디스크립터에 대한 포인터.

**pPutMsgOpts (PMQPMO) - 입출력(I/O)**

메시지 넣기 옵션에 대한 포인터.

**BufferLength (MQLONG) - 입출력(I/O)**

메시지 버퍼 길이.

**pBuffer (PMQBYTE) - 입출력(I/O)**

메시지 버퍼에 대한 포인터.

**CompCode (MQLONG) - 입출력(I/O)**

완료 코드의 올바른 값은 다음과 같습니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_WARNING**

부분 완료입니다.

**MQCC\_FAILED**

호출에 실패했습니다.

**Reason (MQLONG) - 입출력(I/O)**

완료 코드를 규정하는 이유 코드입니다.

완료 코드가 MQCC\_OK인 경우 유일하게 올바른 값은 다음과 같습니다.

**MQRC\_NONE**

(0, x'000') 보고할 이유가 없습니다.

완료 코드가 MQCC\_FAILED 또는 MQCC\_WARNING인 경우 엑시트 함수는 이유 코드 필드를 올바른 MQRC\_\* 값으로 설정할 수 있습니다.

**C 언어 호출**

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```

MQAXP      ExitParms;      /* Exit parameter structure */
MQAXC      ExitContext;    /* Exit context structure */
MQHCONN    Hconn;         /* Connection handle */
MQHOBJ     Hobj;          /* Object handle */
PMQMD      pMsgDesc;      /* Ptr to message descriptor */
PMQPMO     pPutMsgOpts;   /* Ptr to put message options */
MQLONG     BufferLength;   /* Message buffer length */
PMQBYTE    pBuffer;      /* Ptr to message data */
MQLONG     CompCode;      /* Completion code */
MQLONG     Reason;       /* Reason code */

```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```

MQ_PUT_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
             &pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)

```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```

void MQENTRY MQ_PUT_EXIT (
  PMQAXP      pExitParms,    /* Address of exit parameter structure */
  PMQAXC      pExitContext,  /* Address of exit context structure */

```

```

PMQHCONN      pHConn,          /* Address of connection handle */
PMQHOBJ       pHObj,           /* Address of object handle */
PMQMD         pMsgDesc,       /* Address of ptr to message descriptor */
PMQPMO        ppPutMsgOpts,   /* Address of ptr to put message options */
MQLONG        pBufferLength,  /* Address of message buffer length */
PMQBYTE       pBuffer,       /* Address of ptr to message buffer */
MQLONG        pCompCode,     /* Address of completion code */
MQLONG        pReason);     /* Address of reason code qualifying
                             completion code */

```

## 사용법 참고

- 큐 관리자에 의해 생성된 보고 메시지는 일반적인 호출 처리를 건너뛵니다. 따라서 이러한 메시지는 MQ\_PUT\_EXIT 함수 또는 MQPUT1 함수에 의해 인터셉트될 수 없습니다. 그러나 메시지 채널 에이전트에 의해 생성된 보고 메시지는 정상적으로 처리되므로 MQ\_PUT\_EXIT 함수 또는 MQ\_PUT1\_EXIT 함수로 인터셉트할 수 있습니다. MCA에 의해 생성된 모든 보고 메시지를 확실하게 인터셉트하려면 MQ\_PUT\_EXIT 및 MQ\_PUT1\_EXIT를 둘 다 사용해야 합니다.

### Put1 - MQ\_PUT1\_EXIT

MQ\_PUT1\_EXIT는 사전 및 사후 MQPUT1 호출 처리를 수행하는 메시지 하나만 넣기 엑시트 함수를 제공합니다. 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_PUT1을 사용하여 사전 및 사후 MQPUT1 호출 엑시트 함수를 등록하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```

MQ_PUT1_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &pMsgDesc,
&pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)

```

여기서 매개변수는 다음과 같습니다.

#### ExitParms (MQAXP) - 입출력(I/O)

엑시트 매개변수 구조입니다.

#### ExitContext (MQAXC) - 입출력(I/O)

엑시트 컨텍스트 구조입니다.

#### Hconn (MQHCONN) - 입력

연결 핸들입니다.

#### pObjDesc (PMQOD) - 입출력(I/O)

오브젝트 디스크립터에 대한 포인터.

#### pMsgDesc (PMQMD) - 입출력(I/O)

메시지 디스크립터에 대한 포인터.

#### pPutMsgOpts (PMQPMO) - 입출력(I/O)

메시지 넣기 옵션에 대한 포인터.

#### BufferLength (MQLONG) - 입출력(I/O)

메시지 버퍼 길이.

#### pBuffer (PMQBYTE) - 입출력(I/O)

메시지 버퍼에 대한 포인터.

#### CompCode (MQLONG) - 입출력(I/O)

완료 코드의 올바른 값은 다음과 같습니다.

##### MQCC\_OK

정상적으로 완료되었습니다.

##### MQCC\_WARNING

부분 완료입니다.

##### MQCC\_FAILED

호출에 실패했습니다.

#### Reason (MQLONG) - 입출력(I/O)

완료 코드를 규정하는 이유 코드입니다.

완료 코드가 MQCC\_OK인 경우 유일하게 올바른 값은 다음과 같습니다.

### **MQRC\_NONE**

(0, x'000') 보고할 이유가 없습니다.

완료 코드가 MQCC\_FAILED 또는 MQCC\_WARNING인 경우 엑시트 함수는 이유 코드 필드를 올바른 MQRC\_\* 값으로 설정할 수 있습니다.

## **C 언어 호출**

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP      ExitParms;      /* Exit parameter structure */
MQAXC      ExitContext;    /* Exit context structure */
MQHCONN    Hconn;         /* Connection handle */
PMQOD      pObjDesc;      /* Ptr to object descriptor */
PMQMD      pMsgDesc;      /* Ptr to message descriptor */
PMQPMO     pPutMsgOpts;   /* Ptr to put message options */
MQLONG     BufferLength;   /* Message buffer length */
PMQBYTE    pBuffer;      /* Ptr to message data */
MQLONG     CompCode;      /* Completion code */
MQLONG     Reason;       /* Reason code */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
MQ_PUT1_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &pMsgDesc,
              &pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
void MQENTRY MQ_PUT1_EXIT (
    PMQAXP      pExitParms,      /* Address of exit parameter structure */
    PMQAXC      pExitContext,    /* Address of exit context structure */
    PMQHCONN    pHconn,         /* Address of connection handle */
    PPMQOD      ppObjDesc,      /* Address of ptr to object descriptor */
    PPMQMD      ppMsgDesc,      /* Address of ptr to message descriptor */
    PPMQPMO     ppPutMsgOpts,   /* Address of ptr to put message options */
    PMQLONG     pBufferLength,   /* Address of message buffer length */
    PPMQBYTE    ppBuffer,       /* Address of ptr to message buffer */
    PMQLONG     pCompCode,      /* Address of completion code */
    PMQLONG     pReason);      /* Address of reason code qualifying
                                completion code */
```

## **설정 - MQ\_SET\_EXIT**

MQ\_SET\_EXIT는 사전 및 사후 MQSET 호출 처리를 수행하는 설정 엑시트 함수를 제공합니다. 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_SET를 사용하여 사전 및 사후 MQSET 호출 엑시트 함수를 등록하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
MQ_SET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
             &pCharAttr, &CompCode, &Reason)
```

여기서 매개변수는 다음과 같습니다.

### **ExitParms (MQAXP) - 입출력(I/O)**

엑시트 매개변수 구조입니다.

### **ExitContext (MQAXC) - 입출력(I/O)**

엑시트 컨텍스트 구조입니다.

### **Hconn (MQHCONN) - 입력**

연결 핸들입니다.

### **Hobj (MQHOBJ) - 입력**

오브젝트 핸들.



**SelectorCount (MQLONG) - 입력**

선택자 수

**pSelectors (PMQLONG) - 입출력(I/O)**

선택자 값 배열에 대한 포인터.

**IntAttrCount (MQLONG) - 입력**

정수 속성의 수.

**pIntAttrs (PMQLONG) - 입출력(I/O)**

정수 속성 값 배열에 대한 포인터.

**CharAttrLength (MQLONG) - 입출력(I/O)**

문자 속성 배열 길이.

**pCharAttrs (PMQCHAR) - 입출력(I/O)**

문자 속성 값에 대한 포인터.

**CompCode (MQLONG) - 입출력(I/O)**

완료 코드의 올바른 값은 다음과 같습니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_WARNING**

부분 완료입니다.

**MQCC\_FAILED**

호출에 실패했습니다.

**Reason (MQLONG) - 입출력(I/O)**

완료 코드를 규정하는 이유 코드입니다.

완료 코드가 MQCC\_OK인 경우 유일하게 올바른 값은 다음과 같습니다.

**MQRC\_NONE**

(0, x'000') 보고할 이유가 없습니다.

완료 코드가 MQCC\_FAILED 또는 MQCC\_WARNING인 경우 엑시트 함수는 이유 코드 필드를 올바른 MQRC\_\* 값으로 설정할 수 있습니다.

**C 언어 호출**

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```

MQAXP    ExitParms;        /* Exit parameter structure */
MQAXC    ExitContext;     /* Exit context structure */
MQHCONN  Hconn;           /* Connection handle */
MQHOBJ   Hobj;            /* Object handle */
MQLONG   SelectorCount;   /* Count of selectors */
PMQLONG  pSelectors;      /* Ptr to array of attribute selectors */
MQLONG   IntAttrCount;    /* Count of integer attributes */
PMQLONG  pIntAttrs;       /* Ptr to array of integer attributes */
MQLONG   CharAttrLength;  /* Length of char attributes array */
PMQCHAR  pCharAttrs;      /* Ptr to character attributes */
MQLONG   CompCode;        /* Completion code */
MQLONG   Reason;          /* Reason code qualifying completion code */

```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```

MQ_SET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
             &pCharAttrs, &CompCode, &Reason)

```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```

void MQENTRY MQ_SET_EXIT (
PMQAXP    pExitParms,      /* Address of exit parameter structure */
PMQAXC    pExitContext,    /* Address of exit context structure */
PMQHCONN  pHconn,         /* Address of connection handle */

```

```

PMQHOBj    pHobj,          /* Address of object handle */
PMQLONG    pSelectorCount, /* Address of selector count */
PPMQLONG   ppSelectors,    /* Address of ptr to array of selectors */
PMQLONG    pIntAttrCount;  /* Address of count of integer attributes */
PPMQLONG   ppIntAttrs,    /* Address of ptr to array of integer attributes */
PMQLONG    pCharAttrLength, /* Address of character attribute length */
PPMQCHAR   ppCharAttrs,   /* Address of ptr to character attributes array */
PMQLONG    pCompCode,     /* Address of completion code */
PMQLONG    pReason);     /* Address of reason code qualifying completion
                           code */

```

## 상태 - MQ\_STAT\_EXIT

MQ\_STAT\_EXIT는 사전 및 사후 MQSTAT 호출 처리를 수행하는 상태 엑시트 함수를 제공합니다. 사전 및 사후 MQSTAT 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_STAT를 사용하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```

MQ_STAT_EXIT (&ExitParms, &ExitContext, &Hconn, &Type, &pStatus
              &CompCode, &Reason)

```

여기서 매개변수는 다음과 같습니다.

### ExitParms (MQAXP) - 입출력(I/O)

엑시트 매개변수 구조입니다.

### ExitContext (MQAXC) - 입출력(I/O)

엑시트 컨텍스트 구조입니다.

### Hconn (MQHCONN) - 입력

연결 핸들입니다.

### Type (MQLONG) - 입력

검색할 상태 정보의 유형.

### pStatus (PMQSTS) - 출력

상태 버퍼에 대한 포인터.

### CompCode (MQLONG) - 입출력(I/O)

완료 코드의 올바른 값은 다음과 같습니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_WARNING

부분 완료입니다.

#### MQCC\_FAILED

호출에 실패했습니다.

### Reason (MQLONG) - 입출력(I/O)

완료 코드를 규정하는 이유 코드입니다.

완료 코드가 MQCC\_OK인 경우 유일하게 올바른 값은 다음과 같습니다.

#### MQRC\_NONE

(0, x'000') 보고할 이유가 없습니다.

완료 코드가 MQCC\_FAILED 또는 MQCC\_WARNING인 경우 엑시트 함수는 이유 코드 필드를 올바른 MQRC\_\* 값으로 설정할 수 있습니다.

## C 언어 호출

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```

void MQENTRY MQ_STAT_EXIT (
PMQAXP     pExitParms,    /* Address of exit parameter structure */
PMQAXC     pExitContext,  /* Address of exit context structure */
PMQHCONN   pHconn,       /* Address of connection handle */

```

```

PMQLONG  pType           /* Address of status type */
PPMQSTS  ppStatus       /* Address of status buffer */
PMQLONG  pCompCode,     /* Address of completion code */
PMQLONG  pReason);     /* Address of reason code qualifying completion
                        code */

```

## 종료 - MQ\_TERM\_EXIT

MQ\_TERM\_EXIT는 함수 ID MQXF\_TERM 및 ExitReason MQXR\_CONNECTION으로 등록된 연결 레벨 종료를 제공합니다. 등록된 경우 모든 연결 끊기 요청에 대해 MQ\_TERM\_EXIT가 한 번 호출됩니다.

종료의 일부로 엑시트에 더 이상 필요하지 않는 스토리지를 릴리스하고 필요한 정리를 수행할 수 있습니다.

MQXCC\_FAILED가 표시되면서 MQ\_TERM\_EXIT가 실패하면, MQCC\_FAILED 및 MQRC\_API\_EXIT\_ERROR가 표시되면서 큐 관리자가 이를 호출한 MQDISC에서 리턴됩니다.

마지막 MQ\_TERM\_EXIT를 호출한 후 API 엑시트 함수 실행 환경을 종료하는 동안 큐 관리자에 오류가 발생하면 MQCC\_FAILED 및 MQRC\_API\_EXIT\_TERM\_ERROR가 표시되면서 큐 관리자가 MQ\_TERM\_EXIT를 호출한 MQDISC 호출에서 리턴됩니다.

이 함수의 인터페이스는 다음과 같습니다.

```
MQ_TERM_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

여기서 매개변수는 다음과 같습니다.

### ExitParms (MQAXP) - 입출력(I/O)

엑시트 매개변수 구조입니다.

### ExitContext (MQAXC) - 입출력(I/O)

엑시트 컨텍스트 구조입니다.

### CompCode (MQLONG) - 입출력(I/O)

완료 코드의 올바른 값은 다음과 같습니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_FAILED

호출에 실패했습니다.

### Reason (MQLONG) - 입출력(I/O)

완료 코드를 규정하는 이유 코드입니다.

완료 코드가 MQCC\_OK인 경우 유일하게 올바른 값은 다음과 같습니다.

#### MQRC\_NONE

(0, x'000') 보고할 이유가 없습니다.

완료 코드가 MQCC\_FAILED이면 엑시트 함수는 이유 코드 필드를 올바른 MQRC\_\* 값으로 설정할 수 있습니다.

애플리케이션에 리턴되는 CompCode 및 Reason은 MQAXP의 ExitResponse 필드 값에 따라 다릅니다.

## C 언어 호출

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```

MQAXP      ExitParms;      /* Exit parameter structure */
MQAXC      ExitContext;    /* Exit context structure */
MQLONG     CompCode;       /* Completion code */
MQLONG     Reason;        /* Reason code */

```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
MQ_TERM_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
void MQENTRY MQ_TERM_EXIT (
PMQAXP      pExitParms,      /* Address of exit parameter structure */
PMQAXC      pExitContext,    /* Address of exit context structure */
PMQLONG     pCompCode,       /* Address of completion code */
PMQLONG     pReason);        /* Address of reason code qualifying
                             completion code */
```

## 사용법 참고

1. MQ\_TERM\_EXIT 함수는 선택사항입니다. 수행해야 할 종료 처리가 없는 경우 엑시트 스위트가 종료 엑시트를 등록할 필요가 없습니다.  
엑시트 스위트에 속한 함수가 연결하는 동안 자원을 확보하면 MQ\_TERM\_EXIT 함수는 이러한 자원을 비우는 (예: 동적으로 확보된 스토리지 비우기) 편리한 지점이 됩니다.
2. MQDISC 호출 발행 시 MQ\_TERM\_EXIT 함수가 등록된 경우, 모든 MQDISC 엑시트 함수를 호출한 후에 엑시트 함수가 호출됩니다.
3. MQ\_TERM\_EXIT가 MQAXP의 ExitResponse 필드에 MQXCC\_FAILED를 리턴하거나 어떤 식으로든 실패하면 MQ\_TERM\_EXIT 호출을 유발한 MQDISC 호출도 실패하고 **CompCode** 및 **Reason** 매개변수가 적절한 값으로 설정됩니다.

## 구독 등록 - MQ\_SUB\_EXIT

MQ\_SUB\_EXIT는 사전 및 사후 구독 재등록 처리를 수행하는 엑시트 함수를 제공합니다. 사전 및 사후 구독 재등록 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_SUB를 사용하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
MQ_SUB_EXIT (&ExitParms, &ExitContext, &Hconn, &pSubDesc, &pHobj, &pHsub, &CompCode, &Reason)
```

여기서 매개변수는 다음과 같습니다.

### ExitParms (MQAXP) - 입출력(I/O)

엑시트 매개변수 구조입니다.

### ExitContext (MQAXC) - 입출력(I/O)

엑시트 컨텍스트 구조입니다.

### Hconn(MQHCONN) - 입출력(I/O)

연결 핸들입니다.

### pSubDesc - 입출력(I/O)

속성 선택자의 배열.

### pHobj - 입출력(I/O)

오브젝트 핸들

### pHsub (MQHOBJ) 입출력(I/O)

구독 핸들

### CompCode (MQLONG) - 입출력(I/O)

완료 코드의 올바른 값은 다음과 같습니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_WARNING

부분 완료입니다.

#### MQCC\_FAILED

호출에 실패했습니다.

### Reason (MQLONG) - 입출력(I/O)

완료 코드를 규정하는 이유 코드입니다.

완료 코드가 MQCC\_OK인 경우 유일하게 올바른 값은 다음과 같습니다.

### **MQRC\_NONE**

(0, x'000') 보고할 이유가 없습니다.

완료 코드가 MQCC\_FAILED 또는 MQCC\_WARNING인 경우 엑시트 함수는 이유 코드 필드를 올바른 MQRC\_\* 값으로 설정할 수 있습니다.

## **C 언어 호출**

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP    ExitParms;        /* Exit parameter structure */
MQAXC    ExitContext;     /* Exit context structure */
MQHCONN  Hconn;          /* Connection handle */
PMQSD    pSubDesc;       /* Subscription descriptor */
PMQHOBJS pHobj;          /* Object Handle */
PMQHOBJS pHsub;          /* Subscription handle */
MQLONG   CompCode;       /* Completion code */
MQLONG   Reason;         /* Reason code qualifying completion code */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
MQ_SUB_EXIT (&ExitParms, &ExitContext, &Hconn, &pSubDesc, &pHobj, &pHsub,
             &CompCode, &Reason);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
PMQAXP    pExitParms;     /* Exit parameter structure */
PMQAXC    pExitContext;   /* Exit context structure */
PMQHCONN  pHconn;        /* Connection handle */
PPMQSD    ppSubDesc;     /* Subscription descriptor */
PPMQHOBJS ppHobj;        /* Object Handle */
PPMQHOBJS ppHsub;        /* Subscription handle */
PMQLONG   pCompCode;     /* Completion code */
PMQLONG   pReason;       /* Reason code qualifying completion code */
```

### **구독 요청 - MQ\_SUBRQ\_EXIT**

MQ\_SUBRQ\_EXIT는 사전 및 사후 구독 요청 처리를 수행하는 구독 요청 엑시트 함수를 제공합니다. 사전 및 사후 구독 요청 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_SUBRQ를 사용하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
MQ_SUBRQ_EXIT (&ExitParms, &ExitContext, &Hconn, &pHsub, &Action, &pSubRqOpts,
              &CompCode, &Reason)
```

여기서 매개변수는 다음과 같습니다.

#### **ExitParms (MQAXP) - 입출력(I/O)**

엑시트 매개변수 구조입니다.

#### **ExitContext (MQAXC) - 입출력(I/O)**

엑시트 컨텍스트 구조입니다.

#### **Hconn(MQHCONN) - 입출력(I/O)**

연결 핸들입니다.

#### **pHsub (MQHOBJS) 입출력(I/O)**

구독 핸들

#### **Action (MQLONG) 입출력(I/O)**

Action

#### **pSubRqOpts (MQSRO) 입출력(I/O)**

### CompCode (MQLONG) - 입출력(I/O)

완료 코드의 올바른 값은 다음과 같습니다.

#### **MQCC\_OK**

정상적으로 완료되었습니다.

#### **MQCC\_WARNING**

부분 완료입니다.

#### **MQCC\_FAILED**

호출에 실패했습니다.

### Reason (MQLONG) - 입출력(I/O)

완료 코드를 규정하는 이유 코드입니다.

완료 코드가 MQCC\_OK인 경우 유일하게 올바른 값은 다음과 같습니다.

#### **MQRC\_NONE**

(0, x'000') 보고할 이유가 없습니다.

완료 코드가 MQCC\_FAILED 또는 MQCC\_WARNING인 경우 엑시트 함수는 이유 코드 필드를 올바른 MQRC\_\* 값으로 설정할 수 있습니다.

## C 언어 호출

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP    ExitParms;      /* Exit parameter structure */
MQAXC    ExitContext;   /* Exit context structure */
MQHCONN  Hconn;         /* Connection handle */
PMQLONG  pHsub;         /* Subscription handle */
MQLONG   Action;        /* Action */
PMQSRO   pSubRqOpts;    /* Subscription Request Options */
MQLONG   CompCode;      /* Completion code */
MQLONG   Reason;        /* Reason code qualifying completion code */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
MQ_SUBRQ_EXIT (&ExitParms, &ExitContext, &Hconn, &pHsub, &Action, &pSubRqOpts,
               &CompCode, &Reason);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
void MQENTRY MQ_SUBRQ_EXIT (
PMQAXP    pExitParms,      /* Address of exit parameter structure */
PMQAXC    pExitContext,   /* Address of exit context structure */
PMQHCONN  pHconn,         /* Address of connection handle */
PPMQHOBJ  ppHsub,         /* Address of Subscription handle */
PMQLONG   pAction;        /* Address of Action */
PPMQSRO   ppSubRqOpts;    /* Address of Subscription Request Options */
PMQLONG   pCompCode,      /* Address of completion code */
PMQLONG   pReason);       /* Address of reason code qualifying completion
                           code */
```

### ***xa\_close - XA\_CLOSE\_EXIT***

XA\_CLOSE\_EXIT는 사전 및 사후 xa\_close 처리를 수행하는 xa\_close 엑시트 함수를 제공합니다. 사전 및 사후 xa\_close 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_XACLOSE를 사용하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
XA_CLOSE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode)
```

여기서 매개변수는 다음과 같습니다.

**ExitParms (MQAXP) - 입출력(I/O)**

엑시트 매개변수 구조입니다.

**ExitContext (MQAXC) - 입출력(I/O)**

엑시트 컨텍스트 구조입니다.

**Hconn (MQHCONN) - 입력**

연결 핸들입니다.

**pXa\_info (PMQCHAR) - 입출력(I/O)**

인스턴스 고유 자원 관리자 정보.

**Rmid(MQLONG) - 입출력(I/O)**

자원 관리자 ID입니다.

**Flags(MQLONG) - 입출력(I/O)**

자원 관리자 옵션입니다.

**XARetCode(MQLONG) - 입출력(I/O)**

XA 호출의 응답입니다.

**C 언어 호출**

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```

MQAXP    ExitParms;    /* Exit parameter structure */
MQAXC    ExitContext; /* Exit context structure */
MQHCONN  Hconn;       /* Connection handle */
PMQCHAR  pXa_info;    /* Instance-specific RM info */
MQLONG   Rmid;        /* Resource manager identifier */
MQLONG   Flags;       /* Resource manager options*/
MQLONG   XARetCode;  /* Response from XA call */

```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```

XA_CLOSE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode);

```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```

typedef void MQENTRY XA_CLOSE_EXIT (
    PMQAXP    pExitParms,    /* Address of exit parameter structure */
    PMQAXC    pExitContext, /* Address of exit context structure */
    PMQHCONN  pHconn,       /* Address of connection handle */
    PPMQCHAR  ppXa_info,    /* Address of instance-specific RM info */
    PMQLONG   pRmid,        /* Address of resource manager identifier */
    PMQLONG   pFlags,       /* Address of resource manager options*/
    PMQLONG   pXARetCode); /* Address of response from XA call */

```

**xa\_commit - XA\_COMMIT\_EXIT**

XA\_COMMIT\_EXIT는 사전 및 사후 xa\_commit 처리를 수행하는 xa\_commit 엑시트 함수를 제공합니다. 사전 및 사후 xa\_commit 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_XACOMMIT를 사용하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```

XA_COMMIT_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)

```

여기서 매개변수는 다음과 같습니다.

**ExitParms (MQAXP) - 입출력(I/O)**

엑시트 매개변수 구조입니다.

**ExitContext (MQAXC) - 입출력(I/O)**

엑시트 컨텍스트 구조입니다.

**Hconn (MQHCONN) - 입력**

연결 핸들입니다.

**pXID (MQPTR) - 입출력(I/O)**

트랜잭션 분기 ID입니다.

**Rmid(MQLONG) - 입출력(I/O)**

자원 관리자 ID입니다.

**Flags(MQLONG) - 입출력(I/O)**

자원 관리자 옵션입니다.

**XARetCode(MQLONG) - 입출력(I/O)**

XA 호출의 응답입니다.

**C 언어 호출**

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```

MQAXP   ExitParms;   /* Exit parameter structure */
MQAXC   ExitContext; /* Exit context structure */
MQHCONN Hconn;      /* Connection handle */
MQPTR   pXID;       /* Transaction branch ID */
MQLONG  Rmid;       /* Resource manager identifier */
MQLONG  Flags;      /* Resource manager options*/
MQLONG  XARetCode;  /* Response from XA call */

```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```

XA_COMMIT_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);

```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```

typedef void MQENTRY XA_COMMIT_EXIT (
    PMQAXP   pExitParms, /* Address of exit parameter structure */
    PMQAXC   pExitContext, /* Address of exit context structure */
    PMQHCONN pHconn, /* Address of connection handle */
    PMQPTR   ppXID, /* Address of transaction branch ID */
    PMQLONG  pRmid, /* Address of resource manager identifier */
    PMQLONG  pFlags, /* Address of resource manager options*/
    PMQLONG  pXARetCode); /* Address of response from XA call */

```

***xa\_complete - XA\_COMPLETE\_EXIT***

XA\_COMPLETE\_EXIT는 사전 및 사후 xa\_complete 처리를 수행하는 xa\_complete 엑시트 함수를 제공합니다. 사전 및 사후 xa\_complete 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_XACOMPLETE를 사용하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```

XA_COMPLETE_EXIT (&ExitParms, &ExitContext, &Hconn, &pHandle, &pRetVal, &Rmid, &Flags, &XARetCode)

```

여기서 매개변수는 다음과 같습니다.

**ExitParms (MQAXP) - 입출력(I/O)**

엑시트 매개변수 구조입니다.

**ExitContext (MQAXC) - 입출력(I/O)**

엑시트 컨텍스트 구조입니다.

**Hconn (MQHCONN) - 입력**

연결 핸들입니다.

**pHandle (PMQLONG) - 입출력(I/O)**

비동기 조작에 대한 포인터.



**pRetVal (PMQLONG) - 입출력(I/O)**

비동기 조작의 리턴 값.

**Rmid(MQLONG) - 입출력(I/O)**

자원 관리자 ID입니다.

**Flags(MQLONG) - 입출력(I/O)**

자원 관리자 옵션입니다.

**XARetCode(MQLONG) - 입출력(I/O)**

XA 호출의 응답입니다.

**C 언어 호출**

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```

MQAXP  ExitParms;    /* Exit parameter structure */
MQAXC  ExitContext; /* Exit context structure */
MQHCONN Hconn;      /* Connection handle */
PMQLONG pHandle;    /* Ptr to asynchronous op */
PMQLONG pRetVal;    /* Return value of async op */
MQLONG  Rmid;       /* Resource manager identifier */
MQLONG  Flags;      /* Resource manager options*/
MQLONG  XARetCode; /* Response from XA call */

```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```

XA_COMPLETE_EXIT (&ExitParms, &ExitContext, &Hconn, &pHandle, &pRetVal, &Rmid, &Flags,
&XARetCode);

```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```

typedef void MQENTRY XA_COMPLETE_EXIT (
    PMQAXP  pExitParms, /* Address of exit parameter structure */
    PMQAXC  pExitContext, /* Address of exit context structure */
    PMQHCONN pHconn, /* Address of connection handle */
    PPMQLONG ppHandle, /* Address of ptr to asynchronous op */
    PPMQLONG ppRetVal, /* Address of return value of async op */
    PMQLONG  pRmid, /* Address of resource manager identifier */
    PMQLONG  pFlags, /* Address of resource manager options*/
    PMQLONG  pXARetCode); /* Address of response from XA call */

```

***xa\_end - XA\_END\_EXIT***

XA\_END\_EXIT는 사전 및 사후 xa\_end 처리를 수행하는 xa\_end 엑시트 함수를 제공합니다. 사전 및 사후 xa\_end 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_XAEND를 사용하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```

XA_END_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)

```

여기서 매개변수는 다음과 같습니다.

**ExitParms (MQAXP) - 입출력(I/O)**

엑시트 매개변수 구조입니다.

**ExitContext (MQAXC) - 입출력(I/O)**

엑시트 컨텍스트 구조입니다.

**Hconn (MQHCONN) - 입력**

연결 핸들입니다.

**pXID (MQPTR) - 입출력(I/O)**

트랜잭션 분기 ID입니다.

**Rmid(MQLONG) - 입출력(I/O)**

자원 관리자 ID입니다.

## Flags(MQLONG) - 입출력(I/O)

자원 관리자 옵션입니다.

## XARetCode(MQLONG) - 입출력(I/O)

XA 호출의 응답입니다.

## C 언어 호출

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP  ExitParms; /* Exit parameter structure */
MQAXC  ExitContext; /* Exit context structure */
MQHCONN Hconn; /* Connection handle */
MQPTR  pXID; /* Transaction branch ID */
MQLONG Rmid; /* Resource manager identifier */
MQLONG Flags; /* Resource manager options*/
MQLONG XARetCode; /* Response from XA call */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
XA_END_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
typedef void MQENTRY XA_END_EXIT (
    PMQAXP  pExitParms, /* Address of exit parameter structure */
    PMQAXC  pExitContext, /* Address of exit context structure */
    PMQHCONN pHconn, /* Address of connection handle */
    PMQPTR  ppXID, /* Address of transaction branch ID */
    PMQLONG pRmid, /* Address of resource manager identifier */
    PMQLONG pFlags, /* Address of resource manager options*/
    PMQLONG pXARetCode); /* Address of response from XA call */
```

## *xa\_forget* - XA\_FORGET\_EXIT

XA\_FORGET\_EXIT는 사전 및 사후 *xa\_forget* 처리를 수행하는 *xa\_forget* 엑시트 함수를 제공합니다. 사전 및 사후 *xa\_forget* 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_XAFORGET을 사용하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
XA_FORGET_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

여기서 매개변수는 다음과 같습니다.

### ExitParms (MQAXP) - 입출력(I/O)

엑시트 매개변수 구조입니다.

### ExitContext (MQAXC) - 입출력(I/O)

엑시트 컨텍스트 구조입니다.

### Hconn (MQHCONN) - 입력

연결 핸들입니다.

### pXID (MQPTR) - 입출력(I/O)

트랜잭션 분기 ID입니다.

### Rmid(MQLONG) - 입출력(I/O)

자원 관리자 ID입니다.

### Flags(MQLONG) - 입출력(I/O)

자원 관리자 옵션입니다.

### XARetCode(MQLONG) - 입출력(I/O)

XA 호출의 응답입니다.

## C 언어 호출

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP  ExitParms; /* Exit parameter structure */
MQAXC  ExitContext; /* Exit context structure */
MQHCONN Hconn; /* Connection handle */
MQPTR  pXID; /* Transaction branch ID */
MQLONG Rmid; /* Resource manager identifier */
MQLONG Flags; /* Resource manager options*/
MQLONG XARetCode; /* Response from XA call */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
XA_FORGET_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
typedef void MQENTRY XA_FORGET_EXIT (
    PMQAXP  pExitParms, /* Address of exit parameter structure */
    PMQAXC  pExitContext, /* Address of exit context structure */
    PMQHCONN pHconn, /* Address of connection handle */
    PMQPTR  ppXID, /* Address of transaction branch ID */
    PMQLONG pRmid, /* Address of resource manager identifier */
    PMQLONG pFlags, /* Address of resource manager options*/
    PMQLONG pXARetCode); /* Address of response from XA call */
```

### **xa\_open - XA\_OPEN\_EXIT**

XA\_OPEN\_EXIT는 사전 및 사후 xa\_open 처리를 수행하는 xa\_open 엑시트 함수를 제공합니다. 사전 및 사후 xa\_open 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_XAOPEN을 사용하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
XA_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode)
```

여기서 매개변수는 다음과 같습니다.

#### **ExitParms (MQAXP) - 입출력(I/O)**

엑시트 매개변수 구조입니다.

#### **ExitContext (MQAXC) - 입출력(I/O)**

엑시트 컨텍스트 구조입니다.

#### **Hconn (MQHCONN) - 입력**

연결 핸들입니다.

#### **pXa\_info (PMQCHAR) - 입출력(I/O)**

인스턴스 고유 자원 관리자 정보.

#### **Rmid(MQLONG) - 입출력(I/O)**

자원 관리자 ID입니다.

#### **Flags(MQLONG) - 입출력(I/O)**

자원 관리자 옵션입니다.

#### **XARetCode(MQLONG) - 입출력(I/O)**

XA 호출의 응답입니다.

## C 언어 호출

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP  ExitParms; /* Exit parameter structure */
MQAXC  ExitContext; /* Exit context structure */
MQHCONN Hconn; /* Connection handle */
```

```

PMQCHAR pXa_info; /* Instance-specific RM info */
MQLONG Rmid; /* Resource manager identifier */
MQLONG Flags; /* Resource manager options*/
MQLONG XARetCode; /* Response from XA call */

```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
XA_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```

typedef void MQENTRY XA_OPEN_EXIT (
    PMQAXP pExitParms, /* Address of exit parameter structure */
    PMQAXC pExitContext, /* Address of exit context structure */
    PMQHCONN pHconn, /* Address of connection handle */
    PPMQCHAR ppXa_info, /* Address of instance-specific RM info */
    PMQLONG pRmid, /* Address of resource manager identifier */
    PMQLONG pFlags, /* Address of resource manager options*/
    PMQLONG pXARetCode); /* Address of response from XA call */

```

### ***xa\_prepare - XA\_PREPARE\_EXIT***

XA\_PREPARE\_EXIT는 사전 및 사후 xa\_prepare 처리를 수행하는 xa\_prepare 엑시트 함수를 제공합니다. 사전 및 사후 xa\_prepare 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_XAPREPARE를 사용하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
XA_PREPARE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

여기서 매개변수는 다음과 같습니다.

**ExitParms (MQAXP) - 입출력(I/O)**  
엑시트 매개변수 구조입니다.

**ExitContext (MQAXC) - 입출력(I/O)**  
엑시트 컨텍스트 구조입니다.

**Hconn (MQHCONN) - 입력**  
연결 핸들입니다.

**pXID (MQPTR) - 입출력(I/O)**  
트랜잭션 분기 ID입니다.

**Rmid(MQLONG) - 입출력(I/O)**  
자원 관리자 ID입니다.

**Flags(MQLONG) - 입출력(I/O)**  
자원 관리자 옵션입니다.

**XARetCode(MQLONG) - 입출력(I/O)**  
XA 호출의 응답입니다.

### **C 언어 호출**

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```

MQAXP ExitParms; /* Exit parameter structure */
MQAXC ExitContext; /* Exit context structure */
MQHCONN Hconn; /* Connection handle */
MQPTR pXID; /* Transaction branch ID */
MQLONG Rmid; /* Resource manager identifier */
MQLONG Flags; /* Resource manager options*/
MQLONG XARetCode; /* Response from XA call */

```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
XA_PREPARE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
typedef void MQENTRY XA_PREPARE_EXIT (  
    PMQAXP    pExitParms,    /* Address of exit parameter structure */  
    PMQAXC    pExitContext,  /* Address of exit context structure */  
    PMQHCONN  pHconn,       /* Address of connection handle */  
    PMQPTR    ppXID,        /* Address of transaction branch ID */  
    PMQLONG   pRmid,        /* Address of resource manager identifier */  
    PMQLONG   pFlags,       /* Address of resource manager options*/  
    PMQLONG   pXARetCode);  /* Address of response from XA call */
```

### ***xa\_recover* - XA\_RECOVER\_EXIT**

XA\_RECOVER\_EXIT는 사전 및 사후 *xa\_recover* 처리를 수행하는 *xa\_recover* 엑시트 함수를 제공합니다. 사전 및 사후 *xa\_recover* 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_XARECOVER를 사용하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
XA_RECOVER_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Count, &Rmid, &Flags, &XARetCode)
```

여기서 매개변수는 다음과 같습니다.

#### **ExitParms (MQAXP) - 입출력(I/O)**

엑시트 매개변수 구조입니다.

#### **ExitContext (MQAXC) - 입출력(I/O)**

엑시트 컨텍스트 구조입니다.

#### **Hconn (MQHCONN) - 입력**

연결 핸들입니다.

#### **pXID (MQPTR) - 입출력(I/O)**

트랜잭션 분기 ID입니다.

#### **Count(MQLONG) - 입출력(I/O)**

XID 배열의 최대 XID

#### **Rmid(MQLONG) - 입출력(I/O)**

자원 관리자 ID입니다.

#### **Flags(MQLONG) - 입출력(I/O)**

자원 관리자 옵션입니다.

#### **XARetCode(MQLONG) - 입출력(I/O)**

XA 호출의 응답입니다.

## **C 언어 호출**

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP    ExitParms;    /* Exit parameter structure */  
MQAXC    ExitContext; /* Exit context structure */  
MQHCONN  Hconn;       /* Connection handle */  
MQPTR    pXID;        /* Transaction branch ID */  
MQLONG   Count;       /* Max XIDs in XID array */  
MQLONG   Rmid;        /* Resource manager identifier */  
MQLONG   Flags;       /* Resource manager options*/  
MQLONG   XARetCode;   /* Response from XA call */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
XA_RECOVER_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Count, &Rmid, &Flags, &XARetCode);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
typedef void MQENTRY XA_RECOVER_EXIT (  
    PMQAXP  pExitParms, /* Address of exit parameter structure */  
    PMQAXC  pExitContext, /* Address of exit context structure */  
    PMQHCONN pHconn, /* Address of connection handle */  
    PMQPTR  ppXID, /* Address of transaction branch ID */  
    PMQLONG pCount, /* Address of max XIDs in XID array */  
    PMQLONG pRmid, /* Address of resource manager identifier */  
    PMQLONG pFlags, /* Address of resource manager options*/  
    PMQLONG pXARetCode); /* Address of response from XA call */
```

### **xa\_rollback - XA\_ROLLBACK\_EXIT**

XA\_ROLLBACK\_EXIT는 사전 및 사후 xa\_rollback 처리를 수행하는 xa\_rollback 엑시트 함수를 제공합니다. 사전 및 사후 xa\_rollback 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_XAROLLBACK을 사용하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
XA_ROLLBACK_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

여기서 매개변수는 다음과 같습니다.

#### **ExitParms (MQAXP) - 입출력(I/O)**

엑시트 매개변수 구조입니다.

#### **ExitContext (MQAXC) - 입출력(I/O)**

엑시트 컨텍스트 구조입니다.

#### **Hconn (MQHCONN) - 입력**

연결 핸들입니다.

#### **pXID (MQPTR) - 입출력(I/O)**

트랜잭션 분기 ID입니다.

#### **Rmid(MQLONG) - 입출력(I/O)**

자원 관리자 ID입니다.

#### **Flags(MQLONG) - 입출력(I/O)**

자원 관리자 옵션입니다.

#### **XARetCode(MQLONG) - 입출력(I/O)**

XA 호출의 응답입니다.

## **C 언어 호출**

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP  ExitParms; /* Exit parameter structure */  
MQAXC  ExitContext; /* Exit context structure */  
MQHCONN Hconn; /* Connection handle */  
MQPTR  pXID; /* Transaction branch ID */  
MQLONG Rmid; /* Resource manager identifier */  
MQLONG Flags; /* Resource manager options*/  
MQLONG XARetCode; /* Response from XA call */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
XA_ROLLBACK_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
typedef void MQENTRY XA_ROLLBACK_EXIT (  
    PMQAXP  pExitParms, /* Address of exit parameter structure */  
    PMQAXC  pExitContext, /* Address of exit context structure */  
    PMQHCONN pHconn, /* Address of connection handle */
```

```

PMQPTR  ppXID,          /* Address of transaction branch ID */
PMQLONG  pRmid,         /* Address of resource manager identifier */
PMQLONG  pFlags,       /* Address of resource manager options*/
PMQLONG  pXARetCode); /* Address of response from XA call */

```

### **xa\_start - XA\_START\_EXIT**

XA\_START\_EXIT는 사전 및 사후 xa\_start 처리를 수행하는 xa\_start 엑시트 함수를 제공합니다. 사전 및 사후 xa\_start 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_XASTART를 사용하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
XA_START_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

여기서 매개변수는 다음과 같습니다.

#### **ExitParms (MQAXP) - 입출력(I/O)**

엑시트 매개변수 구조입니다.

#### **ExitContext (MQAXC) - 입출력(I/O)**

엑시트 컨텍스트 구조입니다.

#### **Hconn (MQHCONN) - 입력**

연결 핸들입니다.

#### **pXID (MQPTR) - 입출력(I/O)**

트랜잭션 분기 ID입니다.

#### **Rmid(MQLONG) - 입출력(I/O)**

자원 관리자 ID입니다.

#### **Flags(MQLONG) - 입출력(I/O)**

자원 관리자 옵션입니다.

#### **XARetCode(MQLONG) - 입출력(I/O)**

XA 호출의 응답입니다.

## **C 언어 호출**

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```

MQAXP  ExitParms; /* Exit parameter structure */
MQAXC  ExitContext; /* Exit context structure */
MQHCONN Hconn; /* Connection handle */
MQPTR  pXID; /* Transaction branch ID */
MQLONG Rmid; /* Resource manager identifier */
MQLONG Flags; /* Resource manager options*/
MQLONG XARetCode; /* Response from XA call */

```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
XA_START_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```

typedef void MQENTRY XA_START_EXIT (
  MQAXP  pExitParms, /* Address of exit parameter structure */
  MQAXC  pExitContext, /* Address of exit context structure */
  MQHCONN pHconn, /* Address of connection handle */
  MQPTR  ppXID, /* Address of transaction branch ID */
  PMQLONG pRmid, /* Address of resource manager identifier */
  PMQLONG pFlags, /* Address of resource manager options*/
  PMQLONG pXARetCode); /* Address of response from XA call */

```

## **ax\_reg - AX\_REG\_EXIT**

AX\_REG\_EXIT는 사전 및 사후 ax\_reg 처리를 수행하는 ax\_reg 엑시트 함수를 제공합니다. 사전 및 사후 ax\_reg 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_AXREG를 사용하십시오.

이 함수의 인터페이스는 다음과 같습니다.

```
AX_REG_EXIT (&ExitParms, &ExitContext, &pXID, &Rmid, &Flags, &XARetCode)
```

여기서 매개변수는 다음과 같습니다.

### **ExitParms (MQAXP) - 입출력(I/O)**

엑시트 매개변수 구조입니다.

### **ExitContext (MQAXC) - 입출력(I/O)**

엑시트 컨텍스트 구조입니다.

### **Hconn (MQHCONN) - 입력**

연결 핸들입니다.

### **pXID (MQPTR) - 입출력(I/O)**

트랜잭션 분기 ID입니다.

### **Rmid(MQLONG) - 입출력(I/O)**

자원 관리자 ID입니다.

### **Flags(MQLONG) - 입출력(I/O)**

자원 관리자 옵션입니다.

### **XARetCode(MQLONG) - 입출력(I/O)**

XA 호출의 응답입니다.

## **C 언어 호출**

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP ExitParms; /* Exit parameter structure */
MQAXC ExitContext; /* Exit context structure */
MQPTR pXID; /* Transaction branch ID */
MQLONG Rmid; /* Resource manager identifier */
MQLONG Flags; /* Resource manager options*/
MQLONG XARetCode; /* Response from XA call */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
AX_REG_EXIT (&ExitParms, &ExitContext, &pXID, &Rmid, &Flags, &XARetCode);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
typedef void MQENTRY AX_REG_EXIT (
    PMQAXP pExitParms, /* Address of exit parameter structure */
    PMQAXC pExitContext, /* Address of exit context structure */
    PMQPTR ppXID, /* Address of transaction branch ID */
    PMQLONG pRmid, /* Address of resource manager identifier */
    PMQLONG pFlags, /* Address of resource manager options*/
    PMQLONG pXARetCode); /* Address of response from XA call */
```

## **ax\_unreg - AX\_UNREG\_EXIT**

AX\_UNREG\_EXIT는 사전 및 사후 ax\_unreg 처리를 수행하는 ax\_unreg 엑시트 함수를 제공합니다. 사전 및 사후 ax\_unreg 호출 엑시트 함수를 등록하려면 엑시트 이유 MQXR\_BEFORE 및 MQXR\_AFTER와 함께 함수 ID MQXF\_AXUNREG를 사용하십시오.

이 함수의 인터페이스는 다음과 같습니다.



```
AX_UNREG_EXIT (&ExitParms, &ExitContext, &Rmid, &Flags, &XARetCode);
```

여기서 매개변수는 다음과 같습니다.

**ExitParms (MQAXP) - 입출력(I/O)**

엑시트 매개변수 구조입니다.

**ExitContext (MQAXC) - 입출력(I/O)**

엑시트 컨텍스트 구조입니다.

**Rmid(MQLONG) - 입출력(I/O)**

자원 관리자 ID입니다.

**Flags(MQLONG) - 입출력(I/O)**

자원 관리자 옵션입니다.

**XARetCode(MQLONG) - 입출력(I/O)**

XA 호출의 응답입니다.

## C 언어 호출

큐 관리자가 논리적으로 다음 변수를 정의합니다.

```
MQAXP ExitParms; /* Exit parameter structure */
MQAXC ExitContext; /* Exit context structure */
MQLONG Rmid; /* Resource manager identifier */
MQLONG Flags; /* Resource manager options*/
MQLONG XARetCode; /* Response from XA call */
```

그런 다음 큐 관리자가 다음과 같이 엑시트를 논리적으로 호출합니다.

```
AX_UNREG_EXIT (&ExitParms, &ExitContext, &Rmid, &Flags, &XARetCode);
```

사용자의 엑시트는 다음과 같은 C 함수 프로토타입과 일치해야 합니다.

```
typedef void MQENTRY AX_UNREG_EXIT (
    PMQAXP pExitParms, /* Address of exit parameter structure */
    PMQAXC pExitContext, /* Address of exit context structure */
    PMQLONG pRmid, /* Address of resource manager identifier */
    PMQLONG pFlags, /* Address of resource manager options*/
    PMQLONG pXARetCode); /* Address of response from XA call */
```

## 엑시트 함수 호출에 대한 일반 정보

이 주제에서는 엑시트를 계획하는 데 유용한 몇 가지 일반 지침, 특히 오류 및 예상치 못한 이벤트 처리와 관련된 정보를 제공합니다.

### 엑시트 실패

엑시트 함수가 동기점에서 벗어난 파괴적 MQGET 호출 이후 메시지가 애플리케이션으로 전달되기 전에 비정상적으로 종료하는 경우, 엑시트 핸들이 장애에서 복구하고 애플리케이션에 제어를 전달할 수 있습니다.

이 경우, 메시지가 손실될 수 있습니다. 이는 큐에서 메시지를 수신한 직후 애플리케이션이 실패하면 나타나는 현상입니다.

MQGET 호출이 MQCC\_FAILED 및 MQRC\_API\_EXIT\_ERROR와 함께 완료될 수 있습니다.

사전 API 호출 엑시트 함수가 비정상적으로 종료되는 경우, 엑시트 핸들러는 API 호출을 처리하지 않고도 장애에서 복구하고 애플리케이션에 제어를 전달할 수 있습니다. 이 경우, 엑시트 함수가 소유한 자원을 복구해야 합니다.

체인으로 연결된 엑시트를 사용 중인 경우, 성공적으로 구동된 사전 API에 대한 사후 API 호출 엑시트가 자체적으로 구동될 수 있습니다. API 호출이 MQCC\_FAILED 및 MQRC\_API\_EXIT\_ERROR와 함께 실패할 수 있습니다.

엑시트 함수의 예제 오류 처리

다음 다이어그램은 오류가 발생할 수 있는 지점(e N)을 보여줍니다. 엑시트가 어떻게 작동하는지 보여주는 예에 불과하므로 다른 표와 함께 확인해야 합니다. 이 예에서는 각 API 호출 전후에 연결된 엑시트의 동작을 보여주기 위해 두 개의 엑시트 함수가 호출됩니다.

Application	ErrPt	Exit function	API call
Start			
MQCONN	-->		
	e1	MQ_INIT_EXIT	
	e2	before MQ_CONNX_EXIT 1	
	e3	before MQ_CONNX_EXIT 2	
	e4		--> MQCONN
	e5	after MQ_CONNX_EXIT 2	
	e6	after MQ_CONNX_EXIT 1	
	e7		
MQOPEN	<-- -->		
	e8	before MQ_OPEN_EXIT 1	
	e9	before MQ_OPEN_EXIT 2	
	e10		--> MQOPEN
	e11	after MQ_OPEN_EXIT 2	
	e12	after MQ_OPEN_EXIT 1	
MQPUT	<-- -->		
	e13	before MQ_PUT_EXIT 1	
	e14	before MQ_PUT_EXIT 2	
	e15		--> MQPUT
	e16	after MQ_PUT_EXIT 2	
	e17	after MQ_PUT_EXIT 1	
MQCLOSE	<-- -->		
	e18	before MQ_CLOSE_EXIT 1	
	e19	before MQ_CLOSE_EXIT 2	
	e20		--> MQCLOSE
	e21	after MQ_CLOSE_EXIT 2	
	e22	after MQ_CLOSE_EXIT 1	
MQDISC	<-- -->		
	e23	before MQ_DISC_EXIT 1	
	e24	before MQ_DISC_EXIT 2	
	e25		--> MQDISC
	e26	after MQ_DISC_EXIT 2	
	e27	after MQ_DISC_EXIT 1	
	<--		
end			

다음 표에는 각 오류 지점에서 취해야 할 조치가 나와 있습니다. 오류 지점의 일부만 다루며, 여기 설명된 규칙을 다른 모든 항목에 적용할 수 있습니다. 각 사례에서 원하는 동작을 지정하는 조치입니다.

표 837. API 엑시트 오류 및 적절한 수행 조치

Err Pt	설명	조치
e1	환경 설정 중에 오류가 발생합니다.	<ol style="list-style-type: none"> <li>1. 필요에 따라 환경 설정 실행 취소</li> <li>2. 엑시트 함수를 구동하지 않음</li> <li>3. MQCC_FAILED, MQRC_API_EXIT_LOAD_ERROR와 함께 MQCONN 실패</li> </ol>
e2	다음과 함께 MQ_INIT_EXIT 함수가 완료됩니다. <ul style="list-style-type: none"> <li>• MQXCC_FAILED</li> <li>• MQXCC_*</li> </ul>	<ul style="list-style-type: none"> <li>• MQXCC_FAILED의 경우:                             <ol style="list-style-type: none"> <li>1. 환경 정리</li> <li>2. MQCC_FAILED, MQRC_API_EXIT_INIT_ERROR와 함께 MQCONN 실패</li> </ol> </li> <li>• MQXCC_*의 경우                             <ol style="list-style-type: none"> <li>1. MQXCC_* 및 MQXR2_*<sup>1</sup>의 값에 대해서와 같이 작동</li> <li>2. 환경 정리</li> </ol> </li> </ul>
e3	다음과 함께 사전 MQ_CONNX_EXIT 1 함수가 완료됩니다. <ul style="list-style-type: none"> <li>• MQXCC_FAILED</li> <li>• MQXCC_*</li> </ul>	<ul style="list-style-type: none"> <li>• MQXCC_FAILED의 경우:                             <ol style="list-style-type: none"> <li>1. MQ_TERM_EXIT 함수 구동</li> <li>2. 환경 정리</li> <li>3. MQCC_FAILED, MQRC_API_EXIT_ERROR와 함께 MQCONN 호출 실패</li> </ol> </li> <li>• MQXCC_*의 경우                             <ol style="list-style-type: none"> <li>1. MQXCC_* 및 MQXR2_*<sup>1</sup>의 값에 대해서와 같이 작동</li> <li>2. 필요한 경우 MQ_TERM_EXIT 함수 구동</li> <li>3. 필요한 경우 환경 정리</li> </ol> </li> </ul>
e4	다음과 함께 사전 MQ_CONNX_EXIT 2 함수가 완료됩니다. <ul style="list-style-type: none"> <li>• MQXCC_FAILED</li> <li>• MQXCC_*</li> </ul>	<ul style="list-style-type: none"> <li>• MQXCC_FAILED의 경우:                             <ol style="list-style-type: none"> <li>1. 사후 MQ_CONNX_EXIT 1 함수 구동</li> <li>2. MQ_TERM_EXIT 함수 구동</li> <li>3. 환경 정리</li> <li>4. MQCC_FAILED, MQRC_API_EXIT_ERROR와 함께 MQCONN 호출 실패</li> </ol> </li> <li>• MQXCC_*의 경우                             <ol style="list-style-type: none"> <li>1. MQXCC_* 및 MQXR2_*<sup>1</sup>의 값에 대해서와 같이 작동</li> <li>2. 엑시트가 차단되지 않은 경우 사후 MQ_CONNX_EXIT 1 함수 구동</li> <li>3. 필요한 경우 MQ_TERM_EXIT 함수 구동</li> <li>4. 필요한 경우 환경 정리</li> </ol> </li> </ul>

표 837. API 엑시트 오류 및 적절한 수행 조치 (계속)

Err Pt	설명	조치
e5	MQCONN 호출이 실패합니다.	<ol style="list-style-type: none"> <li>1. MQCONN CompCode 및 Reason 전달</li> <li>2. 사전 MQ_CONNX_EXIT 2가 성공하고 엑시트가 차단되지 않은 경우 사후 MQ_CONNX_EXIT 2 함수 구동</li> <li>3. 사전 MQ_CONNX_EXIT 1이 성공하고 엑시트가 차단되지 않은 경우 사후 MQ_CONNX_EXIT 1 함수 구동</li> <li>4. MQ_TERM_EXIT 함수 구동</li> <li>5. 환경 정리</li> </ol>
e6	다음과 함께 사후 MQ_CONNX_EXIT 2 함수가 완료됩니다. <ul style="list-style-type: none"> <li>• MQXCC_FAILED</li> <li>• MQXCC_*</li> </ul>	<ul style="list-style-type: none"> <li>• MQXCC_FAILED의 경우:                             <ol style="list-style-type: none"> <li>1. 사후 MQ_CONNX_EXIT 1 함수 구동</li> <li>2. MQCC_FAILED, MQRC_API_EXIT_ERROR와 함께 MQCONN 호출 완료</li> </ol> </li> <li>• MQXCC_*의 경우                             <ol style="list-style-type: none"> <li>1. MQXCC_* 및 MQXR2_*<sup>1</sup>의 값에 대해서와 같이 작동</li> <li>2. 필요한 경우 사후 MQ_CONNX_EXIT 1 함수 구동</li> </ol> </li> </ul>
e7	다음과 함께 사후 MQ_CONNX_EXIT 1 함수가 완료됩니다. <ul style="list-style-type: none"> <li>• MQXCC_FAILED</li> <li>• MQXCC_*</li> </ul>	<ul style="list-style-type: none"> <li>• MQXCC_FAILED의 경우, MQCC_FAILED, MQRC_API_EXIT_ERROR와 함께 MQCONN 호출 완료</li> <li>• MQXCC_*의 경우, MQXCC_* 및 MQXR2_*<sup>1</sup>의 값에 대해서와 같이 작동</li> </ul>
e8	다음과 함께 사전 MQ_OPEN_EXIT 1 함수가 완료됩니다. <ul style="list-style-type: none"> <li>• MQXCC_FAILED</li> <li>• MQXCC_*</li> </ul>	<ul style="list-style-type: none"> <li>• MQXCC_FAILED의 경우, MQCC_FAILED, MQRC_API_EXIT_ERROR와 함께 MQOPEN 호출 완료</li> <li>• MQXCC_*의 경우, MQXCC_* 및 MQXR2_*<sup>1</sup>의 값에 대해서와 같이 작동</li> </ul>
e9	다음과 함께 사전 MQ_OPEN_EXIT 2 함수가 완료됩니다. <ul style="list-style-type: none"> <li>• MQXCC_FAILED</li> <li>• MQXCC_*</li> </ul>	<ul style="list-style-type: none"> <li>• MQXCC_FAILED의 경우:                             <ol style="list-style-type: none"> <li>1. 사후 MQ_OPEN_EXIT 1 함수 구동</li> <li>2. MQCC_FAILED, MQRC_API_EXIT_ERROR와 함께 MQOPEN 호출 완료</li> </ol> </li> <li>• MQXCC_*의 경우, MQXCC_* 및 MQXR2_*<sup>1</sup>의 값에 대해서와 같이 작동</li> </ul>
e10	MQOPEN 호출이 실패합니다.	<ol style="list-style-type: none"> <li>1. MQOPEN CompCode 및 Reason 전달</li> <li>2. 엑시트가 차단되지 않은 경우 사후 MQ_OPEN_EXIT 2 함수 구동</li> <li>3. 엑시트가 차단되지 않고 연결된 엑시트가 차단되지 않는 경우 사후 MQ_OPEN_EXIT 1 함수 구동</li> </ol>

표 837. API 엑시트 오류 및 적절한 수행 조치 (계속)		
Err Pt	설명	조치
e1 1	다음과 함께 사후 MQ_OPEN_EXIT 2 함수가 완료됩니다. <ul style="list-style-type: none"> <li>• MQXCC_FAILED</li> <li>• MQXCC_*</li> </ul>	<ul style="list-style-type: none"> <li>• MQXCC_FAILED의 경우: <ol style="list-style-type: none"> <li>1. 사후 MQ_OPEN_EXIT 1 함수 구동</li> <li>2. MQCC_FAILED, MQRC_API_EXIT_ERROR와 함께 MQOPEN 호출 완료</li> </ol> </li> <li>• MQXCC_*의 경우 <ol style="list-style-type: none"> <li>1. MQXCC_* 및 MQXR2_*<sup>1</sup>의 값에 대해서와 같이 작동</li> <li>2. 엑시트가 차단되지 않은 경우 사후 MQ_OPEN_EXIT 1 함수 구동</li> </ol> </li> </ul>
e2 5	다음과 함께 사후 MQ_DISC_EXIT 2 함수가 완료됩니다. <ul style="list-style-type: none"> <li>• MQXCC_FAILED</li> <li>• MQXCC_*</li> </ul>	<ul style="list-style-type: none"> <li>• MQXCC_FAILED의 경우: <ol style="list-style-type: none"> <li>1. 사후 MQ_DISC_EXIT 1 함수 구동</li> <li>2. MQ_TERM_EXIT 함수 구동</li> <li>3. 엑시트 실행 환경 정리</li> <li>4. MQCC_FAILED, MQRC_API_EXIT_ERROR와 함께 MQDISC 호출 완료</li> </ol> </li> <li>• MQXCC_*의 경우 <ol style="list-style-type: none"> <li>1. MQXCC_* 및 MQXR2_*<sup>1</sup>의 값에 대해서와 같이 작동</li> <li>2. MQ_TERM_EXIT 함수 구동</li> <li>3. 엑시트 실행 환경 정리</li> </ol> </li> </ul>

**참고:**

1. MQXCC\_\* 및 MQXR2\_\*의 값과 관련 조치는 큐 관리자의 엑시트 함수 처리 방법에 정의되어 있습니다.

**잘못 설정된 ExitResponse 필드**

이 주제는 ExitResponse 필드가 지원되는 값이 아닌 다른 값으로 설정될 때 발생하는 일에 대한 정보를 제공합니다.

ExitResponse 필드를 지원되는 값이 아닌 다른 값으로 설정한 경우, 다음 조치가 적용됩니다.

- 사전 MQCONN 또는 MQDISC API 엑시트 함수:
  - ExitResponse2 값이 무시됩니다.
  - 엑시트 체인의 추가 사전 엑시트 함수(있는 경우)가 호출되지 않습니다. API 호출도 발행되지 않습니다.
  - 성공적으로 호출된 사전 엑시트의 경우, 사후 엑시트가 역순으로 호출됩니다.
  - 등록된 경우, 성공적으로 호출된 체인의 사전 MQCONN 또는 MQDISC 엑시트 함수에 대한 종료 엑시트 함수가 구동되어 이 엑시트 함수 이후 정리를 수행합니다.
  - MQCONN 또는 MQDISC 호출이 MQRC\_API\_EXIT\_ERROR와 함께 실패합니다.
- MQCONN 또는 MQDISC 이외의 사전 IBM MQ API 엑시트 함수:
  - ExitResponse2 값이 무시됩니다.
  - 엑시트 체인의 추가 사전 또는 사후 데이터 변환 함수(있는 경우)가 호출되지 않습니다.
  - 성공적으로 호출된 사전 엑시트의 경우, 사후 엑시트가 역순으로 호출됩니다.
  - IBM MQ API 호출도 발행되지 않습니다.
  - IBM MQ API 호출이 MQRC\_API\_EXIT\_ERROR와 함께 실패합니다.
- 사후 MQCONN 또는 MQDISC API 엑시트 함수:

- ExitResponse2 값이 무시됩니다.
- API 호출 전에 성공적으로 호출된 남아 있는 엑시트 함수가 역순으로 호출됩니다.
- 등록된 경우, 성공적으로 호출된 체인의 사전 또는 사후 MQCONN 또는 MQDISC 엑시트 함수에 대한 종료 엑시트 함수가 구동되어 엑시트 이후 정리를 수행합니다.
- 엑시트에서 리턴한 CompCode 및 MQCC\_WARNING 중 더 심각한 CompCode가 애플리케이션에 리턴됩니다.
- MQRC\_API\_EXIT\_ERROR의 이유가 애플리케이션에 리턴됩니다.
- IBM MQ API 호출이 성공적으로 발행됩니다.
- MQCONN 또는 MQDISC 이외의 사후 IBM MQ API 호출 엑시트 함수:
  - ExitResponse2 값이 무시됩니다.
  - API 호출 전에 성공적으로 호출된 남아 있는 엑시트 함수가 역순으로 호출됩니다.
  - 엑시트에서 리턴한 CompCode 및 MQCC\_WARNING 중 더 심각한 CompCode가 애플리케이션에 리턴됩니다.
  - MQRC\_API\_EXIT\_ERROR의 이유가 애플리케이션에 리턴됩니다.
  - IBM MQ API 호출이 성공적으로 발행됩니다.
- Get 엑시트 함수에서 사전 데이터 변환:
  - ExitResponse2 값이 무시됩니다.
  - API 호출 전에 성공적으로 호출된 남아 있는 엑시트 함수가 역순으로 호출됩니다.
  - 메시지가 변환되지 않고 변환되지 않은 메시지가 애플리케이션에 리턴됩니다.
  - 엑시트에서 리턴한 CompCode 및 MQCC\_WARNING 중 더 심각한 CompCode가 애플리케이션에 리턴됩니다.
  - MQRC\_API\_EXIT\_ERROR의 이유가 애플리케이션에 리턴됩니다.
  - IBM MQ API 호출이 성공적으로 발행됩니다.

**참고:** 엑시트에 오류가 있으면 MQRC\_NOT\_CONVERTED보다 MQRC\_API\_EXIT\_ERROR를 리턴하는 것이 좋습니다.

엑시트 함수가 ExitResponse2 필드를 지원되는 값이 아닌 다른 값으로 설정하는 경우, MQXR2\_DEFAULT\_CONTINUATION 값이 대신 사용됩니다.

## 설치 가능 서비스 인터페이스 참조 정보

이 주제 컬렉션은 설치 가능 서비스에 대한 참조 정보를 제공합니다.

함수 및 데이터 유형이 각 서비스 유형 그룹에 알파벳순으로 나열되어 있습니다.

### 관련 개념

[!\[\]\(10f8862fc183b400327470ea85afe9ae\_img.jpg\) UNIX, Linux 및 Windows용 설치 가능 서비스 및 컴포넌트](#)

[!\[\]\(e1d6102fe77919492c04879c8450f1f5\_img.jpg\) IBM i용 설치 가능 서비스 및 컴포넌트](#)

[!\[\]\(73002692dd5e7a64e60946be3158e719\_img.jpg\) IBMi에 대한 설치 가능 서비스 인터페이스 참조 정보](#)

### 관련 태스크

[큐 관리자 기능 확장](#)

[!\[\]\(ab4e2b3fc7e7887b7a72f548aa6f5e60\_img.jpg\) 설치 가능 서비스 구성](#)

## 함수를 표시하는 방법

설치 가능 서비스 함수를 기록하는 방법입니다.

함수 ID(MQZEP)를 포함하여 각 함수에 대한 설명이 나와 있습니다.

매개변수는 발생해야 하는 순서대로 표시됩니다. 모든 매개변수가 있어야 합니다.

각 매개변수 이름 뒤에 데이터 유형이 표시됩니다. 요소 데이터 유형에 대한 설명은 [234 페이지의 『기본 데이터 유형』](#)에 나와 있습니다.

매개변수 설명 뒤에 C 언어 호출도 제공됩니다.

## MQZ\_AUTHENTICATE\_USER - 사용자 인증

이 함수는 MQZAS\_VERSION\_5 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 사용자를 인증하거나 ID 컨텍스트 필드를 설정하기 위해 호출합니다. IBM MQ 사용자 애플리케이션 컨텍스트가 설정되면 호출됩니다.

애플리케이션 컨텍스트는 연결 호출 동안 애플리케이션의 사용자 컨텍스트가 초기화되는 지점 및 애플리케이션의 사용자 컨텍스트가 변경되는 각 지점에 설정됩니다. 연결 호출을 작성할 때마다 애플리케이션의 사용자 컨텍스트 정보가 *IdentityContext* 필드에 다시 확보됩니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_AUTHENTICATE\_USER입니다.

## 구문

MQZ\_AUTHENTICATE\_USER ( *QMgrName* , *SecurityParms* , *ApplicationContext* , *IdentityContext* , *CorrelationPtr* , *ComponentData* , *Continuation* , *CompCode* , *Reason* )

## 매개변수

### QMgrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

### SecurityParms

유형: MQCSP - 입력

보안 매개변수. 사용자 ID, 비밀번호 및 인증 유형에 관련된 데이터입니다. MQCSP 구조의 AuthenticationType 속성을 MQCSP\_AUTH\_USER\_ID\_AND\_PWD로 지정하는 경우, 사용자 ID 및 비밀번호를 IdentityContext(MQZIC) 매개변수의 동일 필드와 비교하여 일치 여부를 판별합니다. 자세한 정보는 [329 페이지의 『MQCSP - 보안 매개변수』](#)의 내용을 참조하십시오.

MQCONN MQI 호출 동안 이 매개변수에는 널 또는 기본값이 포함됩니다.

### ApplicationContext

유형: MQZAC - 입력

애플리케이션 컨텍스트. 호출 애플리케이션에 관련된 데이터입니다. 자세한 정보는 [MQZAC - 애플리케이션 컨텍스트](#)의 내용을 참조하십시오.

모든 MQCONN 또는 MQCONNX MQI 호출 동안, MQZAC 구조의 사용자 컨텍스트 정보가 다시 확보됩니다.

### IdentityContext

유형: MQZIC - 입출력(I/O)

ID 컨텍스트. 사용자 인증 함수에 대한 입력에서 현재 ID 컨텍스트를 식별합니다. 사용자 인증 함수는 큐 관리자가 새 ID 컨텍스트를 채택하는 지점에서 이 항목을 변경할 수 있습니다. MQZIC 구조에 대한 자세한 정보 [MQZIC - ID 컨텍스트](#)의 내용을 참조하십시오.

### CorrelationPtr

유형: MQPTR - 출력

상관 포인터. 상관 데이터의 주소를 지정합니다. 이후 이 포인터가 다른 OAM 호출로 전달됩니다.

### ComponentData

유형: MQBYTE x ComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 ComponentDataLength 매개변수에서 큐 관리자에 의해 전달됩니다.

#### Continuation

유형: MQLONG - 출력

연속 플래그입니다. 다음 값을 지정할 수 있습니다.

#### MQZCI\_DEFAULT

기타 컴포넌트에 종속되는 연속입니다.

#### MQZCI\_STOP

다음 컴포넌트를 계속하지 않습니다.

#### CompCode

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_FAILED

호출에 실패했습니다.

#### 원인

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드.

CompCode가 MQCC\_OK인 경우:

#### MQRC\_NONE

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_FAILED인 경우:

#### MQRC\_SERVICE\_ERROR

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

이 이유 코드에 대한 자세한 정보는 [메시지 및 이유 코드를 참조하십시오](#).

## C 호출

```
MQZ_AUTHENTICATE_USER (QMgrName, SecurityParms, ApplicationContext,
                        IdentityContext, &CorrelationPtr, ComponentData,
                        &Continuation, &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언합니다.

```
MQCHAR48  QMgrName;           /* Queue manager name */
MQCSP     SecurityParms;      /* Security parameters */
MQZAC     ApplicationContext; /* Application context */
MQZIC     IdentityContext;    /* Identity context */
MQPTR     CorrelationPtr;     /* Correlation pointer */
MQBYTE    ComponentData[n];   /* Component data */
MQLONG    Continuation;       /* Continuation indicator set by
                               component */
MQLONG    CompCode;           /* Completion code */
MQLONG    Reason;            /* Reason code qualifying CompCode */
```



## MQZ\_CHECK\_AUTHORITY - 권한 검사

이 함수는 MQZAS\_VERSION\_1 권한 서비스 컴포넌트에서 제공되며, 큐 관리자가 지정된 오브젝트에서 특정 조치 수행할 권한이 엔티티에 있는지 여부를 검사하기 위해 시작합니다.

이 함수(MQZEP용)에 대한 함수 ID는 MQZID\_CHECK\_AUTHORITY입니다.

### 구문

`MQZ_CHECK_AUTHORITY( QMgrName , EntityName , EntityType , ObjectName , ObjectType , Authority , ComponentData , Continuation , CompCode , Reason )`

### 매개변수

#### QMgrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

#### EntityName

유형: MQCHAR12 - 입력

엔티티 이름입니다. 오브젝트에 대한 권한을 검사해야 하는 엔티티 이름입니다. 문자열의 최대 길이는 12자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

이 엔티티가 근본적인 보안 서비스에 반드시 알려져야 하는 것은 아닙니다. 알려져 있지 않은 경우, 특별 **nobody** 그룹(모든 엔티티가 속하는 것으로 간주됨)의 권한이 검사에 사용됩니다. 공백으로만 구성된 이름은 유효하며 다음과 같이 사용할 수 있습니다.

#### EntityType

유형: MQLONG - 입력

엔티티 유형입니다. 엔티티 유형은 EntityName으로 지정됩니다. 다음 값 중 하나여야 합니다.

#### MQZAET\_PRINCIPAL

프린시펄입니다.

#### MQZAET\_GROUP

그룹.

#### ObjectName

유형: MQCHAR48 - 입력

오브젝트 이름. 액세스가 필요한 오브젝트 이름. 문자열의 최대 길이는 48자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

*ObjectType*이 MQOT\_Q\_MGR인 경우, 이 이름은 *QMgrName*과 동일합니다.

#### ObjectType

유형: MQLONG - 입력

오브젝트 유형. *ObjectName*로 지정된 엔티티의 유형입니다. 다음 값 중 하나여야 합니다.

#### MQOT\_AUTH\_INFO

인증 정보.

#### MQOT\_CHANNEL

채널.

#### MQOT\_CLNTCONN\_CHANNEL

클라이언트 연결 채널.

#### MQOT\_LISTENER

.

**MQOT\_NAMELIST**

이름 목록.

**MQOT\_PROCESS**

process definition.

**MQOT\_Q**

큐.

**MQOT\_Q\_MGR**

큐 관리자.

**MQOT\_SERVICE**

서비스.

**권한**

유형: MQLONG - 입력

검사할 권한. 하나의 권한이 검사되는 경우 이 필드는 적절한 권한 부여 조작(MQZAO\_\* 상수)과 동일합니다. 둘 이상의 권한이 검사되는 경우 해당 MQZAO\_\* 상수의 비트 단위의 OR입니다.

다음 권한이 MQI 호출 사용에 적용됩니다.

**MQZAO\_CONNECT**

MQCONN 호출을 사용할 수 있음.

**MQZAO\_BROWSE**

찾아보기 옵션과 MQGET 호출을 사용할 수 있음.

MQGMO\_BROWSE\_FIRST, MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR 또는 MQGMO\_BROWSE\_NEXT 옵션을 MQGET 호출에 지정할 수 있습니다.

**MQZAO\_INPUT**

프린시펄입니다. 입력 옵션과 MQGET 호출을 사용할 수 있음.

MQOO\_INPUT\_SHARED, MQOO\_INPUT\_EXCLUSIVE 또는 MQOO\_INPUT\_AS\_Q\_DEF 옵션을 MQOPEN 호출에 지정할 수 있습니다.

**MQZAO\_OUTPUT**

MQPUT 호출을 사용할 수 있음.

MQOO\_OUTPUT 옵션을 MQOPEN 호출에 지정할 수 있습니다.

**MQZAO\_INQUIRE**

MQINQ 호출을 사용할 수 있음.

MQOO\_INQUIRE 옵션을 MQOPEN 호출에 지정할 수 있습니다.

**MQZAO\_SET**

MQSET 호출을 사용할 수 있음.

MQOO\_SET 옵션을 MQOPEN 호출에 지정할 수 있습니다.

**MQZAO\_PASS\_IDENTITY\_CONTEXT**

ID 컨텍스트를 전달할 수 있음.

MQOO\_PASS\_IDENTITY\_CONTEXT 옵션을 MQOPEN 호출에 지정하고, MQPMO\_PASS\_IDENTITY\_CONTEXT 옵션을 MQPUT 및 MQPUT1 호출에 지정할 수 있습니다.

**MQZAO\_PASS\_ALL\_CONTEXT**

모든 컨텍스트를 전달할 수 있음.

MQOO\_PASS\_ALL\_CONTEXT 옵션을 MQOPEN 호출에 지정하고, MQPMO\_PASS\_ALL\_CONTEXT 옵션을 MQPUT 및 MQPUT1 호출에 지정할 수 있습니다.

**MQZAO\_SET\_IDENTITY\_CONTEXT**

ID 컨텍스트를 설정할 수 있음.

MQOO\_SET\_IDENTITY\_CONTEXT 옵션을 MQOPEN 호출에 지정하고, MQPMO\_SET\_IDENTITY\_CONTEXT 옵션을 MQPUT 및 MQPUT1 호출에 지정할 수 있습니다.

**MQZAO\_SET\_ALL\_CONTEXT**

모든 컨텍스트를 설정할 수 있음.

MQOO\_SET\_ALL\_CONTEXT 옵션을 MQOPEN 호출에 지정하고, MQPMO\_SET\_ALL\_CONTEXT 옵션을 MQPUT 및 MQPUT1 호출에 지정할 수 있습니다.

**MQZAO\_ALTERNATE\_USER\_AUTHORITY**

대체 사용자 권한을 사용할 수 있음.

MQOO\_ALTERNATE\_USER\_AUTHORITY 옵션을 MQOPEN 호출에 지정하고, MQPMO\_ALTERNATE\_USER\_AUTHORITY 옵션을 MQPUT1 호출에 지정할 수 있습니다.

**MQZAO\_ALL\_MQI**

모든 MQI 권한.

모든 권한을 부여합니다.

다음 권한이 큐 관리자의 관리에 적용됩니다.

**MQZAO\_CREATE**

지정된 유형의 오브젝트를 작성할 수 있음.

**MQZAO\_DELETE**

지정된 오브젝트를 삭제할 수 있음.

**MQZAO\_DISPLAY**

지정된 오브젝트의 속성을 표시할 수 있음.

**MQZAO\_CHANGE**

지정된 오브젝트의 속성을 변경할 수 있음.

**MQZAO\_CLEAR**

지정된 큐에서 모든 메시지를 삭제할 수 있음.

**MQZAO\_AUTHORIZE**

지정된 오브젝트에 대해 다른 사용자에게 권한을 부여할 수 있음.

**MQZAO\_CONTROL**

리스너, 서비스 또는 클라이언트가 아닌 채널 오브젝트 및 클라이언트가 아닌 채널 오브젝트 ping 기능을 시작하거나 중지할 수 있음.

**MQZAO\_CONTROL\_EXTENDED**

순서 번호를 재설정하거나 클라이언트가 아닌 채널 오브젝트에서 인다우트 메시지를 해석할 수 있음.

**MQZAO\_ALL\_ADMIN**

ID 컨텍스트를 설정할 수 있음.

MQZAO\_CREATE를 제외한 모든 관리 권한.

다음 권한이 MQI 사용 및 큐 관리자의 관리에 모두 적용됩니다.

**MQZAO\_ALL**

MQZAO\_CREATE를 제외한 모든 권한.

**MQZAO\_NONE**

권한 없음.

**ComponentData**

유형: MQBYTE x ComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

**Continuation**

유형: MQLONG - 출력

컴포넌트에서 설정한 연속 표시기입니다. 다음 값을 지정할 수 있습니다.

### **MQZCI\_DEFAULT**

큐 관리자에 따른 연속입니다.

MQZ\_CHECK\_AUTHORITY의 경우 MQZCI\_STOP과 동일한 효과가 있습니다.

### **MQZCI\_CONTINUE**

다음 컴포넌트로 계속됩니다.

### **MQZCI\_STOP**

다음 컴포넌트를 계속하지 않습니다.

컴포넌트에 대한 호출이 실패하고(즉, *CompCode*가 MQCC\_FAILED를 리턴함) *Continuation* 매개변수가 MQZCI\_DEFAULT 또는 MQZCI\_CONTINUE이면 큐 관리자가 다른 컴포넌트를 계속 호출합니다(있는 경우).

호출이 성공하면(즉, *CompCode*가 MQCC\_OK를 리턴함) *Continuation*의 설정에 상관없이 다른 컴포넌트가 호출되지 않습니다.

호출이 실패하고 *Continuation* 매개변수가 MQZCI\_STOP이면 다른 컴포넌트가 호출되지 않고 큐 관리자에 오류가 리턴됩니다. 컴포넌트에 이전 호출에 대한 정보가 없으므로 호출 전에 *Continuation* 매개변수가 항상 MQZCI\_DEFAULT로 설정됩니다.

### **CompCode**

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

#### **MQCC\_OK**

정상적으로 완료되었습니다.

#### **MQCC\_FAILED**

호출에 실패했습니다.

### **원인**

유형: MQLONG - 출력

*CompCode*를 규정하는 이유 코드.

*CompCode*가 MQCC\_OK인 경우:

#### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*가 MQCC\_FAILED인 경우:

#### **MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

#### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드](#)를 참조하십시오.

## **C 호출**

```
MQZ_CHECK_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,
                     ObjectType, Authority, ComponentData,
                     &Continuation, &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48  QMgrName;           /* Queue manager name */
MQCHAR12  EntityName;         /* Entity name */
MQLONG    EntityType;         /* Entity type */
MQCHAR48  ObjectName;        /* Object name */
MQLONG    ObjectType;         /* Object type */
MQLONG    Authority;         /* Authority to be checked */
```

```

MQBYTE   ComponentData[n]; /* Component data */
MQLONG   Continuation;   /* Continuation indicator set by
                           component */
MQLONG   CompCode;       /* Completion code */
MQLONG   Reason;         /* Reason code qualifying CompCode */

```

## MQZ\_CHECK\_AUTHORITY\_2 - 권한 검사(확장)

이 함수는 MQZAS\_VERSION\_2 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 지정된 오브젝트에서 특정 조치를 수행할 권한이 엔티티에 있는지 여부를 검사하기 위해 시작합니다.

이 함수(MQZEP용)에 대한 함수 ID는 MQZID\_CHECK\_AUTHORITY입니다.

MQZ\_CHECK\_AUTHORITY\_2는 MQZ\_CHECK\_AUTHORITY와 비슷하지만, **EntityName** 매개변수가 **EntityData** 매개변수로 대체되었습니다.

### 구문

```

MQZ_CHECK_AUTHORITY_2( QMgrName , EntityData , EntityType , ObjectName ,
ObjectType , Authority , ComponentData , Continuation , CompCode , Reason )

```

### 매개변수

#### QMgrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

#### EntityData

유형: MQZED - 입력

엔티티 데이터. 검사할 오브젝트에 대한 권한이 있는 엔티티에 관련된 데이터입니다. 자세한 내용은 [1603 페이지의 『MQZED - 엔티티 디스크립터』](#)의 내용을 참조하십시오.

이 엔티티가 근본적인 보안 서비스에 반드시 알려져야 하는 것은 아닙니다. 알려져 있지 않은 경우, 특별 **nobody** 그룹(모든 엔티티가 속하는 것으로 간주됨)의 권한이 검사에 사용됩니다. 공백으로만 구성된 이름은 유효하며 다음과 같이 사용할 수 있습니다.

#### EntityType

유형: MQLONG - 입력

엔티티 유형입니다. *EntityData*로 지정된 엔티티 유형입니다. 다음 값 중 하나여야 합니다.

#### MQZAET\_PRINCIPAL

프린시펄입니다.

#### MQZAET\_GROUP

그룹.

#### ObjectName

유형: MQCHAR48 - 입력

오브젝트 이름. 액세스가 필요한 오브젝트 이름. 문자열의 최대 길이는 48자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

*ObjectType*이 MQOT\_Q\_MGR인 경우, 이 이름은 *QMgrName*과 동일합니다.

#### ObjectType

유형: MQLONG - 입력

오브젝트 유형. *ObjectName*로 지정된 엔티티 유형입니다. 다음 값 중 하나여야 합니다.

#### MQOT\_AUTH\_INFO

인증 정보.

**MQOT\_CHANNEL**

채널.

**MQOT\_CLNTCONN\_CHANNEL**

클라이언트 연결 채널.

**MQOT\_LISTENER**

.

**MQOT\_NAMELIST**

이름 목록.

**MQOT\_PROCESS**

process definition.

**MQOT\_Q**

큐.

**MQOT\_Q\_MGR**

큐 관리자.

**MQOT\_SERVICE**

서비스.

**MQOT\_TOPIC**

있습니다.

**권한**

유형: MQLONG - 입력

검사할 권한. 하나의 권한이 검사되는 경우 이 필드는 적절한 권한 부여 조작(MQZAO\_\* 상수)과 동일합니다. 둘 이상의 권한이 검사되는 경우 해당 MQZAO\_\* 상수의 비트 단위의 OR입니다.

다음 권한이 MQI 호출 사용에 적용됩니다.

**MQZAO\_CONNECT**

MQCONN 호출을 사용할 수 있음.

**MQZAO\_BROWSE**

찾아보기 옵션과 MQGET 호출을 사용할 수 있음.

MQGMO\_BROWSE\_FIRST, MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR 또는 MQGMO\_BROWSE\_NEXT 옵션을 MQGET 호출에 지정할 수 있습니다.

**MQZAO\_INPUT**

프린시펄입니다. 입력 옵션과 MQGET 호출을 사용할 수 있음.

MQOO\_INPUT\_SHARED, MQOO\_INPUT\_EXCLUSIVE 또는 MQOO\_INPUT\_AS\_Q\_DEF 옵션을 MQOPEN 호출에 지정할 수 있습니다.

**MQZAO\_OUTPUT**

MQPUT 호출을 사용할 수 있음.

MQOO\_OUTPUT 옵션을 MQOPEN 호출에 지정할 수 있습니다.

**MQZAO\_INQUIRE**

MQINQ 호출을 사용할 수 있음.

MQOO\_INQUIRE 옵션을 MQOPEN 호출에 지정할 수 있습니다.

**MQZAO\_SET**

MQSET 호출을 사용할 수 있음.

MQOO\_SET 옵션을 MQOPEN 호출에 지정할 수 있습니다.

**MQZAO\_PASS\_IDENTITY\_CONTEXT**

ID 컨텍스트를 전달할 수 있음.

MQOO\_PASS\_IDENTITY\_CONTEXT 옵션을 MQOPEN 호출에 지정하고, MQPMO\_PASS\_IDENTITY\_CONTEXT 옵션을 MQPUT 및 MQPUT1 호출에 지정할 수 있습니다.

**MQZAO\_PASS\_ALL\_CONTEXT**

모든 컨텍스트를 전달할 수 있음.

MQOO\_PASS\_ALL\_CONTEXT 옵션을 MQOPEN 호출에 지정하고, MQPMO\_PASS\_ALL\_CONTEXT 옵션을 MQPUT 및 MQPUT1 호출에 지정할 수 있습니다.

**MQZAO\_SET\_IDENTITY\_CONTEXT**

ID 컨텍스트를 설정할 수 있음.

MQOO\_SET\_IDENTITY\_CONTEXT 옵션을 MQOPEN 호출에 지정하고, MQPMO\_SET\_IDENTITY\_CONTEXT 옵션을 MQPUT 및 MQPUT1 호출에 지정할 수 있습니다.

**MQZAO\_SET\_ALL\_CONTEXT**

모든 컨텍스트를 설정할 수 있음.

MQOO\_SET\_ALL\_CONTEXT 옵션을 MQOPEN 호출에 지정하고, MQPMO\_SET\_ALL\_CONTEXT 옵션을 MQPUT 및 MQPUT1 호출에 지정할 수 있습니다.

**MQZAO\_ALTERNATE\_USER\_AUTHORITY**

대체 사용자 권한을 사용할 수 있음.

MQOO\_ALTERNATE\_USER\_AUTHORITY 옵션을 MQOPEN 호출에 지정하고, MQPMO\_ALTERNATE\_USER\_AUTHORITY 옵션을 MQPUT1 호출에 지정할 수 있습니다.

**MQZAO\_ALL\_MQI**

모든 MQI 권한.

모든 권한을 부여합니다.

다음 권한이 큐 관리자의 관리에 적용됩니다.

**MQZAO\_CREATE**

지정된 유형의 오브젝트를 작성할 수 있음.

**MQZAO\_DELETE**

지정된 오브젝트를 삭제할 수 있음.

**MQZAO\_DISPLAY**

지정된 오브젝트의 속성을 표시할 수 있음.

**MQZAO\_CHANGE**

지정된 오브젝트의 속성을 변경할 수 있음.

**MQZAO\_CLEAR**

지정된 큐에서 모든 메시지를 삭제할 수 있음.

**MQZAO\_AUTHORIZE**

지정된 오브젝트에 대해 다른 사용자에게 권한을 부여할 수 있음.

**MQZAO\_CONTROL**

리스너, 서비스 또는 클라이언트가 아닌 채널 오브젝트 및 클라이언트가 아닌 채널 오브젝트 ping 기능을 시작하거나 중지할 수 있음.

**MQZAO\_CONTROL\_EXTENDED**

순서 번호를 재설정하거나 클라이언트가 아닌 채널 오브젝트에서 인다우트 메시지를 해석할 수 있음.

**MQZAO\_ALL\_ADMIN**

ID 컨텍스트를 설정할 수 있음.

MQZAO\_CREATE를 제외한 모든 관리 권한.

다음 권한이 MQI 사용 및 큐 관리자의 관리에 모두 적용됩니다.

**MQZAO\_ALL**

MQZAO\_CREATE를 제외한 모든 권한.

**MQZAO\_NONE**

권한 없음.

**ComponentData**

유형: MQBYTE x ComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

#### Continuation

유형: MQLONG - 출력

컴포넌트에서 설정한 연속 표시기입니다. 다음 값을 지정할 수 있습니다.

#### MQZCI\_DEFAULT

큐 관리자에 따른 연속입니다.

MQZ\_CHECK\_AUTHORITY의 경우 MQZCI\_STOP과 동일한 효과가 있습니다.

#### MQZCI\_CONTINUE

다음 컴포넌트로 계속됩니다.

#### MQZCI\_STOP

다음 컴포넌트를 계속하지 않습니다.

#### CompCode

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_FAILED

호출에 실패했습니다.

#### 원인

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드.

CompCode가 MQCC\_OK인 경우:

#### MQRC\_NONE

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_FAILED인 경우:

#### MQRC\_NOT\_AUTHORIZED

(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

#### MQRC\_SERVICE\_ERROR

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

#### MQRC\_SERVICE\_NOT\_AVAILABLE

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드](#)를 참조하십시오.

## C 호출

```
MQZ_CHECK_AUTHORITY_2 (QMgrName, &EntityData, EntityType,
                        ObjectName, ObjectType, Authority, ComponentData,
                        &Continuation, &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48 QMgrName;          /* Queue manager name */
MQZED EntityData;          /* Entity data */
MQLONG EntityType;        /* Entity type */
MQCHAR48 ObjectName;      /* Object name */
```



```

MQLONG      ObjectType;          /* Object type */
MQLONG      Authority;         /* Authority to be checked */
MQBYTE      ComponentData[n]; /* Component data */
MQLONG      Continuation;     /* Continuation indicator set by
                               component */
MQLONG      CompCode;         /* Completion code */
MQLONG      Reason;          /* Reason code qualifying CompCode */

```

## MQZ\_CHECK\_PRIVILEGED - 사용자에게 권한이 있는지 검사

이 함수는 MQZAS\_VERSION\_6 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 지정된 사용자가 권한이 있는 사용자인지 여부를 판별하기 위해 호출합니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_CHECK\_PRIVILEGED입니다.

### 구문

```

MQZ_CHECK_PRIVILEGED( QMgrName , EntityData , EntityType , ComponentData ,
Continuation , CompCode , Reason )

```

### 매개변수

#### QMgrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

#### EntityData

유형: MQZED - 입력

엔티티 데이터. 검사할 엔티티에 관련된 데이터입니다. 자세한 정보는 [1603 페이지의 『MQZED - 엔티티 디스크립터』](#)의 내용을 참조하십시오.

#### EntityType

유형: MQLONG - 입력

엔티티 유형입니다. EntityData로 지정된 엔티티 유형입니다. 다음 값 중 하나여야 합니다.

#### MQZAET\_PRINCIPAL

프린시펄입니다.

#### MQZAET\_GROUP

그룹.

#### ComponentData

유형: MQBYTEExComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공하는 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

#### Continuation

유형: MQLONG - 출력

컴포넌트에서 설정한 연속 표시기입니다. 다음 값을 지정할 수 있습니다.

#### MQZCI\_DEFAULT

큐 관리자에 따른 연속입니다.

MQZ\_CHECK\_AUTHORITY의 경우 MQZCI\_STOP과 동일한 효과가 있습니다.

## MQZCI\_CONTINUE

다음 컴포넌트로 계속됩니다.

## MQZCI\_STOP

다음 컴포넌트를 계속하지 않습니다.

컴포넌트에 대한 호출이 실패하고(즉, *CompCode*가 MQCC\_FAILED를 리턴함) *Continuation* 매개변수가 MQZCI\_DEFAULT 또는 MQZCI\_CONTINUE이면 큐 관리자가 다른 컴포넌트를 계속 호출합니다(있는 경우).

호출이 성공하면(즉, *CompCode*가 MQCC\_OK를 리턴함) *Continuation*의 설정에 상관없이 다른 컴포넌트가 호출되지 않습니다.

호출이 실패하고 *Continuation* 매개변수가 MQZCI\_STOP이면 다른 컴포넌트가 호출되지 않고 큐 관리자에 오류가 리턴됩니다. 컴포넌트에 이전 호출에 대한 정보가 없으므로 호출 전에 *Continuation* 매개변수가 항상 MQZCI\_DEFAULT로 설정됩니다.

## CompCode

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

## MQCC\_OK

정상적으로 완료되었습니다.

## MQCC\_FAILED

호출에 실패했습니다.

## 원인

유형: MQLONG - 출력

*CompCode*를 규정하는 이유 코드.

*CompCode*가 MQCC\_OK인 경우:

## MQRC\_NONE

(0, X'000') 보고할 이유가 없습니다.

*CompCode*가 MQCC\_FAILED인 경우:

## MQRC\_NOT\_PRIVILEGED

(2584, X'A18') 이 사용자는 권한이 있는 사용자 ID가 아닙니다.

## MQRC\_UNKNOWN\_ENTITY

(2292, X'8F4') 서비스에서 알 수 없는 엔티티입니다.

## MQRC\_SERVICE\_ERROR

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

## MQRC\_SERVICE\_NOT\_AVAILABLE

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드](#)를 참조하십시오.

## C 호출

```
MQZ_CHECK_PRIVILEGED (QMgrName, &EntityData, EntityType,
                      ComponentData, &Continuation,
                      &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48 QMgrName;          /* Queue manager name */
MQZED    EntityData;        /* Entity name */
MQLONG   EntityType;        /* Entity type */
MQBYTE   ComponentData[n]; /* Component data */
MQLONG   Continuation;      /* Continuation indicator set by
                             component */
MQLONG   CompCode;          /* Completion code */
MQLONG   Reason;           /* Reason code qualifying CompCode */
```

## MQZ\_COPY\_ALL\_AUTHORITY - 모든 권한 복사

이 함수는 권한 서비스 컴포넌트에서 제공됩니다. 큐 관리자가 현재 참조 오브젝트에 적용되는 모든 권한을 다른 오브젝트에 복사하기 위해 시작합니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_COPY\_ALL\_AUTHORITY입니다.

### 구문

MQZ\_COPY\_ALL\_AUTHORITY( QMgrName , RefObjectName , ObjectName , ObjectType , ComponentData , Continuation , CompCode , Reason )

### 매개변수

#### QMgrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

#### RefObjectName

유형: MQCHAR48 - 입력

참조 오브젝트 이름. 권한을 복사할 참조 오브젝트의 이름입니다. 문자열의 최대 길이는 48자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

#### ObjectName

유형: MQCHAR48 - 입력

오브젝트 이름. 액세스 권한을 설정할 오브젝트의 이름입니다. 문자열의 최대 길이는 48자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

#### ObjectType

유형: MQLONG - 입력

오브젝트 유형. *RefObjectName* 및 *ObjectName*으로 지정된 엔티티 유형입니다. 다음 값 중 하나여야 합니다.

#### MQOT\_AUTH\_INFO

인증 정보.

#### MQOT\_CHANNEL

채널.

#### MQOT\_CLNTCONN\_CHANNEL

클라이언트 연결 채널.

#### MQOT\_LISTENER

.

#### MQOT\_NAMELIST

이름 목록.

#### MQOT\_PROCESS

process definition.

#### MQOT\_Q

큐.

#### MQOT\_Q\_MGR

큐 관리자.

#### MQOT\_SERVICE

서비스.

## **MQOT\_TOPIC**

있습니다.

### **ComponentData**

유형: MQBYTEExComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 ComponentDataLength 매개변수에서 큐 관리자에 의해 전달됩니다.

### **Continuation**

유형: MQLONG - 출력

컴포넌트에서 설정한 연속 표시기입니다. 다음 값을 지정할 수 있습니다.

#### **MQZCI\_DEFAULT**

큐 관리자에 따른 연속입니다.

MQZ\_CHECK\_AUTHORITY의 경우 MQZCI\_STOP과 동일한 효과가 있습니다.

#### **MQZCI\_CONTINUE**

다음 컴포넌트로 계속됩니다.

#### **MQZCI\_STOP**

다음 컴포넌트를 계속하지 않습니다.

### **CompCode**

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

#### **MQCC\_OK**

정상적으로 완료되었습니다.

#### **MQCC\_FAILED**

호출에 실패했습니다.

### **원인**

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드.

CompCode가 MQCC\_OK인 경우:

#### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_FAILED인 경우:

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

#### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

#### **MQRC\_UNKNOWN\_REF\_OBJECT**

(2294, X'8F6') 알 수 없는 참조 오브젝트.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드를 참조하십시오](#).

## **C 호출**

```
MQZ_COPY_ALL_AUTHORITY (QMgrName, RefObjectName, ObjectName, ObjectType,  
ComponentData, &Continuation, &CompCode,  
&Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48 QMgrName;          /* Queue manager name */
MQCHAR48 RefObjectName;     /* Reference object name */
MQCHAR48 ObjectName;       /* Object name */
MQLONG   ObjectType;       /* Object type */
MQBYTE   ComponentData[n]; /* Component data */
MQLONG   Continuation;     /* Continuation indicator set by
                             component */
MQLONG   CompCode;        /* Completion code */
MQLONG   Reason;         /* Reason code qualifying CompCode */
```

## MQZ\_DELETE\_AUTHORITY - 권한 삭제

이 함수는 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 지정된 오브젝트와 연관된 모든 권한을 삭제하기 위해 시작합니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_DELETE\_AUTHORITY입니다.

### 구문

```
MQZ_DELETE_AUTHORITY( QMgrName , ObjectName , ObjectType , ComponentData ,
Continuation , CompCode , Reason )
```

### 매개변수

#### QMgrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

#### ObjectName

유형: MQCHAR48 - 입력

오브젝트 이름. 액세스 권한을 삭제할 오브젝트의 이름입니다. 문자열의 최대 길이는 48자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

*ObjectType*이 MQOT\_Q\_MGR인 경우, 이 이름은 *QMgrName*과 동일합니다.

#### ObjectType

유형: MQLONG - 입력

오브젝트 유형. *ObjectName*로 지정된 엔티티의 유형입니다. 다음 값 중 하나여야 합니다.

#### MQOT\_AUTH\_INFO

인증 정보.

#### MQOT\_CHANNEL

채널.

#### MQOT\_CLNTCONN\_CHANNEL

클라이언트 연결 채널.

#### MQOT\_LISTENER

.

#### MQOT\_NAMELIST

이름 목록.

#### MQOT\_PROCESS

process definition.

#### MQOT\_Q

큐.

**MQOT\_Q\_MGR**

큐 관리자.

**MQOT\_SERVICE**

서비스.

**MQOT\_TOPIC**

있습니다.

**ComponentData**

유형: MQBYTE x ComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 ComponentDataLength 매개변수에서 큐 관리자에 의해 전달됩니다.

**Continuation**

유형: MQLONG - 출력

컴포넌트에서 설정한 연속 표시기입니다. 다음 값을 지정할 수 있습니다.

**MQZCI\_DEFAULT**

큐 관리자에 따른 연속입니다.

MQZ\_CHECK\_AUTHORITY의 경우 MQZCI\_STOP과 동일한 효과가 있습니다.

**MQZCI\_CONTINUE**

다음 컴포넌트로 계속됩니다.

**MQZCI\_STOP**

다음 컴포넌트를 계속하지 않습니다.

**CompCode**

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_FAILED**

호출에 실패했습니다.

**원인**

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드.

CompCode가 MQCC\_OK인 경우:

**MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_FAILED인 경우:

**MQRC\_SERVICE\_ERROR**

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

**MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드](#)를 참조하십시오.

## C 호출

```
MQZ_DELETE_AUTHORITY (QMGrName, ObjectName, ObjectType, ComponentData,  
&Continuation, &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48 QMGrName;          /* Queue manager name */  
MQCHAR48 ObjectName;       /* Object name */  
MQLONG   ObjectType;       /* Object type */  
MQBYTE   ComponentData[n]; /* Component data */  
MQLONG   Continuation;     /* Continuation indicator set by  
                           component */  
MQLONG   CompCode;         /* Completion code */  
MQLONG   Reason;          /* Reason code qualifying CompCode */
```

## MQZ\_ENUMERATE\_AUTHORITY\_DATA - 권한 데이터 나열

이 함수는 MQZAS\_VERSION\_4 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 첫 번째 호출에서 지정된 선택 기준에 일치하는 모든 권한 데이터를 검색하기 위해 반복적으로 시작합니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_ENUMERATE\_AUTHORITY\_DATA입니다.

### 구문

```
MQZ_ENUMERATE_AUTHORITY_DATA( QMGrName , StartEnumeration , Filter ,  
AuthorityBufferLength , AuthorityBuffer , AuthorityDataLength , ComponentData ,  
Continuation , CompCode , Reason )
```

### 매개변수

#### QMGrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

#### StartEnumeration

유형: MQLONG - 입력

호출에서 나열을 시작할 수 있는지 표시하는 플래그. 이 플래그는 호출이 권한 데이터의 나열을 시작하거나 MQZ\_ENUMERATE\_AUTHORITY\_DATA에 대한 이전 호출로 인해 시작된 권한 데이터 나열을 계속할 수 있는지 여부를 표시합니다. 값은 다음 중 하나입니다.

#### MQZSE\_START

나열 시작. 권한 데이터의 나열을 시작하려면 이 값으로 호출을 시작합니다. **Filter** 매개변수는 이 호출 및 이후 호출에서 리턴된 권한 데이터를 선택하는 데 사용할 선택 기준을 지정합니다.

#### MQZSE\_CONTINUE

나열 계속. 권한 데이터의 나열을 계속하려면 이 값으로 호출을 시작합니다. 이 경우 **Filter** 매개변수는 무시되며 널 포인터로 지정될 수 있습니다 (선택 기준은 *StartEnumeration* 가 MQZSE\_START로 설정된 호출에 의해 지정된 **Filter** 매개변수에 의해 판별됨).

### 필터

유형: MQZAD - 입력

필터. *StartEnumeration*이 MQZSE\_START이면 *Filter*는 리턴할 권한 데이터를 선택하는 데 사용할 선택 기준을 지정합니다. *Filter*가 널 포인터이면 선택 기준이 사용되지 않습니다. 즉, 모든 권한 데이터가 리턴됩니다. 사용할 수 있는 선택 기준에 대한 자세한 정보는 1600 페이지의 『MQZAD - 권한 데이터』의 내용을 참조하십시오.

*StartEnumeration*이 MQZSE\_CONTINUE이면 *Filter*가 무시되고 널 포인터로 지정할 수 있습니다.

### **AuthorityBufferLength**

유형: MQLONG - 입력

*AuthorityBuffer*의 길이. 이는 **AuthorityBuffer** 매개변수의 길이(바이트)입니다. 권한 버퍼는 리턴되는 데이터를 수용할 만큼 충분한 크기여야 합니다.

### **AuthorityBuffer**

유형: MQZAD - 출력

권한 데이터. 권한 데이터가 리턴되는 버퍼입니다. 버퍼는 MQZAD 구조, MQZED 구조, 그리고 정의된 가장 긴 엔티티 이름과 가장 긴 도메인 이름을 수용할 만큼 충분한 크기여야 합니다.

**참고:** 참고: MQZAD가 항상 버퍼 시작 부분에 발생하므로 이 매개변수는 MQZAD로 정의됩니다. 그러나, 버퍼가 MQZAD로 선언되면 버퍼가 너무 작습니다. MQZAD, MQZED 그리고 엔티티 및 도메인 이름을 수용할 수 있도록 MQZAD보다 커야 합니다.

### **AuthorityDataLength**

유형: MQLONG - 출력

*AuthorityBuffer*에서 리턴된 데이터 길이. 권한 버퍼가 너무 작으면 *AuthorityDataLength*를 필요한 버퍼의 길이로 설정합니다. 그러면 호출이 완료 코드 MQCC\_FAILED 및 이유 코드 MQRC\_BUFFER\_LENGTH\_ERROR를 리턴합니다.

### **ComponentData**

유형: MQBYTE x ComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 ComponentDataLength 매개변수에서 큐 관리자에 의해 전달됩니다.

### **Continuation**

유형: MQLONG - 출력

컴포넌트에서 설정한 연속 표시기입니다. 다음 값을 지정할 수 있습니다.

#### **MQZCI\_DEFAULT**

큐 관리자에 따른 연속입니다.

MQZ\_ENUMERATE\_AUTHORITY\_DATA의 경우 이는 MQZCI\_CONTINUE와 동일한 효과가 있습니다.

#### **MQZCI\_CONTINUE**

다음 컴포넌트로 계속됩니다.

#### **MQZCI\_STOP**

다음 컴포넌트를 계속하지 않습니다.

### **CompCode**

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

#### **MQCC\_OK**

정상적으로 완료되었습니다.

#### **MQCC\_FAILED**

호출에 실패했습니다.

### **원인**

유형: MQLONG - 출력

*CompCode*를 규정하는 이유 코드.

*CompCode*가 MQCC\_OK인 경우:



### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_FAILED인 경우:

### **MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'7D5') 버퍼 길이 매개변수가 올바르지 않습니다.

### **MQRC\_NO\_DATA\_AVAILABLE**

(2379, X'94B') 데이터가 없습니다.

### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드](#)를 참조하십시오.

## **C 호출**

```
MQZ_ENUMERATE_AUTHORITY_DATA (QMgrName, StartEnumeration, &Filter,
                               AuthorityBufferLength,
                               &AuthorityBuffer,
                               &AuthorityDataLength, ComponentData,
                               &Continuation, &CompCode,
                               &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48  QMgrName;           /* Queue manager name */
MQLONG    StartEnumeration;  /* Flag indicating whether call should
                               start enumeration */

MQZAD     Filter;            /* Filter */
MQLONG    AuthorityBufferLength; /* Length of AuthorityBuffer */
MQZAD     AuthorityBuffer;   /* Authority data */
MQLONG    AuthorityDataLength; /* Length of data returned in
                               AuthorityBuffer */

MQBYTE    ComponentData[n];  /* Component data */
MQLONG    Continuation;      /* Continuation indicator set by
                               component */

MQLONG    CompCode;         /* Completion code */
MQLONG    Reason;          /* Reason code qualifying CompCode */
```

## **MQZ\_FREE\_USER - 프리 사용자**

이 함수는 MQZAS\_VERSION\_5 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 연관된 할당 자원을 비우기 위해 시작합니다.

애플리케이션이 모든 사용자 컨텍스트(예: MQDISC MQI 호출 동안)에서 실행을 마치면 시작됩니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_FREE\_USER입니다.

### **구문**

```
MQZ_FREE_USER( QMgrName , FreeParms , ComponentData , Continuation , CompCode ,
               Reason )
```

### **매개변수**

#### **QMgrName**

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

## FreeParms

유형: MQZFP - 입력

비우기 매개변수. 비우려는 자원과 관련된 데이터를 포함한 구조입니다. 자세한 내용은 [1605 페이지의 『MQZFP - 매개변수 비우기』](#)의 내용을 참조하십시오.

## ComponentData

유형: MQBYTE x ComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 ComponentDataLength 매개변수에서 큐 관리자에 의해 전달됩니다.

## Continuation

유형: MQLONG - 출력

연속 플래그입니다. 다음 값을 지정할 수 있습니다.

### MQZCI\_DEFAULT

기타 컴포넌트에 종속되는 연속입니다.

### MQZCI\_STOP

다음 컴포넌트를 계속하지 않습니다.

## CompCode

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

### MQCC\_OK

정상적으로 완료되었습니다.

### MQCC\_FAILED

호출에 실패했습니다.

## 원인

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드.

CompCode가 MQCC\_OK인 경우:

### MQRC\_NONE

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_FAILED인 경우:

### MQRC\_SERVICE\_ERROR

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드를 참조하십시오](#).

## C 호출

```
MQZ_AUTHENTICATE_USER (QMgrName, SecurityParms, ApplicationContext,  
IdentityContext, CorrelationPtr, ComponentData,  
&Continuation, &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48  QMgrName;           /* Queue manager name */  
MQZFP     FreeParms;         /* Resource to be freed */  
MQBYTE    ComponentData[n]; /* Component data */  
MQLONG    Continuation;     /* Continuation indicator set by  
                             component */
```

```
MQLONG    CompCode;          /* Completion code */
MQLONG    Reason;           /* Reason code qualifying CompCode */
```

## MQZ\_GET\_AUTHORITY - 권한 가져오기

이 함수는 MQZAS\_VERSION\_1 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 프린시펄이 멤버인 그룹이 소유한 권한(엔티티가 프린시펄인 경우)을 포함하여 지정된 오브젝트에 액세스할 수 있는 엔티티의 권한을 검색하기 위해 시작합니다. 일반 프로파일에 있는 권한은 리턴된 권한 세트에 포함됩니다.

이 함수(MQZEP용)에 대한 함수 ID는 MQZID\_GET\_AUTHORITY입니다.

### 구문

```
MQZ_GET_AUTHORITY( QMgrName , EntityName , EntityType , ObjectName ,  
ObjectType , Authority , ComponentData , Continuation , CompCode , Reason )
```

### 매개변수

#### QMgrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

#### EntityName

유형: MQCHAR12 - 입력

엔티티 이름입니다. 오브젝트에 대한 액세스 권한을 검색할 엔티티의 이름입니다. 문자열의 최대 길이는 12 자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

#### EntityType

유형: MQLONG - 입력

엔티티 유형입니다. 엔티티 유형은 *EntityName*으로 지정됩니다. 다음 값 중 하나여야 합니다.

#### MQZAET\_PRINCIPAL

프린시펄입니다.

#### MQZAET\_GROUP

그룹.

#### ObjectName

유형: MQCHAR48 - 입력

오브젝트 이름. 액세스 권한이 검색되는 오브젝트의 이름입니다. 문자열의 최대 길이는 48자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

*ObjectType*이 MQOT\_Q\_MGR인 경우, 이 이름은 *QMgrName*과 동일합니다.

#### ObjectType

유형: MQLONG - 입력

오브젝트 유형. 엔티티 유형은 *ObjectName*으로 지정됩니다. 다음 값 중 하나여야 합니다.

#### MQOT\_AUTH\_INFO

인증 정보.

#### MQOT\_CHANNEL

채널.

#### MQOT\_CLNTCONN\_CHANNEL

클라이언트 연결 채널.

#### MQOT\_LISTENER

.

**MQOT\_NAMELIST**

이름 목록.

**MQOT\_PROCESS**

process definition.

**MQOT\_Q**

큐.

**MQOT\_Q\_MGR**

큐 관리자.

**MQOT\_SERVICE**

서비스.

**MQOT\_TOPIC**

있습니다.

**권한**

유형: MQLONG - 입력

엔티티의 권한입니다. 엔티티에 하나의 권한이 있는 경우, 이 필드는 적절한 권한 부여 조작(MQZAO\_\* constant)과 동일합니다. 둘 이상의 권한이 있는 경우, 이 필드는 해당되는 MQZAO\_\* 상수의 비트 단위의 OR 입니다.

**ComponentData**

유형: MQBYTE×ComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

**Continuation**

유형: MQLONG - 출력

컴포넌트에서 설정한 연속 표시기입니다. 다음 값을 지정할 수 있습니다.

**MQZCI\_DEFAULT**

큐 관리자에 따른 연속입니다.

MQZ\_GET\_AUTHORITY의 경우 이는 MQZCI\_CONTINUE와 효과가 동일합니다.

**MQZCI\_CONTINUE**

다음 컴포넌트로 계속됩니다.

**MQZCI\_STOP**

다음 컴포넌트를 계속하지 않습니다.

**CompCode**

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_FAILED**

호출에 실패했습니다.

**원인**

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드.

CompCode가 MQCC\_OK인 경우:

**MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_FAILED인 경우:

**MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

**MQRC\_SERVICE\_ERROR**

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

**MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

**MQRC\_UNKNOWN\_ENTITY**

(2292, X'8F4') 서비스에서 알 수 없는 엔티티입니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드](#)를 참조하십시오.

## C 호출

```
MQZ_GET_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,  
ObjectTypes, &Authority, ComponentData,  
&Continuation, &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48 QMgrName;          /* Queue manager name */  
MQCHAR12 EntityName;       /* Entity name */  
MQLONG EntityType;        /* Entity type */  
MQCHAR48 ObjectName;      /* Object name */  
MQLONG ObjectTypes;       /* Object type */  
MQLONG Authority;         /* Authority of entity */  
MQBYTE ComponentData[n];  /* Component data */  
MQLONG Continuation;      /* Continuation indicator set by  
                           component */  
MQLONG CompCode;          /* Completion code */  
MQLONG Reason;            /* Reason code qualifying CompCode */
```

## MQZ\_GET\_AUTHORITY\_2 - 권한 가져오기(확장)

이 함수는 MQZAS\_VERSION\_2 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 지정된 오브젝트에 액세스하는 데 필요한 엔티티의 권한을 검색하기 위해 시작합니다.

이 함수(MQZEP용)에 대한 함수 ID는 MQZID\_GET\_AUTHORITY입니다.

MQZ\_GET\_AUTHORITY\_2는 MQZ\_GET\_AUTHORITY와 비슷하지만, **EntityName** 매개변수가 **EntityData** 매개변수로 대체되었습니다.

### 구문

```
MQZ_GET_AUTHORITY_2( QMgrName , EntityData , EntityType , ObjectName ,  
ObjectTypes , Authority , ComponentData , Continuation , CompCode , Reason )
```

### 매개변수

**QMgrName**

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

**EntityData**

유형: MQZED - 입력

엔티티 데이터. 검색되는 오브젝트에 대한 권한이 있는 엔티티에 관련된 데이터입니다. 자세한 내용은 [1603 페이지](#)의 『MQZED - 엔티티 디스크립터』의 내용을 참조하십시오.

### EntityType

유형: MQLONG - 입력

엔티티 유형입니다. *EntityData*로 지정된 엔티티 유형입니다. 다음 값 중 하나여야 합니다.

#### **MQZAET\_PRINCIPAL**

프린시펄입니다.

#### **MQZAET\_GROUP**

그룹.

### ObjectName

유형: MQCHAR48 - 입력

오브젝트 이름. 엔티티 권한이 검색될 오브젝트의 이름입니다. 문자열의 최대 길이는 48자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

*ObjectType*이 MQOT\_Q\_MGR인 경우, 이 이름은 *QMgrName*과 동일합니다.

### ObjectType

유형: MQLONG - 입력

오브젝트 유형. *ObjectName*로 지정된 엔티티 유형입니다. 다음 값 중 하나여야 합니다.

#### **MQOT\_AUTH\_INFO**

인증 정보.

#### **MQOT\_CHANNEL**

채널.

#### **MQOT\_CLNTCONN\_CHANNEL**

클라이언트 연결 채널.

#### **MQOT\_LISTENER**

.

#### **MQOT\_NAMELIST**

이름 목록.

#### **MQOT\_PROCESS**

process definition.

#### **MQOT\_Q**

큐.

#### **MQOT\_Q\_MGR**

큐 관리자.

#### **MQOT\_SERVICE**

서비스.

#### **MQOT\_TOPIC**

있습니다.

### 권한

유형: MQLONG - 입력

엔티티의 권한입니다. 엔티티에 하나의 권한이 있는 경우, 이 필드는 적절한 권한 부여 조작(MQZAO\_\* constant)과 동일합니다. 둘 이상의 권한이 있는 경우, 이 필드는 해당되는 MQZAO\_\* 상수의 비트 단위의 OR입니다.

### ComponentData

유형: MQBYTE×ComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

#### Continuation

유형: MQLONG - 출력

컴포넌트에서 설정한 연속 표시기입니다. 다음 값을 지정할 수 있습니다.

#### MQZCI\_DEFAULT

큐 관리자에 따른 연속입니다.

MQZ\_CHECK\_AUTHORITY의 경우 MQZCI\_STOP과 동일한 효과가 있습니다.

#### MQZCI\_CONTINUE

다음 컴포넌트로 계속됩니다.

#### MQZCI\_STOP

다음 컴포넌트를 계속하지 않습니다.

#### CompCode

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_FAILED

호출에 실패했습니다.

#### 원인

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드.

CompCode가 MQCC\_OK인 경우:

#### MQRC\_NONE

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_FAILED인 경우:

#### MQRC\_NOT\_AUTHORIZED

(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

#### MQRC\_SERVICE\_ERROR

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

#### MQRC\_SERVICE\_NOT\_AVAILABLE

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

#### MQRC\_UNKNOWN\_ENTITY

(2292, X'8F4') 서비스에서 알 수 없는 엔티티입니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드](#)를 참조하십시오.

## C 호출

```
MQZ_GET_AUTHORITY_2 (QMgrName, &EntityData, EntityType, ObjectName,  
                    ObjectType, &Authority, ComponentData,  
                    &Continuation, &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48 QMgrName;          /* Queue manager name */  
MQZED EntityData;          /* Entity data */  
MQLONG EntityType;        /* Entity type */  
MQCHAR48 ObjectName;       /* Object name */  
MQLONG ObjectType;        /* Object type */  
MQLONG Authority;         /* Authority of entity */
```

```

MQBYTE   ComponentData[n]; /* Component data */
MQLONG   Continuation;   /* Continuation indicator set by
                           component */
MQLONG   CompCode;       /* Completion code */
MQLONG   Reason;         /* Reason code qualifying CompCode */

```

## MQZ\_GET\_EXPLICIT\_AUTHORITY - 명시적 권한 가져오기

이 함수는 MQZAS\_VERSION\_1 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 프린시펄이 멤버인 그룹이 소유한 권한(엔티티가 프린시펄인 경우)을 포함하여 지정된 오브젝트에 액세스할 수 있는 엔티티의 권한을 검색하기 위해 시작합니다. 일반 프로파일에 있는 권한은 리턴된 권한 세트에 포함됩니다.

AIX and Linux에서 내장 IBM MQ 오브젝트 권한 관리자(OAM)에 대해 리턴된 권한은 프린시펄의 1차 그룹만 소유한 권한입니다.

이 함수(MQZEP용)에 대한 함수 ID는 MQZID\_GET\_EXPLICIT\_AUTHORITY입니다.

### 구문

```

MQZ_GET_EXPLICIT_AUTHORITY( QMgrName , EntityName , EntityType , ObjectName ,
ObjectType , Authority , ComponentData , Continuation , CompCode , Reason )

```

### 매개변수

#### QMgrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

#### EntityName

유형: MQCHAR12 - 입력

엔티티 이름입니다. 오브젝트에 대한 액세스 권한을 검색할 엔티티의 이름입니다. 문자열의 최대 길이는 12자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

#### EntityType

유형: MQLONG - 입력

엔티티 유형입니다. 엔티티 유형은 *EntityName*으로 지정됩니다. 다음 값 중 하나여야 합니다.

#### MQZAET\_PRINCIPAL

프린시펄입니다.

#### MQZAET\_GROUP

그룹.

#### ObjectName

유형: MQCHAR48 - 입력

오브젝트 이름. 엔티티 권한이 검색될 오브젝트의 이름입니다. 문자열의 최대 길이는 48자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

*ObjectType*이 MQOT\_Q\_MGR인 경우, 이 이름은 *QMgrName*과 동일합니다.

#### ObjectType

유형: MQLONG - 입력

오브젝트 유형. 엔티티 유형은 *ObjectName*으로 지정됩니다. 다음 값 중 하나여야 합니다.

#### MQOT\_AUTH\_INFO

인증 정보.

#### MQOT\_CHANNEL

채널.



**MQOT\_CLNTCONN\_CHANNEL**

클라이언트 연결 채널.

**MQOT\_LISTENER**

.

**MQOT\_NAMELIST**

이름 목록.

**MQOT\_PROCESS**

process definition.

**MQOT\_Q**

큐.

**MQOT\_Q\_MGR**

큐 관리자.

**MQOT\_SERVICE**

서비스.

**MQOT\_TOPIC**

있습니다.

**권한**

유형: MQLONG - 입력

엔티티의 권한입니다. 엔티티에 하나의 권한이 있는 경우, 이 필드는 적절한 권한 부여 조작(MQZAO\_\* constant)과 동일합니다. 둘 이상의 권한이 있는 경우, 이 필드는 해당되는 MQZAO\_\* 상수의 비트 단위의 OR 입니다.

**ComponentData**

유형: MQBYTE x ComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

**Continuation**

유형: MQLONG - 출력

컴포넌트에서 설정한 연속 표시기입니다. 다음 값을 지정할 수 있습니다.

**MQZCI\_DEFAULT**

큐 관리자에 따른 연속입니다.

MQZ\_GET\_AUTHORITY의 경우 이는 MQZCI\_CONTINUE와 효과가 동일합니다.

**MQZCI\_CONTINUE**

다음 컴포넌트로 계속됩니다.

**MQZCI\_STOP**

다음 컴포넌트를 계속하지 않습니다.

**CompCode**

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_FAILED**

호출에 실패했습니다.

**원인**

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드.

CompCode가 MQCC\_OK인 경우:

**MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_FAILED인 경우:

**MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

**MQRC\_SERVICE\_ERROR**

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

**MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

**MQRC\_UNKNOWN\_ENTITY**

(2292, X'8F4') 서비스에서 알 수 없는 엔티티입니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드를 참조하십시오](#).

## C 호출

```
MQZ_GET_EXPLICIT_AUTHORITY (QMgrName, EntityName, EntityType,  
                             ObjectName, ObjectType, &Authority,  
                             ComponentData, &Continuation,  
                             &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48  QMgrName;           /* Queue manager name */  
MQCHAR12  EntityName;        /* Entity name */  
MQLONG    EntityType;        /* Entity type */  
MQCHAR48  ObjectName;        /* Object name */  
MQLONG    ObjectType;        /* Object type */  
MQLONG    Authority;         /* Authority of entity */  
MQBYTE    ComponentData[n]; /* Component data */  
MQLONG    Continuation;      /* Continuation indicator set by  
                             component */  
MQLONG    CompCode;          /* Completion code */  
MQLONG    Reason;           /* Reason code qualifying CompCode */
```

## MQZ\_GET\_EXPLICIT\_AUTHORITY\_2 - 명시적 권한 가져오기(확장)

이 함수는 MQZAS\_VERSION\_2 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 (**nobody** 그룹의 추가 권한 없이) 지정된 오브젝트에 액세스할 수 있는 특정 그룹의 권한 또는 지정된 오브젝트에 액세스할 수 있는 특정 프린 시펄의 기본 그룹의 권한을 검색하기 위해 시작합니다.

이 함수(MQZEP용)에 대한 함수 ID는 MQZID\_GET\_EXPLICIT\_AUTHORITY입니다.

MQZ\_GET\_EXPLICIT\_AUTHORITY\_2는 MQZ\_GET\_EXPLICIT\_AUTHORITY와 비슷하지만, **EntityName** 매개변수가 **EntityData** 매개변수로 대체되었습니다.

## 구문

MQZ\_GET\_EXPLICIT\_AUTHORITY\_2( QMgrName , EntityData , EntityType , ObjectName , ObjectName , Authority , ComponentData , Continuation , CompCode , Reason )

## 매개변수

### QMgrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

#### **EntityData**

유형: MQZED - 입력

엔티티 데이터. 검색되는 오브젝트에 대한 권한이 있는 엔티티에 관련된 데이터입니다. 자세한 내용은 [1603 페이지](#)의 『MQZED - 엔티티 디스크립터』의 내용을 참조하십시오.

#### **EntityType**

유형: MQLONG - 입력

엔티티 유형입니다. *EntityData*로 지정된 엔티티 유형입니다. 다음 값 중 하나여야 합니다.

#### **MQZAET\_PRINCIPAL**

프린시펄입니다.

#### **MQZAET\_GROUP**

그룹.

#### **ObjectName**

유형: MQCHAR48 - 입력

오브젝트 이름. 엔티티 권한이 검색될 오브젝트의 이름입니다. 문자열의 최대 길이는 48자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

*ObjectType*이 MQOT\_Q\_MGR인 경우, 이 이름은 *QMgrName*과 동일합니다.

#### **ObjectType**

유형: MQLONG - 입력

오브젝트 유형. *ObjectName*로 지정된 엔티티 유형입니다. 다음 값 중 하나여야 합니다.

#### **MQOT\_AUTH\_INFO**

인증 정보.

#### **MQOT\_CHANNEL**

채널.

#### **MQOT\_CLNTCONN\_CHANNEL**

클라이언트 연결 채널.

#### **MQOT\_LISTENER**

.

#### **MQOT\_NAMELIST**

이름 목록.

#### **MQOT\_PROCESS**

process definition.

#### **MQOT\_Q**

큐.

#### **MQOT\_Q\_MGR**

큐 관리자.

#### **MQOT\_SERVICE**

서비스.

#### **MQOT\_TOPIC**

있습니다.

#### **권한**

유형: MQLONG - 입력

엔티티의 권한입니다. 엔티티에 하나의 권한이 있는 경우, 이 필드는 적절한 권한 부여 조작(MQZAO\_\* constant)과 동일합니다. 둘 이상의 권한이 있는 경우, 이 필드는 해당되는 MQZAO\_\* 상수의 비트 단위의 OR입니다.

## ComponentData

유형: MQBYTE×ComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

## Continuation

유형: MQLONG - 출력

컴포넌트에서 설정한 연속 표시기입니다. 다음 값을 지정할 수 있습니다.

### MQZCI\_DEFAULT

큐 관리자에 따른 연속입니다.

MQZ\_CHECK\_AUTHORITY의 경우 MQZCI\_STOP과 동일한 효과가 있습니다.

### MQZCI\_CONTINUE

다음 컴포넌트로 계속됩니다.

### MQZCI\_STOP

다음 컴포넌트를 계속하지 않습니다.

## CompCode

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

### MQCC\_OK

정상적으로 완료되었습니다.

### MQCC\_FAILED

호출에 실패했습니다.

## 원인

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드.

CompCode가 MQCC\_OK인 경우:

### MQRC\_NONE

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_FAILED인 경우:

### MQRC\_NOT\_AUTHORIZED

(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

### MQRC\_SERVICE\_ERROR

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

### MQRC\_SERVICE\_NOT\_AVAILABLE

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

### MQRC\_UNKNOWN\_ENTITY

(2292, X'8F4') 서비스에서 알 수 없는 엔티티입니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드](#)를 참조하십시오.

## C 호출

```
MQZ_GET_EXPLICIT_AUTHORITY_2 (QMgrName, &EntityData, EntityType,  
ObjectNames, ObjectType, &Authority,  
ComponentData, &Continuation,  
&CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48  QMgrName;          /* Queue manager name */
MQZED     EntityData;       /* Entity data */
MQLONG    EntityType;       /* Entity type */
MQCHAR48  ObjectName;       /* Object name */
MQLONG    ObjectType;       /* Object type */
MQLONG    Authority;        /* Authority of entity */
MQBYTE    ComponentData[n]; /* Component data */
MQLONG    Continuation;     /* Continuation indicator set by
                             component */
MQLONG    CompCode;         /* Completion code */
MQLONG    Reason;          /* Reason code qualifying CompCode */
```

## MQZ\_INIT\_AUTHORITY - 권한 서비스 초기화

이 함수는 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 컴포넌트를 구성하는 동안 시작합니다. 큐 관리자에게 정보를 제공하기 위해 MQZEP를 호출할 것으로 예상됩니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_INIT\_AUTHORITY입니다.

### 구문

```
MQZ_INIT_AUTHORITY( Hconfig , Options , QMgrName , ComponentDataLength ,
ComponentData , Version , CompCode , Reason )
```

### 매개변수

#### Hconfig

유형: MQHCONFIG - 입력

구성 핸들입니다. 이 핸들은 초기화되는 특정 컴포넌트를 나타냅니다. 이는 MQZEP 함수로 큐 관리자를 호출할 때 컴포넌트에 의해 사용됩니다.

#### 옵션

유형: MQLONG - 입력

초기화 옵션. 다음 값 중 하나여야 합니다.

#### MQZIO\_PRIMARY

1차 초기화.

#### MQZIO\_SECONDARY

2차 초기화.

#### QMgrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

#### ComponentDataLength

유형: MQLONG - 입력

컴포넌트 데이터의 길이. *ComponentData* 영역의 길이(바이트)입니다. 이 길이는 컴포넌트 구성 데이터에서 정의됩니다.

#### ComponentData

유형: MQBYTE x ComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 컴포넌트 1차 초기화 함수를 호출하기 전에 모두 0으로 초기화됩니다. 이 데이터는 이 특정 컴포넌트 대신 큐 관리자에 의해 보관됩니다. 이 컴포넌트가 제공한 함수(초기화 함수 포함)에 의해 데이터에 작성된 모든 변경사항은 보존되며 다음에 이러한 컴포넌트 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

## 버전

유형: MQLONG - 입출력(I/O)

버전 번호. 초기화 함수에 대한 입력에서 큐 관리자가 지원하는 가장 높은 버전 번호를 식별합니다. 필요한 경우, 초기화 함수는 버전을 지원되는 인터페이스의 버전으로 변경해야 합니다. 리턴 시 큐 관리자가 컴포넌트에서 리턴한 버전을 지원하지 않으면 컴포넌트 MQZ\_TERM\_AUTHORITY 함수를 호출하고 이 컴포넌트를 추가로 사용하지 않습니다.

다음 값이 지원됩니다.

### **MQZAS\_VERSION\_1**

버전 1.

### **MQZAS\_VERSION\_2**

버전 2.

### **MQZAS\_VERSION\_3**

버전 3.

### **MQZAS\_VERSION\_4**

버전 4.

### **MQZAS\_VERSION\_5**

버전 5.

### **MQZAS\_VERSION\_6**

IBM WebSphere MQ 6.

## CompCode

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

### **MQCC\_OK**

정상적으로 완료되었습니다.

### **MQCC\_FAILED**

호출에 실패했습니다.

## 원인

유형: MQLONG - 출력

*CompCode*를 규정하는 이유 코드.

*CompCode*가 MQCC\_OK인 경우:

### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*가 MQCC\_FAILED인 경우:

### **MQRC\_INITIALIZATION\_FAILED**

(2286, X'8EE') 정의되지 않은 이유로 초기화에 실패했습니다.

### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드](#)를 참조하십시오.

## C 호출

```
MQZ_INIT_AUTHORITY (Hconfig, Options, QMgrName, ComponentDataLength,  
                    ComponentData, &Version, &CompCode,  
                    &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```

MQHCONFIG  Hconfig;           /* Configuration handle */
MQLONG     Options;         /* Initialization options */
MQCHAR48   QMgrName;       /* Queue manager name */
MQLONG     ComponentDataLength; /* Length of component data */
MQBYTE     ComponentData[n]; /* Component data */
MQLONG     Version;        /* Version number */
MQLONG     CompCode;       /* Completion code */
MQLONG     Reason;         /* Reason code qualifying CompCode */

```

## MQZ\_INQUIRE - 권한 서비스 조회

이 함수는 MQZAS\_VERSION\_5 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 지원되는 기능을 조회하기 위해 시작합니다.

여러 서비스 컴포넌트를 사용하는 경우, 서비스 컴포넌트가 설치된 순서와 역순으로 호출됩니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_INQUIRE입니다.

### 구문

```

MQZ_INQUIRE( QMgrName , SelectorCount , Selectors , IntAttrCount , IntAttrs ,
CharAttrLength , CharAttrs , SelectorReturned , ComponentData , Continuation ,
CompCode , Reason )

```

### 매개변수

#### QMgrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

#### SelectorCount

유형: MQLONG - 입력

선택자의 수. **Selectors** 매개변수에 제공된 선택자의 수입니다.

해당 값은 0 - 256 범위에 있어야 합니다.

#### 선택자

유형: MQLONGxSelectorCount - 입력

선택자의 배열. 각 선택자는 필수 속성을 식별하며 다음 중 하나여야 합니다.

- MQIACF\_INTERFACE\_VERSION(정수)
- MQIACF\_USER\_ID\_SUPPORT(정수)
- MQCACF\_SERVICE\_COMPONENT(문자)

임의의 순서로 선택자를 지정할 수 있습니다. 배열의 선택자 수는 **SelectorCount** 매개변수로 표시됩니다.

선택자에 의해 식별되는 정수 속성은 **Selectors**에 표시되는 것과 동일한 순서로 **IntAttrs** 매개변수에 리턴됩니다.

선택자에 의해 식별된 문자 속성은 **Selectors**에 표시된 것과 동일한 순서로 **CharAttrs** 매개변수에 리턴됩니다.

#### IntAttrCount

유형: MQLONG - 입력

**IntAttrs** 매개변수에 제공된 정수 속성의 수.

해당 값은 0 - 256 범위에 있어야 합니다.

### **IntAttrs**

유형: MQLONG x IntAttrCount - 출력

정수 속성. 정수 속성의 배열. 정수 속성은 *Selectors* 배열에서 해당 정수 선택자와 동일한 순서대로 리턴됩니다.

### **CharAttrCount**

유형: MQLONG - 입력

문자 속성 버퍼의 길이. **CharAttrs** 매개변수의 길이(바이트)입니다.

값이 최소한 요청된 문자 속성 길이의 합계여야 합니다. 요청한 문자 속성이 없으면 0이 유효한 값입니다.

### **CharAttrs**

유형: MQLONG x CharAttrCount - 출력

문자 속성 버퍼. 함께 연결된 문자 속성을 포함한 버퍼입니다. 문자 속성은 *Selectors* 배열에서 해당 문자 선택자와 동일한 순서대로 리턴됩니다.

버퍼의 길이는 CharAttrCount 매개변수에서 제공합니다.

### **SelectorReturned**

유형: MQLONG x SelectorCount - 입력

리턴되는 선택자. *Selectors* 매개변수의 선택자가 요청한 세트에서 리턴된 속성을 식별하는 값의 배열입니다. 이 배열에서 값의 수는 **SelectorCount** 매개변수로 표시됩니다. 배열의 각 값이 *Selectors* 배열에 있는 해당 위치의 선택자와 관련이 있습니다. 각 값은 다음 중 하나입니다.

#### **MQZSL\_RETURNED**

**Selectors** 매개변수의 해당 선택자가 요청한 속성이 리턴되었습니다.

#### **MQZSL\_NOT\_RETURNED**

**Selectors** 매개변수의 해당 선택자가 요청한 속성이 리턴되지 않았습니다.

배열은 모든 값을 **MQZSL\_NOT\_RETURNED**로 사용하여 초기화됩니다. 권한 부여 서비스 컴포넌트가 속성을 리턴하면 배열의 적절한 값을 **MQZSL\_NOT\_RETURNED**로 설정합니다. 이렇게 하면 조회 호출이 발행되는 다른 권한 부여 서비스 컴포넌트가 이미 리턴된 속성을 식별할 수 있습니다.

### **ComponentData**

유형: MQBYTE x ComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

### **Continuation**

유형: MQLONG - 출력

컴포넌트에서 설정한 연속 표시기입니다. 다음 값을 지정할 수 있습니다.

#### **MQZCI\_DEFAULT**

큐 관리자에 따른 연속입니다.

MQZ\_CHECK\_AUTHORITY의 경우 MQZCI\_STOP과 동일한 효과가 있습니다.

#### **MQZCI\_STOP**

다음 컴포넌트를 계속하지 않습니다.

### **CompCode**

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

#### **MQCC\_OK**

정상적으로 완료되었습니다.



**MQCC\_WARNING**  
부분 완료입니다.

**MQCC\_FAILED**  
호출에 실패했습니다.

#### 원인

유형: MQLONG - 출력

*CompCode*를 규정하는 이유 코드.

*CompCode*가 MQCC\_OK인 경우:

**MQRC\_NONE**  
(0, X'000') 보고할 이유가 없습니다.

*CompCode*가 MQCC\_WARNING인 경우:

**MQRC\_CHAR\_ATTRS\_TOO\_SHORT**  
문자 속성을 위한 공간이 충분하지 않습니다.

**MQRC\_INT\_COUNT\_TOO\_SMALL**  
정수 속성을 위한 공간이 충분하지 않습니다.

*CompCode*가 MQCC\_FAILED인 경우:

**MQRC\_SELECTOR\_COUNT\_ERROR**  
선택자의 수가 올바르지 않습니다.

**MQRC\_SELECTOR\_ERROR**  
속성 선택자가 올바르지 않습니다.

**MQRC\_SELECTOR\_LIMIT\_EXCEEDED**  
지정된 선택자가 너무 많습니다.

**MQRC\_INT\_ATTR\_COUNT\_ERROR**  
정수 속성의 수가 올바르지 않습니다.

**MQRC\_INT\_ATTRS\_ARRAY\_ERROR**  
정수 속성 배열이 올바르지 않습니다.

**MQRC\_CHAR\_ATTR\_LENGTH\_ERROR**  
문자 속성의 수가 올바르지 않습니다.

**MQRC\_CHAR\_ATTRS\_ERROR**  
문자 속성 문자열이 올바르지 않습니다.

**MQRC\_SERVICE\_ERROR**  
(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드](#)를 참조하십시오.

## C 호출

```
MQZ_INQUIRE (QMgrName, SelectorCount, Selectors, IntAttrCount,  
              &IntAttrs, CharAttrLength, &CharAttrs,  
              SelectorReturned, ComponentData, &Continuation,  
              &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48  QMgrName;           /* Queue manager name */  
MQLONG    SelectorCount;     /* Selector count */  
MQLONG    Selectors[n];      /* Selectors */  
MQLONG    IntAttrCount;      /* IntAttrs count */  
MQLONG    IntAttrs[n];       /* Integer attributes */  
MQLONG    CharAttrCount;     /* CharAttrs count */  
MQLONG    CharAttrs[n];     /* Character attributes */  
MQLONG    SelectorReturned[n]; /* Selector returned */  
MQBYTE    ComponentData[n];  /* Component data */  
MQLONG    Continuation;      /* Continuation indicator set by
```

```

MQLONG      component */
MQLONG      CompCode; /* Completion code */
MQLONG      Reason;   /* Reason code qualifying CompCode */

```

## MQZ\_REFRESH\_CACHE - 모든 권한 부여 새로 고치기

이 함수는 MQZAS\_VERSION\_3 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 컴포넌트 내부에 보관된 권한 부여 목록을 새로 고치기 위해 호출합니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_REFRESH\_CACHE(8L)입니다.

### 구문

```
MQZ_REFRESH_CACHE( QMgrName , ComponentData , Continuation , CompCode ,
Reason )
```

### 매개변수

#### QMgrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

#### ComponentData

유형: MQBYTE×ComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

#### Continuation

유형: MQLONG - 출력

컴포넌트에서 설정한 연속 표시기입니다. 다음 값을 지정할 수 있습니다.

##### MQZCI\_DEFAULT

큐 관리자에 따른 연속입니다.

MQZ\_CHECK\_AUTHORITY의 경우 이는 MQZCI\_STOP과 동일한 효과가 있습니다.

##### MQZCI\_CONTINUE

다음 컴포넌트로 계속됩니다.

##### MQZCI\_STOP

다음 컴포넌트를 계속하지 않습니다.

#### CompCode

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

##### MQCC\_OK

정상적으로 완료되었습니다.

##### MQCC\_FAILED

호출에 실패했습니다.

### 원인

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드.

CompCode가 MQCC\_OK인 경우:

**MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_WARNING인 경우:

**MQRC\_SERVICE\_ERROR**

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

## C 호출

```
MQZ_REFRESH_CACHE (QMgrName, ComponentData,  
&Continuation, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQCHAR48  QMgrName;           /* Queue manager name */  
MQBYTE    ComponentData[n];  /* Component data */  
MQLONG    Continuation;      /* Continuation indicator set by  
                             component */  
MQLONG    CompCode;          /* Completion code */  
MQLONG    Reason;           /* Reason code qualifying CompCode */
```

## MQZ\_SET\_AUTHORITY - 권한 설정

이 함수는 MQZAS\_VERSION\_1 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 엔티티에서 지정된 오브젝트에 액세스하는 데 필요한 권한을 설정하기 위해 시작합니다.

이 함수(MQZEP용)에 대한 함수 ID는 MQZID\_SET\_AUTHORITY입니다.

**참고:** 이 함수가 기존 권한을 대체합니다. 기존의 권한을 유지하려면 이 함수로 다시 설정해야 합니다.

### 구문

```
MQZ_SET_AUTHORITY( QMgrName , EntityName , EntityType , ObjectName ,  
ObjectType , Authority , ComponentData , Continuation , CompCode , Reason )
```

### 매개변수

#### QMgrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

#### EntityName

유형: MQCHAR12 - 입력

엔티티 이름입니다. 오브젝트에 대한 액세스 권한을 검색할 엔티티의 이름입니다. 문자열의 최대 길이는 12 자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

#### EntityType

유형: MQLONG - 입력

엔티티 유형입니다. 엔티티 유형은 *EntityName*으로 지정됩니다. 다음 값 중 하나여야 합니다.

#### MQZAET\_PRINCIPAL

프린시펄입니다.

#### MQZAET\_GROUP

그룹.

**ObjectName**

유형: MQCHAR48 - 입력

오브젝트 이름. 액세스가 필요한 오브젝트 이름. 문자열의 최대 길이는 48자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

*ObjectType*이 MQOT\_Q\_MGR인 경우, 이 이름은 *QMgrName*과 동일합니다.

**ObjectType**

유형: MQLONG - 입력

오브젝트 유형. 엔티티 유형은 *ObjectName*으로 지정됩니다. 다음 값 중 하나여야 합니다.

**MQOT\_AUTH\_INFO**

인증 정보.

**MQOT\_CHANNEL**

채널.

**MQOT\_CLNTCONN\_CHANNEL**

클라이언트 연결 채널.

**MQOT\_LISTENER**

.

**MQOT\_NAMELIST**

이름 목록.

**MQOT\_PROCESS**

process definition.

**MQOT\_Q**

큐.

**MQOT\_Q\_MGR**

큐 관리자.

**MQOT\_SERVICE**

서비스.

**MQOT\_TOPIC**

있습니다.

**권한**

유형: MQLONG - 입력

엔티티의 권한입니다. 하나의 권한을 설정하는 경우, 이 필드는 적절한 권한 부여 조작(MQZAO\_\* 상수)과 동일합니다. 둘 이상의 권한을 설정하는 경우, 이 필드는 해당 MQZAO\_\* 상수의 비트 단위 OR입니다.

**ComponentDataname>**

유형: MQBYTEComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

**Continuation**

유형: MQLONG - 출력

컴포넌트에서 설정한 연속 표시기입니다. 다음 값을 지정할 수 있습니다.

**MQZCI\_DEFAULT**

큐 관리자에 따른 연속입니다.

MQZ\_GET\_AUTHORITY의 경우 이는 MQZCI\_CONTINUE와 효과가 동일합니다.

**MQZCI\_CONTINUE**

다음 컴포넌트로 계속됩니다.

## MQZCI\_STOP

다음 컴포넌트를 계속하지 않습니다.

### CompCode

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

### MQCC\_OK

정상적으로 완료되었습니다.

### MQCC\_FAILED

호출에 실패했습니다.

### 원인

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드.

CompCode가 MQCC\_OK인 경우:

### MQRC\_NONE

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_FAILED인 경우:

### MQRC\_NOT\_AUTHORIZED

(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

### MQRC\_SERVICE\_ERROR

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

### MQRC\_SERVICE\_NOT\_AVAILABLE

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

### MQRC\_UNKNOWN\_ENTITY

(2292, X'8F4') 서비스에서 알 수 없는 엔티티입니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드](#)를 참조하십시오.

## C 호출

```
MQZ_SET_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,  
                  ObjectType, Authority, ComponentData,  
                  &Continuation, &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48  QMgrName;           /* Queue manager name */  
MQCHAR12  EntityName;        /* Entity name */  
MQLONG    EntityType;        /* Entity type */  
MQCHAR48  ObjectName;        /* Object name */  
MQLONG    ObjectType;        /* Object type */  
MQLONG    Authority;         /* Authority to be checked */  
MQBYTE    ComponentData[n]; /* Component data */  
MQLONG    Continuation;      /* Continuation indicator set by  
                             component */  
MQLONG    CompCode;          /* Completion code */  
MQLONG    Reason;            /* Reason code qualifying CompCode */
```

## MQZ\_SET\_AUTHORITY\_2 - 권한 설정(확장)

이 함수는 MQZAS\_VERSION\_2 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 엔티티에서 지정된 오브젝트에 액세스하는 데 필요한 권한을 설정하기 위해 시작합니다.

이 함수(MQZEP용)에 대한 함수 ID는 MQZID\_SET\_AUTHORITY입니다.

**참고:** 이 함수가 기존 권한을 대체합니다. 기존의 권한을 유지하려면 이 함수로 다시 설정해야 합니다.

MQZ\_SET\_AUTHORITY\_2는 MQZ\_SET\_AUTHORITY와 비슷하지만, **EntityName** 매개변수가 **EntityData** 매개변수로 대체되었습니다.

## 구문

MQZ\_SET\_AUTHORITY\_2( *QMgrName* , *EntityData* , *EntityType* , *ObjectName* , *ObjectType* , *Authority* , *ComponentData* , *Continuation* , *CompCode* , *Reason* )

## 매개변수

### QMgrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

### EntityData

유형: MQZED - 입력

엔티티 데이터. 설정할 오브젝트에 대한 권한이 있는 엔티티에 관련된 데이터입니다. 자세한 내용은 [1603 페이지의 『MQZED - 엔티티 디스크립터』](#)의 내용을 참조하십시오.

### EntityType

유형: MQLONG - 입력

엔티티 유형입니다. *EntityData*로 지정된 엔티티 유형입니다. 다음 값 중 하나여야 합니다.

#### MQZAET\_PRINCIPAL

프린시펄입니다.

#### MQZAET\_GROUP

그룹.

### ObjectName

유형: MQCHAR48 - 입력

오브젝트 이름. 엔티티 권한이 설정될 오브젝트의 이름입니다. 문자열의 최대 길이는 48자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

*ObjectType*이 MQOT\_Q\_MGR인 경우, 이 이름은 *QMgrName*과 동일합니다.

### ObjectType

유형: MQLONG - 입력

오브젝트 유형. *ObjectName*로 지정된 엔티티 유형입니다. 다음 값 중 하나여야 합니다.

#### MQOT\_AUTH\_INFO

인증 정보.

#### MQOT\_CHANNEL

채널.

#### MQOT\_CLNTCONN\_CHANNEL

클라이언트 연결 채널.

#### MQOT\_LISTENER

.

#### MQOT\_NAMELIST

이름 목록.

#### MQOT\_PROCESS

process definition.

#### MQOT\_Q

큐.

**MQOT\_Q\_MGR**

큐 관리자.

**MQOT\_SERVICE**

서비스.

**MQOT\_TOPIC**

있습니다.

**권한**

유형: MQLONG - 입력

엔티티의 권한입니다. 하나의 권한을 설정하는 경우, 이 필드는 적절한 권한 부여 조작(MQZAO\_\* 상수)과 동일합니다. 둘 이상의 권한을 설정하는 경우, 이 필드는 해당 MQZAO\_\* 상수의 비트 단위 OR입니다.

**ComponentData**

유형: MQBYTE×ComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

**Continuation**

유형: MQLONG - 출력

컴포넌트에서 설정한 연속 표시기입니다. 다음 값을 지정할 수 있습니다.

**MQZCI\_DEFAULT**

큐 관리자에 따른 연속입니다.

MQZ\_CHECK\_AUTHORITY의 경우 MQZCI\_STOP과 동일한 효과가 있습니다.

**MQZCI\_CONTINUE**

다음 컴포넌트로 계속됩니다.

**MQZCI\_STOP**

다음 컴포넌트를 계속하지 않습니다.

**CompCode**

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_FAILED**

호출에 실패했습니다.

**원인**

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드.

CompCode가 MQCC\_OK인 경우:

**MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_FAILED인 경우:

**MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

**MQRC\_SERVICE\_ERROR**

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

## MQRC\_SERVICE\_NOT\_AVAILABLE

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

## MQRC\_UNKNOWN\_ENTITY

(2292, X'8F4') 서비스에서 알 수 없는 엔티티입니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드를 참조하십시오](#).

## C 호출

```
MQZ_SET_AUTHORITY_2 (QMgrName, &EntityData, EntityType, ObjectName,  
                    ObjectType, Authority, ComponentData,  
                    &Continuation, &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48  QMgrName;           /* Queue manager name */  
MQZED     EntityData;        /* Entity data */  
MQLONG    EntityType;        /* Entity type */  
MQCHAR48  ObjectName;        /* Object name */  
MQLONG    ObjectType;        /* Object type */  
MQLONG    Authority;         /* Authority to be checked */  
MQBYTE    ComponentData[n]; /* Component data */  
MQLONG    Continuation;      /* Continuation indicator set by  
                             component */  
MQLONG    CompCode;          /* Completion code */  
MQLONG    Reason;            /* Reason code qualifying CompCode */
```

## MQZ\_TERM\_AUTHORITY - 권한 서비스 종료

이 함수는 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 이 컴포넌트의 서비스를 더 이상 필요로 하지 않을 때 시작합니다. 함수는 컴포넌트에서 필요한 모든 정리를 수행해야 합니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_TERM\_AUTHORITY입니다.

## 구문

```
MQZ_TERM_AUTHORITY( Hconfig , Options , QMgrName , ComponentData , CompCode ,  
Reason )
```

## 매개변수

### Hconfig

유형: MQHCONFIG - 입력

구성 핸들입니다. 이 핸들은 종료되는 특정 컴포넌트를 나타냅니다. 이는 MQZEP 함수로 큐 관리자를 호출할 때 컴포넌트에 의해 사용됩니다.

### 옵션

유형: MQLONG - 입력

종료 옵션. 다음 값 중 하나여야 합니다.

### MQZTO\_PRIMARY

1차 종료.

### MQZTO\_SECONDARY

제2차 종료.

### QMgrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.



큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

### ComponentData

유형: MQBYTE x ComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 큐 관리자에 의해 MQZ\_INIT\_AUTHORITY 호출의 ComponentDataLength 매개변수에 전달됩니다.

MQZ\_TERM\_AUTHORITY 호출이 완료되면 큐 관리자는 이 데이터를 제거합니다.

### CompCode

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_FAILED

호출에 실패했습니다.

### 원인

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드.

CompCode가 MQCC\_OK인 경우:

#### MQRC\_NONE

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_FAILED인 경우:

#### MQRC\_SERVICE\_NOT\_AVAILABLE

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

#### MQRC\_TERMINATION\_FAILED

(2287, X'8FF') 정의되지 않은 이유로 종료가 실패했습니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드](#)를 참조하십시오.

## C 호출

```
MQZ_TERM_AUTHORITY (Hconfig, Options, QMgrName, ComponentData,
                    &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQHCONFIG  Hconfig;          /* Configuration handle */
MQLONG     Options;          /* Termination options */
MQCHAR48   QMgrName;        /* Queue manager name */
MQBYTE     ComponentData[n]; /* Component data */
MQLONG     CompCode;        /* Completion code */
MQLONG     Reason;          /* Reason code qualifying CompCode */
```

## MQZ\_DELETE\_NAME - 이름 삭제

이 함수는 이름 서비스 컴포넌트에서 제공되며 큐 관리자가 지정된 큐의 항목을 삭제하기 위해 시작합니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_DELETE\_NAME입니다.

## 구문

MQZ\_DELETE\_NAME( QMgrName , QName , ComponentData , Continuation , CompCode , Reason )

## 매개변수

### QMgrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

### QName

유형: MQCHAR48 - 입력

큐 이름입니다. 입력 항목을 삭제하는 큐의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

### ComponentData

유형: MQBYTE x ComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 큐 관리자에 의해 MQZ\_INIT\_NAME 호출의 ComponentDataLength 매개변수에 전달됩니다.

### Continuation

유형: MQLONG - 출력

컴포넌트에서 설정한 연속 표시기입니다. 다음 값 중 하나여야 합니다.

#### MQZCI\_DEFAULT

큐 관리자에 따른 연속입니다.

#### MQZCI\_STOP

다음 컴포넌트를 계속하지 않습니다.

**MQZ\_DELETE\_NAME** 명령의 경우, 큐 관리자는 **Continuation** 매개변수에 어떤 값이 리턴되더라도 다른 컴포넌트를 시작하려고 하지 않습니다.

### CompCode

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_WARNING

경고(일부 완료).

#### MQCC\_FAILED

호출에 실패했습니다.

### 원인

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드.

CompCode가 MQCC\_OK인 경우:

#### MQRC\_NONE

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_WARNING인 경우:

#### **MQRC\_UNKNOWN\_NAME**

(2288, X'8F0') 큐 이름을 찾을 수 없습니다.

**참고:** 이 경우에서 기본 서비스가 성공으로 응답하면 이 코드를 리턴하지 못할 수 있습니다.

CompCode가 MQCC\_FAILED인 경우:

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

#### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드](#)를 참조하십시오.

## C 호출

```
MQZ_DELETE_NAME (QMGrName, QName, ComponentData, &Continuation,  
&CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48 QMGrName;          /* Queue manager name */  
MQCHAR48 QName;            /* Queue name */  
MQBYTE ComponentData[n];  /* Component data */  
MQLONG Continuation;      /* Continuation indicator set by  
                           component */  
MQLONG CompCode;          /* Completion code */  
MQLONG Reason;            /* Reason code qualifying CompCode */
```

## MQZ\_INIT\_NAME - 이름 서비스 초기화

이 함수는 이름 서비스 컴포넌트에서 제공되며 큐 관리자가 컴포넌트를 구성하는 동안 시작합니다. 큐 관리자에게 정보를 제공하기 위해 MQZEP를 호출할 것으로 예상됩니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_INIT\_NAME입니다.

## 구문

```
MQZ_INIT_NAME( Hconfig , Options , QMGrName , ComponentDataLength ,  
ComponentData , Version , CompCode , Reason )
```

## 매개변수

### Hconfig

유형: MQHCONFIG - 입력

구성 핸들입니다. 이 핸들은 초기화되는 특정 컴포넌트를 나타냅니다. 이는 MQZEP 함수로 큐 관리자를 호출할 때 컴포넌트에 의해 사용됩니다.

### 옵션

유형: MQLONG - 입력

초기화 옵션. 다음 값 중 하나여야 합니다.

#### **MQZIO\_PRIMARY**

1차 초기화.

#### **MQZIO\_SECONDARY**

2차 초기화.

### QMGrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

### **ComponentDataLength**

유형: MQLONG - 입력

컴포넌트 데이터의 길이. *ComponentData* 영역의 길이(바이트)입니다. 이 길이는 컴포넌트 구성 데이터에서 정의됩니다.

### **ComponentData**

유형: MQBYTE x ComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 컴포넌트 1차 초기화 함수를 호출하기 전에 모두 0으로 초기화됩니다. 이 데이터는 이 특정 컴포넌트 대신 큐 관리자에 의해 보관됩니다. 이 컴포넌트가 제공한 함수(초기화 함수 포함)에 의해 데이터에 작성된 모든 변경사항은 보존되며 다음에 이러한 컴포넌트 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

### **버전**

유형: MQLONG - 입출력(I/O)

버전 번호. 초기화 함수에 대한 입력에서 큐 관리자가 지원하는 가장 높은 버전 번호를 식별합니다. 필요한 경우, 초기화 함수는 버전을 지원되는 인터페이스의 버전으로 변경해야 합니다. 리턴 시 큐 관리자가 컴포넌트에서 리턴한 버전을 지원하지 않으면 컴포넌트 MQZ\_TERM\_AUTHORITY 함수를 호출하고 이 컴포넌트를 추가로 사용하지 않습니다.

다음 값이 지원됩니다.

#### **MQZAS\_VERSION\_1**

버전 1.

### **CompCode**

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

#### **MQCC\_OK**

정상적으로 완료되었습니다.

#### **MQCC\_FAILED**

호출에 실패했습니다.

### **원인**

유형: MQLONG - 출력

*CompCode*를 규정하는 이유 코드.

*CompCode*가 MQCC\_OK인 경우:

#### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*가 MQCC\_FAILED인 경우:

#### **MQRC\_INITIALIZATION\_FAILED**

(2286, X'8EE') 정의되지 않은 이유로 초기화에 실패했습니다.

#### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드](#)를 참조하십시오.

## C 호출

```
MQZ_INIT_NAME (Hconfig, Options, QMgrName, ComponentDataLength,  
                ComponentData, &Version, &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQHCONFIG Hconfig;           /* Configuration handle */  
MQLONG Options;             /* Initialization options */  
MQCHAR48 QMgrName;         /* Queue manager name */  
MQLONG ComponentDataLength; /* Length of component data */  
MQBYTE ComponentData[n];   /* Component data */  
MQLONG Version;           /* Version number */  
MQLONG CompCode;         /* Completion code */  
MQLONG Reason;           /* Reason code qualifying CompCode */
```

## MQZ\_INSERT\_NAME - 이름 삽입

이 함수는 이름 서비스 컴포넌트에서 제공되며 큐 관리자가 큐를 소유한 큐 관리자의 이름을 포함하여 지정된 큐에 대해 항목을 삽입하기 위해 시작합니다. 큐가 서비스에서 이미 정의된 경우, 호출에 실패합니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_INSERT\_NAME입니다.

### 구문

```
MQZ_INSERT_NAME( QMgrName , QName , ResolvedQMgrName , ComponentData ,  
Continuation , CompCode , Reason )
```

### 매개변수

#### QMgrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

#### QName

유형: MQCHAR48 - 입력

큐 이름입니다. 입력 항목을 삽입할 큐의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

#### ResolvedQMgrName

유형: MQCHAR48 - 입력

해석된 큐 관리자 이름. 큐가 해석되는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

#### ComponentData

유형: MQBYTE×ComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 이 특정 컴포넌트 대신 큐 관리자에 의해 보관됩니다. 이 컴포넌트가 제공한 함수(초기화 함수 포함)에 의해 데이터에 작성된 모든 변경사항은 보존되며 다음에 이러한 컴포넌트 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 큐 관리자에 의해 MQZ\_INIT\_NAME 호출의 **ComponentDataLength** 매개변수에 전달됩니다.

#### Continuation

유형: MQLONG - 입출력(I/O)

컴포넌트에서 설정한 연속 표시기입니다. MQZ\_INSERT\_NAME의 경우, 큐 관리자는 **Continuation** 매개 변수에 어떤 값이 리턴되더라도 다른 컴포넌트를 시작하려고 하지 않습니다.

다음 값이 지원됩니다.

**MQZCI\_DEFAULT**

큐 관리자에 따른 연속입니다.

**MQZCI\_STOP**

다음 컴포넌트를 계속하지 않습니다.

**CompCode**

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_FAILED**

호출에 실패했습니다.

**원인**

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드.

CompCode가 MQCC\_OK인 경우:

**MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_FAILED인 경우:

**MQRC\_Q\_ALREADY\_EXISTS**

(2290, X'8F2') 큐 오브젝트가 이미 존재합니다.

**MQRC\_SERVICE\_ERROR**

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

**MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드](#)를 참조하십시오.

## C 호출

```
MQZ_INSERT_NAME (QMgrName, QName, ResolvedQMgrName, ComponentData,
                 &Continuation, &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48  QMgrName;           /* Queue manager name */
MQCHAR48  QName;              /* Queue name */
MQCHAR48  ResolvedQMgrName;   /* Resolved queue manager name */
MQBYTE    ComponentData[n];   /* Component data */
MQLONG    Continuation;       /* Continuation indicator set by
                               component */
MQLONG    CompCode;           /* Completion code */
MQLONG    Reason;            /* Reason code qualifying CompCode */
```

## MQZ\_LOOKUP\_NAME - 이름 검색

이 함수는 이름 서비스 컴포넌트에서 제공되며 큐 관리자가 지정된 큐에 대해 소유 큐 관리자의 이름을 검색하기 위해 시작합니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_LOOKUP\_NAME입니다.

## 구문

`MQZ_LOOKUP_NAME( QMgrName , QName , ResolvedQMGrName , ComponentData , Continuation , CompCode , Reason )`

## 매개변수

### QMGrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

### QName

유형: MQCHAR48 - 입력

큐 이름입니다. 입력 항목이 해석되는 큐의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

### ResolvedQMGrName

유형: MQCHAR48 - 출력

해석된 큐 관리자 이름. 함수가 성공적으로 완료된 경우, 큐를 소유한 큐 관리자의 이름입니다.

서비스 컴포넌트가 리턴한 이름은 매개변수의 전체 길이에 맞도록 오른쪽을 공백으로 채워야 합니다. 이름은 널 문자에 의해 종료되지 않으며, 앞이나 중간에 공백이 포함되어 있지 않아야 합니다.

### ComponentData

유형: MQBYTEComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 이 특정 컴포넌트 대신 큐 관리자에 의해 보관됩니다. 이 컴포넌트가 제공한 함수(초기화 함수 포함)에 의해 데이터에 작성된 모든 변경사항은 보존되며 다음에 이러한 컴포넌트 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 큐 관리자에 의해 MQZ\_INIT\_NAME 호출의 **ComponentDataLength** 매개변수에 전달됩니다.

### Continuation

유형: MQLONG - 출력

컴포넌트에서 설정한 연속 표시기입니다. MQZ\_LOOKUP\_NAME의 경우, 큐 관리자는 다음과 같이 다른 이름 서비스 컴포넌트의 시작 여부를 지정합니다.

- *CompCode*가 MQCC\_OK이면 *Continuation*에 어떤 값이 리턴되든 추가 컴포넌트가 시작되지 않습니다.
- *CompCode*가 MQCC\_OK가 아니면 *Continuation*이 MQZCI\_STOP인 경우를 제외하고 추가 컴포넌트가 시작됩니다.

다음 값이 지원됩니다.

#### MQZCI\_DEFAULT

큐 관리자에 따른 연속입니다.

#### MQZCI\_CONTINUE

다음 컴포넌트로 계속됩니다.

#### MQZCI\_STOP

다음 컴포넌트를 계속하지 않습니다.

### CompCode

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

## MQCC\_FAILED

호출에 실패했습니다.

### 원인

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드.

CompCode가 MQCC\_OK인 경우:

## MQRC\_NONE

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_FAILED인 경우:

## MQRC\_SERVICE\_ERROR

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

## MQRC\_SERVICE\_NOT\_AVAILABLE

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

## MQRC\_UNKNOWN\_Q\_NAME

(2288, X'8F0') 큐 이름을 찾을 수 없습니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드를 참조하십시오](#).

## C 호출

```
MQZ_LOOKUP_NAME (QMgrName, QName, ResolvedQMgrName, ComponentData,  
&Continuation, &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48 QMgrName;          /* Queue manager name */  
MQCHAR48 QName;             /* Queue name */  
MQCHAR48 ResolvedQMgrName;  /* Resolved queue manager name */  
MQBYTE ComponentData[n];    /* Component data */  
MQLONG Continuation;        /* Continuation indicator set by  
                             component */  
MQLONG CompCode;            /* Completion code */  
MQLONG Reason;              /* Reason code qualifying CompCode */
```

## MQZ\_TERM\_NAME - 이름 서비스 종료

이 함수는 이름 서비스 컴포넌트에서 제공되며 큐 관리자가 이 컴포넌트의 서비스를 더 이상 필요로 하지 않을 때 시작합니다. 함수는 컴포넌트에서 필요한 모든 정리를 수행해야 합니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_TERM\_NAME입니다.

### 구문

```
MQZ_TERM_NAME( Hconfig , Options , QMgrName , ComponentData , CompCode ,  
Reason )
```

### 매개변수

#### Hconfig

유형: MQHCONFIG - 입력

구성 행들입니다. 이 행들은 종료되는 특정 컴포넌트를 나타냅니다. MQZEP 함수로 큐 관리자를 호출할 때 컴포넌트에 사용됩니다.

#### 옵션

유형: MQLONG - 입력

종료 옵션. 다음 값 중 하나여야 합니다.



## **MQZTO\_PRIMARY**

1차 종료.

## **MQZTO\_SECONDARY**

제2차 종료.

### **QMgrName**

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

### **ComponentData**

유형: MQBYTE x ComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 이 특정 컴포넌트 대신 큐 관리자에 의해 보관됩니다. 이 컴포넌트가 제공한 함수(초기화 함수 포함)에 의해 데이터에 작성된 모든 변경사항은 보존되며 다음에 이러한 컴포넌트 함수 중 하나가 호출될 때 표시됩니다.

컴포넌트 데이터는 모든 프로세스에 액세스할 수 있는 공유 메모리에 있습니다.

이 데이터 영역의 길이는 큐 관리자에 의해 MQZ\_INIT\_NAME 호출의 **ComponentDataLength** 매개변수에 전달됩니다.

MQZ\_TERM\_NAME 호출이 완료되면 큐 관리자는 이 데이터를 제거합니다.

### **CompCode**

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

#### **MQCC\_OK**

정상적으로 완료되었습니다.

#### **MQCC\_FAILED**

호출에 실패했습니다.

### **원인**

유형: MQLONG - 출력

*CompCode*를 규정하는 이유 코드.

*CompCode*가 MQCC\_OK인 경우:

#### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*가 MQCC\_FAILED인 경우:

#### **MQRC\_TERMINATION\_FAILED**

(2287, X'8FF') 정의되지 않은 이유로 종료가 실패했습니다.

#### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드를 참조하십시오](#).

## **C 호출**

```
MQZ_TERM_NAME (Hconfig, Options, QMgrName, ComponentData, &CompCode,  
&Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQHCONFIG Hconfig; /* Configuration handle */
```

```

MQLONG      Options;          /* Termination options */
MQCHAR48    QMgrName;        /* Queue manager name */
MQBYTE      ComponentData[n]; /* Component data */
MQLONG      CompCode;        /* Completion code */
MQLONG      Reason;          /* Reason code qualifying CompCode */

```

## MQZAC - 애플리케이션 컨텍스트

MQZAC 구조는 MQZ\_AUTHENTICATE\_USER 호출에서 *ApplicationContext* 매개변수에 사용됩니다. 이 매개변수는 호출 애플리케이션과 관련된 데이터를 지정합니다.

표 1 은 이 구조의 필드를 요약하여 보여줍니다.

표 838. MQZAC의 필드	
필드	설명
<u>StrucId</u>	구조 ID
<u>Version</u>	구조 버전 번호
<u>ProcessId</u>	프로세스 ID
<u>ThreadId</u>	스레드 ID
<u>ApplName</u>	애플리케이션 이름
<u>UserID</u>	사용자 ID
<u>EffectiveUserID</u>	유효한 사용자 ID
<u>환경</u>	환경
<u>CallerType</u>	호출자 유형
<u>AuthenticationType</u>	인증 유형
<u>BindType</u>	바인드 유형

### 필드

#### StrucId

유형: MQCHAR4 - 입력

구조 ID입니다. 값은 다음과 같습니다.

#### MQZAC\_STRUC\_ID

애플리케이션 컨텍스트 구조의 ID.

C 프로그래밍 언어의 경우 MQZAC\_STRUC\_ID\_ARRAY 상수도 정의됩니다. 이 상수는 MQZAC\_STRUC\_ID와 동일한 값을 갖지만 문자열 대신 문자의 배열입니다.

#### 버전

유형: MQLONG - 입력

구조 버전 번호입니다. 값은 다음과 같습니다.

#### MQZAC\_VERSION\_1

버전-1 애플리케이션 컨텍스트 구조. 상수 MQZAC\_CURRENT\_VERSION은 현재 버전의 버전 번호를 지정합니다.

#### ProcessId

유형: MQPID - 입력

애플리케이션의 프로세스 ID.

#### ThreadId

유형: MQTID - 입력

애플리케이션의 스레드 ID.

**ApplName**

유형: MQCHAR28 - 입력

애플리케이션 이름.

**UserID**

유형: MQCHAR12 - 입력

사용자 ID. AIX and Linux에서 이 필드는 애플리케이션의 실제 사용자 ID를 지정합니다. Windows에서 이 필드는 애플리케이션의 사용자 ID를 지정합니다.

**EffectiveUserID**

유형: MQCHAR12 - 입력

유효한 사용자 ID. AIX and Linux에서 이 필드는 애플리케이션의 유효 사용자 ID를 지정합니다. Windows에서 이 필드는 공백입니다.

**환경**

유형: MQLONG - 입력

환경. 이 필드는 호출이 작성된 환경을 지정합니다. 필드는 다음 중 하나입니다.

**MQXE\_COMMAND\_SERVER**

명령 서버

**MQXE\_MQSC**

runmqsc 명령 해석기

**MQXE\_MCA**

메시지 채널 에이전트 MQXE\_OTHER

**MQXE\_OTHER**

정의되지 않은 환경

**CallerType**

유형: MQLONG - 입력

호출자 유형. 이 필드는 호출한 프로그램의 유형을 지정합니다. 필드는 다음 중 하나입니다.

**MQXACT\_EXTERNAL**

호출이 큐 관리자 외부에 있습니다.

**MQXACT\_INTERNAL**

호출이 큐 관리자 내부에 있습니다.

**AuthenticationType**

유형: MQLONG - 입력

인증 유형. 이 필드는 수행되는 인증의 유형을 지정합니다. 필드는 다음 중 하나입니다.

**MQZAT\_INITIAL\_CONTEXT**

사용자 컨텍스트 초기화로 인한 인증 호출입니다. 이 값은 MQCONN 또는 MQCONNX 호출 동안 사용됩니다.

**MQZAT\_CHANGE\_CONTEXT**

사용자 컨텍스트 변경으로 인한 인증 호출입니다. 이 값은 MCA가 사용자 컨텍스트를 변경할 때 사용됩니다. 상위 주제: MQZAC -

**BindType**

유형: MQLONG - 입력

바인드 유형. 이 필드는 사용 중인 바인딩의 유형을 지정합니다. 필드는 다음 중 하나입니다.

**MQCNO\_FASTPATH\_BINDING**

빠른 경로 바인딩.

**MQCNO\_SHARED\_BINDING**

공유 바인딩.

**MQCNO\_ISOLATED\_BINDING**

격리된 바인딩.

## C 선언

다음과 같이 구조의 필드를 선언하십시오.

```
typedef struct tagMQZAC MQZAC;  
struct tagMQZAC {  
    MQCHAR4    StrucId;           /* Structure identifier */  
    MQLONG     Version;          /* Structure version number */  
    MQPID      ProcessId;        /* Process identifier */  
    MQTID      ThreadId;         /* Thread identifier */  
    MQCHAR28   ApplName;         /* Application name */  
    MQCHAR12   UserID;           /* User identifier */  
    MQCHAR12   EffectiveUserID;  /* Effective user identifier */  
    MQLONG     Environment;      /* Environment */  
    MQLONG     CallerType;       /* Caller type */  
    MQLONG     AuthenticationType; /* Authentication type */  
    MQLONG     BindType;         /* Bind type */  
};
```

## MQZAD - 권한 데이터

MQZAD 구조는 MQZ\_ENUMERATE\_AUTHORITY\_DATA 호출에서 두 개 매개변수, 입력과 출력에 사용됩니다.

**Filter** 및 **AuthorityBuffer** 매개변수에 대한 자세한 정보는 [1563 페이지의 『MQZ\\_ENUMERATE\\_AUTHORITY\\_DATA - 권한 데이터 나열』](#)의 내용을 참조하십시오.

- 호출에 입력되는 **Filter** 매개변수에는 MQZAD가 사용됩니다. 이 매개변수는 호출로 리턴된 권한 데이터를 선택하는 데 사용할 선택 기준을 지정합니다.
- 호출에서 출력된 **AuthorityBuffer** 매개변수에도 MQZAD가 사용됩니다. 이 매개변수는 프로파일 이름, 오브젝트 유형 및 엔티티의 조합에 대한 권한을 지정합니다.

표 1. 구조의 필드를 요약합니다.

표 839. MQZAD의 필드	
필드	설명
<u>StrucId</u>	구조 ID
<u>Version</u>	구조 버전 번호
<u>ProfileName</u>	프로파일 이름
<u>ObjectType</u>	오브젝트 유형
<u>Authority</u>	권한
<u>EntityDataPtr</u>	엔티티 데이터에 대한 포인터
<u>EntityType</u>	엔티티 유형
<u>옵션</u>	옵션

## 필드

### StrucId

유형: MQCHAR4 - 입력

구조 ID입니다. 값은 다음과 같습니다.

### MQZAD\_STRUC\_ID

권한 데이터 구조 ID.

C 프로그래밍 언어의 경우, 상수 MQZAD\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQZAD\_STRUC\_ID와 동일한 값을 갖지만, 문자열이 아니라 문자 배열입니다.

### 버전

유형: MQLONG - 입력

구조 버전 번호입니다. 값은 다음과 같습니다.

**MQZAD\_VERSION\_1**

버전-1 애플리케이션 컨텍스트 구조. 상수 MQZAD\_CURRENT\_VERSION은 현재 버전의 버전 번호를 지정합니다.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

**MQZAD\_CURRENT\_VERSION**

권한 데이터 구조의 현재 버전.

**ProfileName**

유형: MQCHAR48 - 입력

프로파일 이름.

**Filter** 매개변수의 경우, 이 필드는 권한 데이터가 필요한 프로파일 이름입니다. 이름이 필드 끝까지 또는 첫 번째 널(null) 문자까지 완전히 공백인 경우, 모든 프로파일 이름의 권한 데이터가 리턴됩니다.

**AuthorityBuffer** 매개변수의 경우, 이 필드는 지정된 선택 기준과 일치하는 프로파일의 이름입니다.

**ObjectType**

유형: MQLONG - 입력

오브젝트 유형.

**Filter** 매개변수의 경우, 이 필드는 권한 데이터가 필요한 오브젝트 유형입니다. 값이 MQOT\_ALL이면 모든 오브젝트 유형의 권한 데이터가 리턴됩니다.

**AuthorityBuffer** 매개변수의 경우, 이 필드는 **ProfileName** 매개변수로 식별된 프로파일이 적용되는 오브젝트 유형입니다.

값은 다음 중 하나입니다. **Filter** 매개변수의 경우, 값 MQOT\_ALL도 유효합니다.

**MQOT\_AUTH\_INFO**

인증 정보

**MQOT\_CHANNEL**

채널

**MQOT\_CLNTCONN\_CHANNEL**

클라이언트 연결 채널

**MQOT\_LISTENER**

리스너

**MQOT\_NAMELIST**

이름 목록

**MQOT\_PROCESS**

프로세스 정의

**MQOT\_Q**

큐

**MQOT\_Q\_MGR**

큐 관리자

**MQOT\_SERVICE**

서비스

**권한**

유형: MQLONG - 입력

권한.

**Filter** 매개변수의 경우, 이 필드가 무시됩니다.

**AuthorityBuffer** 매개변수의 경우, 이 필드는 **ProfileName** 및 **ObjectType**으로 식별된 오브젝트에 대한 엔티티의 권한을 나타냅니다. 엔티티에 하나의 권한만 있는 경우, 이 필드는 적절한 권한 부여 값

(MQZAO\_\* 상수)과 동일합니다. 엔티티에 둘 이상의 권한이 있는 경우, 이 필드는 해당되는 MQZAO\_\* 상수의 비트 단위 OR입니다.

### EntityDataPtr

유형: PMQZED - 입력

엔티티를 식별하는 MQZED 구조의 주소.

**Filter** 매개변수의 경우, 이 필드는 권한 데이터가 필요한 엔티티를 식별하는 MQZED 구조를 가리킵니다. **EntityDataPtr**가 널(null) 포인터인 경우, 모든 엔티티의 권한 데이터가 리턴됩니다.

**AuthorityBuffer** 매개변수의 경우, 이 필드는 권한 데이터가 리턴된 엔티티를 식별하는 MQZED 구조를 가리킵니다.

### EntityType

유형: MQLONG - 입력

엔티티 유형입니다.

**Filter** 매개변수의 경우, 이 필드는 권한 데이터가 필요한 엔티티 유형을 지정합니다. 값이 MQZAET\_NONE이면 모든 엔티티 유형의 권한 데이터가 리턴됩니다.

**AuthorityBuffer** 매개변수의 경우, 이 필드는 **EntityDataPtr** 매개변수가 가리키는 MQZED 구조로 식별되는 엔티티의 유형을 지정합니다.

값은 다음 중 하나입니다. **Filter** 매개변수의 경우, 값 MQZAET\_NONE도 유효합니다.

#### MQZAET\_PRINCIPAL

프린시펄

#### MQZAET\_GROUP

그룹

### 옵션

유형: MQAUTHOPT - 입력

옵션. 이 필드는 표시된 프로파일을 제어하는 옵션을 지정합니다. 다음 값 중 하나를 지정해야 합니다.

#### MQAUTHOPT\_NAME\_ALL\_MATCHING

모든 프로파일 표시

#### MQAUTHOPT\_NAME\_EXPLICIT

**ProfileName** 필드에 지정된 것과 정확하게 동일한 이름의 프로파일을 표시합니다.

또한, 다음 중 하나도 지정해야 합니다.

#### MQAUTHOPT\_ENTITY\_SET

**ProfileName** 매개변수가 지정한 오브젝트에 대해 엔티티가 갖는 누적 권한을 계산하는 데 사용되는 모든 프로파일을 표시합니다. **ProfileName** 매개변수는 와일드카드 문자를 포함하지 않아야 합니다.

- 지정된 엔티티가 프린시펄인 경우, 세트{entity, groups}의 각 멤버에 대해 오브젝트에 적용되는 가장 적합한 프로파일이 표시됩니다.
- 지정된 엔티티가 그룹인 경우, 오브젝트에 적용되는 그룹의 가장 적합한 프로파일이 표시됩니다.
- 이 값이 지정되면 **ProfileName**, **ObjectType**, **EntityType**의 값 및 **EntityDataPtr** MQZED 구조에 지정된 엔티티 이름은 모두 공백이 아니어야 합니다.

MQAUTHOPT\_NAME\_ALL\_MATCHING을 지정한 경우, 다음 값을 지정할 수도 있습니다.

#### MQAUTHOPT\_ENTITY\_EXPLICIT

**EntityDataPtr** MQZED 구조에 지정된 엔티티 이름과 정확하게 동일한 엔티티 이름의 프로파일을 표시합니다.

## C 선언

```
typedef struct tagMQZAD MQZAD;  
struct tagMQZAD {
```

```

MQCHAR4  StrucId;          /* Structure identifier */
MQLONG   Version;        /* Structure version number */
MQCHAR48 ProfileName;    /* Profile name */
MQLONG   ObjectType;     /* Object type */
MQLONG   Authority;      /* Authority */
PMQZED   EntityDataPtr;  /* Address of MQZED structure identifying an
                           entity */
MQLONG   EntityType;     /* Entity type */
MQAUTHOPT Options;      /* Options */
};

```

## MQZED - 엔티티 디스크립터

MQZED 구조는 권한을 검사할 엔티티를 지정하기 위해 많은 권한 서비스 호출에 사용됩니다.

표 1. 구조의 필드를 요약합니다.

표 840. MQZED의 필드	
필드	설명
StrucId	구조 ID
Version	버전
EntityNamePtr	엔티티 이름
EntityDomainPtr	엔티티 도메인 포인터
SecurityId	보안 ID
CorrelationPtr	상관 포인터

### 필드

#### StrucId

유형: MQCHAR4 - 입력

구조 ID입니다. 값은 다음과 같습니다.

#### MQZED\_STRUC\_ID

엔티티 디스크립터 구조 ID.

C 프로그래밍 언어의 경우, 상수 MQZED\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQZED\_STRUC\_ID와 동일한 값을 갖지만, 문자열이 아니라 문자 배열입니다.

#### 버전

유형: MQLONG - 입력

구조 버전 번호입니다. 값은 다음과 같습니다.

#### MQZED\_VERSION\_1

버전-1 엔티티 디스크립터 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### MQZED\_CURRENT\_VERSION

엔티티 디스크립터 구조의 현재 버전.

#### EntityNamePtr

유형: PMQCHAR - 입력

프로파일 이름.

엔티티 이름 주소. 권한을 검사할 엔티티 이름에 대한 포인터입니다.

#### EntityDomainPtr

유형: PMQCHAR - 입력

엔티티 도메인 이름 주소. 권한을 검사할 엔티티 정의가 포함된 도메인 이름에 대한 포인터입니다.

## SecurityId

유형: MQBYTE40 - 입력

권한.

보안 ID. 권한을 검사할 보안 ID입니다.

## CorrelationPtr

유형: MQPTR - 입력

상관 포인터. 사용자 인증 함수 및 다른 적절한 OAM 함수 간에 상관 데이터를 원활하게 전달할 수 있습니다.

## C 선언

```
typedef struct tagMQZED MQZED;
struct tagMQZED {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;          /* Structure version number */
    PMQCHAR    EntityNamePtr;    /* Address of entity name */
    PMQCHAR    EntityDomainPtr; /* Address of entity domain name */
    MQBYTE40   SecurityId;       /* Security identifier */
    MQPTR      CorrelationPtr;   /* Address of correlation data */
}
```

## MQZEP - 컴포넌트 시작점 추가

서비스 컴포넌트가 초기화 중에 이 함수를 시작하여 해당 서비스 컴포넌트의 시작점 벡터에 시작점을 추가합니다.

## 구문

MQZEP ( *Hconfig*, *Function*, *EntryPoint*, *CompCode*, *Reason* )

## 매개변수

### Hconfig

유형: MQHCONFIG - 입력

구성 핸들입니다. 이 핸들은 이러한 특정 설치 가능 서비스에 구성 중인 컴포넌트를 표시합니다. 컴포넌트 초기화 호출에서 큐 관리자가 컴포넌트 구성 함수에 전달한 컴포넌트와 동일해야 합니다.

### Function

유형: MQLONG - 입력

함수 ID. 각 설치 가능 서비스에 대해 올바른 값이 정의됩니다.

동일한 함수에 MQZEP가 두 번 이상 호출되면 마지막 호출은 사용되는 시작점을 제공합니다.

### EntryPoint

유형: PMQFUNC - 입력

함수 시작점. 함수를 수행하기 위해 컴포넌트가 제공한 시작점 주소입니다.

널 값은 올바르며, 이 컴포넌트가 함수를 제공하지 않음을 표시합니다. MQZEP를 사용하여 정의하지 않은 시작점에 널이 사용됩니다.

### CompCode

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_FAILED

호출에 실패했습니다.



## 원인

유형: MQLONG - 출력

CompCode를 규정하는 이유 코드.

CompCode가 MQCC\_OK인 경우:

### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_FAILED인 경우:

### **MQRC\_FUNCTION\_ERROR**

(2281, X'8E9') 함수 ID가 올바르지 않습니다.

### **MQRC\_HCONFIG\_ERROR**

(2280, X'8E8') 구성 핸들이 올바르지 않습니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드](#)를 참조하십시오.

## C 호출

```
MQZEP (Hconfig, Function, EntryPoint, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONFIG  Hconfig;      /* Configuration handle */
MQLONG     Function;     /* Function identifier */
PMQFUNC    EntryPoint;   /* Function entry point */
MQLONG     CompCode;     /* Completion code */
MQLONG     Reason;      /* Reason code qualifying CompCode */
```

## MQZFP - 매개변수 비우기

MQZFP 구조는 MQZ\_FREE\_USER 호출에서 *FreeParms* 매개변수에 사용됩니다. 이 매개변수는 비우려는 자원과 관련된 데이터를 지정합니다.

표 1. 구조의 필드를 요약합니다.

표 841. MQZFP의 필드	
필드	설명
<u>StrucId</u>	구조 ID
<u>Version</u>	버전
<u>Reserved</u>	Reserved 필드
<u>CorrelationPtr</u>	상관 포인터

## 필드

### **StrucId**

유형: MQCHAR4 - 입력

구조 ID입니다. 값은 다음과 같습니다.

### **MQZIC\_STRUC\_ID**

ID 컨텍스트 구조의 ID. C 프로그래밍 언어의 경우, MQZIC\_STRUC\_ID\_ARRAY 상수도 정의됩니다. 이 상수는 MQZIC\_STRUC\_ID와 동일한 값을 갖지만 문자열 대신 문자의 배열입니다.

### **버전**

유형: MQLONG - 입력

구조 버전 번호입니다. 값은 다음과 같습니다.

## MQZFP\_VERSION\_1

버전-1 비우기 매개변수 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

## MQZFP\_CURRENT\_VERSION

비우기 매개변수 구조의 현재 버전.

### 예약됨

유형: MQBYTE8 - 입력

예약된 필드. 초기값은 널입니다.

### CorrelationPtr

유형: MQPTR - 입력

상관 포인터. 비우려는 자원과 관련된 상관 데이터의 주소입니다.

## C 선언

```
typedef struct tagMQZFP MQZFP;
struct tagMQZFP {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;          /* Structure version number */
    MQBYTE8    Reserved;        /* Reserved field */
    MQPTR      CorrelationPtr;   /* Address of correlation data */
};
```

## MQZIC - ID 컨텍스트

MQZIC 구조는 MQZ\_AUTHENTICATE\_USER 호출에서 *IdentityContext* 매개변수에 사용됩니다.

MQZIC 구조는 메시지를 큐에 처음 넣은 애플리케이션의 사용자를 식별하는 ID 컨텍스트 정보를 포함합니다.

- 큐 관리자가 *UserIdentifier* 필드에 사용자 식별 이름을 입력하는데, 큐 관리자가 이 작업을 수행하는 방법은 애플리케이션이 실행되는 환경에 따라 다릅니다.
- 큐 관리자가 메시지를 넣은 애플리케이션에서 판별한 토큰 또는 번호를 *AccountingToken* 필드에 입력합니다.
- 애플리케이션은 사용자에게 대해 포함하려는 추가 정보(예: 암호화된 비밀번호)에 이 *ApplIdentityData* 필드를 사용할 수 있습니다.

적절한 권한이 있는 애플리케이션은 MQZ\_AUTHENTICATE\_USER 함수를 사용하여 ID 컨텍스트를 설정할 수 있습니다.

Windows 시스템 보안 ID (SID) 는 메시지가 IBM MQ for Windows아래에 작성될 때 *AccountingToken* 필드에 저장됩니다. SID는 *UserIdentifier* 필드를 보충하고 사용자의 신임 정보를 설정하는 데 사용할 수 있습니다.

표 1. 구조의 필드를 요약합니다.

표 842. MQZIC의 필드	
필드	설명
<u>StrucId</u>	구조 ID
<u>Version</u>	버전
<u>UserIdentifier</u>	사용자 ID
<u>AccountingToken</u>	계정 토큰
<u>ApplIdentityData</u>	애플리케이션 ID 데이터

### 필드

#### StrucId

유형: MQCHAR4 - 입력

구조 ID입니다. 값은 다음과 같습니다.

#### **MQZIC\_STRUC\_ID**

ID 컨텍스트 구조의 ID. C 프로그래밍 언어의 경우, MQZIC\_STRUC\_ID\_ARRAY 상수도 정의됩니다. 이 상수는 MQZIC\_STRUC\_ID와 동일한 값을 갖지만 문자열 대신 문자의 배열입니다.

#### **버전**

유형: MQLONG - 입력

구조 버전 번호입니다. 값은 다음과 같습니다.

#### **MQZIC\_VERSION\_1**

버전-1 ID 컨텍스트 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### **MQZIC\_CURRENT\_VERSION**

ID 컨텍스트 구조의 현재 버전.

#### **UserIdentifier**

유형: MQCHAR12 - 입력

사용자 ID. 메시지의 ID 컨텍스트에 포함됩니다. *UserIdentifier*는 메시지를 생성한 애플리케이션의 사용자 ID를 지정합니다. 큐 관리자는 이 정보를 문자 데이터로 처리하지만 그 형식은 정의하지 않습니다.

*UserIdentifier* 필드에 대한 자세한 정보는 [445 페이지](#)의 『MQMD의 경우 UserIdentifier (MQCHAR12)』의 내용을 참조하십시오.

#### **AccountingToken**

유형: MQBYTE32 - 입력

계정 토큰. 메시지의 ID 컨텍스트에 포함됩니다. *AccountingToken*은 애플리케이션이 메시지의 결과로 완료된 작업에 적절히 청구할 수 있도록 합니다. 큐 관리자는 이 정보를 비트 문자열로 처리하며 그 콘텐츠를 검사하지 않습니다. *AccountingToken* 필드에 대한 자세한 정보는 [446 페이지](#)의 『MQMD의 경우 AccountingToken (MQBYTE32)』의 내용을 참조하십시오.

#### **ApplIdentityData**

유형: MQCHAR32 - 입력

ID와 관련된 애플리케이션 데이터. 메시지의 ID 컨텍스트에 포함됩니다. *ApplIdentityData*는 메시지 원본에 대한 추가 정보를 제공하는 데 사용하는 애플리케이션 스위트가 정의한 정보입니다. 예를 들어, ID 데이터가 신뢰되는지 여부를 표시하기 위해 적절한 사용자 권한을 갖고 실행하는 애플리케이션이 이를 설정할 수 있습니다. *ApplIdentityData* 필드에 대한 자세한 정보는 [448 페이지](#)의 『MQMD의 ApplIdentity데이터 (MQCHAR32)』의 내용을 참조하십시오.

## **C 선언**

```
typedef struct tagMQZED MQZED;
struct tagMQZED {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQCHAR12   UserIdentifier;   /* User identifier */
    MQBYTE32   AccountingToken; /* Accounting token */
    MQCHAR32   ApplIdentityData; /* Application data relating to identity */
};
```

## **IBM i IBM i에 설치 가능 서비스 인터페이스 참조 정보**

이 정보를 사용하여 IBM i의 설치 가능 서비스에 대한 참조 정보를 이해합니다.

함수 ID(MQZEP)를 포함하여 각 함수에 대한 설명이 나와 있습니다.

매개변수는 발생해야 하는 순서대로 표시됩니다. 모든 매개변수가 있어야 합니다.

각 매개변수 이름 뒤에 괄호로 묶은 데이터 유형이 표시됩니다. 요소 데이터 유형에 대한 설명은 [957 페이지](#)의 『기본 데이터 유형』에 나와 있습니다.

매개변수 설명 뒤에 C 언어 호출도 제공됩니다.

## 관련 개념

**IBM i** [IBM i용 설치 가능 서비스 및 컴포넌트](#)

**ALW** [UNIX, Linux 및 Windows용 설치 가능 서비스 및 컴포넌트](#)

## 관련 참조

1546 페이지의 『[설치 가능 서비스 인터페이스 참조 정보](#)』

이 주제 컬렉션은 설치 가능 서비스에 대한 참조 정보를 제공합니다.

## **IBM i** IBM i의 MQZEP(컴포넌트 시작점 추가)

이 함수는 해당 서비스 컴포넌트의 시작점 벡터에 시작점을 추가하기 위해 초기화 동안 서비스 컴포넌트에 의해 호출됩니다.

## 구문

```
MQZEP (Hconfig, Function, EntryPoint, CompCode, Reason)
```

## 매개변수

MQZEP 호출은 다음 매개변수를 포함합니다.

### Hconfig(MQHCONFIG) - 입력

구성 핸들입니다.

이 핸들은 이러한 특정 설치 가능 서비스에 구성 중인 컴포넌트를 표시합니다. 컴포넌트 초기화 호출에서 큐 관리자가 컴포넌트 구성 함수에 전달한 컴포넌트와 동일해야 합니다.

### Function (MQLONG) - 입력

함수 ID.

각 설치 가능 서비스에 대해 올바른 값이 정의됩니다. 동일한 함수에 MQZEP가 두 번 이상 호출되면 마지막 호출은 사용되는 시작점을 제공합니다.

### EntryPoint (PMQFUNC) - 입력

함수 시작점.

함수를 수행하기 위해 컴포넌트가 제공한 시작점 주소입니다. NULL 값은 올바르며, 이 컴포넌트가 함수를 제공하지 않음을 표시합니다. MQZEP를 사용하여 정의하지 않은 시작점에 NULL이 사용됩니다.

### CompCode(MQLONG) - 출력

완료 코드.

다음 중 하나입니다.

#### MQCC\_OK

정상적으로 완료되었습니다.

#### MQCC\_FAILED

호출에 실패했습니다.

### Reason(MQLONG) - 출력

CompCode을(를) 규정하는 이유 코드.

CompCode이(가) MQCC\_OK인 경우:

#### MQRC\_NONE

(0, X'000') 보고할 이유가 없습니다.

CompCode이(가) MQCC\_FAILED인 경우:

#### MQRC\_FUNCTION\_ERROR

(2281, X'8E9') 함수 ID가 올바르지 않습니다.

## MQRC\_HCONFIG\_ERROR

(2280, X'8E8') 구성 핸들이 올바르지 않습니다.

이 이유 코드에 대한 자세한 정보는 [메시지 및 이유 코드를 참조하십시오](#).

## C 호출

```
MQZEP (Hconfig, Function, EntryPoint, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQHCONFIG Hconfig; /* Configuration handle */
MQLONG Function; /* Function identifier */
PMQFUNC EntryPoint; /* Function entry point */
MQLONG CompCode; /* Completion code */
MQLONG Reason; /* Reason code qualifying CompCode */
```

## IBM i IBM i의 MQHCONFIG(구성 핸들)

MQHCONFIG 데이터 유형은 특정 설치 가능 서비스에 대해 구성 중인 컴포넌트인 구성 핸들을 표시합니다. 구성 핸들은 자연적인 경계에 맞춰야 합니다.

애플리케이션은 동일성에 대해서만 이 변수 유형을 테스트해야 합니다.

## C 선언

```
typedef void MQPOINTER MQHCONFIG;
```

## IBM i IBM i의 PMQFUNC(Pointer to function)

함수 포인터입니다.

## C 선언

```
typedef void MQPOINTER PMQFUNC;
```

## IBM i IBM i의 MQZ\_AUTHENTICATE\_USER(사용자 인증)

이 함수는 MQZAS\_VERSION\_5 권한 서비스 컴포넌트에서 제공됩니다. 큐 관리자가 사용자를 인증하거나 ID 컨텍스트 필드를 설정하기 위해 호출합니다.

IBM MQ 사용자 애플리케이션 컨텍스트가 설정되면 호출됩니다. 연결 호출 동안 애플리케이션의 사용자 컨텍스트가 초기화되는 지점 및 애플리케이션의 사용자 컨텍스트가 변경되는 각 지점에서 발생합니다. 연결 호출을 작성할 때마다 애플리케이션의 사용자 컨텍스트 정보가 *IdentityContext* 필드에 다시 확보됩니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_AUTHENTICATE\_USER입니다.

## 구문

```
MQZ_AUTHENTICATE_USER (QMgrName, SecurityParms, ApplicationContext,
IdentityContext, CorrelationPtr, ComponentData, Continuation, CompCode,
Reason)
```

## 매개변수

MQZ\_AUTHENTICATE\_USER 호출의 매개변수는 다음과 같습니다.

**QMgrName(MQCHAR48) - 입력**

큐 관리자 이름.

컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다. 큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

**SecurityParms (MQCSP) - 입력**

보안 매개변수.

사용자 ID, 비밀번호 및 인증 유형에 관련된 데이터입니다.

MQCONN MQI 호출 동안 이 매개변수에는 널 또는 기본값이 포함됩니다.

**ApplicationContext (MQZAC) - 입력**

애플리케이션 컨텍스트.

호출 애플리케이션에 관련된 데이터입니다. 자세한 내용은 1638 페이지의 『IBM i의 MQZAC(애플리케이션 컨텍스트)』의 내용을 참조하십시오. 모든 MQCONN 또는 MQCONNX MQI 호출 동안, MQZAC 구조의 사용자 컨텍스트 정보가 다시 확보됩니다.

**IdentityContext (MQZIC) - 입출력(I/O)**

ID 컨텍스트.

사용자 인증 함수에 대한 입력에서 현재 ID 컨텍스트를 식별합니다. 사용자 인증 함수는 큐 관리자가 새 ID 컨텍스트를 채택하는 지점에서 이 항목을 변경할 수 있습니다. MQZIC 구조에 대한 자세한 정보 1645 페이지의 『IBM i의 MQZIC(ID 컨텍스트)』의 내용을 참조하십시오.

**CorrelationPtr (MQPTR) - 출력**

상관 포인터.

상관 데이터의 주소를 지정합니다. 이후 이 포인터가 다른 OAM 호출로 전달됩니다.

**ComponentData (MQBYTE x ComponentDataLength) - 입출력(I/O)**

컴포넌트 데이터입니다.

이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다. 이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

**Continuation (MQLONG) - 출력**

연속 플래그입니다.

다음 값을 지정할 수 있습니다.

**MQZCI\_DEFAULT**

기타 컴포넌트에 종속되는 연속입니다.

**MQZCI\_STOP**

다음 컴포넌트를 계속하지 않습니다.

**CompCode(MQLONG) - 출력**

완료 코드.

다음 중 하나입니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_FAILED**

호출에 실패했습니다.

**Reason(MQLONG) - 출력**

CompCode을(를) 규정하는 이유 코드.

CompCode이(가) MQCC\_OK인 경우:

## MQRC\_NONE

(0, X'000') 보고할 이유가 없습니다.

CompCode이(가) MQCC\_FAILED인 경우:

## MQRC\_SERVICE\_ERROR

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

이 이유 코드에 대한 자세한 정보는 [메시지 및 이유 코드를 참조하십시오](#).

## C 호출

```
MQZ_AUTHENTICATE_USER (QMgrName, SecurityParms, ApplicationContext,
                        IdentityContext, &CorrelationPtr, ComponentData,
                        &Continuation, &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48  QMgrName;           /* Queue manager name */
MQCSP     SecurityParms;      /* Security parameters */
MQZAC     ApplicationContext; /* Application context */
MQZIC     IdentityContext;    /* Identity context */
MQPTR     CorrelationPtr;     /* Correlation pointer */
MQBYTE    ComponentData[n];  /* Component data */
MQLONG    Continuation;      /* Continuation indicator set by
                             component */
MQLONG    CompCode;          /* Completion code */
MQLONG    Reason;           /* Reason code qualifying CompCode */
```

## IBM i

### IBM i의 MQZ\_CHECK\_AUTHORITY(권한 검사)

이 함수는 MQZAS\_VERSION\_1 권한 서비스 컴포넌트에서 제공되며, 큐 관리자가 지정된 오브젝트에서 특정 조치를 수행할 권한이 엔티티에 있는지 여부를 검사하기 위해 호출합니다.

이 함수(MQZEP용)에 대한 함수 ID는 MQZID\_CHECK\_AUTHORITY입니다.

## 구문

```
MQZ_CHECK_AUTHORITY (QMgrName, EntityName, EntityType,
                     ObjectName, ObjectType, Authority, ComponentData, Continuation, CompCode,
                     Reason)
```

## 매개변수

MQZ\_CHECK\_AUTHORITY 호출의 매개변수는 다음과 같습니다.

### QMgrName(MQCHAR48) - 입력

큐 관리자 이름.

컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다. 큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

### EntityName (MQCHAR12) - 입력

엔티티 이름입니다.

오브젝트에 대한 권한을 검사해야 하는 엔티티 이름입니다. 문자열의 최대 길이는 12자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

이 엔티티가 근본적인 보안 서비스에 반드시 알려져야 하는 것은 아닙니다. 알려져 있지 않은 경우, 특별 **nobody** 그룹(모든 엔티티가 속하는 것으로 간주됨)의 권한이 검사에 사용됩니다. 공백으로만 구성된 이름은 유효하며 다음과 같이 사용할 수 있습니다.

**EntityType (MQLONG) - 입력**

엔티티 유형입니다.

엔티티 유형은 *EntityName*으로 지정됩니다. 다음 중 하나입니다.

**MQZAET\_PRINCIPAL**

프린시펄입니다.

**MQZAET\_GROUP**

그룹.

**ObjectName (MQCHAR48) - 입력**

오브젝트 이름.

액세스가 필요한 오브젝트 이름. 문자열의 최대 길이는 48자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

*ObjectType*이 MQOT\_Q\_MGR인 경우, 이 이름은 *QMgrName*과 동일합니다.

**ObjectType (MQLONG) - 입력**

오브젝트 유형.

엔티티 유형은 *ObjectName*으로 지정됩니다. 다음 중 하나입니다.

**MQOT\_AUTH\_INFO**

인증 정보.

**MQOT\_CHANNEL**

채널.

**MQOT\_CLNTCONN\_CHANNEL**

클라이언트 연결 채널.

**MQOT\_LISTENER**

.

**MQOT\_NAMELIST**

이름 목록.

**MQOT\_PROCESS**

process definition.

**MQOT\_Q**

큐.

**MQOT\_Q\_MGR**

큐 관리자.

**MQOT\_SERVICE**

서비스.

**Authority (MQLONG) - 입력**

검사할 권한.

하나의 권한이 검사되는 경우 이 필드는 적절한 권한 부여 조작(MQZAO\_\* 상수)과 동일합니다. 둘 이상의 권한이 검사되는 경우 해당 MQZAO\_\* 상수의 비트 단위의 OR입니다.

다음 권한이 MQI 호출 사용에 적용됩니다.

**MQZAO\_CONNECT**

MQCONN 호출을 사용할 수 있음.

**MQZAO\_BROWSE**

찾아보기 옵션과 MQGET 호출을 사용할 수 있음.

MQGMO\_BROWSE\_FIRST, MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR 또는 MQGMO\_BROWSE\_NEXT 옵션을 MQGET 호출에 지정할 수 있습니다.

**MQZAO\_INPUT**

입력 옵션과 MQGET 호출을 사용할 수 있음.



MQOO\_INPUT\_SHARED, MQOO\_INPUT\_EXCLUSIVE 또는 MQOO\_INPUT\_AS\_Q\_DEF 옵션을 MQOPEN 호출에 지정할 수 있습니다.

#### **MQZAO\_OUTPUT**

MQPUT 호출을 사용할 수 있음.

MQOO\_OUTPUT 옵션을 MQOPEN 호출에 지정할 수 있습니다.

#### **MQZAO\_INQUIRE**

MQINQ 호출을 사용할 수 있음.

MQOO\_INQUIRE 옵션을 MQOPEN 호출에 지정할 수 있습니다.

#### **MQZAO\_SET**

MQSET 호출을 사용할 수 있음.

MQOO\_SET 옵션을 MQOPEN 호출에 지정할 수 있습니다.

#### **MQZAO\_PASS\_IDENTITY\_CONTEXT**

ID 컨텍스트를 전달할 수 있음.

MQOO\_PASS\_IDENTITY\_CONTEXT 옵션을 MQOPEN 호출에 지정하고, MQPMO\_PASS\_IDENTITY\_CONTEXT 옵션을 MQPUT 및 MQPUT1 호출에 지정할 수 있습니다.

#### **MQZAO\_PASS\_ALL\_CONTEXT**

모든 컨텍스트를 전달할 수 있음.

MQOO\_PASS\_ALL\_CONTEXT 옵션을 MQOPEN 호출에 지정하고, MQPMO\_PASS\_ALL\_CONTEXT 옵션을 MQPUT 및 MQPUT1 호출에 지정할 수 있습니다.

#### **MQZAO\_SET\_IDENTITY\_CONTEXT**

ID 컨텍스트를 설정할 수 있음.

MQOO\_SET\_IDENTITY\_CONTEXT 옵션을 MQOPEN 호출에 지정하고, MQPMO\_SET\_IDENTITY\_CONTEXT 옵션을 MQPUT 및 MQPUT1 호출에 지정할 수 있습니다.

#### **MQZAO\_SET\_ALL\_CONTEXT**

모든 컨텍스트를 설정할 수 있음.

MQOO\_SET\_ALL\_CONTEXT 옵션을 MQOPEN 호출에 지정하고, MQPMO\_SET\_ALL\_CONTEXT 옵션을 MQPUT 및 MQPUT1 호출에 지정할 수 있습니다.

#### **MQZAO\_ALTERNATE\_USER\_AUTHORITY**

대체 사용자 권한을 사용할 수 있음.

MQOO\_ALTERNATE\_USER\_AUTHORITY 옵션을 MQOPEN 호출에 지정하고, MQPMO\_ALTERNATE\_USER\_AUTHORITY 옵션을 MQPUT1 호출에 지정할 수 있습니다.

#### **MQZAO\_ALL\_MQI**

모든 MQI 권한.

이전에 설명된 모든 권한을 부여합니다.

다음 권한이 큐 관리자의 관리에 적용됩니다.

#### **MQZAO\_CREATE**

지정된 유형의 오브젝트를 작성할 수 있음.

#### **MQZAO\_DELETE**

지정된 오브젝트를 삭제할 수 있음.

#### **MQZAO\_DISPLAY**

지정된 오브젝트의 속성을 표시할 수 있음.

#### **MQZAO\_CHANGE**

지정된 오브젝트의 속성을 변경할 수 있음.

#### **MQZAO\_CLEAR**

지정된 큐에서 모든 메시지를 삭제할 수 있음.

**MQZAO\_AUTHORIZE**

지정된 오브젝트에 대해 다른 사용자에게 권한을 부여할 수 있음.

**MQZAO\_CONTROL**

클라이언트가 아닌 채널 오브젝트를 시작, 중지 또는 ping할 수 있음.

**MQZAO\_CONTROL\_EXTENDED**

순서 번호를 재설정하거나 클라이언트가 아닌 채널 오브젝트에서 인다우트 메시지를 해석할 수 있음.

**MQZAO\_ALL\_ADMIN**

MQZAO\_CREATE를 제외한 모든 관리 권한.

다음 권한이 MQI 사용 및 큐 관리자의 관리에 모두 적용됩니다.

**MQZAO\_ALL**

MQZAO\_CREATE를 제외한 모든 권한.

**MQZAO\_NONE**

권한 없음.

**ComponentData (MQBYTE x ComponentDataLength) - 입출력(I/O)**

컴포넌트 데이터입니다.

이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

**Continuation (MQLONG) - 출력**

컴포넌트에서 설정한 연속 표시기입니다.

다음 값을 지정할 수 있습니다.

**MQZCI\_DEFAULT**

큐 관리자에 따른 연속입니다.

MQZ\_CHECK\_AUTHORITY의 경우 이는 MQZCI\_STOP과 동일한 효과가 있습니다.

**MQZCI\_CONTINUE**

다음 컴포넌트로 계속됩니다.

**MQZCI\_STOP**

다음 컴포넌트를 계속하지 않습니다.

**CompCode(MQLONG) - 출력**

완료 코드.

다음 중 하나입니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_FAILED**

호출에 실패했습니다.

**Reason(MQLONG) - 출력**

*CompCode*을(를) 규정하는 이유 코드.

*CompCode*이(가) MQCC\_OK인 경우:

**MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_FAILED인 경우:

**MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

**MQRC\_SERVICE\_ERROR**

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

## MQRC\_SERVICE\_NOT\_AVAILABLE

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

이 이유 코드에 대한 자세한 정보는 [메시지 및 이유 코드를 참조하십시오](#).

## C 호출

```
MQZ_CHECK_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,  
                    ObjectType, Authority, ComponentData,  
                    &Continuation, &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48 QMgrName;          /* Queue manager name */  
MQCHAR12 EntityName;       /* Entity name */  
MQLONG   EntityType;       /* Entity type */  
MQCHAR48 ObjectName;      /* Object name */  
MQLONG   ObjectType;      /* Object type */  
MQLONG   Authority;       /* Authority to be checked */  
MQBYTE   ComponentData[n]; /* Component data */  
MQLONG   Continuation;    /* Continuation indicator set by  
                           component */  
MQLONG   CompCode;        /* Completion code */  
MQLONG   Reason;          /* Reason code qualifying CompCode */
```

## MQZ\_CHECK\_PRIVILEGED - 사용자에게 권한이 있는지 검사

이 함수는 MQZAS\_VERSION\_6 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 지정된 사용자가 권한이 있는 사용자인지 여부를 판별하기 위해 호출합니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_CHECK\_PRIVILEGED입니다.

## 구문

```
MQZ_CHECK_PRIVILEGED( QMgrName , EntityData , EntityType , ComponentData ,  
Continuation , CompCode , Reason )
```

## 매개변수

### QMgrName

유형: MQCHAR48 - 입력

큐 관리자 이름. 컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

### EntityData

유형: MQZED - 입력

엔티티 데이터. 검사할 엔티티에 관련된 데이터입니다. 자세한 정보는 [1603 페이지의 『MQZED - 엔티티 디스크립터』](#)의 내용을 참조하십시오.

### EntityType

유형: MQLONG - 입력

엔티티 유형입니다. EntityData로 지정된 엔티티 유형입니다. 다음 값 중 하나여야 합니다.

#### MQZAET\_PRINCIPAL

프린시펄입니다.

#### MQZAET\_GROUP

그룹.

## ComponentData

유형: MQBYTEComponentDataLength - 입출력(I/O)

컴포넌트 데이터입니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

## Continuation

유형: MQLONG - 출력

컴포넌트에서 설정한 연속 표시기입니다. 다음 값을 지정할 수 있습니다.

### MQZCI\_DEFAULT

큐 관리자에 따른 연속입니다.

MQZ\_CHECK\_AUTHORITY의 경우 MQZCI\_STOP과 동일한 효과가 있습니다.

### MQZCI\_CONTINUE

다음 컴포넌트로 계속됩니다.

### MQZCI\_STOP

다음 컴포넌트를 계속하지 않습니다.

컴포넌트에 대한 호출이 실패하고(즉, *CompCode*가 MQCC\_FAILED를 리턴함) *Continuation* 매개변수가 MQZCI\_DEFAULT 또는 MQZCI\_CONTINUE이면 큐 관리자가 다른 컴포넌트를 계속 호출합니다(있는 경우).

호출이 성공하면(즉, *CompCode*가 MQCC\_OK를 리턴함) *Continuation*의 설정에 상관없이 다른 컴포넌트가 호출되지 않습니다.

호출이 실패하고 *Continuation* 매개변수가 MQZCI\_STOP이면 다른 컴포넌트가 호출되지 않고 큐 관리자에 오류가 리턴됩니다. 컴포넌트에 이전 호출에 대한 정보가 없으므로 호출 전에 *Continuation* 매개변수가 항상 MQZCI\_DEFAULT로 설정됩니다.

## CompCode

유형: MQLONG - 출력

완료 코드. 다음 값 중 하나여야 합니다.

### MQCC\_OK

정상적으로 완료되었습니다.

### MQCC\_FAILED

호출에 실패했습니다.

## 원인

유형: MQLONG - 출력

*CompCode*를 규정하는 이유 코드.

*CompCode*가 MQCC\_OK인 경우:

### MQRC\_NONE

(0, X'000') 보고할 이유가 없습니다.

*CompCode*가 MQCC\_FAILED인 경우:

### MQRC\_NOT\_PRIVILEGED

(2584, X'A18') 이 사용자는 권한이 있는 사용자 ID가 아닙니다.

### MQRC\_UNKNOWN\_ENTITY

(2292, X'8F4') 서비스에서 알 수 없는 엔티티입니다.

### MQRC\_SERVICE\_ERROR

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

### MQRC\_SERVICE\_NOT\_AVAILABLE

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

이러한 이유 코드에 대한 자세한 정보는 [API 완료 및 이유 코드를 참조하십시오](#).

## C 호출

```
MQZ_CHECK_PRIVILEGED (QMgrName, &EntityData, EntityType,  
                      ComponentData, &Continuation,  
                      &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48  QMgrName;          /* Queue manager name */  
MQZED     EntityData;        /* Entity name */  
MQLONG    EntityType;        /* Entity type */  
MQBYTE    ComponentData[n]; /* Component data */  
MQLONG    Continuation;      /* Continuation indicator set by  
                             component */  
MQLONG    CompCode;          /* Completion code */  
MQLONG    Reason;           /* Reason code qualifying CompCode */
```

## IBM i IBM i의 MQZ\_COPY\_ALL\_AUTHORITY(모든 권한 복사)

이 함수는 권한 서비스 컴포넌트에서 제공됩니다. 큐 관리자가 현재 참조 오브젝트에 적용되는 모든 권한을 다른 오브젝트에 복사하기 위해 호출합니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_COPY\_ALL\_AUTHORITY입니다.

## 구문

```
MQZ_COPY_ALL_AUTHORITY (QMgrName, RefObjectName, ObjectName,  
                        ObjectType, ComponentData, Continuation, CompCode, Reason)
```

## 매개변수

MQZ\_COPY\_ALL\_AUTHORITY 호출의 매개변수는 다음과 같습니다.

### QMgrName(MQCHAR48) - 입력

큐 관리자 이름.

컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

### RefObjectName (MQCHAR48) - 입력

참조 오브젝트 이름.

권한을 복사할 참조 오브젝트의 이름입니다. 문자열의 최대 길이는 48자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

### ObjectName (MQCHAR48) - 입력

오브젝트 이름.

액세스 권한을 설정할 오브젝트의 이름입니다. 문자열의 최대 길이는 48자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

### ObjectType (MQLONG) - 입력

오브젝트 유형.

RefObjectName 및 ObjectName으로 지정된 오브젝트 유형입니다. 다음 중 하나입니다.

### MQOT\_AUTH\_INFO

인증 정보.

**MQOT\_CHANNEL**

채널.

**MQOT\_CLNTCONN\_CHANNEL**

클라이언트 연결 채널.

**MQOT\_LISTENER**

.

**MQOT\_NAMELIST**

이름 목록.

**MQOT\_PROCESS**

process definition.

**MQOT\_Q**

큐.

**MQOT\_Q\_MGR**

큐 관리자.

**MQOT\_SERVICE**

서비스.

**ComponentData (MQBYTE x ComponentDataLength) - 입출력(I/O)**

컴포넌트 데이터입니다.

이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

**Continuation (MQLONG) - 출력**

컴포넌트에서 설정한 연속 표시기입니다.

다음 값을 지정할 수 있습니다.

**MQZCI\_DEFAULT**

큐 관리자에 따른 연속입니다.

MQZ\_COPY\_ALL\_AUTHORITY의 경우 이는 MQZCI\_STOP과 동일한 효과가 있습니다.

**MQZCI\_CONTINUE**

다음 컴포넌트로 계속됩니다.

**MQZCI\_STOP**

다음 컴포넌트를 계속하지 않습니다.

**CompCode(MQLONG) - 출력**

완료 코드.

다음 중 하나입니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_FAILED**

호출에 실패했습니다.

**Reason(MQLONG) - 출력**

*CompCode*을(를) 규정하는 이유 코드.

*CompCode*이(가) MQCC\_OK인 경우:

**MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_FAILED인 경우:

**MQRC\_SERVICE\_ERROR**

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

## **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

## **MQRC\_UNKNOWN\_REF\_OBJECT**

(2294, X'8F6') 알 수 없는 참조 오브젝트.

이 이유 코드에 대한 자세한 정보는 [메시지 및 이유 코드를 참조하십시오](#).

## **C 호출**

```
MQZ_COPY_ALL_AUTHORITY (QMgrName, RefObjectName, ObjectName, ObjectType,  
                        ComponentData, &Continuation, &CompCode,  
                        &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48 QMgrName;           /* Queue manager name */  
MQCHAR48 RefObjectName;      /* Reference object name */  
MQCHAR48 ObjectName;        /* Object name */  
MQLONG   ObjectType;        /* Object type */  
MQBYTE   ComponentData[n];  /* Component data */  
MQLONG   Continuation;      /* Continuation indicator set by  
                           component */  
MQLONG   CompCode;          /* Completion code */  
MQLONG   Reason;           /* Reason code qualifying CompCode */
```

## **IBM i IBM i의 MQZ\_DELETE\_AUTHORITY(권한 삭제)**

이 함수는 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 지정된 오브젝트와 연관된 모든 권한을 삭제하기 위해 호출합니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_DELETE\_AUTHORITY입니다.

## **구문**

```
MQZ_DELETE_AUTHORITY (QMgrName, ObjectName, ObjectType,  
                    ComponentData, Continuation, CompCode, Reason)
```

## **매개변수**

MQZ\_DELETE\_AUTHORITY 호출의 매개변수는 다음과 같습니다.

### **QMgrName(MQCHAR48) - 입력**

큐 관리자 이름.

컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

### **ObjectName (MQCHAR48) - 입력**

오브젝트 이름.

액세스 권한을 삭제할 오브젝트의 이름입니다. 문자열의 최대 길이는 48자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

*ObjectType*이 MQOT\_Q\_MGR인 경우, 이 이름은 *QMgrName*과 동일합니다.

### **ObjectType (MQLONG) - 입력**

오브젝트 유형.

*ObjectName*로 지정된 엔티티의 유형입니다. 다음 중 하나입니다.

**MQOT\_AUTH\_INFO**

인증 정보.

**MQOT\_CHANNEL**

채널.

**MQOT\_CLNTCONN\_CHANNEL**

클라이언트 연결 채널.

**MQOT\_LISTENER**

.

**MQOT\_NAMELIST**

이름 목록.

**MQOT\_PROCESS**

process definition.

**MQOT\_Q**

큐.

**MQOT\_Q\_MGR**

큐 관리자.

**MQOT\_SERVICE**

서비스.

**ComponentData (MQBYTE x ComponentDataLength) - 입출력(I/O)**

컴포넌트 데이터입니다.

이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

**Continuation (MQLONG) - 출력**

컴포넌트에서 설정한 연속 표시기입니다.

다음 값을 지정할 수 있습니다.

**MQZCI\_DEFAULT**

큐 관리자에 따른 연속입니다.

MQZ\_DELETE\_AUTHORITY의 경우 이는 MQZCI\_STOP과 동일한 효과가 있습니다.

**MQZCI\_CONTINUE**

다음 컴포넌트로 계속됩니다.

**MQZCI\_STOP**

다음 컴포넌트를 계속하지 않습니다.

**CompCode(MQLONG) - 출력**

완료 코드.

다음 중 하나입니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_FAILED**

호출에 실패했습니다.

**Reason(MQLONG) - 출력**

*CompCode*을(를) 규정하는 이유 코드.

*CompCode*이(가) MQCC\_OK인 경우:

**MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_FAILED인 경우:



## **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

## **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

이 이유 코드에 대한 자세한 정보는 [메시지 및 이유 코드를 참조하십시오](#).

## **C 호출**

```
MQZ_DELETE_AUTHORITY (QMgrName, ObjectName, ObjectType, ComponentData,  
&Continuation, &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48 QMgrName;          /* Queue manager name */  
MQCHAR48 ObjectName;       /* Object name */  
MQLONG   ObjectType;       /* Object type */  
MQBYTE   ComponentData[n]; /* Component data */  
MQLONG   Continuation;     /* Continuation indicator set by  
                             component */  
MQLONG   CompCode;         /* Completion code */  
MQLONG   Reason;          /* Reason code qualifying CompCode */
```

## **IBM i** IBM i의 MQZ\_ENUMERATE\_AUTHORITY\_DATA(권한 데이터 나열)

이 함수는 MQZAS\_VERSION\_4 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 첫 번째 호출에서 지정된 선택 기준에 일치하는 모든 권한 데이터를 검색하기 위해 반복적으로 호출합니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_ENUMERATE\_AUTHORITY\_DATA입니다.

## **구문**

```
MQZ_ENUMERATE_AUTHORITY_DATA (QMgrName, StartEnumeration,  
Filter, AuthorityBufferLength, AuthorityBuffer, AuthorityDataLength,  
ComponentData, Continuation, CompCode, Reason)
```

## **매개변수**

MQZ\_ENUMERATE\_AUTHORITY\_DATA 호출의 매개변수는 다음과 같습니다.

### **QMgrName(MQCHAR48) - 입력**

큐 관리자 이름.

컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

### **StartEnumeration (MQLONG) - 입력**

호출에서 나열을 시작해야 하는지 표시하는 플래그.

이 플래그는 호출이 권한 데이터의 나열을 시작하거나 MQZ\_ENUMERATE\_AUTHORITY\_DATA에 대한 이전 호출로 인해 시작된 권한 데이터 나열을 계속해야 하는지 여부를 표시합니다. 값은 다음 중 하나입니다.

#### **MQZSE\_START**

나열 시작.

권한 데이터의 나열을 시작하려면 이 값으로 호출을 호출합니다. **Filter** 매개변수는 이 호출 및 이후 호출에서 리턴된 권한 데이터를 선택하는 데 사용할 선택 기준을 지정합니다.

#### **MQZSE\_CONTINUE**

나열 계속.

권한 데이터의 나열을 계속하려면 이 값으로 호출을 호출합니다. 이 경우 **Filter** 매개변수는 무시되며 널 포인터로 지정될 수 있습니다 (선택 기준은 *StartEnumeration* 가 MQZSE\_START로 설정된 호출에 의해 지정된 **Filter** 매개변수에 의해 판별됨).

#### **Filter (MQZAD) - 입력**

필터.

*StartEnumeration*이 MQZSE\_START이면 *Filter*는 리턴할 권한 데이터를 선택하는 데 사용할 선택 기준을 지정합니다. *Filter*가 널 포인터이면 선택 기준이 사용되지 않습니다. 즉, 모든 권한 데이터가 리턴됩니다. 사용할 수 있는 선택 기준에 대한 자세한 정보는 [1640 페이지의 『IBM i의 MQZAD\(권한 데이터\)』](#)의 내용을 참조하십시오.

*StartEnumeration*이 MQZSE\_CONTINUE이면 *Filter*가 무시되고 널 포인터로 지정할 수 있습니다.

#### **AuthorityBufferLength (MQLONG) - 입력**

*AuthorityBuffer*의 길이.

이는 **AuthorityBuffer** 매개변수의 길이(바이트)입니다. 권한 버퍼는 리턴되는 데이터를 수용할 만큼 충분한 크기여야 합니다.

#### **AuthorityBuffer (MQZAD) - 출력**

권한 데이터.

권한 데이터가 리턴되는 버퍼입니다. 버퍼는 MQZAD 구조, MQZED 구조, 그리고 정의된 가장 긴 엔티티 이름과 가장 긴 도메인 이름을 수용할 만큼 충분한 크기여야 합니다.

**참고:** MQZAD가 항상 버퍼 시작 부분에 발생하므로 이 매개변수는 MQZAD로 정의됩니다. 그러나, 버퍼가 실제 MQZAD로 선언되면 버퍼가 너무 작습니다. MQZAD, MQZED 그리고 엔티티 및 도메인 이름을 수용할 수 있도록 MQZAD보다 커야 합니다.

#### **AuthorityDataLength (MQLONG) - 출력**

*AuthorityBuffer*에서 리턴된 데이터 길이.

*AuthorityBuffer*에서 리턴된 데이터의 길이입니다. 권한 버퍼가 너무 작으면 *AuthorityDataLength*를 필요한 버퍼의 길이로 설정합니다. 그러면 호출이 완료 코드 MQCC\_FAILED 및 이유 코드 MQRC\_BUFFER\_LENGTH\_ERROR를 리턴합니다.

#### **ComponentData (MQBYTE x ComponentDataLength) - 입출력(I/O)**

컴포넌트 데이터입니다.

이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

#### **Continuation (MQLONG) - 출력**

컴포넌트에서 설정한 연속 표시기입니다.

다음 값을 지정할 수 있습니다.

##### **MQZCI\_DEFAULT**

큐 관리자에 따른 연속입니다.

MQZ\_ENUMERATE\_AUTHORITY\_DATA의 경우 이는 MQZCI\_CONTINUE와 동일한 효과가 있습니다.

##### **MQZCI\_CONTINUE**

다음 컴포넌트로 계속됩니다.

##### **MQZCI\_STOP**

다음 컴포넌트를 계속하지 않습니다.

#### **CompCode(MQLONG) - 출력**

완료 코드.

다음 중 하나입니다.

## **MQCC\_OK**

정상적으로 완료되었습니다.

## **MQCC\_FAILED**

호출에 실패했습니다.

### **Reason(MQLONG) - 출력**

*CompCode*을(를) 규정하는 이유 코드.

*CompCode*이(가) MQCC\_OK인 경우:

## **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_FAILED인 경우:

## **MQRC\_BUFFER\_LENGTH\_ERROR**

(2005, X'7D5') 버퍼 길이 매개변수가 올바르지 않습니다.

## **MQRC\_NO\_DATA\_AVAILABLE**

(2379, X'94B') 데이터가 없습니다.

## **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

이 이유 코드에 대한 자세한 정보는 [메시지 및 이유 코드를 참조하십시오](#).

## **C 호출**

```
MQZ_ENUMERATE_AUTHORITY_DATA (QMgrName, StartEnumeration, &Filter,
                               AuthorityBufferLength,
                               &AuthorityBuffer,
                               &AuthorityDataLength, ComponentData,
                               &Continuation, &CompCode,
                               &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48  QMgrName;           /* Queue manager name */
MQLONG    StartEnumeration;  /* Flag indicating whether call should
                               start enumeration */

MQZAD     Filter;            /* Filter */
MQLONG    AuthorityBufferLength; /* Length of AuthorityBuffer */
MQZAD     AuthorityBuffer;   /* Authority data */
MQLONG    AuthorityDataLength; /* Length of data returned in
                               AuthorityBuffer */

MQBYTE    ComponentData[n];  /* Component data */
MQLONG    Continuation;      /* Continuation indicator set by
                               component */

MQLONG    CompCode;          /* Completion code */
MQLONG    Reason;            /* Reason code qualifying CompCode */
```

## **MQZ\_FREE\_USER - 프리 사용자**

이 함수는 MQZAS\_VERSION\_5 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 연관된 할당 자원을 비우기 위해 호출합니다. 애플리케이션이 모든 사용자 컨텍스트(예: MQDISC MQI 호출 동안)에서 실행을 마치면 호출됩니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_FREE\_USER입니다.

## **IBM i IBM i의 MQZ\_GET\_AUTHORITY(권한 가져오기)**

이 함수는 MQZAS\_VERSION\_1 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 지정된 오브젝트에 액세스할 수 있는 엔티티의 권한을 검색하기 위해 호출합니다.

이 함수(MQZEP용)에 대한 함수 ID는 MQZID\_GET\_AUTHORITY입니다.

## 구문

**MQZ\_GET\_AUTHORITY** (*QMgrName, EntityName, EntityType, ObjectName, ObjectType, Authority, ComponentData, Continuation, CompCode, Reason*)

## 매개변수

MQZ\_GET\_AUTHORITY 호출의 매개변수는 다음과 같습니다.

### **QMgrName(MQCHAR48) - 입력**

큐 관리자 이름.

컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

### **EntityName (MQCHAR12) - 입력**

엔티티 이름입니다.

오브젝트에 대한 액세스 권한을 검색할 엔티티의 이름입니다. 문자열의 최대 길이는 12자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

### **EntityType (MQLONG) - 입력**

엔티티 유형입니다.

엔티티 유형은 *EntityName*으로 지정됩니다. 다음 값을 지정할 수 있습니다.

#### **MQZAET\_PRINCIPAL**

프린시פל입니다.

#### **MQZAET\_GROUP**

그룹.

### **ObjectName (MQCHAR48) - 입력**

오브젝트 이름.

엔티티 권한이 검색될 오브젝트의 이름입니다. 문자열의 최대 길이는 48자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

*ObjectType*이 MQOT\_Q\_MGR인 경우, 이 이름은 *QMgrName*과 동일합니다.

### **ObjectType (MQLONG) - 입력**

오브젝트 유형.

엔티티 유형은 *ObjectName*으로 지정됩니다. 다음 중 하나입니다.

#### **MQOT\_AUTH\_INFO**

인증 정보.

#### **MQOT\_CHANNEL**

채널.

#### **MQOT\_CLNTCONN\_CHANNEL**

클라이언트 연결 채널.

#### **MQOT\_LISTENER**

.

#### **MQOT\_NAMELIST**

이름 목록.

#### **MQOT\_PROCESS**

process definition.

#### **MQOT\_Q**

큐.

**MQOT\_Q\_MGR**

큐 관리자.

**MQOT\_SERVICE**

서비스.

**Authority (MQLONG) - 출력**

엔티티의 권한입니다.

엔티티에 하나의 권한이 있는 경우, 이 필드는 적절한 권한 부여 조작(MQZAO\_\* constant)과 동일합니다. 둘 이상의 권한이 있는 경우, 이 필드는 해당되는 MQZAO\_\* 상수의 비트 단위의 OR입니다.

**ComponentData (MQBYTE x ComponentDataLength) - 입출력(I/O)**

컴포넌트 데이터입니다.

이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

**Continuation (MQLONG) - 출력**

컴포넌트에서 설정한 연속 표시기입니다.

다음 값을 지정할 수 있습니다.

**MQZCI\_DEFAULT**

큐 관리자에 따른 연속입니다.

MQZ\_GET\_AUTHORITY의 경우 이는 MQZCI\_CONTINUE와 동일한 효과가 있습니다.

**MQZCI\_CONTINUE**

다음 컴포넌트로 계속됩니다.

**MQZCI\_STOP**

다음 컴포넌트를 계속하지 않습니다.

**CompCode(MQLONG) - 출력**

완료 코드.

다음 중 하나입니다.

**MQCC\_OK**

정상적으로 완료되었습니다.

**MQCC\_FAILED**

호출에 실패했습니다.

**Reason(MQLONG) - 출력**

CompCode을(를) 규정하는 이유 코드.

CompCode이(가) MQCC\_OK인 경우:

**MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode이(가) MQCC\_FAILED인 경우:

**MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

**MQRC\_SERVICE\_ERROR**

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

**MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

**MQRC\_UNKNOWN\_ENTITY**

(2292, X'8F4') 서비스에서 알 수 없는 엔티티입니다.

이 이유 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

## C 호출

```
MQZ_GET_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,  
                  ObjectType, &Authority, ComponentData,  
                  &Continuation, &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48  QMgrName;           /* Queue manager name */  
MQCHAR12  EntityName;        /* Entity name */  
MQLONG    EntityType;        /* Entity type */  
MQCHAR48  ObjectName;        /* Object name */  
MQLONG    ObjectType;        /* Object type */  
MQLONG    Authority;         /* Authority of entity */  
MQBYTE    ComponentData[n]; /* Component data */  
MQLONG    Continuation;      /* Continuation indicator set by  
                             component */  
MQLONG    CompCode;          /* Completion code */  
MQLONG    Reason;            /* Reason code qualifying CompCode */
```

### IBM i IBM i의 MQZ\_GET\_EXPLICIT\_AUTHORITY(명시적 권한 가져오기)

이 함수는 MQZAS\_VERSION\_1 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 (**nobody** 그룹의 추가 권한 없이) 지정된 오브젝트에 액세스할 수 있는 특정 그룹의 권한 또는 지정된 오브젝트에 액세스할 수 있는 특정 프린시펄의 기본 그룹의 권한을 검색하기 위해 호출합니다.

이 함수(MQZEP용)에 대한 함수 ID는 MQZID\_GET\_EXPLICIT\_AUTHORITY입니다.

## 구문

```
MQZ_GET_EXPLICIT_AUTHORITY (QMgrName, EntityName, EntityType,  
                             ObjectName, ObjectType, Authority, ComponentData, Continuation, CompCode,  
                             Reason)
```

## 매개변수

MQZ\_GET\_EXPLICIT\_AUTHORITY 호출의 매개변수는 다음과 같습니다.

### QMgrName(MQCHAR48) - 입력

큐 관리자 이름.

컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

### EntityName (MQCHAR12) - 입력

엔티티 이름입니다.

오브젝트에 대한 액세스 권한을 검색할 엔티티의 이름입니다. 문자열의 최대 길이는 12자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

### EntityType (MQLONG) - 입력

엔티티 유형입니다.

엔티티 유형은 *EntityName*으로 지정됩니다. 다음 값을 지정할 수 있습니다.

#### MQZAET\_PRINCIPAL

프린시펄입니다.

#### MQZAET\_GROUP

그룹.

**ObjectName (MQCHAR48) - 입력**

오브젝트 이름.

엔티티 권한이 검색될 오브젝트의 이름입니다. 문자열의 최대 길이는 48자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

*ObjectType*이 MQOT\_Q\_MGR인 경우, 이 이름은 *QMGrName*과 동일합니다.

**ObjectType (MQLONG) - 입력**

오브젝트 유형.

엔티티 유형은 *ObjectName*으로 지정됩니다. 다음 중 하나입니다.

**MQOT\_AUTH\_INFO**

인증 정보.

**MQOT\_CHANNEL**

채널.

**MQOT\_CLNTCONN\_CHANNEL**

클라이언트 연결 채널.

**MQOT\_LISTENER**

.

**MQOT\_NAMELIST**

이름 목록.

**MQOT\_PROCESS**

process definition.

**MQOT\_Q**

큐.

**MQOT\_Q\_MGR**

큐 관리자.

**MQOT\_SERVICE**

서비스.

**Authority (MQLONG) - 출력**

엔티티의 권한입니다.

엔티티에 하나의 권한이 있는 경우, 이 필드는 적절한 권한 부여 조작(MQZAO\_\* constant)과 동일합니다. 둘 이상의 권한이 있는 경우, 이 필드는 해당되는 MQZAO\_\* 상수의 비트 단위의 OR입니다.

**ComponentData (MQBYTE x ComponentDataLength) - 입출력(I/O)**

컴포넌트 데이터입니다.

이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

**Continuation (MQLONG) - 출력**

컴포넌트에서 설정한 연속 표시기입니다.

다음 값을 지정할 수 있습니다.

**MQZCI\_DEFAULT**

큐 관리자에 따른 연속입니다.

MQZ\_GET\_EXPLICIT\_AUTHORITY의 경우 이는 MQZCI\_CONTINUE와 동일한 효과가 있습니다.

**MQZCI\_CONTINUE**

다음 컴포넌트로 계속됩니다.

**MQZCI\_STOP**

다음 컴포넌트를 계속하지 않습니다.

### CompCode(MQLONG) - 출력

완료 코드.

다음 중 하나입니다.

#### **MQCC\_OK**

정상적으로 완료되었습니다.

#### **MQCC\_FAILED**

호출에 실패했습니다.

### Reason(MQLONG) - 출력

CompCode을(를) 규정하는 이유 코드.

CompCode이(가) MQCC\_OK인 경우:

#### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode이(가) MQCC\_FAILED인 경우:

#### **MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

#### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

#### **MQRC\_UNKNOWN\_ENTITY**

(2292, X'8F4') 서비스에서 알 수 없는 엔티티입니다.

이 이유 코드에 대한 자세한 정보는 [메시지 및 이유 코드를 참조하십시오](#).

## C 호출

```

MQZ_GET_EXPLICIT_AUTHORITY (QMgrName, EntityName, EntityType,
                             ObjectName, ObjectType, &Authority,
                             ComponentData, &Continuation,
                             &CompCode, &Reason);

```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```

MQCHAR48  QMgrName;           /* Queue manager name */
MQCHAR12  EntityName;        /* Entity name */
MQLONG    EntityType;        /* Entity type */
MQCHAR48  ObjectName;       /* Object name */
MQLONG    ObjectType;       /* Object type */
MQLONG    Authority;        /* Authority of entity */
MQBYTE    ComponentData[n]; /* Component data */
MQLONG    Continuation;     /* Continuation indicator set by
                             component */
MQLONG    CompCode;         /* Completion code */
MQLONG    Reason;          /* Reason code qualifying CompCode */

```

## IBM i IBM i의 MQZ\_INIT\_AUTHORITY(권한 서비스 초기화)

이 함수는 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 컴포넌트를 구성하는 동안 호출합니다. 큐 관리자에게 정보를 제공하기 위해 MQZEP를 호출할 것으로 예상됩니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_INIT\_AUTHORITY입니다.

## 구문

```

MQZ_INIT_AUTHORITY (Hconfig, Options, QMgrName, ComponentDataLength,
                   ComponentData, Version, CompCode, Reason)

```



## 매개변수

MQZ\_INIT\_AUTHORITY 호출의 매개변수는 다음과 같습니다.

### Hconfig(MQHCONFIG) - 입력

구성 핸들입니다.

이 핸들은 초기화되는 특정 컴포넌트를 나타냅니다. 이는 MQZEP 함수로 큐 관리자를 호출할 때 컴포넌트에 의해 사용됩니다.

### Options (MQLONG) - 입력

초기화 옵션.

다음 중 하나입니다.

#### MQZIO\_PRIMARY

1차 초기화.

#### MQZIO\_SECONDARY

2차 초기화.

### QMgrName(MQCHAR48) - 입력

큐 관리자 이름.

컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

### ComponentDataLength (MQLONG) - 입력

컴포넌트 데이터의 길이.

*ComponentData* 영역의 길이(바이트)입니다. 이 길이는 컴포넌트 구성 데이터에서 정의됩니다.

### ComponentData (MQBYTE x ComponentDataLength) - 입출력(I/O)

컴포넌트 데이터입니다.

컴포넌트 1차 초기화 함수를 호출하기 전에 모두 0으로 초기화됩니다. 이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수(초기화 함수 포함)에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

### Version (MQLONG) - 입출력(I/O)

버전 번호.

초기화 함수에 대한 입력에서 큐 관리자가 지원하는 가장 높은 버전 번호를 식별합니다. 초기화 함수는 필요한 경우 초기화 권한 서비스 호출에서 지원하는 인터페이스의 버전으로 변경해야 합니다. 리턴 시 큐 관리자가 컴포넌트에서 리턴한 버전을 지원하지 않으면 컴포넌트의 MQZ\_TERM\_AUTHORITY 함수를 호출하고 이 컴포넌트를 추가로 사용하지 않습니다.

다음 값이 지원됩니다.

#### MQZAS\_VERSION\_1

버전 1.

#### MQZAS\_VERSION\_2

버전 2.

#### MQZAS\_VERSION\_3

버전 3.

#### MQZAS\_VERSION\_4

버전 4.

#### MQZAS\_VERSION\_5

버전 5.

#### MQZAS\_VERSION\_6

IBM WebSphere MQ 6.

### CompCode(MQLONG) - 출력

완료 코드.

다음 중 하나입니다.

#### **MQCC\_OK**

정상적으로 완료되었습니다.

#### **MQCC\_FAILED**

호출에 실패했습니다.

### Reason(MQLONG) - 출력

CompCode을(를) 규정하는 이유 코드.

CompCode이(가) MQCC\_OK인 경우:

#### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode이(가) MQCC\_FAILED인 경우:

#### **MQRC\_INITIALIZATION\_FAILED**

(2286, X'8EE') 정의되지 않은 이유로 초기화에 실패했습니다.

#### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

이 이유 코드에 대한 자세한 정보는 [메시지 및 이유 코드](#)를 참조하십시오.

## C 호출

```
MQZ_INIT_AUTHORITY (Hconfig, Options, QMgrName, ComponentDataLength,
                    ComponentData, &Version, &CompCode,
                    &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQHCONFIG  Hconfig;           /* Configuration handle */
MQLONG     Options;           /* Initialization options */
MQCHAR48   QMgrName;         /* Queue manager name */
MQLONG     ComponentDataLength; /* Length of component data */
MQBYTE     ComponentData[n]; /* Component data */
MQLONG     Version;          /* Version number */
MQLONG     CompCode;         /* Completion code */
MQLONG     Reason;           /* Reason code qualifying CompCode */
```

## IBM i IBM i의 MQZ\_INQUIRE(권한 서비스 조회)

이 함수는 MQZAS\_VERSION\_5 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 지원되는 기능을 조회하기 위해 호출합니다. 여러 서비스 컴포넌트를 사용하는 경우, 서비스 컴포넌트가 설치된 순서와 역순으로 호출됩니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_INQUIRE입니다.

## 구문

### **MQZ\_INQUIRE**

(QMgrName, SelectorCount, Selectors, IntAttrCount, IntAttrs, CharAttrLength, CharAttrs, SelectorReturned, ComponentData, Continuation, CompCode, Reason)

## 매개변수

MQZ\_INQUIRE 호출의 매개변수는 다음과 같습니다.

### **QMgrName(MQCHAR48) - 입력**

큐 관리자 이름.

컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

#### **SelectorCount (MQLONG) - 입력**

선택자의 수.

Selectors 매개변수에 제공된 선택자의 수입니다.

값은 0 - 256 사이에 있어야 합니다.

#### **Selectors (MQLONG x SelectorCount) - 입력**

선택자.

선택자의 배열. 각 선택자는 필수 속성을 식별하며 다음 유형 중 하나여야 합니다.

- MQIACF\_\* (정수)
- MQCACF\_\* (문자)

임의의 순서로 선택자를 지정할 수 있습니다. 배열의 선택자 수는 SelectorCount 매개변수로 표시됩니다.

선택자가 식별한 정수 속성이 Selectors에 표시되는 순서와 동일한 순서대로 IntAttrs 매개변수에 리턴됩니다.

선택자가 식별한 문자 속성이 Selectors에 표시되는 순서와 동일한 순서대로 CharAttrs 매개변수에 리턴됩니다.

#### **IntAttrCount (MQLONG) - 입력**

정수 속성의 수.

IntAttrs 매개변수에 제공된 정수 속성의 수.

해당 값은 0 - 256 범위에 있어야 합니다.

#### **IntAttrs (MQLONG x IntAttrCount) - 출력**

정수 속성.

정수 속성의 배열. 정수 속성은 Selectors 배열에서 해당 정수 선택자와 동일한 순서대로 리턴됩니다.

#### **CharAttrCount (MQLONG) - 입력**

문자 속성 버퍼의 길이.

CharAttrs 매개변수의 길이(바이트).

값이 최소한 요청된 문자 속성 길이의 합계여야 합니다. 요청한 문자 속성이 없으면 0이 유효한 값입니다.

#### **CharAttrs (MQLONG x CharAttrCount) - 출력**

문자 속성 버퍼.

함께 연결된 문자 속성을 포함한 버퍼입니다. 문자 속성은 Selectors 배열에서 해당 문자 선택자와 동일한 순서대로 리턴됩니다.

버퍼의 길이는 CharAttrCount 매개변수에서 제공합니다.

#### **SelectorReturned (MQLONGxSelectorCount) - 입력**

리턴되는 선택자.

Selectors 매개변수의 선택자가 요청한 세트에서 리턴된 속성을 식별하는 값의 배열입니다. 배열의 선택자 수는 SelectorCount 매개변수로 표시됩니다. 배열의 각 값이 Selectors 배열에 있는 해당 위치의 선택자와 관련이 있습니다. 각 값은 다음 중 하나입니다.

##### **MQZSL\_RETURNED**

Selectors 매개변수의 해당 선택자가 요청한 속성이 리턴되었습니다.

##### **MQZSL\_NOT\_RETURNED**

Selectors 매개변수의 해당 선택자가 요청한 속성이 리턴되지 않았습니다.

배열은 모든 값을 `MQZSL_NOT_RETURNED`로 사용하여 초기화됩니다. 권한 부여 서비스 컴포넌트가 속성을 리턴하면 배열의 적절한 값을 `MQZSL_RETURNED`로 설정합니다. 이렇게 하면 조회 호출이 발행되는 다른 권한 부여 서비스 컴포넌트가 이미 리턴된 속성을 식별할 수 있습니다.

#### **ComponentData (MQBYTE x ComponentDataLength) - 입출력(I/O)**

컴포넌트 데이터입니다.

이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 `MQZ_INIT_AUTHORITY` 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

#### **Continuation (MQLONG) - 출력**

연속 플래그입니다.

다음 값을 지정할 수 있습니다.

##### **MQZCI\_DEFAULT**

기타 컴포넌트에 종속되는 연속입니다.

##### **MQZCI\_STOP**

다음 컴포넌트를 계속하지 않습니다.

#### **CompCode(MQLONG) - 출력**

완료 코드.

다음 중 하나입니다.

##### **MQCC\_OK**

정상적으로 완료되었습니다.

##### **MQCC\_WARNING**

부분 완료입니다.

##### **MQCC\_FAILED**

호출에 실패했습니다.

#### **Reason(MQLONG) - 출력**

*CompCode*을(를) 규정하는 이유 코드.

*CompCode*이(가) `MQCC_OK`인 경우:

##### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) `MQCC_WARNING`인 경우:

##### **MQRC\_CHAR\_ATTRS\_TOO\_SHORT**

문자 속성을 위한 공간이 충분하지 않습니다.

##### **MQRC\_INT\_COUNT\_TOO\_SMALL**

정수 속성을 위한 공간이 충분하지 않습니다.

*CompCode*이(가) `MQCC_FAILED`인 경우:

##### **MQRC\_SELECTOR\_COUNT\_ERROR**

선택자의 수가 올바르지 않습니다.

##### **MQRC\_SELECTOR\_ERROR**

속성 선택자가 올바르지 않습니다.

##### **MQRC\_SELECTOR\_LIMIT\_EXCEEDED**

지정된 선택자가 너무 많습니다.

##### **MQRC\_INT\_ATTR\_COUNT\_ERROR**

정수 속성의 수가 올바르지 않습니다.

##### **MQRC\_INT\_ATTRS\_ARRAY\_ERROR**

정수 속성 배열이 올바르지 않습니다.

### **MQRC\_CHAR\_ATTR\_LENGTH\_ERROR**

문자 속성의 수가 올바르지 않습니다.

### **MQRC\_CHAR\_ATTRS\_ERROR**

문자 속성 문자열이 올바르지 않습니다.

### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

## **C 호출**

```
MQZ_INQUIRE (QMgrName, SelectorCount, Selectors, IntAttrCount,
              &IntAttrs, CharAttrLength, &CharAttrs,
              SelectorReturned, ComponentData, &Continuation,
              &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48  QMgrName;           /* Queue manager name */
MQLONG    SelectorCount;     /* Selector count */
MQLONG    Selectors[n];      /* Selectors */
MQLONG    IntAttrCount;      /* IntAttrs count */
MQLONG    IntAttrs[n];       /* Integer attributes */
MQLONG    CharAttrCount;     /* CharAttrs count */
MQLONG    CharAttrs[n];      /* Character attributes */
MQLONG    SelectorReturned[n]; /* Selector returned */
MQBYTE    ComponentData[n];  /* Component data */
MQLONG    Continuation;      /* Continuation indicator set by
                               component */
MQLONG    CompCode;          /* Completion code */
MQLONG    Reason;           /* Reason code qualifying CompCode */
```

## **IBM i** IBM i의 MQZ\_REFRESH\_CACHE(모든 권한 새로 고치기)

이 함수는 MQZAS\_VERSION\_3 권한 서비스 컴포넌트에서 제공됩니다. 큐 관리자가 컴포넌트 내부에 보관된 권한 부여 목록을 새로 고치기 위해 호출합니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_REFRESH\_CACHE(8L)입니다.

## **구문**

### **MQZ\_REFRESH\_CACHE**

(QMgrName, ComponentData, Continuation, CompCode, Reason)

## **매개변수**

### **QMgrName (MQCHAR48) - 입력**

큐 관리자 이름.

컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

### **ComponentData (MQBYTE x ComponentDataLength) - 입출력(I/O)**

컴포넌트 데이터입니다.

이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 ComponentDataLength 매개변수로 큐 관리자가 전달합니다.

### **Continuation (MQLONG) - 출력**

컴포넌트에서 설정한 연속 표시기입니다.

다음 값을 지정할 수 있습니다.

#### **MQZCI\_DEFAULT**

큐 관리자에 따른 연속입니다.

MQZ\_REFRESH\_CACHE의 경우 이는 MQZCI\_CONTINUE와 동일한 효과가 있습니다.

#### **MQZCI\_CONTINUE**

다음 컴포넌트로 계속됩니다.

#### **MQZCI\_STOP**

다음 컴포넌트를 계속하지 않습니다.

### **CompCode(MQLONG) - 출력**

완료 코드.

다음 중 하나입니다.

#### **MQCC\_OK**

정상적으로 완료되었습니다.

#### **MQCC\_FAILED**

호출에 실패했습니다.

### **Reason(MQLONG) - 출력**

CompCode를 규정하는 이유 코드.

CompCode가 MQCC\_OK인 경우:

#### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

CompCode가 MQCC\_FAILED인 경우:

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

## **C 호출**

```
MQZ_REFRESH_CACHE (QMgrName, ComponentData,  
                  &Continuation, &CompCode, &Reason);
```

매개변수를 다음과 같이 선언합니다.

```
MQCHAR48  QMgrName;           /* Queue manager name */  
MQBYTE    ComponentData[n];  /* Component data */  
MQLONG    Continuation;      /* Continuation indicator set by  
                             component */  
MQLONG    CompCode;          /* Completion code */  
MQLONG    Reason;           /* Reason code qualifying CompCode */
```

## **IBM i IBM i의 MQZ\_SET\_AUTHORITY(권한 설정)**

이 함수는 MQZAS\_VERSION\_1 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 엔티티에서 지정된 오브젝트에 액세스하는 데 필요한 권한을 설정하기 위해 호출합니다.

이 함수(MQZEP용)에 대한 함수 ID는 MQZID\_SET\_AUTHORITY입니다.

**참고:** 이 함수가 기존 권한을 대체합니다. 기존의 권한을 유지하려면 이 함수로 다시 설정해야 합니다.

## **구문**

**MQZ\_SET\_AUTHORITY** (*QMgrName, EntityName, EntityType, ObjectName, ObjectType, Authority, ComponentData, Continuation, CompCode, Reason*)

## 매개변수

MQZ\_SET\_AUTHORITY 호출의 매개변수는 다음과 같습니다.

### **QMgrName(MQCHAR48)** - 입력

큐 관리자 이름.

컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

### **EntityName (MQCHAR12)** - 입력

엔티티 이름입니다.

오브젝트에 대한 액세스 권한을 설정할 엔티티의 이름입니다. 문자열의 최대 길이는 12자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

### **EntityType (MQLONG)** - 입력

엔티티 유형입니다.

엔티티 유형은 *EntityName*으로 지정됩니다. 다음 값을 지정할 수 있습니다.

#### **MQZAET\_PRINCIPAL**

프린시펄입니다.

#### **MQZAET\_GROUP**

그룹.

### **ObjectName (MQCHAR48)** - 입력

오브젝트 이름.

액세스가 필요한 오브젝트 이름. 문자열의 최대 길이는 48자입니다. 길이가 더 짧을 경우에는 오른쪽을 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

*ObjectType*이 MQOT\_Q\_MGR인 경우, 이 이름은 *QMgrName*과 동일합니다.

### **ObjectType (MQLONG)** - 입력

오브젝트 유형.

엔티티 유형은 *ObjectName*으로 지정됩니다. 다음 중 하나입니다.

#### **MQOT\_AUTH\_INFO**

인증 정보.

#### **MQOT\_CHANNEL**

채널.

#### **MQOT\_CLNTCONN\_CHANNEL**

클라이언트 연결 채널.

#### **MQOT\_LISTENER**

.

#### **MQOT\_NAMELIST**

이름 목록.

#### **MQOT\_PROCESS**

process definition.

#### **MQOT\_Q**

큐.

#### **MQOT\_Q\_MGR**

큐 관리자.

## **MQOT\_SERVICE**

서비스.

### **Authority (MQLONG) - 입력**

검사할 권한.

하나의 권한을 설정하는 경우, 이 필드는 적절한 권한 부여 조작(MQZAO\_\* 상수)과 동일합니다. 둘 이상의 권한을 설정하는 경우, 해당 MQZAO\_\* 상수의 비트 단위 OR입니다.

### **ComponentData (MQBYTE x ComponentDataLength) - 입출력(I/O)**

컴포넌트 데이터입니다.

이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수로 큐 관리자가 전달합니다.

### **Continuation (MQLONG) - 출력**

컴포넌트에서 설정한 연속 표시기입니다.

다음 값을 지정할 수 있습니다.

#### **MQZCI\_DEFAULT**

큐 관리자에 따른 연속입니다.

MQZ\_SET\_AUTHORITY의 경우 이는 MQZCI\_STOP과 동일한 효과가 있습니다.

#### **MQZCI\_CONTINUE**

다음 컴포넌트로 계속됩니다.

#### **MQZCI\_STOP**

다음 컴포넌트를 계속하지 않습니다.

### **CompCode(MQLONG) - 출력**

완료 코드.

다음 중 하나입니다.

#### **MQCC\_OK**

정상적으로 완료되었습니다.

#### **MQCC\_FAILED**

호출에 실패했습니다.

### **Reason(MQLONG) - 출력**

*CompCode*을(를) 규정하는 이유 코드.

*CompCode*이(가) MQCC\_OK인 경우:

#### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_FAILED인 경우:

#### **MQRC\_NOT\_AUTHORIZED**

(2035, X'7F3') 액세스할 권한이 부여되지 않았습니다.

#### **MQRC\_SERVICE\_ERROR**

(2289, X'8F1') 서비스에 액세스하는 중 예상치 못한 오류가 발생했습니다.

#### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

#### **MQRC\_UNKNOWN\_ENTITY**

(2292, X'8F4') 서비스에서 알 수 없는 엔티티입니다.

## **C 호출**



```
MQZ_SET_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,
                   ObjectType, Authority, ComponentData,
                   &Continuation, &CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQCHAR48  QMgrName;           /* Queue manager name */
MQCHAR12  EntityName;        /* Entity name */
MQLONG    EntityType;        /* Entity type */
MQCHAR48  ObjectName;       /* Object name */
MQLONG    ObjectType;       /* Object type */
MQLONG    Authority;        /* Authority to be checked */
MQBYTE    ComponentData[n]; /* Component data */
MQLONG    Continuation;     /* Continuation indicator set by
                             component */
MQLONG    CompCode;         /* Completion code */
MQLONG    Reason;          /* Reason code qualifying CompCode */
```

## MQZ\_TERM\_AUTHORITY - 권한 서비스 종료

이 함수는 권한 서비스 컴포넌트에서 제공되며 큐 관리자가 이 컴포넌트의 서비스를 더 이상 필요로 하지 않을 때 호출합니다. 함수는 컴포넌트에서 필요한 모든 정리를 수행해야 합니다.

이 함수(MQZEP)에 대한 함수 ID는 MQZID\_TERM\_AUTHORITY입니다.

### 구문

**MQZ\_TERM\_AUTHORITY** (*Hconfig, Options, QMgrName, ComponentData, CompCode, Reason*)

### 매개변수

MQZ\_TERM\_AUTHORITY 호출의 매개변수는 다음과 같습니다.

#### **Hconfig(MQHCONFIG) - 입력**

구성 핸들입니다.

이 핸들은 종료되는 특정 컴포넌트를 나타냅니다.

#### **Options (MQLONG) - 입력**

종료 옵션.

다음 중 하나입니다.

##### **MQZTO\_PRIMARY**

1차 종료.

##### **MQZTO\_SECONDARY**

제2차 종료.

#### **QMgrName(MQCHAR48) - 입력**

큐 관리자 이름.

컴포넌트를 호출하는 큐 관리자의 이름입니다. 이 이름은 매개변수의 전체 길이를 공백으로 채웁니다. 이름은 널 문자로 끝나지 않습니다.

큐 관리자 이름이 정보용으로 컴포넌트에 전달되며, 권한 서비스 인터페이스의 경우 컴포넌트가 큐 관리자 이름을 정의된 방식으로 사용하지 않아도 됩니다.

#### **ComponentData (MQBYTE x ComponentDataLength) - 입출력(I/O)**

컴포넌트 데이터입니다.

이 데이터는 현재 특정 컴포넌트 대신 큐 관리자에 보관됩니다. 이 컴포넌트가 제공한 함수에 의해 데이터에 작성된 모든 변경사항은 보존되었다가 다음에 이 컴포넌트의 함수 중 하나가 호출될 때 표시됩니다.

이 데이터 영역의 길이는 MQZ\_INIT\_AUTHORITY 호출의 **ComponentDataLength** 매개변수에서 큐 관리자에 의해 전달됩니다.

MQZ\_TERM\_AUTHORITY 호출이 완료되면 큐 관리자는 이 데이터를 제거합니다.

#### **CompCode(MQLONG) - 출력**

완료 코드.

다음 중 하나입니다.

##### **MQCC\_OK**

정상적으로 완료되었습니다.

##### **MQCC\_FAILED**

호출에 실패했습니다.

#### **Reason(MQLONG) - 출력**

*CompCode*을(를) 규정하는 이유 코드.

*CompCode*이(가) MQCC\_OK인 경우:

##### **MQRC\_NONE**

(0, X'000') 보고할 이유가 없습니다.

*CompCode*이(가) MQCC\_FAILED인 경우:

##### **MQRC\_SERVICE\_NOT\_AVAILABLE**

(2285, X'8ED') 사용 불가능한 근본적인 서비스입니다.

##### **MQRC\_TERMINATION\_FAILED**

(2287, X'8FF') 정의되지 않은 이유로 종료가 실패했습니다.

이 이유 코드에 대한 자세한 정보는 [메시지 및 이유 코드를 참조하십시오](#).

## C 호출

```
MQZ_TERM_AUTHORITY (Hconfig, Options, QMgrName, ComponentData,  
&CompCode, &Reason);
```

서비스에 전달된 매개변수는 다음과 같이 선언됩니다.

```
MQHCONFIG  Hconfig;           /* Configuration handle */  
MQLONG     Options;          /* Termination options */  
MQCHAR48   QMgrName;        /* Queue manager name */  
MQBYTE     ComponentData[n]; /* Component data */  
MQLONG     CompCode;        /* Completion code */  
MQLONG     Reason;          /* Reason code qualifying CompCode */
```

## IBM i IBM i의 MQZAC(애플리케이션 컨텍스트)

이 매개변수는 호출 애플리케이션과 관련된 데이터를 지정합니다.

MQZAC 구조는 MQZ\_AUTHENTICATE\_USER 호출에서 **ApplicationContext** 매개변수에 사용됩니다.

### 필드

#### **StrucId (MQCHAR4)**

구조 ID입니다.

값은 다음과 같습니다.

##### **MQZAC\_STRUC\_ID**

애플리케이션 컨텍스트 구조의 ID.

C 프로그래밍 언어의 경우 MQZAC\_STRUC\_ID\_ARRAY 상수도 정의됩니다. 이 상수는 MQZAC\_STRUC\_ID와 동일한 값을 갖지만 문자열 대신 문자의 배열입니다.

이 필드는 서비스의 입력 필드입니다.

**Version (MQLONG)**

구조 버전 번호입니다.

값은 다음과 같습니다.

**MQZAC\_VERSION\_1**

버전-1 애플리케이션 컨텍스트 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

**MQZAC\_CURRENT\_VERSION**

애플리케이션 컨텍스트 구조의 현재 버전.

이 필드는 서비스의 입력 필드입니다.

**ProcessId (MQPID)**

프로세스 ID.

애플리케이션의 프로세스 ID.

**ThreadId (MQTID)**

스레드 ID.

애플리케이션의 스레드 ID.

**ApplName (MQCHAR28)**

애플리케이션 이름.

애플리케이션 이름.

**UserID (MQCHAR12)**

사용자 ID.

IBM i 시스템의 경우 애플리케이션 작업을 작성할 때 적용하는 사용자 프로파일입니다. IBM i에서 애플리케이션 작업에서 QWTSETP API를 사용해 프로파일 스왑을 완료하는 경우 현재 사용자 프로파일이 리턴됩니다.

**EffectiveUserID (MQCHAR12)**

유효한 사용자 ID.

IBM i 시스템의 경우 애플리케이션 작업의 현재 사용자 프로파일입니다.

**Environment (MQLONG)**

환경.

이 필드는 호출이 작성된 환경을 지정합니다.

값은 다음 중 하나일 수 있습니다.

**MQXE\_COMMAND\_SERVER**

명령 서버.

**MQXE\_MQSC**

runmqsc 명령 해석기.

**MQXE\_MCA**

메시지 채널 에이전트

**MQXE\_OTHER**

정의되지 않은 환경

**CallerType (MQLONG)**

호출자 유형.

이 필드는 호출한 프로그램의 유형을 지정합니다.

값은 다음 중 하나일 수 있습니다.

**MQXACT\_EXTERNAL**

호출이 큐 관리자 외부에 있습니다.

## MQXACT\_INTERNAL

호출이 큐 관리자 내부에 있습니다.

## AuthenticationType (MQLONG)

인증 유형.

이 필드는 수행되는 인증의 유형을 지정합니다.

값은 다음 중 하나일 수 있습니다.

## MQZAT\_INITIAL\_CONTEXT

사용자 컨텍스트 초기화로 인한 인증 호출입니다. 이 값은 MQCONN 또는 MQCONNX 호출 동안 사용됩니다.

## MQZAT\_CHANGE\_CONTEXT

사용자 컨텍스트 변경으로 인한 인증 호출입니다. 이 값은 MCA가 사용자 컨텍스트를 변경할 때 사용됩니다.

v

## BindType (MQLONG)

바인드 유형.

이 필드는 사용 중인 바인딩의 유형을 지정합니다.

값은 다음 중 하나일 수 있습니다.

## MQCNO\_FASTPATH\_BINDING

빠른 경로 바인딩.

## MQCNO\_SHARED\_BINDING

공유 바인딩.

## MQCNO\_ISOLATED\_BINDING

격리된 바인딩.

## C 선언

```
typedef struct tagMQZAC MQZAC;
struct tagMQZAC {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG     Version;          /* Structure version number */
    MQPID      ProcessId;        /* Process identifier */
    MQTID      ThreadId;         /* Thread identifier */
    MQCHAR28   ApplName;         /* Application name */
    MQCHAR12   UserID;           /* User identifier */
    MQCHAR12   EffectiveUserID;  /* Effective user identifier */
    MQLONG     Environment;      /* Environment */
    MQLONG     CallerType;       /* Caller type */
    MQLONG     AuthenticationType; /* Authentication type */
    MQLONG     BindType;         /* Bind type */
};
```

## IBM i IBM i의 MQZAD(권한 데이터)

MQZAD 구조는 MQZ\_ENUMERATE\_AUTHORITY\_DATA 호출에서 두 개의 매개변수에 사용됩니다.

**Filter** 및 **AuthorityBuffer** 매개변수에 대한 자세한 정보는 [1621 페이지의 『IBM i의 MQZ\\_ENUMERATE\\_AUTHORITY\\_DATA\(권한 데이터 나열\)』](#)의 내용을 참조하십시오.

- 호출에 입력되는 **Filter** 매개변수에는 MQZAD가 사용됩니다. 이 매개변수는 호출로 리턴된 권한 데이터를 선택하는 데 사용할 선택 기준을 지정합니다.
- 호출에서 출력된 **AuthorityBuffer** 매개변수에도 MQZAD가 사용됩니다. 이 매개변수는 프로파일 이름, 오브젝트 유형 및 엔티티의 조합에 대한 권한을 지정합니다.

## 필드

### StrucId (MQCHAR4)

구조 ID입니다.

값은 다음과 같습니다.

### MQZAD\_STRUC\_ID

권한 데이터 구조 ID.

C 프로그래밍 언어의 경우, 상수 MQZAD\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQZAD\_STRUC\_ID와 동일한 값을 갖지만, 문자열이 아니라 문자 배열입니다.

이 필드는 서비스의 입력 필드입니다.

### Version(MQLONG)

구조 버전 번호입니다.

값은 다음과 같습니다.

### MQZAD\_VERSION\_1

버전-1 권한 데이터 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

### MQZAD\_CURRENT\_VERSION

권한 데이터 구조의 현재 버전.

이 필드는 서비스의 입력 필드입니다.

### ProfileName (MQCHAR48)

프로파일 이름.

**Filter** 매개변수의 경우, 이 필드는 권한 데이터가 필요한 프로파일 이름입니다. 이름이 필드 끝까지 또는 첫 번째 널(null) 문자까지 완전히 공백인 경우, 모든 프로파일 이름의 권한 데이터가 리턴됩니다.

**AuthorityBuffer** 매개변수의 경우, 이 필드는 지정된 선택 기준과 일치하는 프로파일의 이름입니다.

### ObjectType (MQLONG)

오브젝트 유형.

**Filter** 매개변수의 경우, 이 필드는 권한 데이터가 필요한 오브젝트 유형입니다. 값이 MQOT\_ALL이면 모든 오브젝트 유형의 권한 데이터가 리턴됩니다.

**AuthorityBuffer** 매개변수의 경우 이 필드는 **ProfileName** 로 식별되는 프로파일이 적용되는 오브젝트 유형입니다.

값은 다음 중 하나입니다. **Filter** 매개변수의 경우, 값 MQOT\_ALL도 유효합니다.

### MQOT\_AUTH\_INFO

인증 정보.

### MQOT\_CHANNEL

채널.

### MQOT\_CLNTCONN\_CHANNEL

클라이언트 연결 채널.

### MQOT\_LISTENER

.

### MQOT\_NAMELIST

이름 목록.

### MQOT\_PROCESS

process definition.

### MQOT\_Q

큐.

**MQOT\_Q\_MGR**

큐 관리자.

**MQOT\_SERVICE**

서비스.

**Authority (MQLONG)**

권한.

**Filter** 매개변수의 경우, 이 필드가 무시됩니다.

**AuthorityBuffer** 매개변수의 경우, 이 필드는 **ProfileName** 및 **ObjectType**으로 식별된 오브젝트에 대한 엔티티의 권한을 나타냅니다. 엔티티에 하나의 권한만 있는 경우, 이 필드는 적절한 권한 부여 값 (MQZAO\_\* 상수)과 동일합니다. 엔티티에 둘 이상의 권한이 있는 경우, 이 필드는 해당되는 MQZAO\_\* 상수의 비트 단위 OR입니다.

**EntityDataPtr (PMQZED)**

엔티티를 식별하는 MQZED 구조의 주소.

**Filter** 매개변수의 경우, 이 필드는 권한 데이터가 필요한 엔티티를 식별하는 MQZED 구조를 가리킵니다. **EntityDataPtr**가 널(null) 포인터인 경우, 모든 엔티티의 권한 데이터가 리턴됩니다.

**AuthorityBuffer** 매개변수의 경우, 이 필드는 리턴된 권한 데이터가 있었던 엔티티를 식별하는 MQZED 구조를 가리킵니다.

**EntityType (MQLONG)**

엔티티 유형입니다.

**Filter** 매개변수의 경우, 이 필드는 권한 데이터가 필요한 엔티티 유형을 지정합니다. 값이 MQZAET\_NONE이면 모든 엔티티 유형의 권한 데이터가 리턴됩니다.

**AuthorityBuffer** 매개변수의 경우, 이 필드는 **EntityDataPtr**가 가리키는 MQZED 구조로 식별되는 엔티티의 유형을 지정합니다.

값은 다음 중 하나입니다. **Filter** 매개변수의 경우, 값 MQZAET\_NONE도 유효합니다.

**MQZAET\_PRINCIPAL**

프린시펄입니다.

**MQZAET\_GROUP**

그룹.

**Options (MQAUTHOPT)**

옵션.

이 필드는 표시된 프로파일을 제어하는 옵션을 지정합니다.

다음 중 하나를 지정해야 합니다.

**MQAUTHOPT\_NAME\_ALL\_MATCHING**

모든 프로파일 표시

**MQAUTHOPT\_NAME\_EXPLICIT**

**ProfileName** 필드에 지정된 것과 정확하게 동일한 이름의 프로파일을 표시합니다.

또한, 다음 중 하나도 지정해야 합니다.

**MQAUTHOPT\_ENTITY\_SET**

엔티티가 **ProfileName**로 지정된 오브젝트에 대해 갖는 누적 권한을 계산하는 데 사용되는 모든 프로파일을 표시합니다. **ProfileName** 필드는 와일드카드 문자를 포함하지 않아야 합니다.

- 지정된 엔티티가 프린시펄인 경우, 세트{entity, groups}의 각 멤버에 대해 오브젝트에 적용되는 가장 적합한 프로파일이 표시됩니다.
- 지정된 엔티티가 그룹인 경우, 오브젝트에 적용되는 그룹의 가장 적합한 프로파일이 표시됩니다.
- 이 값이 지정되면 **ProfileName**, **ObjectType**, **EntityType**의 값 및 **EntityDataPtr** MQZED 구조에 지정된 엔티티 이름은 모두 공백이 아니어야 합니다.

**MQAUTHOPT\_NAME\_ALL\_MATCHING**을 지정한 경우, 다음을 지정할 수도 있습니다.

## MQAUTHOPT\_ENTITY\_EXPLICIT

**EntityDataPtr** MQZED 구조에 지정된 엔티티 이름과 정확하게 동일한 엔티티 이름의 프로파일을 표시합니다.

## C 선언

```
typedef struct tagMQZAD MQZAD;  
struct tagMQZAD {  
    MQCHAR4    StrucId;           /* Structure identifier */  
    MQLONG     Version;          /* Structure version number */  
    MQCHAR48   ProfileName;     /* Profile name */  
    MQLONG     ObjectType;       /* Object type */  
    MQLONG     Authority;        /* Authority */  
    PMQZED     EntityDataPtr;    /* Address of MQZED structure identifying an  
                                entity */  
    MQLONG     EntityType;       /* Entity type */  
    MQAUTHOPT  Options;         /* Options */  
};
```

## IBM i IBM i의 MQZED(엔티티 디스크립터)

MQZED 구조는 권한을 검사할 엔티티를 지정하기 위해 많은 권한 서비스 호출에 사용됩니다.

## 필드

### StrucId (MQCHAR4)

구조 ID입니다.

값은 다음과 같습니다.

### MQZED\_STRUC\_ID

엔티티 디스크립터 구조 ID.

C 프로그래밍 언어의 경우, 상수 MQZED\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQZED\_STRUC\_ID와 동일한 값을 갖지만, 문자열이 아니라 문자 배열입니다.

이 필드는 서비스의 입력 필드입니다.

### Version(MQLONG)

구조 버전 번호입니다.

값은 다음과 같습니다.

### MQZED\_VERSION\_1

버전-1 엔티티 디스크립터 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

### MQZED\_CURRENT\_VERSION

엔티티 디스크립터 구조의 현재 버전.

이 필드는 서비스의 입력 필드입니다.

### EntityNamePtr (PMQCHAR)

엔티티 이름 주소.

권한을 검사할 엔티티 이름에 대한 포인터입니다.

### EntityDomainPtr (PMQCHAR)

엔티티 도메인 이름 주소.

권한을 검사할 엔티티 정의가 포함된 도메인 이름에 대한 포인터입니다.

### SecurityId (MQBYTE40)

보안 ID.

권한을 검사할 보안 ID입니다.

## CorrelationPtr (MQPTR)

상관 포인터.

사용자 인증 함수 및 다른 적절한 OAM 함수 간에 상관 데이터를 원활하게 전달할 수 있습니다.

## C 선언

```
typedef struct tagMQZED MQZED;  
struct tagMQZED {  
    MQCHAR4    StrucId;           /* Structure identifier */  
    MQLONG     Version;          /* Structure version number */  
    PMQCHAR    EntityNamePtr;    /* Address of entity name */  
    PMQCHAR    EntityDomainPtr; /* Address of entity domain name */  
    MQBYTE40   SecurityId;       /* Security identifier */  
    MQPTR      CorrelationPtr;   /* Address of correlation data */  
};
```

## IBM i IBM i의 MQZFP(비우기 매개변수)

이 매개변수는 비우려는 자원과 관련된 데이터를 지정합니다.

MQZFP 구조는 MQZ\_FREE\_USER 호출에서 **FreeParms** 매개변수에 사용됩니다.

## 필드

### StrucId (MQCHAR4)

구조 ID입니다.

값은 다음과 같습니다.

### MQZFP\_STRUC\_ID

비우기 매개변수 구조의 ID.

C 프로그래밍 언어의 경우, 상수 MQZFP\_STRUC\_ID\_ARRAY도 정의됩니다. 이는 MQZFP\_STRUC\_ID와 동일한 값을 갖지만, 문자열이 아니라 문자 배열입니다.

이 필드는 서비스의 입력 필드입니다.

### Version(MQLONG)

구조 버전 번호입니다.

값은 다음과 같습니다.

### MQZFP\_VERSION\_1

버전-1 비우기 매개변수 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

### MQZFP\_CURRENT\_VERSION

비우기 매개변수 구조의 현재 버전.

이 필드는 서비스의 입력 필드입니다.

### Reserved (MQBYTE8)

예약된 필드.

초기값은 널입니다.

### CorrelationPtr (MQPTR)

상관 포인터.

비우려는 자원과 관련된 상관 데이터의 주소입니다.

## C 선언

```
typedef struct tagMQZFP MQZFP;  
struct tagMQZFP {
```



```

MQCHAR4  StrucId;          /* Structure identifier */
MQLONG   Version;        /* Structure version number */
MQBYTE8  Reserved;       /* Reserved field */
MQPTR    CorrelationPtr; /* Address of correlation data */
};

```

## IBM i IBM i의 MQZIC(ID 컨텍스트)

MQZIC 구조는 MQZ\_AUTHENTICATE\_USER 호출에서 **IdentityContext** 매개변수에 사용됩니다.

MQZIC 구조는 메시지를 큐에 처음 넣은 애플리케이션의 사용자를 식별하는 ID 컨텍스트 정보를 포함합니다.

- 큐 관리자가 **UserIdentifier** 필드에 사용자 식별 이름을 입력하는데, 큐 관리자가 이 작업을 수행하는 방법은 애플리케이션이 실행되는 환경에 따라 다릅니다.
- 큐 관리자가 메시지를 넣은 애플리케이션에서 판별한 토큰 또는 번호를 **AccountingToken** 필드에 입력합니다.
- 애플리케이션은 사용자에 대해 포함하려는 추가 정보(예: 암호화된 비밀번호)에 이 **AppIdentityData** 필드를 사용할 수 있습니다.

적절한 권한이 있는 애플리케이션은 MQZ\_AUTHENTICATE\_USER 함수를 사용하여 ID 컨텍스트를 설정할 수 있습니다.

Windows 시스템 보안 ID (SID) 는 메시지가 IBM MQ for Windows 아래에 작성될 때 **AccountingToken** 필드에 저장됩니다. SID는 **UserIdentifier** 필드를 보충하고 사용자의 신임 정보를 설정하는 데 사용할 수 있습니다.

### 필드

#### StrucId (MQCHAR4)

구조 ID입니다.

값은 다음과 같습니다.

#### MQZIC\_STRUC\_ID

ID 컨텍스트 구조의 ID.

C 프로그래밍 언어의 경우, MQZIC\_STRUC\_ID\_ARRAY 상수도 정의됩니다. 이 상수는 MQZIC\_STRUC\_ID와 동일한 값을 갖지만 문자열 대신 문자의 배열입니다.

이 필드는 서비스의 입력 필드입니다.

#### Version(MQLONG)

구조 버전 번호입니다.

값은 다음과 같습니다.

#### MQZIC\_VERSION\_1

버전-1 ID 컨텍스트 구조.

다음 상수는 현재 버전의 버전 번호를 지정합니다.

#### MQZIC\_CURRENT\_VERSION

ID 컨텍스트 구조의 현재 버전.

이 필드는 서비스의 입력 필드입니다.

#### UserIdentifier (MQCHAR12)

사용자 ID.

메시지의 **ID 컨텍스트**의 부분입니다.

*UserIdentifier*는 메시지를 생성한 애플리케이션의 사용자 ID를 지정합니다. 큐 관리자는 이 정보를 문자 데이터로 처리하지만 그 형식은 정의하지 않습니다. *UserIdentifier* 필드에 대한 자세한 정보는 [445 페이지의 『MQMD의 경우 UserIdentifier \(MQCHAR12\)』](#)의 내용을 참조하십시오.

#### AccountingToken (MQBYTE32)

계정 토큰.

메시지의 **ID 컨텍스트**의 부분입니다.

*AccountingToken*은 애플리케이션이 메시지의 결과로 완료된 작업에 적절히 청구할 수 있도록 합니다. 큐 관리자는 이 정보를 비트 문자열로 처리하며 그 콘텐츠를 검사하지 않습니다. *AccountingToken* 필드에 대한 자세한 정보는 [446 페이지의 『MQMD의 경우 AccountingToken \(MQBYTE32\)』](#)의 내용을 참조하십시오.

### ApplIdentityData (MQCHAR32)

ID와 관련된 애플리케이션 데이터.

메시지의 **ID 컨텍스트**의 부분입니다.

*ApplIdentityData*는 애플리케이션 스위트에 의해 정의된 정보이며, 메시지 원본에 대한 추가 정보를 제공하는 데 사용될 수 있습니다. 예를 들어, ID 데이터가 신뢰되는지 여부를 표시하기 위해 적절한 사용자 권한을 갖고 실행하는 애플리케이션이 이를 설정할 수 있습니다. *ApplIdentityData* 필드에 대한 자세한 정보는 [448 페이지의 『MQMD의 ApplIdentity데이터 \(MQCHAR32\)』](#)의 내용을 참조하십시오.

## C 선언

```
typedef struct tagMQZED MQZED;
struct tagMQZED {
    MQCHAR4    StrucId;           /* Structure identifier */
    MQLONG    Version;          /* Structure version number */
    MQCHAR12   UserIdentifier;   /* User identifier */
    MQBYTE32   AccountingToken; /* Accounting token */
    MQCHAR32   ApplIdentityData; /* Application data relating to identity */
};
```

## IBM MQ .NET 클래스 및 인터페이스

IBM MQ .NET 클래스 및 인터페이스가 알파벳순으로 나열되어 있습니다. 특성, 메소드 및 구성자에 대해 설명합니다.

### MQAsyncStatus.NET 클래스

이전 MQI 활동의 상태에 조회하려면 *MQAsyncStatus*를 사용하십시오. 예를 들어, 이전 비동기 Put 조작의 성공을 조회합니다. *MQAsyncStatus*는 MQSTS 데이터 구조의 기능을 캡슐화합니다.

### 클래스

```
System.Object
├── IBM.WMQ.MQBase
│   └── IBM.WMQ.MQBaseObject
│       └── IBM.WMQ.MQAsyncStatus
```

```
public class IBM.WMQ.MQAsyncStatus extends IBM.WMQ.MQBaseObject;
```

- [1646 페이지의 『특성』](#)
- [1647 페이지의 『구성자』](#)

### 특성

특성을 가져올 때 전달되는 *MQException*에 대한 테스트.

```
public static int CompCode {get;}
```

첫 번째 오류 또는 경고의 완료 코드.

```
public static int Reason {get;}
```

첫 번째 오류 또는 경고의 이유 코드.

```
public static int PutSuccessCount {get;}
```

성공적인 비동기 MQI Put 호출의 수.

```
public static int PutWarningCount {get;}
```

경고와 함께 성공한 비동기 MQI Put 호출의 수.

```
public static int PutFailureCount {get;}
```

실패한 비동기 MQI Put 호출의 수.

```
public static int ObjectType {get;}
```

첫 번째 오류에 대한 오브젝트 유형. 가능한 값은 다음과 같습니다.

- MQC.MQOT\_ALIAS\_Q
- MQC.MQOT\_LOCAL\_Q
- MQC.MQOT\_MODEL\_Q
- MQC.MQOT\_Q
- MQC.MQOT\_REMOTE\_Q
- MQC.MQOT\_TOPIC
- 0. 리턴된 오브젝트가 없음을 의미합니다.

```
public static string ObjectName {get;}
```

오브젝트 이름.

```
public static string ObjectQMgrName {get;}
```

오브젝트 큐 관리자 이름.

```
public static string ResolvedObjectName {get;}
```

해석된 오브젝트 이름.

```
public static string ResolvedObjectQMgrName {get;}
```

해석된 오브젝트 큐 관리자 이름.

## 구성자

```
public MQAsyncStatus() throws MQException;
```

구성자 메소드로, 0 또는 공백으로 적절하게 초기화된 필드를 사용하여 오브젝트를 구성합니다.

## MQAuthenticationInformationRecord.NET 클래스

MQAuthenticationInformationRecord 를 사용하여 IBM MQ TLS 클라이언트 연결에서 사용될 인증자에 대한 정보를 지정하십시오. MQAuthenticationInformationRecord는 인증 정보 레코드, MQAIR을 캡슐화합니다.

## 클래스

```
System.Object
├── IBM.WMQ.MQAuthenticationInformationRecord
```

```
public class IBM.WMQ.MQAuthenticationInformationRecord extends System.Object;
```

- [1648 페이지의 『특성』](#)

- 1648 페이지의 『구성자』

## 특성

특성을 가져올 때 전달되는 MQException에 대한 테스트.

```
public long Version {get; set;}
```

구조 버전 번호입니다.

```
public long AuthInfoType {get; set;}
```

인증 정보의 유형입니다. 이 속성은 다음 값 중 하나로 설정해야 합니다.

- OCSP - OCSP를 사용하여 인증서 폐기 상태 검사가 완료되었습니다.
- CRLLDAP - LDAP 서버의 인증서 폐기 목록을 사용하여 인증서 폐기 상태 검사가 완료되었습니다.

```
public string AuthInfoConnName {get; set;}
```

선택적 포트 번호가 있는 LDP 서버가 실행 중인 호스트의 DNS 이름 또는 IP 주소. 이 키워드는 필수입니다.

```
public string LDAPPassword {get; set;}
```

LDAP 서버에 액세스하는 사용자의 식별 이름과 연관된 비밀번호. 이 특성은 **AuthInfoType**이(가) CRLLDAP로 설정된 경우에만 적용됩니다.

```
public string LDAPUserName {get; set;}
```

LDAP 서버에 액세스하는 사용자의 식별 이름. 이 특성을 설정할 때 LDAPUserNameLength 및 LDAPUserNamePtr은 자동으로 올바르게 설정됩니다. 이 특성은 AuthInfoType이 CRLLDAP로 설정된 경우에만 적용됩니다.

```
public string OCSPResponderURL {get; set;}
```

OCSP 응답자에게 연락할 수 있는 URL. 이 특성은 AuthInfoType이 OCSP로 설정될 때에만 적용됩니다.

이 필드는 대소문자를 구분합니다. 이는 소문자로 된 http:// 문자열로 시작해야 합니다. 나머지 URL은 OCSP 서버 구현에 따라 대소문자를 구분할 수도 있습니다.

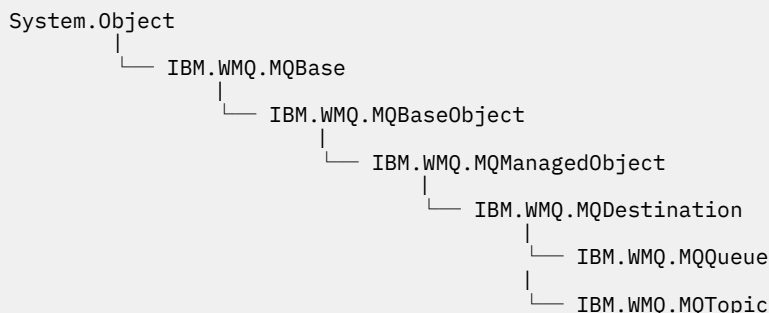
## 구성자

```
MQAuthenticationInformationRecord();
```

## MQDestination.NET 클래스

MQQueue 및 MQTopic에 공통되는 메소드에 액세스하려면 MQDestination을 사용하십시오. MQDestination은 추상 기본 클래스이고 인스턴스화할 수 없습니다.

## 클래스



```
public class IBM.WMQ.MQDestination extends IBM.WMQ.MQManagedObject;
```

- [1649 페이지의 『특성』](#)
- [1649 페이지의 『메소드』](#)
- [1650 페이지의 『구성자』](#)

## 특성

특성을 가져올 때 전달되는 MQException에 대한 테스트.

**public DateTime CreationDateTime {get;}**

큐 또는 토픽이 작성된 날짜 및 시간. 원래 MQQueue에 포함되어 있던 이 특성이 기본 MQDestination 클래스로 이동했습니다.

기본값은 없습니다.

**public int DestinationType {get;}**

사용 중인 목적지의 유형을 설명하는 정수 값. 서브클래스 구성자 MQQueue 또는 MQTopic에서 초기화된 이 값은 다음 값 중 하나를 사용할 수 있습니다.

- MQOT\_Q
- MQOT\_TOPIC

기본값은 없습니다.

## 메소드

**public void Get(MQMessage message);**

**public void Get(MQMessage message, MQGetMessageOptions getMessageOptions);**

**public void Get(MQMessage message, MQGetMessageOptions getMessageOptions, int MaxMsgSize);**

MQException을 전달합니다.

목적지가 MQQueue 오브젝트이면 큐에서, 목적지가 MQTopic 오브젝트이면 토픽에서 메시지를 가져오며, 이 가져오기를 수행하기 위해 MQGetMessageOptions의 기본 인스턴스를 사용합니다.

가져오기에 실패하면 MQMessage 오브젝트가 변경되지 않습니다. 가져오기에 성공하면 MQMessage의 메시지 데이터 부분이 수신되는 메시지의 메시지 디스크립터 및 메시지 데이터로 대체됩니다.

특정 MQQueueManager 에서 IBM MQ 로의 모든 호출은 동기적입니다. 따라서, Get(대기 포함)을 수행하는 경우 동일한 MQQueueManager를 사용하는 다른 모든 스레드가 Get 호출이 완료될 때까지 추가 IBM MQ 호출하지 못하도록 차단됩니다. 동시에 IBM MQ에 액세스하기 위해 여러 개의 스레드가 필요한 경우, 각 스레드가 고유의 MQQueueManager 오브젝트를 작성해야 합니다.

### 메시지

메시지 디스크립터 및 리턴된 메시지 데이터를 포함합니다. 메시지 디스크립터의 필드 중 일부는 입력 매개변수입니다. MessageId 및 CorrelationId 입력 매개변수가 필요에 따라 설정되었는지 확인해야 합니다.

다시 연결 가능한 클라이언트는 다시 연결에 성공한 후 MQGM\_SYNCPOINT에 따라 수신된 메시지에 대해 이유 코드 MQRC\_BACKED\_OUT을 리턴합니다.

### getMessageOptions

Get 조치를 제어하는 옵션.

MQC.MQGMO\_CONVERT 옵션을 사용하면 1바이트 문자 코드에서 2바이트 코드로 변환할 때 이유 코드 MQC.MQRC\_CONVERTED\_STRING\_TOO\_BIG과 함께 예외가 발생할 수 있습니다. 이 경우, 메시지가 변환 없이 버퍼로 복사됩니다.

getMessageOptions가 지정되지 않은 경우, 사용된 메시지 옵션은 MQGMO\_NOWAIT입니다.

reconnectable 클라이언트에서 MQGMO\_LOGICAL\_ORDER 옵션을 사용하는 경우 MQRC\_RECONNECT\_INCOMPATIBLE 이유 코드가 리턴됩니다.

## MaxMsgSize

이 메시지 오브젝트가 수신하는 최대 메시지. 큐의 메시지가 이 크기보다 크면 다음 두 가지 중 하나가 발생합니다.

- MQGMO\_ACCEPT\_TRUNCATED\_MSG 플래그가 MQGetMessageOptions 오브젝트에 설정된 경우, 메시지에 최대한 많은 메시지 데이터가 입력됩니다. MQCC\_WARNING 완료 코드 및 MQRC\_TRUNCATED\_MSG\_ACCEPTED 이유 코드와 함께 예외가 전달됩니다.
- MQGMO\_ACCEPT\_TRUNCATED\_MSG 플래그가 설정되지 않은 경우, 메시지가 큐에 유지됩니다. MQCC\_WARNING 완료 코드 및 MQRC\_TRUNCATED\_MSG\_FAILED 이유 코드와 함께 예외가 전달됩니다.

MaxMsgSize를 지정하지 않으면 전체 메시지가 검색됩니다.

```
public void Put(MQMessage message);  
public void Put(MQMessage message, MQPutMessageOptions putMessageOptions);
```

MQException을 전달합니다.

목적지가 MQQueue 오브젝트이면 메시지를 큐에 넣고, 목적지가 MQTopic 오브젝트이면 메시지를 토픽에 발행합니다.

Put 호출이 완료된 후 MQMessage 오브젝트를 수정해도 IBM MQ 큐 또는 발행물 토픽의 실제 메시지에는 아무런 영향이 없습니다.

Put은 MQMessage 오브젝트의 MessageId 및 CorrelationId 특성을 업데이트하고 메시지 데이터를 지우지 않습니다. 추가적인 Put 또는 Get 호출은 MQMessage 오브젝트의 업데이트된 정보를 참조합니다. 예를 들면, 다음 코드 스니펫에서 첫 번째 메시지에 a와 두 번째 ab가 들어 있습니다.

```
msg.WriteString("a");  
q.Put(msg, pmo);  
msg.WriteString("b");  
q.Put(msg, pmo);
```

## 메시지

메시지 디스크립터 데이터와 송신할 메시지를 포함한 MQMessage 오브젝트. 이 메소드의 결과로 메시지 디스크립터가 대체될 수 있습니다. 이 메소드가 완료된 직후 메시지 디스크립터의 값이 큐에 넣거나 토픽에 발행한 값입니다.

다음 이유 코드가 다시 연결 가능한 클라이언트에 리턴됩니다.

- 지속 메시지에서 Put 호출을 실행하는 동안 연결이 끊기고 다시 연결에 성공한 경우 MQRC\_CALL\_INTERRUPTED.
- 비지속 메시지에서 Put 호출을 실행하는 동안 연결에 성공한 경우 MQRC\_NONE(애플리케이션 복구 참조).

## putMessageOptions

Put 조치를 제어하는 옵션.

putMessageOptions를 지정하지 않으면 MQPutMessageOptions의 기본 인스턴스가 사용됩니다.

reconnectable 클라이언트에서 MQPMO\_LOGICAL\_ORDER 옵션을 사용하는 경우 MQRC\_RECONNECT\_INCOMPATIBLE 이유 코드가 리턴됩니다.

**참고:** 간편함과 성능을 위해 단일 메시지를 큐에 넣으려는 경우 MQQueueManager.Put 오브젝트를 사용하십시오. 이를 위해서는 MQQueue 오브젝트가 있어야 합니다.

## 구성자

MQDestination은 추상 기본 클래스이고 인스턴스화할 수 없습니다. MQQueue 및 MQTopic 구성자를 사용하거나, MQQueueManager.AccessQueue 및 MQQueueManager.AccessTopic methods를 사용하여 목적지에 액세스하십시오.

## MQEnvironment.NET 클래스

MQEnvironment 를 사용하여 MQQueueManager 생성자가 호출되는 방법을 제어하고 IBM MQ MQI client 연결을 선택하십시오. MQEnvironment 클래스는 IBM MQ의 동작을 제어하는 특성을 포함합니다.

### 클래스

```
System.Object
└── IBM.WMQ.MQEnvironment
```

```
public class IBM.WMQ.MQEnvironment extends System.Object;
```

- [1651 페이지의 『특성 - 클라이언트만 해당』](#)
- [1652 페이지의 『특성』](#)
- [1653 페이지의 『구성자』](#)

### 특성 - 클라이언트만 해당

특성을 가져올 때 전달되는 MQException에 대한 테스트.

```
public static int CertificateValPolicy {get; set;}
```

리모트 파트너 시스템에서 수신된 디지털 인증서의 유효성을 검증하는 데 사용하는 TLS 인증서 유효성 검증 정책을 설정합니다. 올바른 값은 다음과 같습니다.

- MQC.CERTIFICATE\_VALIDATION\_POLICY\_ANY
- MQC.CERTIFICATE\_VALIDATION\_POLICY\_RFC5280

```
public static ArrayList EncryptionPolicySuiteB {get; set;}
```

Suite B 준수 암호화 레벨을 설정합니다. 올바른 값은 다음과 같습니다.

- MQC.MQ\_SUITE\_B\_NONE - 기본값입니다.
- MQC.MQ\_SUITE\_B\_128\_BIT
- MQC.MQ\_SUITE\_B\_192\_BIT

```
public static string Channel {get; set;}
```

대상 큐 관리자에 연결하기 위한 채널의 이름. 클라이언트 모드에서 MQQueueManager 인스턴스를 인스턴스화하기 전에 채널 특성을 설정해야 합니다.

```
public static int FipsRequired {get; set;}
```

암호화가 IBM MQ에서 수행되는 경우 FIPS 인증 알고리즘만 사용하려면 MQC.MQSSL\_FIPS\_YES 를 지정하십시오. 기본값은 MQC.MQSSL\_FIPS\_NO입니다.

암호화 하드웨어가 구성된 경우 사용되는 암호화 모듈은 하드웨어 제품에서 제공하는 모듈입니다. 사용 중인 하드웨어에 따라 특정 레벨로 FIPS 인증되지 않았을 수 있습니다.

```
public static string Hostname {get; set;}
```

IBM MQ 서버가 상주하는 컴퓨터의 TCP/IP 호스트 이름. 호스트 이름을 설정되지 않았고 설정된 대체 특성이 없는 경우, 서버 바인딩 모드를 사용하여 로컬 큐 관리자에 연결합니다.

```
public static int Port {get; set;}
```

연결할 포트. IBM MQ 서버가 수신되는 연결 요청을 대기하는 포트입니다. 기본값은 1414입니다.

```
public static string SSLCipherSpec {get; set;}
```

연결에 TLS를 사용하려면 SVRCONN 채널에 설정된 CipherSpec의 값으로 SSLCipherSpec을 설정합니다. 기본값은 null이고 TLS가 연결에 사용되지 않습니다.



**public static string sslPeerName {get; set;}**

식별 이름 패턴. sslCipherSpec을 설정한 경우, 이 변수를 사용하여 올바른 큐 관리자가 사용되는지 확인할 수 있습니다. 널(기본값)로 설정한 경우, 큐 관리자의 DN이 수행되지 않습니다. sslCipherSpec이 널이면 sslPeerName이 무시됩니다.

**public static string SSLKeyRepository {get; set;}**

키 저장소에 액세스하기 위한 일반 텍스트 또는 암호화된 비밀번호 문구입니다. 키 저장소 비밀번호 문구는 **runmqicred** 유틸리티를 사용하여 클라이언트 애플리케이션에서 사용할 수 있도록 암호화됩니다.

SSLKeyRepositoryPassword가 널(기본값)로 설정되면 **MQKEYRPWD** 환경 변수의 값 또는 클라이언트 구성 파일의 **SSLKeyRepositoryPassword** 속성이 사용됩니다.

**public static string InitialKey {get; set;}**

SSLKeyRepositoryPassword에 지정된 키 저장소 비밀번호 문구를 암호화하는 데 사용된 초기 키입니다.

**runmqicred** 유틸리티를 사용하여 키 저장소 비밀번호 문구를 암호화할 때 초기 키 파일을 지정한 경우 초기 키를 지정해야 합니다.

## 특성

특성을 가져올 때 전달되는 MQException에 대한 테스트.

**public static ArrayList HdrCompList {get; set;}**

헤더 데이터 압축 목록

**public static int KeyResetCount {get; set;}**

비밀 키를 재협상하기 전에 TLS 대화 안에서 송신 및 수신된 암호화되지 않은 바이트 수를 표시합니다.

**public static ArrayList MQAIRArray {get; set;}**

MQAuthenticationInformationRecord 오브젝트의 어레이.

**public static ArrayList MsgCompList {get; set;}**

메시지 데이터 압축 목록

**public static string Password {get; set;}**

인증될 비밀번호. MQCSP 구조에서 참조된 비밀번호는 이 비밀번호 특성을 설정하면 입력됩니다.

**public static string ReceiveExit {get; set;}**

수신 엑시트는 큐 관리자로부터 수신된 데이터를 조사 및 대체할 수 있도록 합니다. 일반적으로 큐 관리자의 해당 송신 엑시트와 함께 사용됩니다. ReceiveExit를 널로 설정하면 수신 엑시트가 호출되지 않습니다.

**public static string ReceiveUserData {get; set;}**

수신 엑시트와 연관된 사용자 데이터. 32자로 제한됩니다.

**public static string SecurityExit {get; set;}**

보안 엑시트는 큐 관리자에 연결하려고 할 때 발생하는 보안 플로우를 사용자 정의할 수 있도록 합니다. SecurityExit를 널로 설정하면 보안 엑시트가 호출되지 않습니다.

**public static string SecurityUserData {get; set;}**

보안 엑시트와 연관된 사용자 데이터. 32자로 제한됩니다.

**public static string SendExit {get; set;}**

송신 엑시트는 큐 관리자로부터 송신된 데이터를 조사 및 대체할 수 있도록 합니다. 일반적으로 큐 관리자의 해당 수신 엑시트와 함께 사용됩니다. SendExit를 널로 설정하면 송신 엑시트가 호출되지 않습니다.

**public static string SendUserData {get; set;}**

송신 엑시트와 연관된 사용자 데이터. 32자로 제한됩니다.

**public static string SharingConversations {get; set;}**

SharingConversations 필드는 이 애플리케이션이 클라이언트 채널 정의 테이블(CCDT)을 사용하지 않을 때 .NET 애플리케이션에서 연결하는 경우 사용됩니다.

SharingConversations는 이 연결과 연관된 소켓에서 공유할 수 있는 최대 대화 수를 판별합니다.

값이 0이면 대화 공유, 미리 읽기, 하트비트와 관련하여 채널이 IBM WebSphere MQ 7.0 이전처럼 작동합니다.



필드는 IBM MQ 큐 관리자를 인스턴스화할 때 특성의 해시 테이블에서 SHARING\_CONVERSATIONS\_PROPERTY로 전달됩니다.

SharingConversations를 지정하지 않으면 기본값 10이 사용됩니다.

**public static string SSLCryptoHardware {get; set;}**

시스템에 있는 암호화 하드웨어를 구성하는 데 필요한 매개변수 문자열의 이름을 설정합니다. sslCipherSpec이 널이면 SSLCryptoHardware가 무시됩니다.

**public static string SSLKeyRepository {get; set;}**

키 저장소의 완전한 파일 이름을 설정합니다.

파일 확장자를 제공하지 않으면 .kdb 로 가정합니다.

SSLKeyRepository를 널(기본값)로 설정한 경우, 키 저장소를 찾는 데 인증서 MQSSLKEYR 환경 변수가 사용됩니다.

**public static string UserId {get; set;}**

인증될 사용자 ID. MQCSP 구조에서 참조된 사용자 ID는 UserId를 설정하면 입력됩니다. API 또는 보안 엑시트를 사용하여 UserId을 인증하십시오.

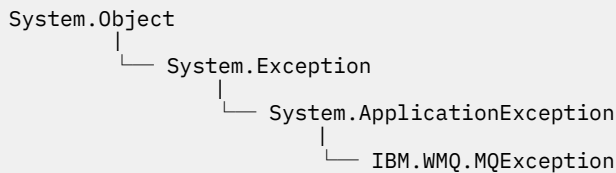
## 구성자

**public MQEnvironment()**

## MQException.NET 클래스

MQException 를 사용하여 실패한 IBM MQ 함수의 완료 및 이유 코드를 찾으십시오. IBM MQ 오류가 발생할 때 마다 MQException 이 발생합니다.

## 클래스



```
public class IBM.WMQ.MQException extends System.ApplicationException;
```

- [1653 페이지의 『특성』](#)
- [1654 페이지의 『구성자』](#)

## 특성

**public int CompletionCode {get; set;}**

오류와 연관된 IBM MQ 완료 코드. 가능한 값은 다음과 같습니다.

- MQException.MQCC\_OK
- MQException.MQCC\_WARNING
- MQException.MQCC\_FAILED

**public int ReasonCode {get; set;}**

오류를 설명하는 IBM MQ 이유 코드.

## 구성자

```
public MQException(int completionCode, int reasonCode)
```

**completionCode**

IBM MQ 완료 코드.

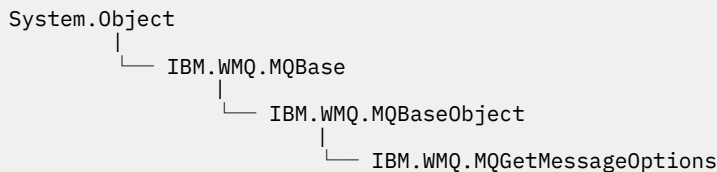
**reasonCode**

IBM MQ 완료 코드.

## MQGetMessageOptions.NET 클래스

메시지의 검색 방법을 지정하려면 MQGetMessageOptions을 사용하십시오. MQDestination.Get의 동작을 수정합니다.

## 클래스



```
public class IBM.WMQ.MQGetMessageOptions extends IBM.WMQ.MQBaseObject;
```

- [1654 페이지의 『특성』](#)
- [1656 페이지의 『구성자』](#)

## 특성

**참고:** 이 클래스에 사용할 수 있는 일부 옵션의 동작은 사용 환경에 따라 다릅니다. 이러한 요소는 별표 \*로 표시됩니다.

특성을 가져올 때 전달되는 MQException에 대한 테스트.

```
public int GroupStatus {get;}*
```

GroupStatus는 검색된 메시지가 그룹에 있는지 그리고 그룹에서 마지막 항목인지를 표시합니다. 가능한 값은 다음과 같습니다.

**MQC.MQGS\_LAST\_MSG\_IN\_GROUP**

메시지가 그룹에 있는 마지막 또는 유일한 메시지입니다.

**MQC.MQGS\_MSG\_IN\_GROUP**

메시지가 그룹에 있지만 그룹의 마지막이 아닙니다.

**MQC.MQGS\_NOT\_IN\_GROUP**

메시지는 그룹에 없습니다.

```
public int MatchOptions {get; set;}*
```

MatchOptions는 메시지의 선택 방법을 판별합니다. 설정할 수 있는 일치 옵션은 다음과 같습니다.

**MQC.MQMO\_MATCH\_CORREL\_ID**

일치하는 상관 ID.

**MQC.MQMO\_MATCH\_GROUP\_ID**

일치하는 그룹 ID.

**MQC.MQMO\_MATCH\_MSG\_ID**

일치하는 메시지 ID.

**MQC.MQMO\_MATCH\_MSG\_SEQ\_NUMBER**

메시지 순서 번호에 일치합니다.

**MQC.MQMO\_NONE**

일치할 필요가 없습니다.

**public int Options {get; set;}**

Options는 MQQueue.get 조치를 제어합니다. 다음 값을 지정할 수 있습니다. 둘 이상의 옵션이 필요하다면 값을 추가하거나 비트 단위의 OR 연산자를 사용하여 결합할 수 있습니다.

**MQC.MQGMO\_ACCEPT\_TRUNCATED\_MSG**

메시지 데이터의 잘림을 허용하십시오.

**MQC.MQGMO\_ALL\_MSGS\_AVAILABLE\***

그룹의 모든 메시지가 사용 가능한 경우에만 그룹에서 메시지를 검색합니다.

**MQC.MQGMO\_ALL\_SEGMENTS\_AVAILABLE\***

그룹의 모든 세그먼트가 사용 가능한 경우에만 논리 메시지 세그먼트를 검색합니다.

**MQC.MQGMO\_BROWSE\_FIRST**

큐의 시작로부터 찾아보기.

**MQC.MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR\***

찾아보기 커서 아래의 메시지를 찾으십시오.

**MQC.MQGMO\_BROWSE\_NEXT**

큐의 현재 위치에서 찾아봅니다.

**MQC.MQGMO\_COMPLETE\_MSG\***

완료된 논리 메시지만 검색합니다.

**MQC.MQGMO\_CONVERT**

데이터를 메시지 버퍼에 복사하기 전에 MQMessage의 CharSet 및 Encoding 속성에 맞도록 애플리케이션 데이터의 변환을 요청합니다. 메시지 버퍼에서 데이터를 검색할 때에도 데이터 변환이 적용되기 때문에 애플리케이션은 이 옵션을 설정하지 않습니다.

이 옵션을 사용하면 1바이트 문자 세트를 2바이트 문자 세트로 변환할 때 문제가 발생할 수 있습니다. 대신, 메시지를 전달한 후 readString, readLine 및 writeString 메소드를 사용하여 변환을 수행하십시오.

**MQC.MQGMO\_FAIL\_IF QUIESCING**

큐 관리자가 정지 중인 경우 실패합니다.

**MQC.MQGMO\_LOCK\***

찾아보는 메시지를 잠급니다.

**MQC.MQGMO\_LOGICAL\_ORDER\***

메시지를 그룹으로 또는 논리 메시지의 세그먼트를 논리 순서대로 리턴합니다.

다시 연결 가능한 클라이언트에서 MQGMO\_LOGICAL\_ORDER 옵션을 사용하는 경우 MQRC\_RECONNECT\_INCOMPATIBLE 이유 코드가 애플리케이션에 리턴됩니다.

**MQC.MQGMO\_MARK\_SKIP\_BACKOUT\***

큐에서 메시지를 다시 인스턴스화하지 않고 작업 단위를 백아웃할 수 있습니다.

**MQC.MQGMO\_MSG\_UNDER\_CURSOR**

찾아보기 커서 아래의 메시지를 가져오십시오.

**MQC.MQGMO\_NONE**

다른 옵션이 지정되지 않았습니다. 모든 옵션이 기본값을 사용합니다.

**MQC.MQGMO\_NO\_PROPERTIES**

메시지 디스크립터(또는 확장자)에 포함된 특성을 제외하고, 메시지의 특성이 검색되지 않습니다.

**MQC.MQGMO\_NO\_SYNCPOINT**

동기점 제어가 없는 메시지를 가져오십시오.

**MQC.MQGMO\_NO\_WAIT**

적절한 메시지가 없으면 바로 리턴합니다.

**MQC.MQGMO\_PROPERTIES\_AS\_Q\_DEF**

MQQueue의 PropertyControl 속성에 정의된 대로 메시지 특성을 검색합니다. 메시지 디스크립터 또는 확장자의 메시지 특성에 액세스하는 것은 PropertyControl 속성의 영향을 받지 않습니다.

**MQC.MQGMO\_PROPERTIES\_COMPATIBILITY**

MQRFH2 헤더에서 접두부가 `mcd`, `jms`, `usr` 또는 `mqext`인 메시지 특성을 검색합니다. 메시지 디스크립터 또는 확장자에 포함된 특성을 제외한 메시지의 다른 특성은 제거됩니다.

**MQC.MQGMO\_PROPERTIES\_FORCE\_MQRFH2**

MQRFH2 헤더에서 메시지 디스크립터 또는 확장자에 포함된 특성을 제외한 메시지 특성을 검색합니다. 특성을 검색해야 하지만 메시지 핸들을 사용하도록 변경할 수 없는 애플리케이션에서 MQC.MQGMO\_PROPERTIES\_FORCE\_MQRFH2를 사용하십시오.

**MQC.MQGMO\_PROPERTIES\_IN\_HANDLE**

MsgHandle을 사용하여 메시지 특성을 검색합니다.

**MQC.MQGMO\_SYNCPOINT**

동기점 제어 아래에서 메시지를 가져옵니다. 이 메시지는 다른 애플리케이션에 사용 불가능한 것으로 표시되지만, 작업 단위가 커밋되는 경우에만 큐에서 삭제됩니다. 작업 단위가 백아웃되는 경우 메시지가 다시 사용 가능하게 됩니다.

**MQC.MQGMO\_SYNCPOINT\_IF\_PERSISTENT\***

메시지가 지속적인 경우 동기점 제어를 가진 메시지를 가져오십시오.

**MQC.MQGMO\_UNLOCK\***

이전에 잠근 메시지를 잠금 해제합니다.

**MQC.MQGMO\_WAIT**

메시지가 도착할 때까지 대기합니다.

**public string ResolvedQueueName {get;}**

큐 관리자는 ResolvedQueueName을 메시지가 검색된 큐의 로컬 이름으로 설정합니다. 알리어스 큐 또는 모델 큐가 열린 경우 ResolvedQueueName은 큐를 여는 데 사용하는 이름과 다릅니다.

**public char Segmentation {get;}\***

Segmentation은 검색된 메시지의 세그먼트화 허용 여부를 표시합니다. 가능한 값은 다음과 같습니다.

**MQC.MQSEG\_INHIBITED**

세그먼트화를 허용하지 않습니다.

**MQC.MQSEG\_ALLOWED**

세그먼트화를 허용합니다.

**public byte SegmentStatus {get;}\***

SegmentStatus는 검색된 메시지가 논리 메시지의 세그먼트인지 여부를 표시하는 출력 필드입니다. 메시지가 세그먼트이면 플래그는 마지막 세그먼트인지 여부를 표시합니다. 가능한 값은 다음과 같습니다.

**MQC.MQSS\_LAST\_SEGMENT**

메시지가 논리 메시지의 마지막 또는 유일한 세그먼트입니다.

**MQC.MQSS\_NOT\_A\_SEGMENT**

메시지가 세그먼트가 아닙니다.

**MQC.MQSS\_SEGMENT**

메시지가 세그먼트이지만 논리 메시지의 마지막 세그먼트가 아닙니다.

**public int WaitInterval {get; set;}**

WaitInterval은 적절한 메시지가 도착할 때까지 MQQueue.get 호출이 대기하는 최대 시간(밀리초)입니다. WaitInterval을 MQC.MQGMO\_WAIT과 함께 사용하십시오. 메시지를 무제한 대기하려면 MQC.MQWI\_UNLIMITED 값을 설정하십시오.

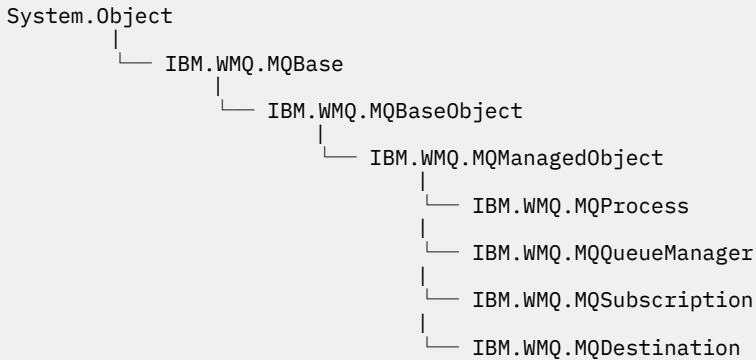
**구성자****public MQGetMessageOptions()**

Options을 MQC.MQGMO\_NO\_WAIT으로, WaitInterval을 0으로, ResolvedQueueName을 공백으로 설정한 상태에서 새 MQGetMessageOptions 오브젝트를 구성합니다.

## MQManagedObject.NET 클래스

MQDestination, MQProcess, MQQueueManager 및 MQSubscription의 속성을 조회하고 설정하려면 MQManagedObject를 사용하십시오. MQManagedObject는 이러한 클래스의 수퍼클래스입니다.

### 클래스



```
public class IBM.WMQ.MQManagedObject extends IBM.WMQ.MQBaseObject;
```

- [1657 페이지의 『특성』](#)
- [1658 페이지의 『메소드』](#)
- [1659 페이지의 『구성자』](#)

### 특성

특성을 가져올 때 전달되는 MQException에 대한 테스트.

```
public string AlternateUserId {get; set;}
```

자원을 열 때 설정한 대체 사용자 ID(있는 경우). 열린 오브젝트에 대해 발행된 경우에는 AlternateUserID.set가 무시됩니다. AlternateUserId가 구독에는 유효하지 않습니다.

```
public int CloseOptions {get; set;}
```

자원을 닫는 방법을 제어하려면 이 속성을 설정합니다. 기본값은 MQC.MQCO\_NONE입니다. MQC.MQCO\_NONE은 영구적 동적 큐, 임시 동적 큐, 구독 및 이를 작성한 오브젝트에서 액세스하는 토픽을 제외한 모든 자원에 허용되는 유일한 값입니다.

큐와 토픽에 대해서는 다음과 같은 추가 값을 사용할 수 있습니다.

#### MQC.MQCO\_DELETE

메시지가 없는 경우 큐를 삭제합니다.

#### MQC.MQCO\_DELETE\_PURGE

큐의 메시지를 영구 제거하여 큐를 삭제하십시오.

#### MQC.MQCO\_QUIESCE

큐 닫기를 요청하고, 메시지가 남아 있으면 경고를 수신합니다(최종 닫기 전에 검색될 수 있도록 허용함).

구독에 대해서는 다음과 같은 추가 값을 사용할 수 있습니다.

#### MQC.MQCO\_KEEP\_SUB

구독은 삭제되지 않습니다. 최초 구독이 지속되는 경우에만 이 옵션이 유효합니다. MQC.MQCO\_KEEP\_SUB는 지속적 토픽의 기본값입니다.

#### MQC.MQCO\_REMOVE\_SUB

구독이 삭제됩니다. MQC.MQCO\_REMOVE\_SUB는 비지속적 관리되지 않는 토픽의 기본값입니다.

#### MQC.MQCO\_PURGE\_SUB

구독이 삭제됩니다. MQC.MQCO\_PURGE\_SUB는 비지속적 관리되는 주제의 기본값입니다.

**public MQQueueManager ConnectionReference {get;}**

이 자원이 속한 큐 관리자.

**public string MQDescription {get;}**

큐 관리자에 보유된 자원에 대한 설명. MQDescription은 구독 및 토픽에 대한 빈 문자열을 리턴합니다.

**public boolean IsOpen {get;}**

자원이 현재 열려 있는지 여부를 표시합니다.

**public string Name {get;}**

자원의 이름. 이 이름은 액세스 메소드에 제공되거나, 동적 큐를 위해 큐 관리자에서 할당합니다.

**public int OpenOptions {get; set;}**

IBM MQ 오브젝트가 열려 있으면 OpenOptions가 설정됩니다. OpenOptions.set 메소드는 무시되고, 오류를 일으키지 않습니다. 구독에는 OpenOptions가 없습니다.

## 메소드

**public virtual void Close();**

MQException을 전달합니다.

오브젝트를 닫습니다. Close를 호출한 후 이 자원에 대해 어떤 추가적인 작업도 허용되지 않습니다. Close 메소드의 동작을 변경하려면 closeOptions 속성을 설정하십시오.

**public string GetAttributeString(int selector, int length);**

MQException을 전달합니다.

속성 문자열을 가져옵니다.

**선택기**

조회하는 속성을 표시하는 정수.

**length**

필요한 문자열의 길이를 표시하는 정수.

**public void Inquire(int[] selectors, int[] intAttrs, byte[] charAttrs);**

MQException을 전달합니다.

큐, 프로세스 또는 큐 관리자의 속성을 포함하는 문자열 세트와 정수 배열을 리턴합니다. 조회할 속성은 선택자 배열에 지정됩니다.

**참고:** MQManagedObject, MQQueue 및 MQQueueManager에 정의된 Get 메소드를 사용하여 더 많은 공통 속성을 조회할 수 있습니다.

**선택자**

조회할 값을 포함한 속성을 식별하는 정수 배열.

**intAttrs**

정수 속성 값이 리턴되는 배열. 정수 속성 값은 선택자 배열에 있는 정수 속성 선택자와 같은 순서대로 리턴됩니다.

**charAttrs**

문자 속성이 리턴되고 연결되는 버퍼. 문자 속성은 선택자 배열에 있는 문자 속성 선택자와 같은 순서대로 리턴됩니다. 각 속성 문자열의 길이는 속성별로 수정됩니다.

**public void Set(int[] selectors, int[] intAttrs, byte[] charAttrs);**

MQException을 전달합니다.

선택자의 벡터에 정의된 속성을 설정합니다. 설정할 속성이 선택자 배열에 지정됩니다.

**선택자**

설정할 값을 포함한 속성을 식별하는 정수 배열.

**intAttrs**

설정할 정수 속성 값의 배열. 이 값은 선택자 배열에 있는 정수 속성 선택자와 같은 순서여야 합니다.

**charAttrs**

설정할 문자 속성이 연결되는 버퍼. 이 값은 선택자 배열에 있는 문자 속성 선택자와 같은 순서여야 합니다. 각 문자 속성의 길이는 수정됩니다.

```
public void SetAttributeString(int selector, string value, int length);
```

MQException을 전달합니다.

속성 문자열을 설정합니다.

**선택기**

설정하는 속성을 표시하는 정수.

**값**

속성 값으로 설정할 문자열.

**length**

필요한 문자열의 길이를 표시하는 정수.

## 구성자

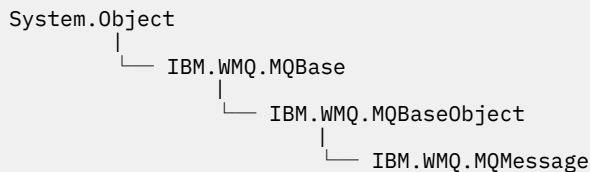
```
protected MQManagedObject()
```

구성자 메소드. 이 오브젝트는 자체 인스턴스화할 수 없는 추상 기본 클래스입니다.

## MQMessage.NET 클래스

MQMessage 를 사용하여 IBM MQ 메시지에 대한 데이터 및 메시지 디스크립터에 액세스하십시오. MQMessage 는 IBM MQ 메시지를 캡슐화합니다.

## 클래스



```
public class IBM.WMQ.MQMessage extends IBM.WMQ.MQBaseObject;
```

MQMessage 오브젝트를 작성한 다음 Read 및 Write 메소드를 사용하여 메시지와 애플리케이션의 다른 오브젝트 간에 데이터를 전송하십시오. MQDestination, MQQueue 및 MQTopic 클래스의 Put 및 Get 메소드를 사용하여 MQMessage 오브젝트를 송신 및 수신하십시오.

MQMessage의 특성을 사용하여 메시지 디스크립터의 특성을 가져오고 설정하십시오. SetProperty 및 GetProperty 메소드를 사용하여 확장된 메시지 특성을 설정하고 가져오십시오.

- [1659 페이지의 『특성』](#)
- [1665 페이지의 『Read 및 Write 메시지 메소드』](#)
- [1668 페이지의 『버퍼 메소드』](#)
- [1668 페이지의 『특성 메소드』](#)
- [1671 페이지의 『구성자』](#)

## 특성

특성을 가져올 때 전달되는 MQException에 대한 테스트.

```
public string AccountingToken {get; set;}
```

메시지 ID 컨텍스트의 부분으로, 애플리케이션이 메시지의 결과로 완료된 작업에 대해 청구할 수 있도록 합니다. 기본값은 MQC.MQACT\_NONE입니다.

```
public string ApplicationIdData {get; set;}
```

메시지 ID 컨텍스트의 부분입니다. ApplicationIdData는 애플리케이션 스위트에 의해 정의된 정보이며, 메시지 또는 메시지 진원지에 대한 추가 정보를 제공하는 데 사용될 수 있습니다. 기본값은 ""입니다.

**public string ApplicationOriginData {get; set;}**

메시지 원본에 대한 추가 정보를 제공하는 데 사용될 수 있는 애플리케이션에 의해 정의된 정보입니다. 기본값은 ""입니다.

**public int BackoutCount {get;}**

메시지가 이전에 작업 단위의 일부로 MQQueue.Get 호출에 의해 리턴되고 백아웃된 횟수입니다. 기본값은 0입니다.

**public int CharacterSet {get; set;}**

메시지에 있는 문자 데이터에 대한 코드화 문자 세트 ID.

메시지에 있는 문자 데이터의 문자 세트를 식별하려면 CharacterSet을 설정하십시오. 메시지에 있는 문자 데이터를 인코딩하는 데 사용된 문자 세트가 무엇인지 확인하려면 CharacterSet을 가져오십시오.

.NET 애플리케이션은 항상 유니코드에서 실행되는 반면, 다른 환경에서는 애플리케이션이 큐 관리자가 실행되는 동일한 문자 세트에서 실행됩니다.

ReadString 및 ReadLine 메소드는 사용자를 위해 메시지의 문자 데이터를 유니코드로 변환합니다.

WriteString 메소드는 유니코드에서 CharacterSet에서 인코딩된 문자 세트로 변환합니다.

CharacterSet가 0인 기본값 MQC.MQCCSI\_Q\_MGR로 설정된 경우, 변환이 발생하지 않고 CharacterSet가 1200으로 설정됩니다. CharacterSet를 몇 가지 다른 값으로 설정하면 WriteString이 유니코드에서 대체 값으로 변환합니다.

**참고:** 다른 읽기 및 쓰기 메소드는 CharacterSet를 사용하지 않습니다.

- ReadChar 및 WriteChar는 변환 없이 메시지 버퍼(부터) 유니코드 문자를 읽고 씁니다.
- ReadUTF 및 WriteUTF는 애플리케이션의 유니코드 문자열 및 2바이트 길이 필드가 접두부로 붙은 메시지 버퍼의 UTF-8 문자열 간에 변환합니다.
- 바이트 메소드는 대체 없이 애플리케이션과 메시지 버퍼 간에 바이트를 전송합니다.

**public byte[] CorrelationId {get; set;}**

• MQQueue.Get 호출의 경우, 검색할 메시지의 상관 ID입니다. 큐 관리자는 메시지 디스크립터 필드와 일치하는 메시지 ID 및 상관 ID를 포함한 첫 번째 메시지를 리턴합니다. 기본값 MQC.MQCI\_NONE을 사용하면 모든 상관 ID가 일치하도록 할 수 있습니다.

• MQQueue.Put 호출의 경우, 설정할 상관 ID입니다.

**public int DataLength {get;}**

읽기 대상으로 남아 있는 메시지 데이터의 바이트 수.

**public int DataOffset {get; set;}**

메시지 데이터 안에서 현재 커서 위치. 읽고 및 쓰기는 현재 위치에서 적용됩니다.

**public int Encoding {get; set;}**

애플리케이션 메시지 데이터에서 숫자 값에 사용되는 표현. Encoding은 2진, 팩형 10진수 및 부동 소수점 데이터에 적용됩니다. 이러한 숫자 형식의 읽기 및 쓰기 메소드의 동작이 그에 따라 대체됩니다. 각 세 개 섹션의 값을 하나씩 추가하여 인코딩 필드의 값을 구성하십시오. 또는, 비트 단위의 OR 연산자를 사용하여 각 세 개 섹션의 값을 결합하는 값을 구성하십시오.

## 1. 2진 정수

**MQC.MQENC\_INTEGER\_NORMAL**

Big Endian 정수.

**MQC.MQENC\_INTEGER\_REVERSED**

Intel 아키텍처에 사용되는 Little Endian 정수.

## 2. 팩형 10진수

**MQC.MQENC\_DECIMAL\_NORMAL**

z/OS에 사용되는 Big Endian 팩형 10진수.

**MQC.MQENC\_DECIMAL\_REVERSED**

Little Endian 팩형 10진수.



### 3. 부동 소수점

#### **MQC.MQENC\_FLOAT\_IEEE\_NORMAL**

Big Endian IEEE Float.

#### **MQC.MQENC\_FLOAT\_IEEE\_REVERSED**

Intel 아키텍처에 사용되는 Little Endian IEEE Float.

#### **MQC.MQENC\_FLOAT\_S390**

z/OS 형식 부동 소수점.

기본값은 다음과 같습니다.

```
MQC.MQENC_INTEGER_REVERSED |
MQC.MQENC_DECIMAL_REVERSED |
MQC.MQENC_FLOAT_IEEE_REVERSED
```

기본 설정에서는 WriteInt가 Little Endian 정수를 쓰고 ReadInt가 Little Endian 정수를 읽습니다.

MQC.MQENC\_INTEGER\_NORMAL 플래그를 대신 설정하면 WriteInt가 Big Endian 정수를 쓰고 ReadInt가 Big Endian 정수를 읽습니다.

**참고:** IEEE 형식 부동 소수점에서 zSeries 형식 부동 소수점으로 변환하면 정밀도가 저하될 수 있습니다.

#### **public int Expiry {get; set;}**

메시지를 넣는 애플리케이션에 의해 설정되는 10분의 1초로 표시된 만기 시간. 메시지의 만기 시간이 경과된 후에는 큐 관리자가 메시지를 제거할 수 있습니다. 메시지가 MQC.MQRO\_EXPIRATION 플래그 중 하나를 지정한 경우, 메시지를 제거하면 보고서가 생성됩니다. 기본값은 메시지가 만기되지 않음을 의미하는 MQC.MQEI\_UNLIMITED입니다.

#### **public int Feedback {get; set;}**

보고서의 네이처를 표시하려면 MQC.MQMT\_REPORT 유형의 메시지와 함께 Feedback을 사용하십시오. 시스템에서 정의한 피드백 코드는 다음과 같습니다.

- MQC.MQFB\_EXPIRATION
- MQC.MQFB\_COA
- MQC.MQFB\_COD
- MQC.MQFB\_QUIT
- MQC.MQFB\_PAN
- MQC.MQFB\_NAN
- MQC.MQFB\_DATA\_LENGTH\_ZERO
- MQC.MQFB\_DATA\_LENGTH\_NEGATIVE
- MQC.MQFB\_DATA\_LENGTH\_TOO\_BIG
- MQC.MQFB\_BUFFER\_OVERFLOW
- MQC.MQFB\_LENGTH\_OFF\_BY\_ONE
- MQC.MQFB\_IIH\_ERROR

MQC.MQFB\_APPL\_FIRST ~ MQC.MQFB\_APPL\_LAST 범위의 애플리케이션 정의의 피드백 값을 사용할 수도 있습니다. 이 필드의 기본값은 피드백을 제공하지 않음을 의미하는 MQC.MQFB\_NONE입니다.

#### **public string Format {get; set;}**

메시지 송신자가 수신자에게 메시지 내에 있는 데이터의 네이처를 표시하기 위해 사용하는 형식 이름. 고유 형식 이름을 사용할 수 있으며, MQ 문자로 시작하는 이름은 큐 관리자에서 정의한 것이라는 의미입니다. 큐 관리자 내장 형식은 다음과 같습니다.

#### **MQC.MQFMT\_ADMIN**

명령 서버 요청/응답 메시지입니다.

#### **MQC.MQFMT\_COMMAND\_1**

유형 1 명령 응답 메시지입니다.

**MQC.MQFMT\_COMMAND\_2**

유형 2 명령 응답 메시지입니다.

**MQC.MQFMT\_DEAD\_LETTER\_HEADER**

데드-레터 헤더입니다.

**MQC.MQFMT\_EVENT**

이벤트 메시지입니다.

**MQC.MQFMT\_NONE**

형식 이름이 없습니다.

**MQC.MQFMT\_PCF**

프로그래밍 가능한 명령 형식의 사용자 정의 메시지.

**MQC.MQFMT\_STRING**

완전히 문자로 구성된 메시지입니다.

**MQC.MQFMT\_TRIGGER**

트리거 메시지

**MQC.MQFMT\_XMIT\_Q\_HEADER**

전송 큐 헤더입니다.

기본값은 MQC.MQFMT\_NONE입니다.

**public byte[] GroupId {get; set;}**

물리적 메시지가 속한 메시지 그룹을 식별하는 바이트 문자열. 기본값은 MQC.MQGI\_NONE입니다.

**public int MessageFlags {get; set;}**

메시지의 세그먼트화 및 상태를 제어하는 플래그.

**public byte[] MessageId {get; set;}**

MQQueue.Get 호출의 경우, 이 필드는 검색할 메시지의 메시지 ID를 지정합니다. 일반적으로, 큐 관리자는 메시지 디스크립터 필드와 일치하는 메시지 ID 및 상관 ID를 포함한 첫 번째 메시지를 리턴합니다. 특수 값 MQC.MQMI\_NONE을 사용하여 모든 메시지 ID가 일치하도록 하십시오.

MQQueue.Put 호출의 경우, 이 필드는 사용할 메시지 ID를 지정합니다. MQC.MQMI\_NONE을 지정한 경우, 메시지를 넣으면 큐 관리자가 고유한 메시지 ID를 생성합니다. 넣기 동작 후 사용된 메시지 ID를 표시하기 위해 이 멤버 변수의 값이 업데이트됩니다. 기본값은 MQC.MQMI\_NONE입니다.

**public int MessageLength {get;}**

MQMessage 오브젝트에 있는 메시지 데이터의 바이트 수.

**public int MessageSequenceNumber {get; set;}**

그룹 내의 논리 메시지의 순서 번호.

**public int MessageType {get; set;}**

메시지의 유형을 표시합니다. 현재 시스템에서 정의한 값은 다음과 같습니다.

- MQC.MQMT\_DATAGRAM
- MQC.MQMT\_REPLY
- MQC.MQMT\_REPORT
- MQC.MQMT\_REQUEST

MQC.MQMT\_APPL\_FIRST ~ MQC.MQMT\_APPL\_LAST 범위에서는 애플리케이션 정의 값을 사용할 수도 있습니다. 이 필드의 기본값은 MQC.MQMT\_DATAGRAM입니다.

**public int Offset {get; set;}**

세그먼트화된 메시지에서 논리 메시지 시작 시 물리적 메시지 데이터의 오프셋.

**public int OriginalLength {get; set;}**

세그먼트화된 메시지의 원본 길이.

**public int Persistence {get; set;}**

메시지 지속성. 다음 값이 정의됩니다.

- MQC.MQPER\_NOT\_PERSISTENT

이 옵션을 다시 연결 가능한 클라이언트에 설정하는 경우, 연결에 성공하면 MQRC\_NONE 이유 코드가 애플리케이션에 리턴됩니다.

- MQC.MQPER\_PERSISTENT

이 옵션을 다시 연결 가능한 클라이언트에 설정하는 경우, 연결에 성공하면 MQRC\_CALL\_INTERRUPTED 이유 코드가 애플리케이션에 리턴됩니다.

- MQC.MQPER\_PERSISTENCE\_AS\_Q\_DEF

기본값은 MQC.MQPER\_PERSISTENCE\_AS\_Q\_DEF이며, 목적지 큐의 기본 지속성 속성에서 메시지의 지속성을 가져옵니다.

**public int Priority {get; set;}**

메시지 우선순위. 아웃바운드 메시지에는 특수 값 MQC.MQPRI\_PRIORITY\_AS\_Q\_DEF를 설정할 수도 있습니다. 그런 다음 메시지의 우선순위는 목적지 큐의 기본 우선순위 속성에서 가져옵니다. 기본값은 MQC.MQPRI\_PRIORITY\_AS\_Q\_DEF입니다.

**public int PropertyValidation {get; set;}**

메시지 특성을 설정할 때 특성의 유효성을 검증하는지 여부를 지정합니다. 가능한 값은 다음과 같습니다.

- MQCMHO\_DEFAULT\_VALIDATION
- MQCMHO\_VALIDATE
- MQCMHO\_NO\_VALIDATION

기본값은 MQCMHO\_DEFAULT\_VALIDATION입니다.

**public string PutApplicationName {get; set;}**

메시지를 넣는 애플리케이션 이름. 기본값은 ""입니다.

**public int PutApplicationType {get; set;}**

메시지를 넣는 애플리케이션의 유형. PutApplicationType은 시스템에서 정의하거나 사용자가 정의하는 값입니다. 시스템에서 정의한 값은 다음과 같습니다.

- MQC.MQAT\_AIX
- MQC.MQAT\_CICS
- MQC.MQAT\_DOS
- MQC.MQAT\_IMS
- MQC.MQAT\_MVS
- MQC.MQAT\_OS2
- MQC.MQAT\_OS400
- MQC.MQAT\_QMGR
- MQC.MQAT\_UNIX
- MQC.MQAT\_WINDOWS
- MQC.MQAT\_JAVA

기본값은 메시지에 컨텍스트 정보가 없음을 의미하는 MQC.MQAT\_NO\_CONTEXT입니다.

**public DateTime PutDateTime {get; set;}**

메시지를 넣은 시간 및 날짜.

**public string ReplyToQueueManagerName {get; set;}**

응답 또는 보고 메시지를 송신하는 큐 관리자의 이름. 기본값은 ""이고, 큐 관리자는 ReplyToQueueManagerName을 제공합니다.

**public string ReplyToQueueName {get; set;}**

메시지에 대한 Get 요청을 발행한 애플리케이션이 MQC.MQMT\_REPLY 및 MQC.MQMT\_REPORT 메시지를 송신하는 메시지 큐의 이름. 기본 ReplyToQueueName은 ""입니다.

**public int Report {get; set;}**

보고 및 응답 메시지에 대한 옵션을 지정하려면 Report를 사용하십시오.

- 보고가 필요한지 여부.
- 애플리케이션 메시지 데이터가 보고에 포함되는지 여부.
- 보고 또는 응답에서 메시지 및 상관 ID를 설정하는 방법.

4가지 보고 유형을 어떤 조합으로도 요청할 수 있습니다.

- 4가지 보고 유형의 조합을 지정하십시오. 애플리케이션 메시지 데이터가 보고 메시지에 포함되는지 여부에 따라 각 보고 유형에 대해 3가지 옵션 중 아무거나 선택합니다.

1. 도착 시 확인

- MQC.MQRO\_COA
- MQC.MQRO\_COA\_WITH\_DATA
- MQC.MQRO\_COA\_WITH\_FULL\_DATA \*\*

2. COD(Confirm on delivery)

- MQC.MQRO\_COD
- MQC.MQRO\_COD\_WITH\_DATA
- MQC.MQRO\_COD\_WITH\_FULL\_DATA \*\*

3. 예외

- MQC.MQRO\_EXCEPTION
- MQC.MQRO\_EXCEPTION\_WITH\_DATA
- MQC.MQRO\_EXCEPTION\_WITH\_FULL\_DATA \*\*

4. 만기

- MQC.MQRO\_EXPIRATION
- MQC.MQRO\_EXPIRATION\_WITH\_DATA
- MQC.MQRO\_EXPIRATION\_WITH\_FULL\_DATA \*\*

**참고:** 목록에서 \*\*로 표시된 값은 z/OS 큐 관리자에서 지원되지 않습니다. 애플리케이션이 실행 중인 플랫폼에 상관없이 애플리케이션이 z/OS 큐 관리자에 액세스하는 경우 이 항목을 사용하지 마십시오.

- 보고 또는 응답 메시지의 메시지 ID가 생성되는 방법을 제어하려면 다음 중 하나를 지정하십시오.
  - MQC.MQRO\_NEW\_MSG\_ID
  - MQC.MQRO\_PASS\_MSG\_ID
- 보고 또는 응답 메시지의 상관 ID가 설정되는 방법을 제어하려면 다음 중 하나를 지정하십시오.
  - MQC.MQRO\_COPY\_MSG\_ID\_TO\_CORREL\_ID
  - MQC.MQRO\_PASS\_CORREL\_ID
- 원래 메시지를 목적지 큐에 전달할 수 없을 때 처리 방법을 제어하려면 다음 중 하나를 지정하십시오.
  - MQC.MQRO\_DEAD\_LETTER\_Q
  - MQC.MQRO\_DISCARD\_MSG \*\*
- 보고 옵션이 지정되지 않은 경우, 기본값은 다음과 같습니다.

```
MQC.MQRO_NEW_MSG_ID |
MQC.MQRO_COPY_MSG_ID_TO_CORREL_ID |
MQC.MQRO_DEAD_LETTER_Q
```

- 수신 애플리케이션이 긍정적인 조치 또는 부정적인 조치 보고 메시지를 송신하도록 요청하려면 다음 중 하나 또는 둘 다를 지정합니다.
  - MQC.MQRO\_PAN
  - MQC.MQRO\_NAN

**public int TotalMessageLength {get;}**

수신된 이 메시지가 있었던 메시지 큐에 저장된 메시지의 총 바이트 수.

**public string UserId {get; set;}**

UserId는 메시지 ID 컨텍스트의 일부입니다. 큐 관리자에서 일반적으로 값을 제공합니다. ID 컨텍스트를 설정하는 권한이 있으면 값을 대체할 수 있습니다.

**public int Version {get; set;}**

사용 중인 MQMD 구조의 버전.

## Read 및 Write 메시지 메소드

Read 및 Write 메소드는 .NET System.IO 네임스페이스에 있는 BinaryReader 및 BinaryWriter 클래스의 멤버와 동일한 기능을 수행합니다. 전체 언어 구문과 사용법 예는 MSDN을 참조하십시오. 메소드는 메시지 버퍼의 현재 위치에서 읽어오거나 씁니다. 읽거나 쓴 바이트 수만큼 현재 위치를 앞으로 이동합니다.

**참고:** 메시지 데이터에 MQRFH 또는 MQRFH2 헤더가 포함되어 있으면 ReadBytes 메소드를 사용하여 데이터를 읽어야 합니다.

- 모든 메소드는 IOException을 전달합니다.
- ReadFully 메소드는 메시지에 정확히 맞도록 대상 byte 또는 sbyte 어레이의 크기를 자동으로 조정합니다. 널 어레이의 크기도 조정됩니다.
- Read 메소드는 EndOfStreamException을 전달합니다.
- WriteDecimal 메소드는 MQException을 전달합니다.
- ReadString, ReadLine 및 WriteString 메소드는 유니코드와 메시지 문자 세트 간에 변환합니다. [CharacterSet](#)를 참조하십시오.
- Decimal 메소드는 Encoding의 값에 따라 Big Endian MQC.MQENC\_DECIMAL\_NORMAL 또는 Little Endian MQC.MQENC\_DECIMAL\_REVERSE 형식으로 인코딩된 팩형 10진수를 읽고 씁니다. 10진수 범위 및 해당 .NET 유형은 다음과 같습니다.

### Decimal2/short

-999에서 999까지

### Decimal4/int

-9999999에서 9999999까지

### Decimal8/long

-9999999999999999에서 9999999999999999까지

- Double 및 Float 메소드는 Encoding의 값에 따라 IEEE Big Endian 및 Little Endian 형식, MQC.MQENC\_FLOAT\_IEEE\_NORMAL 및 MQC.MQENC\_FLOAT\_IEEE\_REVERSED 또는 S/390 형식, MQC.MQENC\_FLOAT\_S390으로 인코딩된 부동 값을 읽고 씁니다.
- Int 메소드는 Encoding의 값에 따라 Big Endian, MQC.MQENC\_INTEGER\_NORMAL 또는 Little Endian, MQC.MQENC\_INTEGER\_REVERSED 형식으로 인코딩된 정수 값을 읽고 씁니다. 부호 없는 2바이트 정수 유형을 더하는 경우를 제외하고, 모든 정수에는 부호가 있습니다. 정수 크기, .NET 및 IBM MQ 유형은 다음과 같습니다.

### 2바이트

short, Int2, ushort, UInt2

### 4바이트

int, Int4

### 8바이트

long, Int8

- WriteObject는 오브젝트의 클래스, 임시가 아니고 정적이 아닌 필드의 값, 수퍼유형의 필드를 메시지 버퍼로 전송합니다.
- ReadObject는 오브젝트의 클래스, 클래스의 서명, 임시가 아니고 정적이 아닌 필드의 값, 수퍼유형의 필드에서 오브젝트를 작성합니다.

표 843. 읽기 및 쓰기 메시지 메소드

대상 유형	메소드 서명
<b>Boolean</b>	<pre>public bool ReadBoolean();  public void WriteBoolean(bool value);</pre>
<b>Byte</b>	<pre>public byte ReadByte() public byte ReadUnsignedByte()  public void Write(int value) public void WriteByte(int value) public void WriteByte(byte value) public void WriteByte(sbyte value)</pre>
<b>Bytes</b>	<pre>public byte[] ReadBytes(int count) public void ReadFully(ref byte[] value) public void ReadFully(ref sbyte[] value) public void ReadFully(ref byte[] value, int offset, int length) public void ReadFully(ref sbyte[] value, int offset, int length)  public void Write(byte[] value) public void Write(sbyte[] value) public void Write(byte[] value, int offset, int length) public void Write(sbyte[] value, int offset, int length) public void WriteBytes(string value)</pre>
<b>Decimal2</b>	<pre>public void WriteDecimal2(short value)</pre>
<b>Decimal4</b>	<pre>public void WriteDecimal4(short value)</pre>
<b>Decimal8</b>	<pre>public void WriteDecimal8(short value)</pre>
<b>Double</b>	<pre>public double ReadDouble()  public void WriteDouble(double value)</pre>
<b>Float</b>	<pre>public float ReadFloat()  public void WriteFloat(float value)</pre>
<b>Int2</b>	<pre>public void WriteInt2(int value)</pre>

표 843. 읽기 및 쓰기 메시지 메소드 (계속)

대상 유형	메소드 서명
<b>Int4</b>	<pre>public int readDecimal4() public int ReadInt() public int ReadInt4()  public void WriteInt(int value) public void WriteInt4(int value)</pre>
<b>Int8</b>	<pre>public void WriteInt8(long value)</pre>
<b>Long</b>	<pre>public long ReadDecimal8() public long ReadLong() public long ReadInt8()  public void WriteLong(long value)</pre>
<b>Object</b>	<pre>public Object ReadObject()  public void WriteObject(Object object)</pre>
<b>Short</b>	<pre>public short ReadShort() public short ReadDecimal2() public short ReadInt2()  public void WriteShort(int value)</pre>
<b>string</b>	<pre>public string ReadString(int length)  public void WriteString(string string)</pre>
<b>Unsigned Short</b>	<pre>public ushort ReadUnsignedShort() public ushort ReadUInt2()</pre>
<b>Unicode</b>	<pre>public string ReadLine() public char ReadChar()  public void WriteChar(int value) public void WriteChars(string string)</pre>

표 843. 읽기 및 쓰기 메시지 메소드 (계속)

대상 유형	메소드 서명
UTF	public string ReadUTF()
	public void WriteUTF(string <i>string</i> )

## 버퍼 메소드

### public void ClearMessage();

IOException을 전달합니다.

메시지 버퍼에서 데이터를 제거하고, 데이터 오프셋을 0으로 설정합니다.

### public void ResizeBuffer(int *size*)

IOException을 전달합니다.

후속 Get 조작에 필요할 수 있는 버퍼 크기에 대한 MQMessage 오브젝트 힌트. 메시지에 현재 메시지 데이터가 포함되어 있고 새 크기가 현재 크기보다 작으면 메시지 데이터가 잘립니다.

### public void Seek(int *pos*)

IOException, ArgumentOutOfRangeException, ArgumentException을 전달합니다.

*pos*에 제공된 메시지 버퍼의 절대 위치로 커서를 이동시킵니다. 후속 읽기 및 쓰기는 버퍼의 이 위치에서 수행합니다.

### public int SkipBytes(int *i*)

IOException, EndOfStreamException을 전달합니다.

메시지 버퍼에서 *n*바이트를 진행하고 건너뛰는 바이트 수, *n*을 리턴합니다.

SkipBytes 메소드는 다음 이벤트 중 하나가 발생할 때까지 차단됩니다.

- 모든 바이트를 건너뛰
- 메시지 버퍼의 끝이 감지됨
- 예외가 처리됨

## 특성 메소드

### public void DeleteProperty(string *name*);

MQException을 전달합니다.

메시지에서 지정된 이름의 특성을 삭제합니다.

**이름**

삭제할 특성 이름입니다.

### public System.Collections.IEnumerator GetPropertyNames(string *name*)

MQException을 전달합니다.

지정된 이름과 일치하는 모든 특성 이름의 IEnumerator를 리턴합니다. 이름 끝에 와일드카드 문자로 퍼센트 부호 '%'를 사용하면 마침표(.)를 포함하여 0개 이상의 문자와 일치하는 항목을 찾아 메시지의 특성을 필터링할 수 있습니다.

**이름**

일치하는 항목을 찾을 특성의 이름.



## SetProperty 및 GetProperty 메소드

모든 SetProperty 및 GetProperty 메소드는 MQException을 처리합니다.

MQMessage.NET 클래스의 SetProperty 메소드는 특성이 아직 없는 경우 새 특성을 추가합니다. 그러나 특성이 있는 경우 제공된 특성 값은 목록의 마지막에 추가됩니다. SetProperty를 사용하여 여러 개의 값이 특성 이름에 설정된 경우 해당 이름에 대한 GetProperty를 호출하면 해당 값이 설정된 순서 대로 값이 리턴됩니다.

동작은 모든 Set\*Property 및 Get\*Property 유형 메소드(예: GetLongProperty, SetLongProperty, GetBooleanProperty, SetBooleanProperty, GetStringProperty 및 SetStringProperty)에 대해 동일합니다.

표 844. SetProperty 및 GetProperty 메소드	
유형	메소드 서명
<b>Boolean</b>	<pre>public boolean GetBooleanProperty(string name); public boolean GetBooleanProperty(string name, MQPropertyDescriptor pd);  public void SetBooleanProperty(string name, boolean value); public void SetBooleanProperty(string name, MQPropertyDescriptor pd, boolean value);</pre>
<b>Byte</b>	<pre>public sbyte GetByteProperty(string name); public sbyte GetByteProperty(string name, MQPropertyDescriptor pd);  public void SetByteProperty(string name, sbyte value); public void SetByteProperty(string name, MQPropertyDescriptor pd, sbyte value);</pre>
<b>Bytes</b>	<pre>public sbyte[] GetBytesProperty(string name); public sbyte[] GetBytesProperty(string name, MQPropertyDescriptor pd);  public void SetBytesProperty(string name, sbyte[] value); public void SetBytesProperty(string name, MQPropertyDescriptor pd, sbyte[] value);</pre>
<b>Double</b>	<pre>public double GetDoubleProperty(string name); public double GetDoubleProperty(string name, MQPropertyDescriptor pd);  public void SetDoubleProperty(string name, double value); public void SetDoubleProperty(string name, MQPropertyDescriptor pd, double value);</pre>
<b>Float</b>	<pre>public float GetFloatProperty(string name); public float GetFloatProperty(string name, MQPropertyDescriptor pd);  public void SetFloatProperty(string name, float value); public void SetFloatProperty(string name, MQPropertyDescriptor pd, float value);</pre>

표 844. SetProperty 및 GetProperty 메소드 (계속)

유형	메소드 서명
<b>Int2</b>	<pre>public short GetInt2Property(string name); public short GetInt2Property(string name, MQPropertyDescriptor pd);  public void SetInt2Property(string name, short value); public void SetInt2Property(string name, MQPropertyDescriptor pd, short value);</pre>
<b>Int4</b>	<pre>public int GetInt4Property(string name); public int GetInt4Property(string name, MQPropertyDescriptor pd);  public void SetInt4Property(string name, int value); public void SetInt4Property(string name, MQPropertyDescriptor pd, int value);</pre>
<b>Int8</b>	<pre>public long GetInt8Property(string name); public long GetInt8Property(string name, MQPropertyDescriptor pd);  public void SetInt8Property(string name, long value); public void SetInt8Property(string name, MQPropertyDescriptor pd, long value);</pre>
<b>Long</b>	<pre>public long GetLongProperty(string name); public long GetLongProperty(string name, MQPropertyDescriptor pd);  public void SetLongProperty(string name, long value); public void SetLongProperty(string name, MQPropertyDescriptor pd, long value);</pre>
<b>Object</b>	<pre>public Object GetObjectProperty(string name); public Object GetObjectProperty(string name, MQPropertyDescriptor pd);  public void SetObjectProperty(string name, Object value); public void SetObjectProperty(string name, MQPropertyDescriptor pd, Object value);</pre>
<b>Short</b>	<pre>public short GetShortProperty(string name); public short GetShortProperty(string name, MQPropertyDescriptor pd);  public void SetShortProperty(string name, short value); public void SetShortProperty(string name, MQPropertyDescriptor pd, short value);</pre>
<b>string</b>	<pre>public string GetStringProperty(string name); public string GetStringProperty(string name, MQPropertyDescriptor pd);  public void SetStringProperty(string name, string value); public void SetStringProperty(string name, MQPropertyDescriptor pd, string value);</pre>

## 구성자

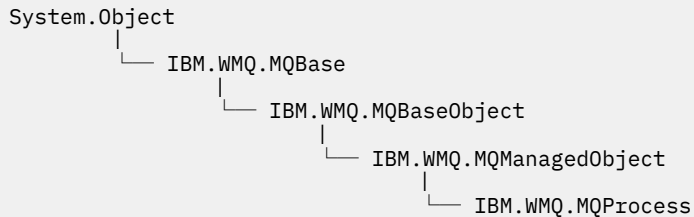
**public MQMessage();**

기본 메시지 디스크립터 정보와 빈 메시지 버퍼를 사용하여 MQMessage 오브젝트를 작성합니다.

## MQProcess.NET 클래스

MQProcess 를 사용하여 IBM MQ 프로세스의 속성을 조회하십시오. 구성자 또는 MQQueueManager AccessProcess 메소드를 사용하여 MQProcess 오브젝트를 작성하십시오.

## 클래스



```
public class IBM.WMQ.MQProcess extends IBM.WMQ.MQManagedObject;
```

- [1671 페이지의 『특성』](#)
- [1672 페이지의 『구성자』](#)

## 특성

특성을 가져올 때 전달되는 MQException에 대한 테스트.

```
public string ApplicationId {get;}
```

시작할 애플리케이션을 식별하는 문자열을 가져옵니다. ApplicationId은 트리거 모니터 애플리케이션에 사용됩니다. ApplicationId가 트리거 메시지의 일부로 이니시에이션 큐에 송신됩니다.

기본값은 널입니다.

```
public int ApplicationType {get;}
```

트리거 모니터 애플리케이션을 통해 시작되는 프로세스의 유형을 식별합니다. 표준 유형이 정의되어 있지만 다른 유형을 사용할 수 있습니다.

- MQAT\_AIX
- MQAT\_CICS
- MQAT\_IMS
- MQAT\_MVS
- MQAT\_NATIVE
- MQAT\_OS400
- MQAT\_UNIX
- MQAT\_WINDOWS
- MQAT\_JAVA
- MQAT\_USER\_FIRST
- MQAT\_USER\_LAST

기본값은 MQAT\_NATIVE입니다.

```
public string EnvironmentData {get;}
```

시작할 애플리케이션 환경에 대한 정보를 가져옵니다.

기본값은 널입니다.

```
public string UserData {get;}
```

시작할 애플리케이션에 대해 사용자가 제공한 정보를 가져옵니다.

기본값은 널입니다.

## 구성자

```
public MQProcess(MQQueueManager queueManager, string processName, int openOptions);
```

```
public MQProcess(MQQueueManager qMgr, string processName, int openOptions, string queueManagerName, string alternateUserId);
```

MQException을 전달합니다.

큐 관리자 *qMgr*의 IBM MQ 프로세스에 액세스하여 프로세스 속성을 조회합니다.

**qMgr**

액세스할 큐 관리자.

**processName**

열려는 프로세스의 이름.

**openOptions**

프로세스 열기를 제어하는 옵션. 비트 단위 OR을 사용하여 추가하거나 결합할 수 있는 유효한 옵션은 다음과 같습니다.

- MQC.MQ00\_FAIL\_IF QUIESCING
- MQC.MQ00\_INQUIRE
- MQC.MQ00\_SET
- MQC.MQ00\_ALTERNATE\_USER\_AUTHORITY

**queueManagerName**

프로세스가 정의된 큐 관리자의 이름. 큐 관리자가 프로세스에서 액세스하는 것과 동일하면 큐 관리자 이름을 공백으로 두거나 널로 지정할 수 있습니다.

**alternateUserId**

MQC.MQ00\_ALTERNATE\_USER\_AUTHORITY가 **openOptions** 매개변수에 지정되면, *alternateUserId*는 조치의 권한을 확인하는 데 사용되는 대체 사용자 ID를 지정합니다. MQ00\_ALTERNATE\_USER\_AUTHORITY가 지정되지 않으면, *alternateUserId*는 공백이거나 널일 수 있습니다.

MQC.MQ00\_ALTERNATE\_USER\_AUTHORITY를 지정하지 않은 경우, 연결에 기본 사용자 권한이 사용됩니다.

```
public MQProcess MQQueueManager.AccessProcess(string processName, int openOptions);
```

```
public MQProcess MQQueueManager.AccessProcess(string processName, int openOptions, string queueManagerName, string alternateUserId);
```

MQException을 전달합니다.

이 큐 관리자의 IBM MQ 프로세스에 액세스하여 프로세스 속성에 대해 조사하십시오.

**processName**

열려는 프로세스의 이름.

### openOptions

프로세스 열기를 제어하는 옵션. 비트 단위 OR을 사용하여 추가하거나 결합할 수 있는 유효한 옵션은 다음과 같습니다.

- MQC.MQOO\_FAIL\_IF\_QUIESCING
- MQC.MQOO\_INQUIRE
- MQC.MQOO\_SET
- MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY

### queueManagerName

프로세스가 정의된 큐 관리자의 이름. 큐 관리자가 프로세스에서 액세스하는 것과 동일하면 큐 관리자 이름을 공백으로 두거나 널로 지정할 수 있습니다.

### alternateUserId

MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY가 **openOptions** 매개변수에 지정되면, *alternateUserId*는 조치의 권한을 확인하는 데 사용되는 대체 사용자 ID를 지정합니다. MQOO\_ALTERNATE\_USER\_AUTHORITY가 지정되지 않으면, *alternateUserId*는 공백이거나 널일 수 있습니다.

MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY를 지정하지 않은 경우, 연결에 기본 사용자 권한이 사용됩니다.

## MQPropertyDescriptor.NET 클래스

MQPropertyDescriptor를 MQMessage GetProperty 및 SetProperty 메소드에 대한 매개변수로 사용하십시오. MQPropertyDescriptor는 MQMessage 특성에 대해 설명합니다.

### 클래스

```
System.Object
└── IBM.WMQ.MQPropertyDescriptor
```

```
public class IBM.WMQ.MQPropertyDescriptor extends System.Object;
```

- [1673 페이지의 『특성』](#)
- [1674 페이지의 『구성자』](#)

### 특성

특성을 가져올 때 전달되는 MQException에 대한 테스트.

```
public int Context {get; set;}
```

특성이 속한 메시지 컨텍스트. 가능한 값은 다음과 같습니다.

#### MQC.MQPD\_NO\_CONTEXT

특성은 메시지 컨텍스트와 연관되지 않습니다.

#### MQC.MQPD\_USER\_CONTEXT

특성은 사용자 컨텍스트와 연관됩니다.

사용자에게 권한이 부여된 경우, 메시지가 검색될 때 사용자 컨텍스트와 연관된 특성이 저장됩니다. 저장된 컨텍스트를 참조하는 후속 Put 메소드는 특성을 새 메시지로 전달할 수 있습니다.

```
public int CopyOptions {get; set;}
```

CopyOptions는 특성이 복사될 수 있는 메시지의 유형을 설명합니다.

큐 관리자가 올바르지 않다고 인식하는 IBM MQ 정의 특성이 포함된 메시지를 큐 관리자가 수신하는 경우, 큐 관리자는 Context 필드의 값을 조정합니다.

다음 옵션의 조합을 지정할 수 있습니다. 값을 추가하거나 비트 단위의 OR을 사용하여 옵션을 결합하십시오.

#### **MQC.MQCOPY\_ALL**

이 특성은 모든 유형의 후속 메시지에 복사됩니다.

#### **MQC.MQCOPY\_FORWARD**

이 특성은 전달되는 메시지에 복사됩니다.

#### **MQC.MQCOPY\_PUBLISH**

이 특성은 메시지가 발행되는 경우 구독자가 수신하는 메시지에 복사됩니다.

#### **MQC.MQCOPY\_REPLY**

이 특성은 응답 메시지에 복사됩니다.

#### **MQC.MQCOPY\_REPORT**

이 특성은 보고 메시지에 복사됩니다.

#### **MQC.MQCOPY\_DEFAULT**

이 값은 다른 복사 옵션이 지정되지 않았음을 표시합니다. 특성과 후속 메시지 간에 어떤 관계도 존재하지 않습니다. 메시지 디스크립터 특성에 대해서는 항상 MQC.MQCOPY\_DEFAULT가 리턴됩니다.

#### **MQC.MQCOPY\_NONE**

MQC.MQCOPY\_DEFAULT와 동일합니다.

```
public int Options { set; }
```

Options의 기본값은 CMQC.MQPD\_NONE입니다. 다른 값을 설정할 수 없습니다.

```
public int Support { get; set; }
```

IBM MQ정의 메시지 특성에 필요한 지원 레벨을 지정하려면 지원 을 설정하십시오. 그 외의 다른 모든 특성에 대한 지원은 선택사항입니다. 다음 값 중 아무거나 지정하거나 지정하지 않아도 됩니다.

#### **MQC.MQPD\_SUPPORT\_OPTIONAL**

지원되지 않는 경우라도 특성은 큐 관리자에 의해 허용됩니다. 특성은 메시지 특성을 지원하지 않는 큐 관리자로 메시지를 플로우하기 위해 특성을 제거할 수 있습니다. 또한 이 값은 IBM MQ에서 정의하지 않은 특성에도 지정됩니다.

#### **MQC.MQPD\_SUPPORT\_REQUIRED**

특성에 대한 지원은 필수입니다. IBM MQ 정의 특성을 지원하지 않는 큐 관리자에 메시지를 넣는 경우, 메소드가 실패합니다. 완료 코드 MQC.MQCC\_FAILED 및 이유 코드 MQC.MQRC\_UNSUPPORTED\_PROPERTY를 리턴합니다.

#### **MQC.MQPD\_SUPPORT\_REQUIRED\_IF\_LOCAL**

메시지가 로컬 큐로 이동하는 경우, 이 특성은 반드시 지원해야 합니다. IBM MQ 정의 특성을 지원하지 않는 큐 관리자의 로컬 큐에 메시지를 넣는 경우, 메소드가 실패합니다. 완료 코드 MQC.MQCC\_FAILED 및 이유 코드 MQC.MQRC\_UNSUPPORTED\_PROPERTY를 리턴합니다.

메시지를 리모트 큐 관리자에 넣는 경우에는 검사가 수행되지 않습니다.

## 구성자

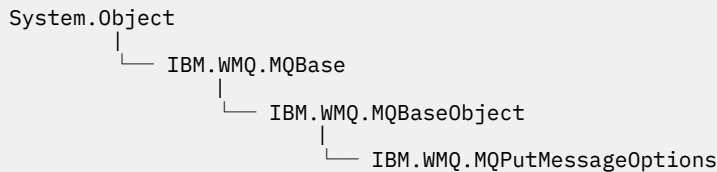
```
PropertyDescriptor();
```

특성 디스크립터를 작성합니다.

## MQPutMessageOptions.NET 클래스

메시지 송신 방법을 지정하려면 MQPutMessageOptions를 사용하십시오. MQDestination.Put의 동작을 수정합니다.

## 클래스



```
public class IBM.WMQ.MQPutMessageOptions extends IBM.WMQ.MQBaseObject;
```

• [1675 페이지의 『특성』](#) [1677 페이지의 『구성자』](#)

## 특성

특성을 가져올 때 전달되는 MQException에 대한 테스트.

**참고:** 이 클래스에 사용할 수 있는 일부 옵션의 동작은 사용 환경에 따라 다릅니다. 이러한 요소는 별표 \*로 표시됩니다.

### **public MQQueue ContextReference {get; set;}**

options 필드에 MQC.MQPMO\_PASS\_IDENTITY\_CONTEXT 또는 MQC.MQPMO\_PASS\_ALL\_CONTEXT가 있으면 이 필드가 컨텍스트 정보를 가져오는 MQQueue를 참조하도록 설정하십시오.

이 필드의 초기값은 널입니다.

### **public int InvalidDestCount {get;} \***

일반적으로, 분배 목록에 사용되는 InvalidDestCount는 분배 목록에 있는 큐로 송신할 수 없는 메시지의 수를 표시합니다. 이 수에는 열지 못한 큐와 성공적으로 열었으나 Put 조작이 실패한 큐도 포함됩니다.

.NET은 분배 목록을 지원하지 않지만 단일 큐를 여는 경우 InvalidDestCount가 설정됩니다.

### **public int KnownDestCount {get;} \***

일반적으로 분배 목록에 사용되는 KnownDestCount는 현재 호출이 로컬 큐로 해석되는 큐로 성공적으로 송신한 메시지의 수를 표시합니다.

.NET은 분배 목록을 지원하지 않지만 단일 큐를 여는 경우 InvalidDestCount가 설정됩니다.

### **public int Options {get; set;}**

MQDestination.put 및 MQQueueManager.put의 조치를 제어하는 옵션. 다음 값 중 아무거나 지정하거나 지정하지 않아도 됩니다. 둘 이상의 옵션이 필요하면 값을 추가하거나 비트 단위의 OR 연산자를 사용하여 결합할 수 있습니다.

#### **MQC.MQPMO\_ASYNC\_RESPONSE**

이 옵션은 몇 가지 응답 데이터를 통해 MQDestination.put 호출이 비동기적으로 실행되도록 합니다.

#### **MQC.MQPMO\_DEFAULT\_CONTEXT**

기본 컨텍스트를 메시지와 연관시킵니다.

#### **MQC.MQPMO\_FAIL\_IF QUIESCING**

큐 관리자가 정지 중인 경우 실패합니다.

#### **MQC.MQPMO\_LOGICAL\_ORDER \***

메시지 그룹의 논리 메시지와 세그먼트를 논리 순서대로 넣습니다.

다시 연결 가능한 클라이언트에서 MQPMO\_LOGICAL\_ORDER 옵션을 사용하는 경우 MQRC\_RECONNECT\_INCOMPATIBLE 이유 코드가 애플리케이션에 리턴됩니다.

#### **MQC.MQPMO\_NEW\_CORREL\_ID \***

각 송신 데이터마다 새 상관 ID를 생성합니다.

#### **MQC.MQPMO\_NEW\_MSG\_ID \***

각 수신 데이터마다 새 메시지 ID를 생성합니다.

#### **MQC.MQPMO\_NONE**

옵션이 지정되지 않았습니다. 다른 옵션과 함께 사용하지 마십시오.

**MQC.MQPMO\_NO\_CONTEXT**

메시지와 연관된 컨텍스트가 없습니다.

**MQC.MQPMO\_NO\_SYNCPOINT**

동기점 제어 없이 메시지를 넣습니다. 동기점 제어 옵션을 지정하지 않으면 기본값 동기점 없음이 사용됩니다.

**MQC.MQPMO\_PASS\_ALL\_CONTEXT**

입력 큐 핸들에서 모든 컨텍스트를 전달하십시오.

**MQC.MQPMO\_PASS\_IDENTITY\_CONTEXT**

입력 큐 핸들에서 ID 컨텍스트를 전달하십시오.

**MQC.MQPMO\_RESPONSE\_AS\_Q\_DEF**

MQDestination.put 호출의 경우, 이 옵션은 큐의 DEFPRESP 속성에서 Put 응답 유형을 가져옵니다.

MQQueueManager.put 호출의 경우, 이 옵션은 동기적으로 호출되도록 합니다.

**MQC.MQPMO\_RESPONSE\_AS\_TOPIC\_DEF**

토픽 오브젝트와 함께 사용하는 경우 MQC.MQPMO\_RESPONSE\_AS\_TOPIC\_DEF는 MQC.MQPMO\_RESPONSE\_AS\_Q\_DEF의 동의어입니다.

**MQC.MQPMO\_RETAIN**

송신되는 발행물을 큐 관리자가 보유하게 됩니다. 이 옵션을 사용해도 발행물을 보유할 수 없는 경우에는, 메시지가 발행되지 않고 호출이 실패하며 MQC.MQRC\_PUT\_NOT\_RETAINED가 발생합니다.

발행된 시간 이후에 MQSubscription.RequestPublicationUpdate 메소드를 호출하여 이 발행물의 사본을 요청합니다. 저장된 발행물은 MQC.MQSO\_NEW\_PUBLICATIONS\_ONLY 옵션을 설정하지 않고 구독을 작성하는 애플리케이션으로 송신됩니다. 수신 시 발행물의 MQIsRetained 메시지 특성을 확인하여 보유된 발행물이었는지 알아봅니다.

구독자가 보유된 발행물을 요청하는 경우, 사용된 구독의 토픽 문자열에 와일드카드를 포함할 수 있습니다. 구독과 일치하는 보유된 발행물이 토픽 트리에 여러 개 있으면 모두 송신됩니다.

**MQC.MQPMO\_SET\_ALL\_CONTEXT**

애플리케이션의 모든 컨텍스트를 설정하십시오.

**MQC.MQPMO\_SET\_IDENTITY\_CONTEXT**

애플리케이션의 ID 컨텍스트를 설정하십시오.

**MQC.MQPMO\_SYNC\_RESPONSE**

이 옵션은 전체 응답 데이터를 통해 MQDestination.put 또는 MQQueueManager.put 호출이 동기적으로 실행되도록 합니다.

**MQC.MQPMO\_SUPPRESS\_REPLYTO**

발행물의 ReplyToQueueName 및 ReplyToQueueManagerName 필드에 입력되는 정보는 구독자에게 전달되지 않습니다. 이 옵션을 ReplyToQueueName이 필요한 보고 옵션과 함께 사용하는 경우, MQC.MQRC\_MISSING\_REPLY\_TO\_Q가 표시되면서 호출이 실패합니다.

**MQC.MQPMO\_SYNCPOINT**

동기점 제어로 메시지를 넣습니다. 작업 단위가 커밋될 때까지 작업 단위의 외부에는 메시지가 표시되지 않습니다. 작업 단위가 백아웃되면 메시지가 삭제됩니다.

**public int RecordFields {get; set;} \***

분배 목록에 대한 정보. 분배 목록은 .NET에서 지원되지 않습니다.

**public string ResolvedQueueManagerName {get;}**

큐 관리자가 리모트 큐 이름으로 지정된 큐를 소유한 큐 관리자의 이름으로 설정한 출력 필드. ResolvedQueueManagerName은 큐가 리모트 큐일 때 큐에 액세스한 큐 관리자의 이름과 다를 수 있습니다.

오브젝트가 단일 큐이면 공백이 아닌 값이 리턴됩니다. 오브젝트가 분배 목록 또는 토픽이면 리턴되는 값이 정의되지 않습니다.



**public string ResolvedQueueName {get;}**

큐 관리자에서 메시지가 놓이는 큐의 이름으로 설정한 출력 필드. ResolvedQueueName은 열린 큐가 알리어스 또는 모델 큐인 경우 큐를 여는 데 사용한 이름과 다를 수 있습니다.

오브젝트가 단일 큐인 경우에만 공백이 아닌 값이 리턴됩니다. 오브젝트가 분배 목록 또는 토픽이면 리턴되는 값이 정의되지 않습니다.

**public int UnknownDestCount {get;} \***

일반적으로 분배 목록에 사용되는 UnknownDestCount는 큐 관리자가 설정한 출력 필드입니다. 현재 호출이 리모트 큐로 해석되는 큐로 성공적으로 송신한 메시지의 수를 보고합니다.

.NET은 분배 목록을 지원하지 않지만 단일 큐를 여는 경우 InvalidDestCount가 설정됩니다.

## 구성자

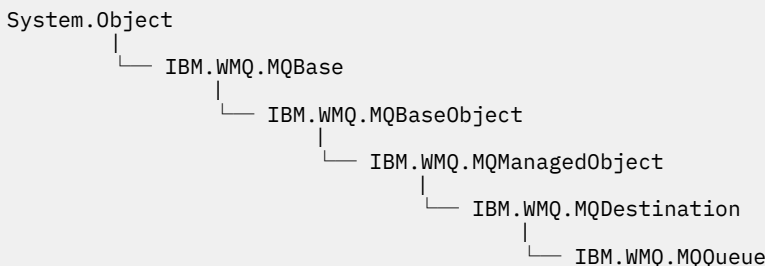
**public MQPutMessageOptions();**

공백 ResolvedQueueName 및 ResolvedQueueManagerName, 옵션 세트가 없는 새 MQPutMessageOptions 오브젝트를 구성합니다.

## MQQueue.NET 클래스

MQQueue 를 사용하여 메시지를 송수신하고 IBM MQ 큐의 속성을 조회하십시오. 구성자 또는 MQQueueManager.AccessProcess 메소드를 사용하여 MQQueue 오브젝트를 작성하십시오.

## 클래스



```
public class IBM.WMQ.MQQueue extends IBM.WMQ.MQDestination;
```

- [1677 페이지의 『특성』](#)
- [1679 페이지의 『메소드』](#)
- [1682 페이지의 『구성자』](#)

## 특성

특성을 가져올 때 전달되는 MQException에 대한 테스트.

**public int ClusterWorkLoadPriority {get;}**

큐의 우선순위를 지정합니다. 이 매개변수는 로컬, 리모트 및 알리어스 큐에 대해서만 유효합니다.

**public int ClusterWorkLoadRank {get;}**

큐의 순위를 지정합니다. 이 매개변수는 로컬, 리모트 및 알리어스 큐에 대해서만 유효합니다.

**public int ClusterWorkLoadUseQ {get;}**

대상 큐에 로컬 인스턴스 및 최소한 하나의 리모트 클러스터 인스턴스가 있는 경우에 MQPUT 조작의 동작을 지정합니다. MQPUT이 클러스터 채널에서 생성되는 경우에는 이 매개변수가 적용되지 않습니다. 이 매개변수는 로컬 큐에 대해서만 유효합니다.

**public DateTime CreationDateTime {get;}**

이 큐가 작성된 날짜 및 시간.

**public int CurrentDepth {get;}**

현재 큐에 있는 메시지의 수를 가져옵니다. Put 호출과, Get 호출의 백아웃 동안에는 이 값이 증가합니다. 찾아보기가 아닌 Get 호출과, Put 호출의 백아웃 동안에는 감소합니다.

**public int DefinitionType {get;}**

큐가 정의된 방법. 가능한 값은 다음과 같습니다.

- MQC.MQQDT\_PREDEFINED
- MQC.MQQDT\_PERMANENT\_DYNAMIC
- MQC.MQQDT\_TEMPORARY\_DYNAMIC

**public int InhibitGet {get; set;}**

이 큐에 또는 이 토픽에 대해 메시지를 가져올 수 있는지 여부를 제어합니다. 가능한 값은 다음과 같습니다.

- MQC.MQQA\_GET\_INHIBITED
- MQC.MQQA\_GET\_ALLOWED

**public int InhibitPut {get; set;}**

이 큐에 또는 이 토픽에 대해 메시지를 넣을 수 있는지 여부를 제어합니다. 가능한 값은 다음과 같습니다.

- MQQA\_PUT\_INHIBITED
- MQQA\_PUT\_ALLOWED

**public int MaximumDepth {get;}**

한 번에 큐에 존재할 수 있는 메시지의 최대 수. 이렇게 많은 메시지가 이미 들어 있는 큐에 메시지를 넣으려고 하면 이유 코드 MQC.MQRC\_Q\_FULL이 표시되면서 실패합니다.

**public int MaximumMessageLength {get;}**

이 큐에서 각 메시지에 존재할 수 있는 애플리케이션 데이터의 최대 길이. 이 값보다 큰 메시지를 넣으려고 하면 이유 코드 MQC.MQRC\_MSG\_TOO\_BIG\_FOR\_Q가 표시되면서 실패합니다.

**public int NonPersistentMessageClass {get;}**

이 큐에 넣은 비지속 메시지에 대한 신뢰성의 레벨.

**public int OpenInputCount {get;}**

현재 큐에서 메시지를 제거할 수 있는 핸들의 수. OpenInputCount는 단순히 애플리케이션에서 작성한 핸들이 아니라 로컬 큐 관리자에 인식되는 올바른 입력 핸들의 합계입니다.

**public int OpenOutputCount {get;}**

현재 큐에 메시지를 추가할 수 있는 핸들의 수. OpenOutputCount는 단순히 애플리케이션에서 작성한 핸들이 아니라 로컬 큐 관리자에 인식되는 올바른 출력 핸들의 합계입니다.

**public int QueueAccounting {get;}**

큐의 계정 정보 수집을 사용할 수 있는지 여부를 지정합니다.

**public int QueueMonitoring {get;}**

큐에 대한 모니터링을 사용할 수 있는지 여부를 지정합니다.

**public int QueueStatistics {get;}**

큐 통계 수집을 사용할 수 있는지 여부를 지정합니다.

**public int QueueType {get;}**

이 큐의 유형은 다음 값 중 하나입니다.

- MQC.MQQT\_ALIAS
- MQC.MQQT\_LOCAL
- MQC.MQQT\_REMOTE
- MQC.MQQT\_CLUSTER

**public int Shareability {get;}**

입력을 위해 큐를 여러 번 열 수 있는지 여부. 가능한 값은 다음과 같습니다.

- MQC.MQQA\_SHAREABLE
- MQC.MQQA\_NOT\_SHAREABLE

**public string TPIPE {get;}**

IBM MQ IMS 브릿지를 사용하여 OTMA와 통신하는 데 사용되는 TPIPE 이름입니다.

**public int TriggerControl {get; set;}**

큐 제공을 위해 애플리케이션을 시작하도록 트리거 메시지가 이니시에이션 큐에 기록되는지 여부. 가능한 값은 다음과 같습니다.

- MQC.MQTC\_OFF
- MQC.MQTC\_ON

**public string TriggerData {get; set;}**

큐 관리자가 트리거 메시지에 삽입하는 자유 양식 데이터. 이 큐에 도착하는 메시지로 인해 트리거 메시지가 이니시에이션 큐에 기록되는 경우 TriggerData를 삽입합니다. 허용 가능한 최대 문자열 길이는 MQC.MQ\_TRIGGER\_DATA\_LENGTH를 통해 지정됩니다.

**public int TriggerDepth {get; set;}**

트리거 유형이 MQC.MQTT\_DEPTH로 설정되면 트리거 메시지가 기록되기 전에 큐에 있어야 하는 메시지의 수.

**public int TriggerMessagePriority {get; set;}**

메시지가 트리거 메시지 생성의 원인이 되지 않는 메시지 우선순위. 즉, 트리거의 생성 여부를 결정할 때 큐 관리자는 이러한 메시지를 무시합니다. 값이 0이면 모든 메시지가 트리거 메시지 생성의 원인이 됩니다.

**public int TriggerType {get; set;}**

이 큐에 도착한 메시지의 결과로 트리거 메시지가 기록되는 조건. 가능한 값은 다음과 같습니다.

- MQC.MQTT\_NONE
- MQC.MQTT\_FIRST
- MQC.MQTT EVERY
- MQC.MQTT\_DEPTH

## 메소드

**public void Get(MQMessage message);**

**public void Get(MQMessage message, MQGetMessageOptions getMessageOptions);**

**public void Get(MQMessage message, MQGetMessageOptions getMessageOptions, int MaxMsgSize);**

MQException을 전달합니다.

큐에서 메시지를 가져옵니다.

가져오기에 실패하면 MQMessage 오브젝트가 변경되지 않습니다. 가져오기에 성공하면 MQMessage의 메시지 데이터 부분이 수신되는 메시지의 메시지 디스크립터 및 메시지 데이터로 대체됩니다.

특정 MQQueueManager 에서 IBM MQ 로의 모든 호출은 동기적입니다. 따라서, Get(대기 포함)을 수행하는 경우 동일한 MQQueueManager를 사용하는 다른 모든 스레드가 Get 호출이 완료될 때까지 추가 IBM MQ 호출하지 못하도록 차단됩니다. 동시에 IBM MQ에 액세스하기 위해 여러 개의 스레드가 필요한 경우, 각 스레드가 고유의 MQQueueManager 오브젝트를 작성해야 합니다.

## 메시지

메시지 디스크립터 및 리턴된 메시지 데이터를 포함합니다. 메시지 디스크립터의 필드 중 일부는 입력 매개변수입니다. MessageId 및 CorrelationId 입력 매개변수가 필요에 따라 설정되었는지 확인해야 합니다.

다시 연결 가능한 클라이언트는 다시 연결에 성공한 후 MQGM\_SYNCPOINT에 따라 수신된 메시지에 대해 이유 코드 MQRC\_BACKED\_OUT을 리턴합니다.

## getMessageOptions

Get 조치를 제어하는 옵션.

MQC.MQGMO\_CONVERT 옵션을 사용하면 1바이트 문자 코드에서 2바이트 코드로 변환할 때 이유 코드 MQC.MQRC\_CONVERTED\_STRING\_TOO\_BIG과 함께 예외가 발생할 수 있습니다. 이 경우, 메시지가 변환 없이 버퍼로 복사됩니다.

*getMessageOptions*가 지정되지 않은 경우, 사용된 메시지 옵션은 MQGMO\_NOWAIT입니다.

reconnectable 클라이언트에서 MQGMO\_LOGICAL\_ORDER 옵션을 사용하는 경우 MQRC\_RECONNECT\_INCOMPATIBLE 이유 코드가 리턴됩니다.

### MaxMsgSize

이 메시지 오브젝트가 수신하는 최대 메시지. 큐의 메시지가 이 크기보다 크면 다음 두 가지 중 하나가 발생합니다.

- MQGMO\_ACCEPT\_TRUNCATED\_MSG 플래그가 MQGetMessageOptions 오브젝트에 설정된 경우, 메시지에 최대한 많은 메시지 데이터가 입력됩니다. MQCC\_WARNING 완료 코드 및 MQRC\_TRUNCATED\_MSG\_ACCEPTED 이유 코드와 함께 예외가 전달됩니다.
- MQGMO\_ACCEPT\_TRUNCATED\_MSG 플래그가 설정되지 않은 경우, 메시지가 큐에 유지됩니다. MQCC\_WARNING 완료 코드 및 MQRC\_TRUNCATED\_MSG\_FAILED 이유 코드와 함께 예외가 전달됩니다.

*MaxMsgSize*를 지정하지 않으면 전체 메시지가 검색됩니다.

```
public void Put(MQMessage message);  
public void Put(MQMessage message, MQPutMessageOptions putMessageOptions);
```

MQException을 전달합니다.

큐에 메시지를 넣습니다.

Put 호출이 완료된 후 MQMessage 오브젝트를 수정해도 IBM MQ 큐 또는 발행물 토픽의 실제 메시지에는 아무런 영향이 없습니다.

Put은 MQMessage 오브젝트의 MessageId 및 CorrelationId 특성을 업데이트하고 메시지 데이터를 지우지 않습니다. 추가적인 Put 또는 Get 호출은 MQMessage 오브젝트의 업데이트된 정보를 참조합니다. 예를 들면, 다음 코드 스니펫에서 첫 번째 메시지에 a와 두 번째 ab가 들어 있습니다.

```
msg.WriteString("a");  
q.Put(msg, pmo);  
msg.WriteString("b");  
q.Put(msg, pmo);
```

### 메시지

메시지 디스크립터 데이터와 송신할 메시지를 포함한 MQMessage 오브젝트. 이 메소드의 결과로 메시지 디스크립터가 대체될 수 있습니다. 이 메소드가 완료된 직후 메시지 디스크립터의 값이 큐에 넣거나 토픽에 발행한 값입니다.

다음 이유 코드가 다시 연결 가능한 클라이언트에 리턴됩니다.

- 지속 메시지에서 Put 호출을 실행하는 동안 연결이 끊기고 다시 연결에 성공한 경우 MQRC\_CALL\_INTERRUPTED.
- 비지속 메시지에서 Put 호출을 실행하는 동안 연결에 성공한 경우 MQRC\_NONE(애플리케이션 복구 참조).

### putMessageOptions

Put 조치를 제어하는 옵션.

*putMessageOptions*를 지정하지 않으면 MQPutMessageOptions의 기본 인스턴스가 사용됩니다.

reconnectable 클라이언트에서 MQPMO\_LOGICAL\_ORDER 옵션을 사용하는 경우 MQRC\_RECONNECT\_INCOMPATIBLE 이유 코드가 리턴됩니다.

**참고:** 간편함과 성능을 위해 단일 메시지를 큐에 넣으려는 경우 MQQueueManager.Put 오브젝트를 사용하십시오. 이를 위해서는 MQQueue 오브젝트가 있어야 합니다.

```
public void PutForwardMessage(MQMessage message);  
public void PutForwardMessage(MQMessage message, MQPutMessageOptions  
putMessageOptions);
```

MQException을 전달합니다.

*message*가 원본 메시지인 경우, 전달 중인 메시지를 큐에 넣습니다.

#### 메시지

메시지 디스크립터 데이터와 송신할 메시지를 포함한 MQMessage 오브젝트. 이 메소드의 결과로 메시지 디스크립터가 대체될 수 있습니다. 이 메소드가 완료된 직후 메시지 디스크립터의 값이 큐에 넣거나 토크에 발행한 값입니다.

다음 이유 코드가 다시 연결 가능한 클라이언트에 리턴됩니다.

- 지속 메시지에서 Put 호출을 실행하는 동안 연결이 끊기고 다시 연결에 성공한 경우 MQRC\_CALL\_INTERRUPTED.
- 비지속 메시지에서 Put 호출을 실행하는 동안 연결에 성공한 경우 MQRC\_NONE(애플리케이션 복구 참조).

#### putMessageOptions

Put 조치를 제어하는 옵션.

*putMessageOptions*를 지정하지 않으면 MQPutMessageOptions의 기본 인스턴스가 사용됩니다.

reconnectable 클라이언트에서 MQPMO\_LOGICAL\_ORDER 옵션을 사용하는 경우

MQRC\_RECONNECT\_INCOMPATIBLE 이유 코드가 리턴됩니다.

```
public void PutReplyMessage(MQMessage message)  
public void PutReplyMessage(MQMessage message, MQPutMessageOptions  
putMessageOptions)
```

MQException을 전달합니다.

*message*가 원본 메시지인 경우, 응답 메시지를 큐에 넣습니다.

#### 메시지

메시지 디스크립터 및 리턴된 메시지 데이터를 포함합니다. 메시지 디스크립터의 필드 중 일부는 입력 매개변수입니다. MessageId 및 CorrelationId 입력 매개변수가 필요에 따라 설정되었는지 확인해야 합니다.

다시 연결 가능한 클라이언트는 다시 연결에 성공한 후 MQGM\_SYNCPOINT에 따라 수신된 메시지에 대해 이유 코드 MQRC\_BACKED\_OUT을 리턴합니다.

#### putMessageOptions

Put 조치를 제어하는 옵션.

*putMessageOptions*를 지정하지 않으면 MQPutMessageOptions의 기본 인스턴스가 사용됩니다.

reconnectable 클라이언트에서 MQPMO\_LOGICAL\_ORDER 옵션을 사용하는 경우

MQRC\_RECONNECT\_INCOMPATIBLE 이유 코드가 리턴됩니다.

```
public void PutReportMessage(MQMessage message)  
public void PutReportMessage(MQMessage message, MQPutMessageOptions  
putMessageOptions)
```

MQException을 전달합니다.

*message*가 원본 메시지인 경우, 보고 메시지를 큐에 넣습니다.

#### 메시지

메시지 디스크립터 및 리턴된 메시지 데이터를 포함합니다. 메시지 디스크립터의 필드 중 일부는 입력 매개변수입니다. MessageId 및 CorrelationId 입력 매개변수가 필요에 따라 설정되었는지 확인해야 합니다.

다시 연결 가능한 클라이언트는 다시 연결에 성공한 후 MQGM\_SYNCPOINT에 따라 수신된 메시지에 대해 이유 코드 MQRC\_BACKED\_OUT을 리턴합니다.

## putMessageOptions

Put 조치를 제어하는 옵션.

*putMessageOptions*를 지정하지 않으면 MQPutMessageOptions의 기본 인스턴스가 사용됩니다.

reconnectable 클라이언트에서 MQPMO\_LOGICAL\_ORDER 옵션을 사용하는 경우

MQRC\_RECONNECT\_INCOMPATIBLE 이유 코드가 리턴됩니다.

## 구성자

```
public MQQueue MQQueueManager.AccessQueue(string queueName, int openOptions);  
public MQQueue MQQueueManager.AccessQueue(string queueName, int openOptions,  
string queueManagerName, string dynamicQueueName, string alternateUserId);
```

MQException을 전달합니다.

이 큐 관리자의 큐에 액세스합니다.

메시지를 가져오거나 찾아보고, 메시지를 넣고, 큐 속성에 대해 조회하거나, 큐의 속성을 설정할 수 있습니다.

이름 지정된 큐가 모델 큐이면 동적 로컬 큐가 작성됩니다. 결과적인 MQQueue 오브젝트의 name 속성을 조회하여 동적 큐의 이름을 확인합니다.

## queueName

열려는 큐의 이름.

## openOptions

큐 열기를 제어하는 옵션.

### MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY

지정된 사용자 ID로 유효성 검증합니다.

### MQC.MQOO\_BIND\_AS\_QDEF

큐의 기본 바인딩을 사용합니다.

### MQC.MQOO\_BIND\_NOT\_FIXED

특정 목적지에 바인딩하지 않습니다.

### MQC.MQOO\_BIND\_ON\_OPEN

큐를 열 때 핸들을 목적지에 바인딩합니다.

### MQC.MQOO\_BROWSE

메시지를 열어 찾아봅니다.

### MQC.MQOO\_FAIL\_IF QUIESCING

큐 관리자가 정지 중인 경우 실패합니다.

### MQC.MQOO\_INPUT\_AS\_Q\_DEF

큐 정의의 기본값을 사용하여 메시지를 열어 가져옵니다.

### MQC.MQOO\_INPUT\_SHARED

공유 액세스 권한으로 메시지를 열어 가져옵니다.

### MQC.MQOO\_INPUT\_EXCLUSIVE

독점 액세스 권한으로 메시지를 열어 가져옵니다.

### MQC.MQOO\_INQUIRE

조회를 위한 열기로, 특성을 조회하려는 경우 필수입니다.

### MQC.MQOO\_OUTPUT

메시지를 열어 넣습니다.

### MQC.MQOO\_PASS\_ALL\_CONTEXT

모든 컨텍스트의 전달을 허용합니다.

### MQC.MQOO\_PASS\_IDENTITY\_CONTEXT

ID 컨텍스트의 전달을 허용합니다.

### MQC.MQOO\_SAVE\_ALL\_CONTEXT

메시지가 검색되면 컨텍스트를 저장합니다.

**MQC.MQOO\_SET**

속성 설정을 위한 열기로, 특성을 설정하려는 경우 필수입니다.

**MQC.MQOO\_SET\_ALL\_CONTEXT**

모든 컨텍스트를 설정할 수 있습니다.

**MQC.MQOO\_SET\_IDENTITY\_CONTEXT**

ID 컨텍스트를 설정할 수 있습니다.

**queueManagerName**

큐가 정의된 큐 관리자의 이름. 완전히 공백이거나 널인 이름은 MQQueueManager 오브젝트가 연결되는 큐 관리자를 나타냅니다.

**dynamicQueueName**

queueName이 모델 큐의 이름을 지정하는 경우를 제외하고, *dynamicQueueName*은 무시됩니다. 해당 경우 *dynamicQueueName*은 작성될 동적 큐의 이름을 지정합니다. queueName이 모델 큐의 이름을 지정하면 공백 또는 널 이름은 올바르지 않습니다. 이름에서 공백이 아닌 마지막 문자가 별표 (\*) 인 경우 큐 관리자는 별표를 문자열로 대체합니다. 이 문자는 큐에 생성된 이름이 이 큐 관리자에서 고유하도록 보장합니다.

**alternateUserId**

MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY가 openOptions 매개변수에 지정되면, *alternateUserId*는 열기의 권한을 확인하는 데 사용되는 대체 사용자 ID를 지정합니다.

MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY가 지정되지 않으면, *alternateUserId*는 공백이거나 널일 수 있습니다.

```
public MQQueue(MQQueueManager queueManager, string queueName, int openOptions, string queueManagerName, string dynamicQueueName, string alternateUserId);
```

MQException을 전달합니다.

queueManager의 큐에 액세스합니다.

메시지를 가져오거나 찾아보고, 메시지를 넣고, 큐 속성에 대해 조회하거나, 큐의 속성을 설정할 수 있습니다. 이름 지정된 큐가 모델 큐이면 동적 로컬 큐가 작성됩니다. 결과적인 MQQueue 오브젝트의 name 속성을 조회하여 동적 큐의 이름을 확인합니다.

**queueManager**

액세스하는 큐의 큐 관리자.

**queueName**

열려는 큐의 이름.

**openOptions**

큐 열기를 제어하는 옵션.

**MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY**

지정된 사용자 ID로 유효성 검증합니다.

**MQC.MQOO\_BIND\_AS\_QDEF**

큐의 기본 바인딩을 사용합니다.

**MQC.MQOO\_BIND\_NOT\_FIXED**

특정 목적지에 바인딩하지 않습니다.

**MQC.MQOO\_BIND\_ON\_OPEN**

큐를 열 때 핸들을 목적지에 바인딩합니다.

**MQC.MQOO\_BROWSE**

메시지를 열어 찾아봅니다.

**MQC.MQOO\_FAIL\_IF QUIESCING**

큐 관리자가 정지 중인 경우 실패합니다.

**MQC.MQOO\_INPUT\_AS\_Q\_DEF**

큐 정의 기본값을 사용하여 메시지를 열어 가져옵니다.

**MQC.MQOO\_INPUT\_SHARED**

공유 액세스 권한으로 메시지를 열어 가져옵니다.

**MQC.MQOO\_INPUT\_EXCLUSIVE**

독점 액세스 권한으로 메시지를 열어 가져옵니다.

**MQC.MQOO\_INQUIRE**

조회를 위한 열기로, 특성을 조회하려는 경우 필수입니다.

**MQC.MQOO\_OUTPUT**

메시지를 열어 넣습니다.

**MQC.MQOO\_PASS\_ALL\_CONTEXT**

모든 컨텍스트의 전달을 허용합니다.

**MQC.MQOO\_PASS\_IDENTITY\_CONTEXT**

ID 컨텍스트의 전달을 허용합니다.

**MQC.MQOO\_SAVE\_ALL\_CONTEXT**

메시지가 검색되면 컨텍스트를 저장합니다.

**MQC.MQOO\_SET**

속성 설정을 위한 열기로, 특성을 설정하려는 경우 필수입니다.

**MQC.MQOO\_SET\_ALL\_CONTEXT**

모든 컨텍스트를 설정할 수 있습니다.

**MQC.MQOO\_SET\_IDENTITY\_CONTEXT**

ID 컨텍스트를 설정할 수 있습니다.

**queueManagerName**

큐가 정의된 큐 관리자의 이름. 완전히 공백이거나 널인 이름은 MQQueueManager 오브젝트가 연결되는 큐 관리자를 나타냅니다.

**dynamicQueueName**

queueName이 모델 큐의 이름을 지정하는 경우를 제외하고, *dynamicQueueName*은 무시됩니다. 해당 경우 *dynamicQueueName*은 작성될 동적 큐의 이름을 지정합니다. queueName이 모델 큐의 이름을 지정하면 공백 또는 널 이름은 올바르지 않습니다. 이름에서 공백이 아닌 마지막 문자가 별표 (\*) 인 경우 큐 관리자는 별표를 문자열로 대체합니다. 이 문자는 큐에 생성된 이름이 이 큐 관리자에서 고유하도록 보장합니다.

**alternateUserId**

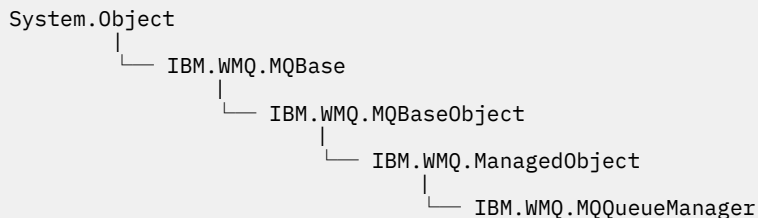
MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY가 openOptions 매개변수에 지정되면, *alternateUserId*는 열기의 권한을 확인하는 데 사용되는 대체 사용자 ID를 지정합니다.

MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY가 지정되지 않으면, *alternateUserId*는 공백이거나 널일 수 있습니다.

## MQQueueManager.NET 클래스

큐 관리자에 연결하고 큐 관리자 오브젝트에 액세스하려면 MQQueueManager를 사용하십시오. 또한 이 클래스는 트랜잭션을 제어합니다. MQQueueManager 구성자는 클라이언트 또는 서버 연결을 작성합니다.

### 클래스



```
public class IBM.WMQ.MQQueueManager extends IBM.WMQ.MQManagedObject;
```

- [1685 페이지의 『특성』](#)
- [1688 페이지의 『메소드』](#)



## 특성

특성을 가져올 때 전달되는 MQException에 대한 테스트.

**public int AccountingConnOverride {get;}**

애플리케이션이 MQI 계정 및 큐 계정 값의 설정을 대체할 수 있는지 여부.

**public int AccountingInterval {get;}**

중간 계정 레코드가 작성되기까지 얼마나 걸렸습니까(초)?

**public int ActivityRecording {get;}**

활동 보고서의 생성을 제어합니다.

**public int AdoptNewMCACheck {get;}**

새 인바운드 채널이 감지될 때 MCA가 채택되는지 여부를 판별하기 위해 확인하는 요소를 지정합니다. 채택되기 위해서는 MCA 이름이 활성 MCA의 이름과 일치해야 합니다.

**public int AdoptNewMCAInterval {get;}**

orphan 채널이 종료될 때까지 새 채널이 대기하는 시간(초).

**public int AdoptNewMCAType {get;}**

AdoptNewMCACheck 값과 일치하는 새 인바운드 채널 요청이 감지되면 orphan MCA 인스턴스가 채택(재시작)되는지 여부.

**public int BridgeEvent {get;}**

IMS 브릿지 이벤트의 생성 여부.

**public int ChannelEvent {get;}**

채널 이벤트의 생성 여부.

**public int ChannelInitiatorControl {get;}**

큐 관리자 시작 시 채널 시작기가 자동으로 시작하는지 여부.

**public int ChannelInitiatorAdapters {get;}**

IBM MQ 호출을 처리하는 어댑터 서브태스크의 수.

**public int ChannelInitiatorDispatchers {get;}**

채널 시작기에 사용할 디스패처 수.

**public int ChannelInitiatorTraceAutoStart {get;}**

채널 시작기 추적이 자동으로 시작하는지 여부를 지정합니다.

**public int ChannelInitiatorTraceTableSize {get;}**

채널 시작기의 추적 데이터 공간 크기(MB).

**public int ChannelMonitoring {get;}**

채널 모니터링의 사용 여부.

**public int ChannelStatistics {get;}**

채널에 대한 통계 데이터의 콜렉션을 제어합니다.

**public int CharacterSet {get;}**

큐 관리자에 대한 코드화 문자 세트 ID(CCSID)를 리턴합니다. CharacterSet는 큐 관리자가 애플리케이션 프로그래밍 인터페이스의 모든 문자열 필드에 사용합니다.

**public int ClusterSenderMonitoring {get;}**

자동 정의 클러스터 송신자 채널에 대한 온라인 모니터링 데이터 콜렉션을 제어합니다.

**public int ClusterSenderStatistics {get;}**

자동 정의 클러스터 송신자 채널에 대한 통계 데이터 콜렉션을 제어합니다.

**public int ClusterWorkLoadMRU {get;}**

아웃바운드 클러스터 채널의 최대 수.

**public int ClusterWorkLoadUseQ {get;}**

QMGR의 값을 지정하는 경우, MQQueue 특성, ClusterWorkLoadUseQ의 기본값.

**public int CommandEvent {get;}**

명령 이벤트의 생성 여부를 지정합니다.

**public string CommandInputQueueName {get;}**

큐 관리자에 정의된 명령 입력 큐의 이름을 리턴합니다. 애플리케이션이 명령을 이 큐로 송신할 수 있습니다 (관련 권한이 부여된 경우).

**public int CommandLevel {get;}**

큐 관리자의 함수 레벨을 나타냅니다. 특정 함수 레벨에 해당되는 함수 세트는 플랫폼에 따라 다릅니다. 특정 플랫폼에서는 모든 큐 관리자에 공통되는 최하위 기능 레벨의 함수를 지원하는 모든 큐 관리자를 사용할 수 있습니다.

**public int CommandLevel {get;}**

큐 관리자 시작 시 명령 서버가 자동으로 시작하는지 여부.

**public string DNSGroup {get;}**

더 이상 사용되지 않습니다.

**public int DNSWLM {get;}**

더 이상 사용되지 않습니다.

**public int IPAddressVersion {get;}**

채널 연결에 사용할 IP 프로토콜(IPv4 또는 IPv6).

**public boolean IsConnected {get;}**

isConnected의 값을 리턴합니다.

true이면, 큐 관리자에 대한 연결이 설정되었고 끊어진 것으로 인식되지 않습니다. IsConnected 호출에서 적극적으로 큐 관리자에 도달하려고 하지 않으므로 물리적 연결이 끊어질 수 있지만 IsConnected는 여전히 true를 리턴할 수 있습니다. IsConnected 상태는 메시지 넣기와 가져오기 같은 활동이 큐 관리자에서 수행될 때에만 업데이트됩니다.

false이면, 큐 관리자에 대한 연결이 설정되지 않았거나, 끊어졌거나, 연결 해제되었습니다.

**public int KeepAlive {get;}**

연결의 다른 쪽 끝이 여전히 사용 가능한지 검사하기 위해 TCP KEEPALIVE 기능을 사용하는지 여부를 지정합니다. 사용 불가능한 경우, 채널이 닫힙니다.

**public int ListenerTimer {get;}**

APPC 또는 TCP/IP 실패 후 IBM MQ가 리스너를 재시작하기 위해 시도하는 사이의 시간 간격(초)입니다.

**public int LoggerEvent {get;}**

로거 이벤트의 생성 여부.

**public string LU62ARMSuffix {get;}**

SYS1.PARMLIB의 APPCPM 멤버 접미부. 이 접미부는 이 채널 시작기의 LUADD를 지정합니다. 자동 재시작 관리자(ARM)가 채널 시작기를 재시작할 때 z/OS command SET APPC=xx가 실행됩니다.

**public string LUGroupName {get; z/os}**

큐 공유 그룹의 인바운드 전송을 핸들링하는 LU 6.2 리스너가 사용할 일반 LU 이름입니다.

**public string LUName {get;}**

아웃바운드 LU 6.2 전송에 사용할 LU 이름.

**public int MaximumActiveChannels {get;}**

언제든지 활성화될 수 있는 최대 채널 수.

**public int MaximumCurrentChannels {get;}**

현재 사용할 수 있는 최대 채널 수(연결된 클라이언트가 있는 서버 연결 채널 포함).

**public int MaximumLU62Channels {get;}**

LU 6.2 전송 프로토콜을 사용하는 현재 실행할 수 있는 채널 또는 연결 가능한 클라이언트의 최대 수.

**public int MaximumMessageLength {get;}**

큐 관리자가 처리할 수 있는 메시지의 최대 길이(바이트)를 리턴합니다. 최대 메시지 길이가 MaximumMessageLength를 초과하는 경우 큐를 정의할 수 없습니다.

**public int MaximumPriority {get;}**

큐 관리자에 지원되는 최대 메시지 우선순위를 리턴합니다. 우선순위의 범위는 0(최저점) ~ 이 값입니다. 큐 관리자와 연결을 끊은 후 이 메소드를 호출하면 MQException을 전달합니다.

**public int MaximumTCPChannels {get;}**

TCP/IP 전송 프로토콜을 사용하는 전송 가능한 채널 또는 연결할 수 있는 클라이언트의 최대 수.

**public int MQIAccounting {get;}**  
MQI 데이터에 대한 계정 정보 콜렉션을 제어합니다.

**public int MQIStatistics {get;}**  
큐 관리자에 대한 통계 모니터링 정보의 콜렉션을 제어합니다.

**public int OutboundPortMax {get;}**  
보내는 채널을 바인딩할 때 사용되는 포트 번호 범위에서 최대값.

**public int OutboundPortMin {get;}**  
보내는 채널을 바인딩할 때 사용되는 포트 번호 범위에서 최소값.

**public int QueueAccounting {get;}**  
클래스 3 계정(스레드 레벨 및 큐 레벨 계정) 데이터가 모든 큐에 사용되는지 여부.

**public int QueueMonitoring {get;}**  
큐에 대한 온라인 모니터링 데이터의 콜렉션을 제어합니다.

**public int QueueStatistics {get;}**  
큐에 대한 통계 데이터 콜렉션을 제어합니다.

**public int ReceiveTimeout {get;}**  
TCP/IP 채널이 비활성 상태로 돌아가기 전에 해당 파트너로부터 하트비트를 포함한 데이터를 수신하기 위해 대기하는 시간입니다.

**public int ReceiveTimeoutMin {get;}**  
TCP/IP 채널이 비활성 상태로 돌아가기 전에 상대방으로부터 하트비트를 포함하여 데이터를 수신하기 위해 대기하는 최소 시간.

**public int ReceiveTimeoutType {get;}**  
ReceiveTimeout의 값에 적용되는 규정자.

**public int SharedQueueQueueManagerName {get;}**  
메시지를 공유 큐로 전달하는 방법을 지정합니다. Put이 대상 큐 관리자와 동일한 큐 공유 그룹이 아닌 다른 큐 관리자를 지정하는 경우, 메시지가 다음 두 가지 방법으로 전달됩니다.

**MQC.MQSQM\_USE**  
공유 큐에 넣기 전에 메시지가 오브젝트 큐 관리자로 전달됩니다.

**MQCMQSQM\_IGNORE**  
메시지를 공유 큐에 직접적으로 넣습니다.

**public int SSLEvent {get;}**  
TLS 이벤트 생성 여부입니다.

**public int SSLFips {get;}**  
암호화 하드웨어가 아닌 IBM MQ에서 암호화가 수행되는 경우 FIPS 인증 알고리즘만 사용하는지 여부.

**public int SSLKeyResetCount {get;}**  
비밀 키를 재협상하기 전에 SSL 대화 안에서 송신 및 수신된 암호화되지 않은 바이트 수를 표시합니다.

**public int ClusterSenderStatistics {get;}**  
연속적인 통계 수집 사이의 분 단위 간격을 지정합니다.

**public int SyncpointAvailability {get;}**  
큐 관리자가 MQQueue.get 및 MQQueue.put 메소드를 사용하여 작업 단위와 동기점을 지원하는지 여부를 표시합니다.

**public string TCPName {get;}**  
TCPStackType 값에 따라 사용되는 유일한 또는 기본 TCP/IP 시스템의 이름.

**public int TCPStackType {get;}**  
채널 시작기가 TCPName에 지정된 TCP/IP 주소 공간만 사용하는지 여부를 지정합니다. 또는, 채널 시작기를 TCP/IP 주소에 바인딩할 수 있습니다.

**public int TraceRouteRecording {get;}**  
라우트 추적 정보의 기록을 제어합니다.

## 메소드

```
public MQProcess AccessProcess(string processName, int openOptions);  
public MQProcess AccessProcess(string processName, int openOptions, string  
queueManagerName, string alternateUserId);
```

MQException을 전달합니다.

이 큐 관리자의 IBM MQ 프로세스에 액세스하여 프로세스 속성에 대해 조사하십시오.

### processName

열려는 프로세스의 이름.

### openOptions

프로세스 열기를 제어하는 옵션. 비트 단위 OR을 사용하여 추가하거나 결합할 수 있는 유효한 옵션은 다음과 같습니다.

- MQC.MQOO\_FAIL\_IF QUIESCING
- MQC.MQOO\_INQUIRE
- MQC.MQOO\_SET
- MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY

### queueManagerName

프로세스가 정의된 큐 관리자의 이름. 큐 관리자가 프로세스에서 액세스하는 것과 동일하면 큐 관리자 이름을 공백으로 두거나 널로 지정할 수 있습니다.

### alternateUserId

MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY가 **openOptions** 매개변수에 지정되면, *alternateUserId*는 조치의 권한을 확인하는 데 사용되는 대체 사용자 ID를 지정합니다.

MQOO\_ALTERNATE\_USER\_AUTHORITY가 지정되지 않으면, *alternateUserId*는 공백이거나 널일 수 있습니다.

MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY를 지정하지 않은 경우, 연결에 기본 사용자 권한이 사용됩니다.

```
public MQQueue AccessQueue(string queueName, int openOptions);  
public MQQueue AccessQueue(string queueName, int openOptions, string  
queueManagerName, string dynamicQueueName, string alternateUserId);
```

MQException을 전달합니다.

이 큐 관리자의 큐에 액세스합니다.

메시지를 가져오거나 찾아보고, 메시지를 넣고, 큐 속성에 대해 조회하거나, 큐의 속성을 설정할 수 있습니다. 이름 지정된 큐가 모델 큐이면 동적 로컬 큐가 작성됩니다. 결과적인 MQQueue 오브젝트의 name 속성을 조회하여 동적 큐의 이름을 확인합니다.

### queueName

열려는 큐의 이름.

### openOptions

큐 열기를 제어하는 옵션.

#### MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY

지정된 사용자 ID로 유효성 검증합니다.

#### MQC.MQOO\_BIND\_AS\_QDEF

큐의 기본 바인딩을 사용합니다.

#### MQC.MQOO\_BIND\_NOT\_FIXED

특정 목적지에 바인딩하지 않습니다.

#### MQC.MQOO\_BIND\_ON\_OPEN

큐를 열 때 핸들을 목적지에 바인딩합니다.

**MQC.MQOO\_BROWSE**

메시지를 열어 찾아봅니다.

**MQC.MQOO\_FAIL\_IF QUIESCING**

큐 관리자가 정지 중인 경우 실패합니다.

**MQC.MQOO\_INPUT\_AS\_Q\_DEF**

큐 정의 기본값을 사용하여 메시지를 열어 가져옵니다.

**MQC.MQOO\_INPUT\_SHARED**

공유 액세스 권한으로 메시지를 열어 가져옵니다.

**MQC.MQOO\_INPUT\_EXCLUSIVE**

독점 액세스 권한으로 메시지를 열어 가져옵니다.

**MQC.MQOO\_INQUIRE**

조회를 위한 열기로, 특성을 조회하려는 경우 필수입니다.

**MQC.MQOO\_OUTPUT**

메시지를 열어 넣습니다.

**MQC.MQOO\_PASS\_ALL\_CONTEXT**

모든 컨텍스트의 전달을 허용합니다.

**MQC.MQOO\_PASS\_IDENTITY\_CONTEXT**

ID 컨텍스트의 전달을 허용합니다.

**MQC.MQOO\_SAVE\_ALL\_CONTEXT**

메시지가 검색되면 컨텍스트를 저장합니다.

**MQC.MQOO\_SET**

속성 설정을 위한 열기로, 특성을 설정하려는 경우 필수입니다.

**MQC.MQOO\_SET\_ALL\_CONTEXT**

모든 컨텍스트를 설정할 수 있습니다.

**MQC.MQOO\_SET\_IDENTITY\_CONTEXT**

ID 컨텍스트를 설정할 수 있습니다.

**queueManagerName**

큐가 정의된 큐 관리자의 이름. 완전히 공백이거나 널인 이름은 MQQueueManager 오브젝트가 연결되는 큐 관리자를 나타냅니다.

**dynamicQueueName**

queueName이 모델 큐의 이름을 지정하는 경우를 제외하고, *dynamicQueueName*은 무시됩니다. 해당 경우 *dynamicQueueName*은 작성될 동적 큐의 이름을 지정합니다. queueName이 모델 큐의 이름을 지정하면 공백 또는 널 이름은 올바르지 않습니다. 이름에서 공백이 아닌 마지막 문자가 별표 (\*) 인 경우 큐 관리자는 별표를 문자열로 대체합니다. 이 문자는 큐에 생성된 이름이 이 큐 관리자에서 고유하도록 보장합니다.

**alternateUserId**

MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY가 openOptions 매개변수에 지정되면, *alternateUserId*는 열기의 권한을 확인하는 데 사용되는 대체 사용자 ID를 지정합니다.

MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY가 지정되지 않으면, *alternateUserId*는 공백이거나 널일 수 있습니다.

```

public MQTopic AccessTopic( MQDestination destination, string topicName, string
topicObject, int options);
public MQTopic AccessTopic( MQDestination destination, string topicName, string
topicObject, int options, string alternateUserId);
public MQTopic AccessTopic( MQDestination destination, string topicName, string
topicObject, int options, string alternateUserId, string subscriptionName);
public MQTopic AccessTopic( MQDestination destination, string topicName, string
topicObject, int options, string alternateUserId, string subscriptionName,
System.Collections.Hashtable properties);
public MQTopic AccessTopic(string topicName, string topicObject, int openAs,
int options);
public MQTopic AccessTopic(string topicName, string topicObject, int openAs,
int options, string alternateUserId);
public MQTopic AccessTopic(string topicName, string topicObject, int options,
string alternateUserId, string subscriptionName);
public MQTopic AccessTopic(string topicName, string topicObject, int options,
string alternateUserId, string subscriptionName, System.Collections.Hashtable
properties);

```

이 큐 관리자에 대한 토픽에 액세스합니다.

MQTopic 오브젝트는 때때로 토픽 오브젝트라고 하는 관리 토픽 오브젝트와 밀접하게 관련이 있습니다. 입력 시, *topicObject*는 관리 토픽 오브젝트를 가리킵니다. MQTopic 구성자는 토픽 오브젝트에서 토픽 문자열을 가져오고 이를 *topicName*과 결합하여 토픽 이름을 작성합니다. *topicObject*와 *topicName* 중 하나 또는 전체가 널일 수 있습니다. 토픽 이름은 토픽 트리와 일치하며, 가장 가깝게 일치하는 관리 토픽 오브젝트의 이름이 *topicObject*에 리턴됩니다.

MQTopic 오브젝트와 연관된 토픽은 두 개의 토픽 문자열을 결합한 결과입니다. 첫 번째 토픽 문자열은 *topicObject*로 식별된 관리 토픽 오브젝트에 의해 정의됩니다. 두 번째 토픽 문자열은 *topicString*입니다. MQTopic 오브젝트와 연관된 결과 토픽 문자열은 와일드카드를 포함하여 여러 토픽을 식별할 수 있습니다.

발행 또는 구독을 위해 토픽을 열었는지에 따라 MQTopic.Put 메소드를 사용하여 토픽에 대해 발행하거나 MQTopic.Get 메소드를 사용하여 토픽 발행물을 수신할 수 있습니다. 동일한 토픽을 발행 및 구독하려는 경우, 발행과 구독을 위해 각각 한 번씩 두 번 토픽에 액세스해야 합니다.

MQDestination 오브젝트를 제공하지 않고 구독용 MQTopic 오브젝트를 작성하는 경우, 관리되는 구독이 사용됩니다. 큐를 MQDestination 오브젝트로 전달하는 경우에는 관리되지 않는 구독이 사용됩니다. 설정한 구독 옵션이 관리 또는 관리되지 않는 구독과 일치하는지 확인해야 합니다.

#### **destination**

*destination*은 MQQueue 인스턴스입니다. *destination*을 제공하면 MQTopic이 관리되지 않는 구독으로 열립니다. 토픽에 대한 발행물은 *destination*로 액세스되는 큐에 전달됩니다.

#### **topicName**

토픽 이름의 두 번째 부분인 토픽 문자열. *topicName*은 *topicObject* 관리 토픽 오브젝트에 정의된 토픽 문자열과 연결되어 있습니다. *topicName*을 널로 설정할 수 있는데, 이 경우 토픽 이름은 *topicObject*의 토픽 문자열을 통해 정의됩니다.

#### **topicObject**

입력 시, *topicObject*는 토픽 이름의 첫 번째 부분을 형성하는 토픽 문자열이 포함된 토픽 오브젝트의 이름입니다. *topicObject*의 토픽 문자열은 *topicName*과 연결되어 있습니다. 토픽 문자열 구성 규칙은 [토픽 문자열 결합](#)에 정의되어 있습니다.

출력 시, *topicObject*는 토픽 트리에서 토픽 문자열로 식별된 토픽에 가장 가깝게 일치하는 관리 토픽 오브젝트의 이름을 포함합니다.

#### **openAs**

발행하거나 구독하기 위한 토픽에 액세스합니다. 이 매개변수는 이러한 옵션의 하나만 포함할 수 있습니다.

- MQC.MQTOPIC\_OPEN\_AS\_SUBSCRIPTION

- MQC.MQTOPIC\_OPEN\_AS\_PUBLICATION

### 옵션

발행 또는 구독을 위해 토픽의 열림을 제어하는 옵션을 결합합니다. 구독할 토픽에 액세스하려면 MQC.MQSO\_\* 상수를 사용하고, 발행할 토픽에 액세스하려면 MQC.MQOO\_\* 상수를 사용하십시오.

둘 이상의 옵션이 필요한 경우, 값을 함께 추가하거나 비트 단위의 OR 연산자를 사용하여 옵션 값을 결합하십시오.

### alternateUserId

조작 완료에 필요한 권한을 확인하는 데 사용되는 대체 사용자 ID를 지정합니다.

MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY 또는 MQC.MQSO\_ALTERNATE\_USER\_AUTHORITY가 옵션 매개변수를 설정되면 *alternateUserId*를 지정해야 합니다.

### subscriptionName

옵션 MQC.MQSO\_DURABLE 또는 MQC.MQSO\_ALTER가 제공되는 경우 *subscriptionName*이 필수입니다. 두 경우 모두 MQTopic은 암시적으로 구독을 위해 열려 있습니다. MQC.MQSO\_DURABLE이 설정되면 구독이 존재하거나 MQC.MQSO\_ALTER가 설정되고 구독이 존재하지 않으면 예외가 전달됩니다.

### 특성

해시 테이블을 사용하여 나열된 특수 구독 특성을 설정합니다. 해시 테이블에 있는 지정된 항목이 출력 값으로 업데이트됩니다. 출력 값을 보고하기 위해 항목이 해시 테이블에 추가되지는 않습니다.

- MQC.MQSUB\_PROP\_ALTERNATE\_SECURITY\_ID
- MQC.MQSUB\_PROP\_SUBSCRIPTION\_EXPIRY
- MQC.MQSUB\_PROP\_SUBSCRIPTION\_USER\_DATA
- MQC.MQSUB\_PROP\_SUBSCRIPTION\_CORRELATION\_ID
- MQC.MQSUB\_PROP\_PUBLICATION\_PRIORITY
- MQC.MQSUB\_PROP\_PUBLICATION\_ACCOUNTING\_TOKEN
- MQC.MQSUB\_PROP\_PUBLICATION\_APPLICATIONID\_DATA

### public MQAsyncStatus GetAsyncStatus();

MQException을 전달합니다.

큐 관리자 연결을 위한 비동기 활동을 나타내는 MQAsyncStatus 오브젝트를 리턴합니다.

### public void Backout();

MQException을 전달합니다.

마지막 동기점 이후 동기점 내에서 읽거나 쓰여진 메시지를 백아웃합니다.

MQC.MQPMO\_SYNCPOINT 플래그가 설정된 상태로 작성된 메시지는 큐에서 제거됩니다.

MQC.MQGMO\_SYNCPOINT 플래그를 사용하여 읽은 메시지는 원래 있던 큐에서 복원됩니다. 메시지가 지속적이면 변경사항이 로그됩니다.

재연결 가능한 클라이언트의 경우, 재연결에 성공한 후 MQRC\_NONE 이유 코드가 클라이언트로 리턴됩니다.

### public void Begin();

MQException을 전달합니다.

Begin은 서버 바인딩 모드에서만 지원됩니다. 글로벌 작업 단위를 시작합니다.

### public void Commit();

MQException을 전달합니다.

마지막 동기점 이후 동기점 내에서 읽거나 쓰여진 메시지를 커밋합니다.

MQC.MQPMO\_SYNCPOINT 플래그 세트로 작성된 메시지는 다른 애플리케이션에서 사용할 수 있습니다. MQC.MQGMO\_SYNCPOINT 플래그 세트로 검색된 메시지가 삭제됩니다. 메시지가 지속적이면 변경사항이 로깅됩니다.

다음 이유 코드가 다시 연결 가능한 클라이언트에 리턴됩니다.

- 커밋 호출을 수행하는 동안 연결이 끊어진 경우 MQRC\_CALL\_INTERRUPTED.
- 커밋 호출이 다시 연결된 후에 실행되는 경우 MQRC\_BACKED\_OUT.

### **Disconnect();**

MQException을 전달합니다.

큐 관리자에 대한 연결을 닫습니다. 이 큐 관리자에서 액세스하는 모든 오브젝트는 더 이상 이 애플리케이션에서 액세스할 수 없습니다. 오브젝트에 다시 액세스하려면 MQQueueManager 오브젝트를 작성하십시오.

일반적으로, 작업 단위의 일부로 수행된 작업은 커밋됩니다. 그러나 작업 단위가 .NET에 의해 관리되는 경우에는 작업 단위가 롤백될 수 있습니다.

```
public void Put(int type, string destinationName, MQMessage message);  
public void Put(int type, string destinationName, MQMessage message  
MQPutMessageOptions putMessageOptions);  
public void Put(int type, string destinationName, string queueManagerName,  
string topicString, MQMessage message);  
public void Put(string queueName, MQMessage message);  
public void Put(string queueName, MQMessage message, MQPutMessageOptions  
putMessageOptions);  
public void Put(string queueName, string queueManagerName, MQMessage message);  
public void Put(string queueName, string queueManagerName, MQMessage message,  
MQPutMessageOptions putMessageOptions);  
public void Put(string queueName, string queueManagerName, MQMessage message,  
MQPutMessageOptions putMessageOptions, string alternateUserId);
```

MQException을 전달합니다.

먼저 MQQueue 또는 MQTopic 오브젝트를 작성하지 않고 단일 메시지를 큐 또는 토픽에 놓습니다.

#### **queueName**

메시지를 놓을 큐의 이름.

#### **destinationName**

목적지 오브젝트의 이름. *type*의 값에 따라 큐 또는 토픽 중 하나입니다.

#### **유형**

목적지 오브젝트의 유형. 옵션은 결합하지 않아야 합니다.

#### **MQC.MQOT\_Q**

큐

#### **MQC.MQOT\_TOPIC**

토픽

#### **queueManagerName**

큐가 정의된 큐 관리자 또는 큐 관리자 알리어스의 이름. MQC.MQOT\_TOPIC 유형이 지정되면 이 매개변수는 무시됩니다.

큐가 모델 큐이고 해석된 큐 관리자 이름이 이 큐 관리자가 아니면 MQException이 전달됩니다.

#### **topicString**

*topicString*은(는) *destinationName* 토픽 오브젝트의 토픽 이름과 결합됩니다.

*destinationName*이(가) 큐인 경우 *topicString*은(는) 무시됩니다.

#### **메시지**

송신하는 메시지. 메시지는 입출력(I/O) 오브젝트입니다.



다음 이유 코드가 다시 연결 가능한 클라이언트에 리턴됩니다.

- 지속 메시지에 대한 넣기 호출을 수행하는 동안 연결이 끊어진 경우 MQRC\_CALL\_INTERRUPTED .
- 비지속 메시지에 대한 넣기 호출을 수행하는 동안 연결에 성공한 경우 MQRC\_NONE([애플리케이션 복구 참조](#)).

### putMessageOptions

Put 조치를 제어하는 옵션.

*putMessageOptions*을(를) 생략하면 *putMessageOptions*의 기본 인스턴스가 작성됩니다.  
*putMessageOptions*은(는) 입출력(I/O) 오브젝트입니다.

reconnectable 클라이언트에서 MQPMO\_LOGICAL\_ORDER 옵션을 사용하는 경우 MQRC\_RECONNECT\_INCOMPATIBLE 이유 코드가 리턴됩니다.

### alternateUserId

메시지를 큐에 놓을 때 권한 확인에 사용하는 대체 사용자 ID를 지정합니다.

*putMessageOptions*에 MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY을(를) 설정하지 않은 경우 *alternateUserId*을(를) 생략할 수 있습니다. MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY을(를) 설정하면 *alternateUserId*도 설정해야 합니다. MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY도 설정하지 않으면 *alternateUserId*은(는) 효과가 없습니다..

## 구성자

```
public MQQueueManager();  
public MQQueueManager(string queueManagerName);  
public MQQueueManager(string queueManagerName, Int options);  
public MQQueueManager(string queueManagerName, Int options, string channel,  
string connName);  
public MQQueueManager(string queueManagerName, string channel, string  
connName);  
public MQQueueManager(string queueManagerName, System.Collections.Hashtable  
properties);
```

MQException을 전달합니다.

큐 관리자에 대한 연결을 작성합니다. 클라이언트 연결 또는 서버 연결 작성 중에서 선택합니다.

큐 관리자에 연결을 시도할 때 큐 관리자에 대한 조회(inq) 권한이 있어야 합니다. 조회 권한이 없으면 연결에 실패합니다.

다음 조건 중 하나가 충족되면 클라이언트 연결이 작성됩니다.

1. *channel* 또는 *connName*이(가) 생성자에 지정되어 있습니다.
2. *HostName*, *Port* 또는 *Channel*이(가) *properties*에 지정되어 있습니다.
3. *MQEnvironment.HostName*, *MQEnvironment.Port* 또는 *MQEnvironment.Channel*이(가) 지정되어 있습니다.

연결 특성의 값은 표시된 순서대로 기본 설정됩니다. 생성자의 *channel* 및 *connName*이(가) 생성자의 특성 값보다 우선합니다. 구성자 특성 값이 MQEnvironment 특성보다 우선합니다.

호스트 이름, 채널 이름 및 포트는 MQEnvironment 클래스에 정의됩니다.

### queueManagerName

연결할 큐 관리자 또는 큐 관리자 그룹의 이름.

기본 큐 관리자를 선택하려면 매개변수를 생략하거나, 널 또는 공백으로 두십시오. 서버의 기본 큐 관리자가 해당 서버의 기본 큐 관리자에 연결됩니다. 클라이언트 연결의 기본 큐 관리자는 리스너가 연결되는 큐 관리자에 연결됩니다.

## 옵션

MQCNO 연결 옵션을 지정하십시오. 이 값은 설정하는 연결의 유형에 적용할 수 있어야 합니다. 예를 들어, 클라이언트 연결에 대해 다음 서버 연결 특성을 지정하면 MQException이 전달됩니다.

- MQC.MQCNO\_FASTPATH\_BINDING
- MQC.MQCNO\_STANDARD\_BINDING

## 특성

특성 매개변수는 MQEnvironment에 의해 설정된 특성을 대체하는 일련의 키/값 쌍을 가져옵니다. 예 (1696 페이지의 『MQEnvironment 특성 대체』)를 참조하십시오. 대체할 수 있는 특성은 다음과 같습니다.

- MQC.CONNECT\_OPTIONS\_PROPERTY
- MQC.CONNECTION\_NAME\_PROPERTY
- MQC.ENCRYPTION\_POLICY\_SUITE\_B
- MQC.HOST\_NAME\_PROPERTY
- MQC.PORT\_PROPERTY
- MQC.CHANNEL\_PROPERTY
- MQC.SSL\_CIPHER\_SPEC\_PROPERTY
- MQC.SSL\_PEER\_NAME\_PROPERTY
- MQC.SSL\_CERT\_STORE\_PROPERTY
- MQC.SSL\_CRYPTOHARDWARE\_PROPERTY
- MQC.SECURITY\_EXIT\_PROPERTY
- MQC.SECURITY\_USERDATA\_PROPERTY
- MQC.SEND\_EXIT\_PROPERTY
- MQC.SEND\_USERDATA\_PROPERTY
- MQC.RECEIVE\_EXIT\_PROPERTY
- MQC.RECEIVE\_USERDATA\_PROPERTY
- MQC.USER\_ID\_PROPERTY
- MQC.PASSWORD\_PROPERTY
- MQC.MQAIR\_ARRAY
- MQC.KEY\_RESET\_COUNT
- MQC.FIPS\_REQUIRED
- MQC.HDR\_CMP\_LIST
- MQC.MSG\_CMP\_LIST
- MQC.TRANSPORT\_PROPERTY

## 채널

서버 연결 채널의 이름

### connName

*HostName* (포트) 형식의 연결 이름입니다.

CONNECTION\_NAME\_PROPERTY를 사용하여 *hostnames* 및 *ports* 목록을 구성자 MQQueueManager (String queueManagerName, Hashtable properties)에 대한 인수로 제공할 수 있습니다.

예를 들면, 다음과 같습니다.

```
ConnectionName = "fred.mq.com(2344),nick.mq.com(3746),tom.mq.com(4288)";  
Hashtable Properties=new Hashtable();
```

```
properties.Add(MQC.CONNECTION_NAME_PROPERTY, ConnectionName);
MQQueueManager qmgr=new MQQueue Manager("qmgrname", properties);
```

연결을 시도할 때 연결 이름 목록이 순서대로 처리됩니다. 첫 번째 호스트 이름 및 포트에 연결하려고 했으나 실패하면 속성의 두 번째 쌍에 연결하려고 시도합니다. 연결에 성공하거나 목록을 소진할 때까지 클라이언트가 이 프로세스를 반복합니다. 목록이 소진되면 적절한 이유 코드와 완료 코드가 클라이언트 애플리케이션으로 리턴됩니다.

연결 이름에 포트 번호가 제공되지 않은 경우 기본 포트(mqclient.ini에 구성됨)가 사용됩니다.

## 연결 목록 설정

자동 클라이언트 다시 연결 옵션을 설정할 때 다음 메소드를 사용하여 연결 목록을 설정할 수 있습니다.

### MQSERVER를 통해 연결 목록 설정

명령 프롬프트를 통해 연결 목록을 설정할 수 있습니다.

명령 프롬프트에서 다음 명령을 설정합니다.

```
MQSERVER=SYSTEM.DEF.SVRCONN/TCP/Hostname1(Port1),Hostname2(Por2),Hostname3(Port3)
```

예를 들면, 다음과 같습니다.

```
MQSERVER=SYSTEM.DEF.SVRCONN/TCP/fred.mq.com(5266),nick.mq.com(6566),jack.mq.com(8413)
```

MQSERVER에서 연결을 설정하는 경우, 애플리케이션에 설정하지 마십시오.

애플리케이션에 연결 목록을 설정하는 경우, MQSERVER 환경 변수에 무엇을 설정하든 애플리케이션에서 덮어씁니다.

### 애플리케이션을 통해 연결 목록 설정

호스트 이름과 포트 특성을 지정하여 애플리케이션에서 연결 목록을 설정할 수 있습니다.

```
String connName = "fred.mq.com(2344), nick.mq.com(3746), chris.mq.com(4288)";
MQQueueManager qm = new MQQueueManager("QM1", "TestChannel", connName);
```

### app.config을(를) 통해 연결 목록을 설정하십시오.

App.config은(는) 키-값 쌍을 지정하는 XML 파일입니다.

연결 목록에 다음을 지정하십시오.

```
<app.Settings>
<add key="Connection1" value="Hostname1(Port1)"/>
<add key="Connection2" value="Hostname2(Port2)"/>
</app.Settings>
```

예를 들면, 다음과 같습니다.

```
<app.Settings>
<add key="Connection1" value="fred.mq.com(2966)"/>
<add key="Connection2" value="alex.mq.com(6533)"/>
</app.Settings>
```

app.config 파일에서 연결 목록을 직접 변경할 수 있습니다.

### MQEnvironment을(를) 통해 연결 목록을 설정하십시오.

MQEnvironment를 통해 연결 목록을 설정하려면 *ConnectionName* 특성을 사용하십시오.

```
MQEnvironment.ConnectionName = "fred.mq.com(4288),alex.mq.com(5211);
```

*ConnectionName* 특성은 MQEnvironment에 설정된 호스트 이름과 포트 특성을 덮어씁니다.

## 클라이언트 연결 작성

다음 예는 큐 관리자에 대한 클라이언트 연결을 작성하는 방법을 보여줍니다. 새 `MQQueueManager` 오브젝트를 작성하기 전에 `MQEnvironment` 변수를 설정하여 클라이언트 연결을 작성할 수 있습니다.

```
MQEnvironment.Hostname = "fred.mq.com"; // host to connect to
MQEnvironment.Port     = 1414;         // port to connect to
                                   // If not explicitly set,
                                   // defaults to 1414
                                   // (the default IBM MQ port)
MQEnvironment.Channel  = "channel.name"; // the case sensitive
                                   // name of the
                                   // SVR CONN channel on
                                   // the queue manager
MQQueueManager qMgr    = new MQQueueManager("MYQM");
```

그림 11. 클라이언트 연결

## MQEnvironment 특성 대체

다음 예는 해시 테이블에 정의된 사용자 ID 및 비밀번호로 큐 관리자를 작성하는 방법을 보여줍니다.

```
Hashtable properties = new Hashtable();

properties.Add( MQC.USER_ID_PROPERTY, "ExampleUserId" );
properties.Add( MQC.PASSWORD_PROPERTY, "ExamplePassword" );

try
{
    MQQueueManager qMgr = new MQQueueManager("qmgrname", properties);
}
catch (MQException mqe)
{
    System.Console.WriteLine("Connect failed with " + mqe.Message);
    return((int)mqe.Reason);
}
```

그림 12. `MQEnvironment` 특성 대체

## 다시 연결 가능한 연결 작성

다음 예는 클라이언트를 큐 관리자에 자동으로 다시 연결하는 방법을 보여줍니다.

```
Hashtable properties = new Hashtable(); // The queue manager name and the
                                   // properties how it has to be connected

properties.Add(MQC.CONNECT_OPTIONS_PROPERTY, MQC.MQCNO_RECONNECT); // Options
                                   // through which reconnection happens

properties.Add(MQC.CONNECTION_NAME_PROPERTY, "fred.mq.com(4789),nick.mq.com(4790)"); // The list
                                   // of queue managers through which reconnection happens

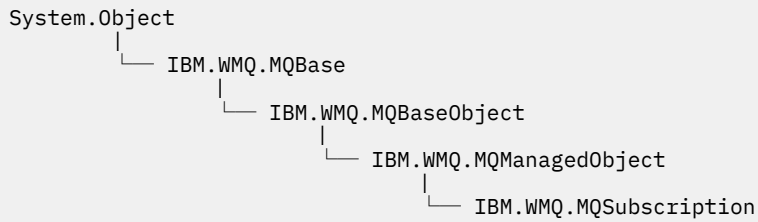
MQ QueueManager qmgr = new MQQueueManager("qmgrname", properties);
```

그림 13. 클라이언트를 큐 관리자에 자동으로 다시 연결

## MQSubscription.NET 클래스

보유된 발행물을 구독자에게 송신하도록 요청하려면 `MQSubscription`을 사용하십시오. `MQSubscription`은 구독을 위해 연 `MQTopic` 오브젝트의 특성입니다.

### 클래스



```
public class IBM.WMQ.MQSubscription extends IBM.WMQ.MQManagedObject;
```

- [1697 페이지의 『특성』](#)
- [1697 페이지의 『메소드』](#)
- [1697 페이지의 『구성자』](#)

## 특성

MQManagedObject 클래스를 사용하여 구독 특성에 액세스합니다. [1657 페이지의 『특성』](#)의 내용을 참조하십시오.

## 메소드

MQManagedObject 클래스를 사용하여 구독 Inquire, Set 및 Get 메소드에 액세스합니다. [1658 페이지의 『메소드』](#)의 내용을 참조하십시오.

```
public int RequestPublicationUpdate(int options);
```

MQException을 전달합니다.

현재 토픽에 대해 업데이트된 발행물을 요청하십시오. 토픽에 대한 보유된 발행물이 큐 관리자에 있으면 구독자에게 송신됩니다.

RequestPublicationUpdate를 호출하기 전에 구독 토픽을 열어 MQSubscription 오브젝트를 가져 오십시오.

일반적으로, MQC.MQSO\_PUBLICATIONS\_ON\_REQUEST 옵션과 함께 구독을 여십시오. 토픽 문자열에 와일드카드가 없으면 이 호출의 결과로 하나의 발행물만 송신된 것입니다. 토픽 문자열에 와일드카드가 있으면 많은 발행물을 송신할 수 있습니다. 메소드가 구독 큐로 송신한 보유된 발행물의 수를 리턴합니다. 특히 비지속 메시징인 경우 이렇게 많은 발행물의 수신이 보장되지 않습니다.

### 옵션

#### **MQC.MQSRO\_FAIL\_IF QUIESCING**

큐 관리자가 정지 상태인 경우 메소드가 실패합니다. z/OS에서 CICS 또는 IMS 애플리케이션에 대해 연결이 정지 상태이면 MQC.MQSRO\_FAIL\_IF QUIESCING에서 메소드를 강제 실패 처리합니다.

#### **MQC.MQSRO\_NONE**

지정된 옵션이 없습니다.

## 구성자

공용 구성자가 없습니다.

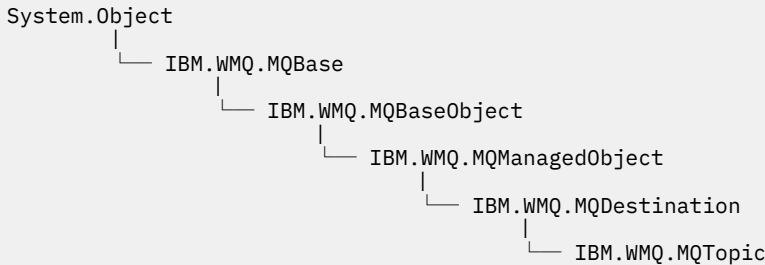
구독 개시된 MQTopic 오브젝트의 SubscriptionReference 특성에 MQSubscription 오브젝트가 리턴됩니다.

RequestPublicationUpdate 메소드를 호출하십시오. MQSubscription은 MQManagedObject의 서브 클래스입니다. MQManagedObject의 메소드와 특성에 액세스하려면 참조를 사용하십시오.

## MQTopic.NET 클래스

토픽의 메시지를 발행 또는 구독하거나 토픽의 속성을 조회 또는 설정하려면 MQTopic을 사용하십시오. 구성자나 MQQueueManager.AccessTopic 메소드를 사용하여 발행 또는 구독을 위한 MQTopic 오브젝트를 작성하십시오.

### 클래스



```
public class IBM.WMQ.MQTopic extends IBM.WMQ.MQDestination;
```

- [1698 페이지의 『특성』](#)
- [1698 페이지의 『메소드』](#)
- [1700 페이지의 『구성자』](#)

### 특성

특성을 가져올 때 전달되는 MQException에 대한 테스트.

```
public Boolean IsDurable {get;}
```

구독이 지속되면 True를, 그렇지 않으면 False를 리턴하는 읽기 전용 특성. 토픽이 발행을 위해 열려 있으면 해당 특성이 무시되고 항상 False를 리턴합니다.

```
public Boolean IsManaged {get;};
```

구독이 큐 관리자를 통해 관리되면 True를, 그렇지 않으면 False를 리턴하는 읽기 전용 특성. 토픽이 발행을 위해 열려 있으면 해당 특성이 무시되고 항상 False를 리턴합니다.

```
public Boolean IsSubscribed {get;};
```

토픽이 구독을 위해 열려 있으면 True를, 발행을 위해 열려 있으면 False를 리턴하는 읽기 전용 특성.

```
public MQSubscription SubscriptionReference {get;};
```

구독을 위해 열린 토픽 오브젝트와 연관된 MQSubscription 오브젝트를 리턴하는 읽기 전용 특성. 닫기 옵션을 수정하거나 오브젝트 메소드를 시작하려는 경우 참조를 사용할 수 있습니다.

```
public MQDestination UnmanagedDestinationReference {get;};
```

관리되지 않는 구독과 연관된 MQQueue를 리턴하는 읽기 전용 특성. 토픽 오브젝트를 작성할 때 지정된 목적지입니다. 이 특성은 발행을 위해 또는 관리되는 구독을 통해 열린 토픽 오브젝트에 대해 널을 리턴합니다.

### 메소드

```
public void Put(MQMessage message);
```

```
public void Put(MQMessage message, MQPutMessageOptions putMessageOptions);
```

MQException을 전달합니다.

메시지를 토픽에 발행합니다.

Put 호출이 완료된 후 MQMessage 오브젝트를 수정해도 IBM MQ 큐 또는 발행물 토픽의 실제 메시지에는 아무런 영향이 없습니다.

Put은 MQMessage 오브젝트의 MessageId 및 CorrelationId 특성을 업데이트하고 메시지 데이터를 지우지 않습니다. 추가적인 Put 또는 Get 호출은 MQMessage 오브젝트의 업데이트된 정보를 참조합니다. 예를 들면, 다음 코드 스니펫에서 첫 번째 메시지에 a와 두 번째 ab가 들어 있습니다.

```
msg.WriteString("a");
q.Put(msg, pmo);
msg.WriteString("b");
q.Put(msg, pmo);
```

## 메시지

메시지 디스크립터 데이터와 송신할 메시지를 포함한 MQMessage 오브젝트. 이 메소드의 결과로 메시지 디스크립터가 대체될 수 있습니다. 이 메소드가 완료된 직후 메시지 디스크립터의 값이 큐에 넣거나 토픽에 발행한 값입니다.

다음 이유 코드가 다시 연결 가능한 클라이언트에 리턴됩니다.

- 지속 메시지에서 Put 호출을 실행하는 동안 연결이 끊기고 다시 연결에 성공한 경우 MQRC\_CALL\_INTERRUPTED.
- 비지속 메시지에서 Put 호출을 실행하는 동안 연결에 성공한 경우 MQRC\_NONE(애플리케이션 복구 참조).

## putMessageOptions

Put 조치를 제어하는 옵션.

putMessageOptions를 지정하지 않으면 MQPutMessageOptions의 기본 인스턴스가 사용됩니다.

reconnectable 클라이언트에서 MQPMO\_LOGICAL\_ORDER 옵션을 사용하는 경우 MQRC\_RECONNECT\_INCOMPATIBLE 이유 코드가 리턴됩니다.

**참고:** 간편함과 성능을 위해 단일 메시지를 큐에 넣으려는 경우 MQQueueManager.Put 오브젝트를 사용하십시오. 이를 위해서는 MQQueue 오브젝트가 있어야 합니다.

```
public void Get(MQMessage message);  
public void Get(MQMessage message, MQGetMessageOptions getMessageOptions);  
public void Get(MQMessage message, MQGetMessageOptions getMessageOptions, int MaxMsgSize);
```

MQException을 전달합니다.

토픽에서 메시지를 검색합니다.

이 메소드는 가져오기를 수행하기 위해 MQGetMessageOptions의 기본 인스턴스를 사용합니다. 사용된 메시지 옵션은 MQGMO\_NOWAIT입니다.

가져오기에 실패하면 MQMessage 오브젝트가 변경되지 않습니다. 가져오기에 성공하면 MQMessage의 메시지 데이터 부분이 수신되는 메시지의 메시지 디스크립터 및 메시지 데이터로 대체됩니다.

특정 MQQueueManager 에서 IBM MQ 로의 모든 호출은 동기적입니다. 따라서, Get(대기 포함)을 수행하는 경우 동일한 MQQueueManager를 사용하는 다른 모든 스레드가 Get 호출이 완료될 때까지 추가 IBM MQ 호출하지 못하도록 차단됩니다. 동시에 IBM MQ에 액세스하기 위해 여러 개의 스레드가 필요한 경우, 각 스레드가 고유의 MQQueueManager 오브젝트를 작성해야 합니다.

## 메시지

메시지 디스크립터 및 리턴된 메시지 데이터를 포함합니다. 메시지 디스크립터의 필드 중 일부는 입력 매개변수입니다. MessageId 및 CorrelationId 입력 매개변수가 필요에 따라 설정되었는지 확인해야 합니다.

다시 연결 가능한 클라이언트는 다시 연결에 성공한 후 MQGM\_SYNCPOINT에 따라 수신된 메시지에 대해 이유 코드 MQRC\_BACKED\_OUT을 리턴합니다.

## getMessageOptions

Get 조치를 제어하는 옵션.

MQC.MQGMO\_CONVERT 옵션을 사용하면 1바이트 문자 코드에서 2바이트 코드로 변환할 때 이유 코드 MQC.MQRC\_CONVERTED\_STRING\_TOO\_BIG과 함께 예외가 발생할 수 있습니다. 이 경우, 메시지가 변환 없이 버퍼로 복사됩니다.

`getMessageOptions`가 지정되지 않은 경우, 사용된 메시지 옵션은 MQGMO\_NOWAIT입니다.

reconnectable 클라이언트에서 MQGMO\_LOGICAL\_ORDER 옵션을 사용하는 경우 MQRC\_RECONNECT\_INCOMPATIBLE 이유 코드가 리턴됩니다.

### MaxMsgSize

이 메시지 오브젝트가 수신하는 최대 메시지. 큐의 메시지가 이 크기보다 크면 다음 두 가지 중 하나가 발생합니다.

- MQGMO\_ACCEPT\_TRUNCATED\_MSG 플래그가 MQGetMessageOptions 오브젝트에 설정된 경우, 메시지에 최대한 많은 메시지 데이터가 입력됩니다. MQCC\_WARNING 완료 코드 및 MQRC\_TRUNCATED\_MSG\_ACCEPTED 이유 코드와 함께 예외가 전달됩니다.
- MQGMO\_ACCEPT\_TRUNCATED\_MSG 플래그가 설정되지 않은 경우, 메시지가 큐에 유지됩니다. MQCC\_WARNING 완료 코드 및 MQRC\_TRUNCATED\_MSG\_FAILED 이유 코드와 함께 예외가 전달됩니다.

`MaxMsgSize`를 지정하지 않으면 전체 메시지가 검색됩니다.

## 구성자

```
public MQTopic(MQQueueManager queueManager, MQDestination destination, string
topicName, string topicObject, int options);
public MQTopic(MQQueueManager queueManager, MQDestination destination, string
topicName, string topicObject, int options, string alternateUserId);
public MQTopic(MQQueueManager queueManager, MQDestination destination, string
topicName, string topicObject, int options, string alternateUserId, string
subscriptionName);
public MQTopic(MQQueueManager queueManager, MQDestination destination, string
topicName, string topicObject, int options, string alternateUserId, string
subscriptionName, System.Collections.Hashtable properties);
public MQTopic(MQQueueManager queueManager, string topicName, string
topicObject, int openAs, int options);
public MQTopic(MQQueueManager queueManager, string topicName, string
topicObject, int openAs, int options, string alternateUserId);
public MQTopic(MQQueueManager queueManager, string topicName, string
topicObject, int options, string alternateUserId, string subscriptionName);
public MQTopic(MQQueueManager queueManager, string topicName, string
topicObject, int options, string alternateUserId, string subscriptionName,
System.Collections.Hashtable properties);
```

`queueManager`의 토픽에 액세스합니다.

`MQTopic` 오브젝트는 때때로 토픽 오브젝트라고 하는 관리 토픽 오브젝트와 밀접하게 관련이 있습니다. 입력 시, `topicObject`는 관리 토픽 오브젝트를 가리킵니다. `MQTopic` 구성자는 토픽 오브젝트에서 토픽 문자열을 가져오고 이를 `topicName`과 결합하여 토픽 이름을 작성합니다. `topicObject`와 `topicName` 중 하나 또는 전체가 널일 수 있습니다. 토픽 이름은 토픽 트리와 일치하며, 가장 가깝게 일치하는 관리 토픽 오브젝트의 이름이 `topicObject`에 리턴됩니다.

`MQTopic` 오브젝트와 연관된 토픽은 두 개의 토픽 문자열을 결합한 결과입니다. 첫 번째 토픽 문자열은 `topicObject`로 식별된 관리 토픽 오브젝트에 의해 정의됩니다. 두 번째 토픽 문자열은 `topicString`입니다. `MQTopic` 오브젝트와 연관된 결과 토픽 문자열은 와일드카드를 포함하여 여러 토픽을 식별할 수 있습니다.

발행 또는 구독을 위해 토픽을 열었는지에 따라 `MQTopic.Put` 메소드를 사용하여 토픽에 대해 발행하거나 `MQTopic.Get` 메소드를 사용하여 토픽 발행물을 수신할 수 있습니다. 동일한 토픽을 발행 및 구독하려는 경우, 발행과 구독을 위해 각각 한 번씩 두 번 토픽에 액세스해야 합니다.



MQDestination 오브젝트를 제공하지 않고 구독용 MQTopic 오브젝트를 작성하는 경우, 관리되는 구독이 사용됩니다. 큐를 MQDestination 오브젝트로 전달하는 경우에는 관리되지 않는 구독이 사용됩니다. 설정한 구독 옵션이 관리 또는 관리되지 않는 구독과 일치하는지 확인해야 합니다.

### queueManager

토픽에 액세스하는 큐 관리자.

### destination

*destination*은 MQQueue 인스턴스입니다. *destination*을 제공하면 MQTopic이 관리되지 않는 구독으로 열립니다. 토픽에 대한 발행물은 *destination*로 액세스되는 큐에 전달됩니다.

### topicName

토픽 이름의 두 번째 부분인 토픽 문자열. *topicName*은 *topicObject* 관리 토픽 오브젝트에 정의된 토픽 문자열과 연결되어 있습니다. *topicName*을 널로 설정할 수 있는데, 이 경우 토픽 이름은 *topicObject*의 토픽 문자열을 통해 정의됩니다.

### topicObject

입력 시, *topicObject*는 토픽 이름의 첫 번째 부분을 형성하는 토픽 문자열이 포함된 토픽 오브젝트의 이름입니다. *topicObject*의 토픽 문자열은 *topicName*과 연결되어 있습니다. 토픽 문자열 구성 규칙은 [토픽 문자열 결합](#)에 정의되어 있습니다.

출력 시, *topicObject*는 토픽 트리에서 토픽 문자열로 식별된 토픽에 가장 가깝게 일치하는 관리 토픽 오브젝트의 이름을 포함합니다.

### openAs

발행하거나 구독하기 위한 토픽에 액세스합니다. 이 매개변수는 이러한 옵션의 하나만 포함할 수 있습니다.

- MQC.MQTOPIC\_OPEN\_AS\_SUBSCRIPTION
- MQC.MQTOPIC\_OPEN\_AS\_PUBLICATION

### 옵션

발행 또는 구독을 위해 토픽의 열림을 제어하는 옵션을 결합합니다. 구독할 토픽에 액세스하려면 MQC.MQSO\_\* 상수를 사용하고, 발행할 토픽에 액세스하려면 MQC.MQOO\_\* 상수를 사용하십시오.

둘 이상의 옵션이 필요한 경우, 값을 함께 추가하거나 비트 단위의 OR 연산자를 사용하여 옵션 값을 결합하십시오.

### alternateUserId

조작 완료에 필요한 권한을 확인하는 데 사용되는 대체 사용자 ID를 지정합니다.

MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY 또는 MQC.MQSO\_ALTERNATE\_USER\_AUTHORITY가 옵션 매개변수를 설정되면 *alternateUserId*를 지정해야 합니다.

### subscriptionName

옵션 MQC.MQSO\_DURABLE 또는 MQC.MQSO\_ALTER가 제공되는 경우 *subscriptionName*이 필수입니다. 두 경우 모두 MQTopic은 암시적으로 구독을 위해 열려 있습니다. MQC.MQSO\_DURABLE이 설정되면 구독이 존재하거나 MQC.MQSO\_ALTER가 설정되고 구독이 존재하지 않으면 예외가 전달됩니다.

### 특성

해시 테이블을 사용하여 나열된 특수 구독 특성을 설정합니다. 해시 테이블에 있는 지정된 항목이 출력 값으로 업데이트됩니다. 출력 값을 보고하기 위해 항목이 해시 테이블에 추가되지는 않습니다.

- MQC.MQSUB\_PROP\_ALTERNATE\_SECURITY\_ID
- MQC.MQSUB\_PROP\_SUBSCRIPTION\_EXPIRY
- MQC.MQSUB\_PROP\_SUBSCRIPTION\_USER\_DATA
- MQC.MQSUB\_PROP\_SUBSCRIPTION\_CORRELATION\_ID
- MQC.MQSUB\_PROP\_PUBLICATION\_PRIORITY
- MQC.MQSUB\_PROP\_PUBLICATION\_ACCOUNTING\_TOKEN
- MQC.MQSUB\_PROP\_PUBLICATION\_APPLICATIONID\_DATA

```

public MQTopic MQQueueManager.AccessTopic(MQDestination destination, string
topicName, string topicObject, int options);
public MQTopic MQQueueManager.AccessTopic(MQDestination destination, string
topicName, string topicObject, int options, string alternateUserId);
public MQTopic MQQueueManager.AccessTopic(MQDestination destination, string
topicName, string topicObject, int options, string alternateUserId, string
subscriptionName);
public MQTopic MQQueueManager.AccessTopic(MQDestination destination, string
topicName, string topicObject, int options, string alternateUserId, string
subscriptionName, System.Collections.Hashtable properties);
public MQTopic MQQueueManager.AccessTopic(string topicName, string topicObject,
int openAs, int options);
public MQTopic MQQueueManager.AccessTopic(string topicName, string topicObject,
int openAs, int options, string alternateUserId);
public MQTopic MQQueueManager.AccessTopic(string topicName, string topicObject,
int options, string alternateUserId, string subscriptionName);
public MQTopic MQQueueManager.AccessTopic(string topicName, string topicObject,
int options, string alternateUserId, string subscriptionName,
System.Collections.Hashtable properties);

```

이 큐 관리자에 대한 토픽에 액세스합니다.

MQTopic 오브젝트는 때때로 토픽 오브젝트라고 하는 관리 토픽 오브젝트와 밀접하게 관련이 있습니다. 입력 시, *topicObject*는 관리 토픽 오브젝트를 가리킵니다. MQTopic 구성자는 토픽 오브젝트에서 토픽 문자열을 가져오고 이를 *topicName*과 결합하여 토픽 이름을 작성합니다. *topicObject*와 *topicName* 중 하나 또는 전체가 널일 수 있습니다. 토픽 이름은 토픽 트리와 일치하며, 가장 가깝게 일치하는 관리 토픽 오브젝트의 이름이 *topicObject*에 리턴됩니다.

MQTopic 오브젝트와 연관된 토픽은 두 개의 토픽 문자열을 결합한 결과입니다. 첫 번째 토픽 문자열은 *topicObject*로 식별된 관리 토픽 오브젝트에 의해 정의됩니다. 두 번째 토픽 문자열은 *topicString*입니다. MQTopic 오브젝트와 연관된 결과 토픽 문자열은 와일드카드를 포함하여 여러 토픽을 식별할 수 있습니다.

발행 또는 구독을 위해 토픽을 열었는지에 따라 MQTopic.Put 메소드를 사용하여 토픽에 대해 발행하거나 MQTopic.Get 메소드를 사용하여 토픽 발행물을 수신할 수 있습니다. 동일한 토픽을 발행 및 구독하려는 경우, 발행과 구독을 위해 각각 한 번씩 두 번 토픽에 액세스해야 합니다.

MQDestination 오브젝트를 제공하지 않고 구독용 MQTopic 오브젝트를 작성하는 경우, 관리되는 구독이 사용됩니다. 큐를 MQDestination 오브젝트로 전달하는 경우에는 관리되지 않는 구독이 사용됩니다. 설정한 구독 옵션이 관리 또는 관리되지 않는 구독과 일치하는지 확인해야 합니다.

#### **destination**

*destination*은 MQQueue 인스턴스입니다. *destination*을 제공하면 MQTopic이 관리되지 않는 구독으로 열립니다. 토픽에 대한 발행물은 *destination*로 액세스되는 큐에 전달됩니다.

#### **topicName**

토픽 이름의 두 번째 부분인 토픽 문자열. *topicName*은 *topicObject* 관리 토픽 오브젝트에 정의된 토픽 문자열과 연결되어 있습니다. *topicName*을 널로 설정할 수 있는데, 이 경우 토픽 이름은 *topicObject*의 토픽 문자열을 통해 정의됩니다.

#### **topicObject**

입력 시, *topicObject*는 토픽 이름의 첫 번째 부분을 형성하는 토픽 문자열이 포함된 토픽 오브젝트의 이름입니다. *topicObject*의 토픽 문자열은 *topicName*과 연결되어 있습니다. 토픽 문자열 구성 규칙은 토픽 문자열 결합에 정의되어 있습니다.

출력 시, *topicObject*는 토픽 트리에서 토픽 문자열로 식별된 토픽에 가장 가깝게 일치하는 관리 토픽 오브젝트의 이름을 포함합니다.

#### **openAs**

발행하거나 구독하기 위한 토픽에 액세스합니다. 이 매개변수는 이러한 옵션의 하나만 포함할 수 있습니다.

- MQC.MQTOPIC\_OPEN\_AS\_SUBSCRIPTION
- MQC.MQTOPIC\_OPEN\_AS\_PUBLICATION

### 옵션

발행 또는 구독을 위해 토픽의 열림을 제어하는 옵션을 결합합니다. 구독할 토픽에 액세스하려면 MQC.MQSO\_\* 상수를 사용하고, 발행할 토픽에 액세스하려면 MQC.MQOO\_\* 상수를 사용하십시오.

둘 이상의 옵션이 필요한 경우, 값을 함께 추가하거나 비트 단위의 OR 연산자를 사용하여 옵션 값을 결합하십시오.

### alternateUserId

조작 완료에 필요한 권한을 확인하는 데 사용되는 대체 사용자 ID를 지정합니다.

MQC.MQOO\_ALTERNATE\_USER\_AUTHORITY 또는 MQC.MQSO\_ALTERNATE\_USER\_AUTHORITY가 옵션 매개변수를 설정되면 *alternateUserId*를 지정해야 합니다.

### subscriptionName

옵션 MQC.MQSO\_DURABLE 또는 MQC.MQSO\_ALTER가 제공되는 경우 *subscriptionName*이 필수입니다. 두 경우 모두 MQTopic은 암시적으로 구독을 위해 열려 있습니다. MQC.MQSO\_DURABLE이 설정되면 구독이 존재하거나 MQC.MQSO\_ALTER가 설정되고 구독이 존재하지 않으면 예외가 전달됩니다.

### 특성

해시 테이블을 사용하여 나열된 특수 구독 특성을 설정합니다. 해시 테이블에 있는 지정된 항목이 출력 값으로 업데이트됩니다. 출력 값을 보고하기 위해 항목이 해시 테이블에 추가되지는 않습니다.

- MQC.MQSUB\_PROP\_ALTERNATE\_SECURITY\_ID
- MQC.MQSUB\_PROP\_SUBSCRIPTION\_EXPIRY
- MQC.MQSUB\_PROP\_SUBSCRIPTION\_USER\_DATA
- MQC.MQSUB\_PROP\_SUBSCRIPTION\_CORRELATION\_ID
- MQC.MQSUB\_PROP\_PUBLICATION\_PRIORITY
- MQC.MQSUB\_PROP\_PUBLICATION\_ACCOUNTING\_TOKEN
- MQC.MQSUB\_PROP\_PUBLICATION\_APPLICATIONID\_DATA

## IMQObjectTrigger.NET 인터페이스

IMQObjectTrigger 를 구현하여 **runmqdmn.NET** 모니터에서 전달된 메시지를 처리하십시오.

### 인터페이스

```
public interface IBM.WMQMonitor.IMQObjectTrigger();
```

**runmqdmn** 명령에 동기점 제어가 지정되었는지 여부에 따라 **Execute** 메소드가 리턴되기 전이나 후에 메시지가 큐에서 제거됩니다.

### 메소드

```
void Execute (MQQueueManager queueManager, MQQueue queue, MQMessage message, string param);
```

#### queueManager

모니터링 대상 큐를 호스팅하는 큐 관리자.

#### 큐

모니터링 대상 큐.

#### 메시지

큐에서 읽은 메시지.

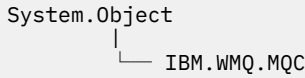
**param**

UserParameter에서 전달된 데이터.

## MQC.NET 인터페이스

상수 이름에 접두부 MQC. 를 붙여 MQI 상수를 참조하십시오. MQC는 MQI에 사용되는 모든 상수를 정의합니다.

### 인터페이스



```
public interface IBM.WMQ.MQC extends System.Object;
```

### 예

```
MQQueue queue;
queue.closeOptions = MQC.MQCO_DELETE;
```

## .NET 애플리케이션의 문자 세트 ID

.NET IBM MQ 메시지를 인코딩하기 위해 선택할 수 있는 문자 세트에 대한 설명입니다.

문자 세트	설명
37	ibm037
437	ibm437 / PC Original
500	ibm500
819	iso-8859-1 / latin1 / ibm819
1200	유니코드
1208	UTF-8
273	ibm273
277	ibm277
278	ibm278
280	ibm280
284	ibm284
285	ibm285
297	ibm297
420	ibm420
424	ibm424
737	ibm737 / PC Greek
775	ibm775 / PC Baltic
813	iso-8859-7 / greek / ibm813
838	ibm838

문자 세트	설명
850	ibm850 / PC Latin 1
852	ibm852 / PC Latin 2
855	ibm855 / PC Cyrillic
856	ibm856
857	ibm857 / PC Turkish
860	ibm860 / PC Portuguese
861	ibm861 / PC Icelandic
862	ibm862 / PC Hebrew
863	ibm863 / PC Canadian French
864	ibm864 / PC Arabic
865	ibm865 / PC Nordic
866	ibm866 / PC Russian
868	ibm868
869	ibm869 / PC Modern Greek
870	ibm870
871	ibm871
874	ibm874
875	ibm875
912	iso-8859-2 / latin2 / ibm912
913	iso-8859-3 / latin3 / ibm913
914	iso-8859-4 / latin4 / ibm914
915	iso-8859-5 / cyrillic / ibm915
916	iso-8859-8 / hebrew / ibm916
918	ibm918
920	iso-8859-9 / latin5 / ibm920
921	ibm921
922	ibm922
930	ibm930
932	PC Japanese
933	ibm933
935	ibm935
937	ibm937
939	ibm939
942	ibm942
943	ibm943

문자 세트	설명
948	ibm948
949	ibm949
950	ibm950 / Big 5 Traditional Chinese
954	EUCJIS
964	ibm964 / CNS 11643 Traditional Chinese
970	ibm970
1006	ibm1006
1025	ibm1025
1026	ibm1026
1089	iso-8859-6 / arabic / ibm1089
1097	ibm1097
1098	ibm1098
1112	ibm1112
1122	ibm1122
1123	ibm1123
1124	ibm1124
1250	Windows 라틴어 2
1251	Windows 키릴 문자
1252	Windows 라틴어 1
1253	Windows 그리스어
1254	Windows 터키어
1255	Windows 히브리어
1256	Windows 아랍어
1257	Windows 발트어
1258	Windows 베트남어
1381	ibm1381
1383	ibm1383
2022	JIS
5601	ksc-5601 Korean
33722	ibm33722

## IBM MQ C++ 클래스

IBM MQ C++ 클래스는 IBM MQ MQI(Message Queue Interface)를 캡슐화합니다. 이러한 클래스를 모두 포함하는 단일 C++ 헤더 파일, **imqi.hpp**가 있습니다.

각 클래스마다 다음 정보가 표시됩니다.

## 클래스 계층 다이어그램

클래스의 직속 상위 클래스(있는 경우)에 대한 상속 관계를 표시하는 클래스 다이어그램.

## 기타 관련 클래스

기타 관련 클래스에 대한 문서 링크(예: 상위 클래스) 및 메소드 서명에 사용된 오브젝트의 클래스.

## 오브젝트 속성

클래스의 속성. 이들은 상위 클래스에 정의된 속성에 추가됩니다. 많은 속성이 IBM MQ 데이터 구조 멤버를 반영합니다(1708 페이지의 『C++ 및 MQI 상호 참조』 참조). 자세한 정보는 768 페이지의 『오브젝트 속성』의 내용을 참조하십시오.

## 구성자

클래스의 오브젝트를 작성하는 데 사용되는 특수 메소드의 서명.

## 오브젝트 메소드(공용)

조작에 클래스 인스턴스를 필요로 하고 사용 제한이 없는 메소드의 서명.

적용되는 경우 다음 정보도 표시됩니다.

## 클래스 메소드(공용)

조작에 클래스 인스턴스를 필요로 하지 않고 사용 제한이 없는 메소드의 서명.

## 오버로드된(상위 클래스) 메소드

상위 클래스에 정의되어 있지만 이 클래스에 대해 서로 다른 다형성 동작을 보여주는 가상 메소드의 서명.

## 오브젝트 메소드(보호됨)

이 조작에 클래스의 인스턴스를 필요로 하고 도출된 클래스 구현에 사용하도록 예약된 메소드의 서명. 이 절은 클래스 사용자가 아닌 클래스 작성자만 관련이 있습니다.

## 오브젝트 데이터(보호됨)

도출된 클래스 구현에서 사용할 수 있는 오브젝트 인스턴스 데이터의 구현 세부사항. 이 절은 클래스 사용자가 아닌 클래스 작성자만 관련이 있습니다.

## 이유 코드

실패한 메소드에서 예상할 수 있는 MQRC\_\* 값 (API 완료 및 이유 코드 참조). 클래스 오브젝트에 대해 발생할 수 있는 이유 코드의 전체 목록은 상위 클래스 문서를 참조하십시오. 클래스의 문서화된 이유 코드 목록에 상위 클래스의 이유 코드는 포함되지 않습니다.

## 참고:

1. 이러한 클래스의 오브젝트는 스레드로부터 안전하지 않습니다. 따라서 최적의 성능을 보장하지만 둘 이상의 스레드에서 오브젝트에 액세스하지 않도록 주의하십시오.
2. 멀티스레드 프로그램의 경우 각 스레드별로 별도의 ImqQueueManager 오브젝트를 사용하는 것이 좋습니다. 각 관리자 오브젝트에는 다른 오브젝트에 대한 자체 독립 컬렉션이 있어야 하며 서로 다른 스레드의 오브젝트가 서로 격리됩니다.

클래스는 다음과 같습니다.

- [1722 페이지의 『ImqAuthenticationRecord C++ 클래스』](#)
- [1725 페이지의 『ImqBinary C++ 클래스』](#)
- [1726 페이지의 『ImqCache C++ 클래스』](#)
- [1729 페이지의 『ImqChannel C++ 클래스』](#)
- [1734 페이지의 『ImqCICSBridgeHeader C++ 클래스』](#)
- [1740 페이지의 『ImqDeadLetterHeader C++ 클래스』](#)
- [1743 페이지의 『ImqDistributionList C++ 클래스』](#)
- [1744 페이지의 『ImqError C++ 클래스』](#)
- [1745 페이지의 『ImqGetMessageOptions C++ 클래스』](#)
- [1748 페이지의 『ImqHeader C++ 클래스』](#)
- [1750 페이지의 『ImqIMSBridgeHeader C++ 클래스』](#)
- [1752 페이지의 『ImqItem C++ 클래스』](#)
- [1754 페이지의 『ImqMessage C++ 클래스』](#)

- [1760 페이지의 『ImqMessageTracker C++ 클래스』](#)
- [1763 페이지의 『ImqNamelist C++ 클래스』](#)
- [1765 페이지의 『ImqObject C++ 클래스』](#)
- [1770 페이지의 『ImqProcess C++ 클래스』](#)
- [1771 페이지의 『ImqPutMessageOptions C++ 클래스』](#)
- [1774 페이지의 『ImqQueue C++ 클래스』](#)
- [1784 페이지의 『ImqQueueManager C++ 클래스』](#)
- [1799 페이지의 『ImqReferenceHeader C++ 클래스』](#)
- [1801 페이지의 『ImqString C++ 클래스』](#)
- [1806 페이지의 『ImqTrigger C++ 클래스』](#)
- [1809 페이지의 『ImqWorkHeader C++ 클래스』](#)

## C++ 및 MQI 상호 참조

이 주제 컬렉션은 C++과 MQI의 상관 관계 정보를 제공합니다.

이 정보와 함께 [234 페이지의 『MQI에서 사용되는 데이터 유형』](#)도 읽어보십시오.

이 표는 MQI 데이터 구조와 C++ 클래스, 포함 파일의 관계를 보여줍니다. 다음 주제는 각 C++ 클래스에 대한 상호 참조 정보를 설명합니다. 이 상호 참조는 기본 IBM MQ 프로시저 인터페이스의 사용과 관련이 있습니다. ImqBinary, ImqDistributionList 및 ImqString 클래스는 이 범주에 속하는 속성이 없으므로 제외됩니다.

표 845. 데이터 구조, 클래스 및 포함 파일 상호 참조		
데이터 구조	클래스	포함 파일
MQAIR	ImqAuthenticationRecord	imqair.hpp
	ImqBinary	imqbin.hpp
	ImqCache	imqcac.hpp
MQCD	ImqChannel	imqchl.hpp
MQCIH	ImqCICSBridgeHeader	imqcih.hpp
MQDLH	ImqDeadLetterHeader	imqdlh.hpp
MQOR	ImqDistributionList	imqdst.hpp
	ImqError	imqerr.hpp
MQGMO	ImqGetMessageOptions	imqgmo.hpp
	ImqHeader	imqhdr.hpp
MQIIH	ImqIMSBridgeHeader	imqiih.hpp
	ImqItem	imqitm.hpp
MQMD	ImqMessage	imqmsg.hpp
	ImqMessageTracker	imqmtr.hpp
	ImqNamelist	imqnml.hpp
MQOD, MQRR	ImqObject	imqobj.hpp
MQPMO, MQPMR, MQRR	ImqPutMessageOptions	imqpmo.hpp
	ImqProcess	imqpro.hpp
	ImqQueue	imqque.hpp



표 845. 데이터 구조, 클래스 및 포함 파일 상호 참조 (계속)		
데이터 구조	클래스	포함 파일
MQBO, MQCNO, MQCSP	ImqQueueManager	imqmgr.hpp
MQRMH	ImqReferenceHeader	imqrfh.hpp
	ImqString	imqstr.hpp
MQTM	ImqTrigger	imqtrg.hpp
MQTMC		
MQTMC2	ImqTrigger	imqtrg.hpp
MQXQH		
MQWIH	ImqWorkHeader	imqwih.hpp

### ImqAuthenticationRecord 상호 참조

ImqAuthenticationRecord C++ 클래스에 대한 속성, 데이터 구조, 필드 및 호출의 상호 참조입니다.

표 846. 속성, 데이터 구조, 필드 및 호출			
속성	데이터 구조	필드	전화로 문의
연결 이름	MQAIR	AuthInfoConnName	MQCONN
비밀번호	MQAIR	LDAPPassword	MQCONN
유형	MQAIR	AuthInfoType	MQCONN
사용자 이름	MQAIR	LDAPUserNamePtr	MQCONN
	MQAIR	LDAPUserNameOffset	MQCONN
	MQAIR	LDAPUserNameLength	MQCONN

### ImqCache 상호 참조

ImqCache C++ 클래스에 대한 속성 및 호출의 상호 참조입니다.

표 847. 속성 및 호출	
속성	전화로 문의
자동 버퍼	MQGET
버퍼 길이	MQGET
버퍼 포인터	MQGET, MQPUT
데이터 길이	MQGET
데이터 오프셋	MQGET
데이터 포인터	MQGET
메시지 길이	MQGET, MQPUT

### ImqChannel 상호 참조

ImqChannel C++ 클래스에 대한 속성, 데이터 구조, 필드 및 호출의 상호 참조입니다.

표 848. 속성, 데이터 구조, 필드 및 호출			
속성	데이터 구조	필드	전화로 문의
배치 하트비트	MQCD	BatchHeartbeat	MQCONN
채널 이름	MQCD	ChannelName	MQCONN
연결 이름	MQCD	ConnectionName	MQCONN
	MQCD	ShortConnectionName	MQCONN
헤더 압축	MQCD	HdrCompList	MQCONN
하트비트 간격	MQCD	HeartbeatInterval	MQCONN
활성 상태 지속 간격	MQCD	KeepAliveInterval	MQCONN
로컬 주소	MQCD	LocalAddress	MQCONN
최대 메시지 길이	MQCD	MaxMsgLength	MQCONN
메시지 압축	MQCD	MsgCompList	MQCONN
모드 이름	MQCD	ModeName	MQCONN
비밀번호	MQCD	비밀번호	MQCONN
수신 엑시트 수	MQCD		MQCONN
수신 엑시트 이름	MQCD	ReceiveExit	MQCONN
	MQCD	ReceiveExitsDefined	MQCONN
	MQCD	ReceiveExitPtr	MQCONN
수신 사용자 데이터	MQCD	ReceiveUserData	MQCONN
	MQCD	ReceiveUserDataPtr	MQCONN
보안 엑시트 이름	MQCD	SecurityExit	MQCONN
보안 사용자 데이터	MQCD	SecurityUserData	MQCONN
송신 엑시트 수	MQCD		MQCONN
송신 엑시트 이름	MQCD	SendExit	MQCONN
	MQCD	SendExitsDefined	MQCONN
	MQCD	SendExitPtr	MQCONN
송신 사용자 데이터	MQCD	SendUserData	MQCONN
	MQCD	SendUserDataPtr	MQCONN
SSL CipherSpec	MQCD	sslCipherSpecification	MQCONN
SSL 클라이언트 인증 유형	MQCD	sslClientAuthentication	MQCONN
SSL 피어 이름	MQCD	sslPeerName	MQCONN
트랜잭션 프로그램 이름	MQCD	TpName	MQCONN
전송 유형	MQCD	TransportType	MQCONN
사용자 ID	MQCD	UserIdentifier	MQCONN

## ImqCICSBridgeHeader 상호 참조

ImqCICSBridgeHeader C++ 클래스에 대한 속성, 데이터 구조 및 필드의 상호 참조입니다.

표 849. 속성, 데이터 구조 및 필드의 맵핑		
속성	데이터 구조	필드
브릿지 이상종료 코드	MQCIH	AbendCode
ADS 디스크립터	MQCIH	AdsDescriptor
주의 ID	MQCIH	AttentionId
인증자	MQCIH	Authenticator
브릿지 완료 코드	MQCIH	BridgeCompletionCode
브릿지 오류 오프셋	MQCIH	ErrorOffset
브릿지 이유 코드	MQCIH	BridgeReason
브릿지 취소 코드	MQCIH	CancelCode
대화식 태스크	MQCIH	ConversationalTask
커서 위치	MQCIH	CursorPosition
기능 토큰	MQCIH	Facility
기능 보관 시간	MQCIH	FacilityKeepTime
기능 유사	MQCIH	FacilityLike
함수	MQCIH	Function
Get 대기 간격	MQCIH	GetWaitInterval
링크 유형	MQCIH	LinkType
다음 트랜잭션 ID	MQCIH	NextTransactionId
출력 데이터 길이	MQCIH	OutputDataLength
회신 형식	MQCIH	ReplyToFormat
브릿지 리턴 코드	MQCIH	ReturnCode
시작 코드	MQCIH	StartCode
태스크 종료 상태	MQCIH	TaskEndStatus
트랜잭션 ID	MQCIH	TransactionId
uow 제어	MQCIH	UowControl
버전	MQCIH	버전

### ImqDeadLetterHeader 상호 참조

ImqDeadLetterHeader C++ 클래스에 대한 속성, 데이터 구조 및 필드의 상호 참조입니다.

표 850. 속성, 데이터 구조 및 필드의 맵핑		
속성	데이터 구조	필드
데드 레터 이유 코드	MQDLH	원인
목적지 큐 관리자 이름	MQDLH	DestQMgrName
목적지 큐 이름	MQDLH	DestQName
Put 애플리케이션 이름	MQDLH	PutApplName
Put 애플리케이션 유형	MQDLH	PutApplType

표 850. 속성, 데이터 구조 및 필드의 �핑 (계속)		
속성	데이터 구조	필드
Put 날짜	MQDLH	PutDate
Put 시간	MQDLH	PutTime

### ImqError 상호 참조

ImqError C++ 클래스에 대한 속성 및 호출의 상호 참조입니다.

표 851. 속성 및 호출	
속성	전화로 문의
완료 코드	MQBACK, MQBEGIN, MQCLOSE, MQCMIT, MQCONN, MQCONNX, MQDISC, MQGET, MQINQ, MQOPEN, MQPUT, MQSET
이유 코드	MQBACK, MQBEGIN, MQCLOSE, MQCMIT, MQCONN, MQCONNX, MQDISC, MQGET, MQINQ, MQOPEN, MQPUT, MQSET

### ImqGetMessageOptions 상호 참조

ImqGetMessageOptions C++ 클래스에 대한 속성, 데이터 구조 및 필드의 상호 참조입니다.

표 852. 속성, 데이터 구조 및 필드의 �핑		
속성	데이터 구조	필드
그룹 상태	MQGMO	GroupStatus
일치 옵션	MQGMO	MatchOptions
메시지 토큰(message token)	MQGMO	MessageToken
옵션	MQGMO	옵션
해석된 큐 이름	MQGMO	ResolvedQName
리턴된 길이	MQGMO	ReturnedLength
세그먼트화(segmentation)	MQGMO	Segmentation
세그먼트 상태	MQGMO	SegmentStatus
	MQGMO	Signal1
	MQGMO	Signal2
동기점 참여	MQGMO	옵션
대기 간격	MQGMO	WaitInterval

### ImqHeader 상호 참조

ImqHeader C++ 클래스에 대한 속성, 데이터 구조 및 필드의 상호 참조입니다.

표 853. 속성, 데이터 구조 및 필드의 �핑		
속성	데이터 구조	필드
문자 세트	MQDLH, MQIIH	CodedCharSetId
encoding	MQDLH, MQIIH	Encoding
형식	MQDLH, MQIIH	형식

표 853. 속성, 데이터 구조 및 필드의 �핑 (계속)		
속성	데이터 구조	필드
헤더 플래그	MQIIH, MQRMH	플래그

### ImqIMSBridgeHeader 상호 참조

ImqAuthenticationRecord C++ 클래스에 대한 필드 및 데이터 구조, 속성의 상호 참조입니다.

표 854. 속성, 데이터 구조 및 필드의 �핑		
속성	데이터 구조	필드
인증자	MQIIH	Authenticator
커미트 모드	MQIIH	CommitMode
논리 터미널 대체	MQIIH	LTermOverride
메시지 형식 서비스 맵 이름	MQIIH	MFSMapName
회신 형식	MQIIH	ReplyToFormat
보안 범위	MQIIH	SecurityScope
트랜잭션 인스턴스 ID	MQIIH	TranInstanceId
트랜잭션 상태	MQIIH	TranState

### ImqItem 상호 참조

ImqItem C++ 클래스에 대한 속성 및 호출의 상호 참조입니다.

표 855. 속성 및 호출	
속성	전화로 문의
구조 ID	MQGET

### ImqMessage 상호 참조

ImqMessage C++ 클래스에 대한 속성, 데이터 구조, 필드 및 호출의 상호 참조입니다.

표 856. 속성, 데이터 구조, 필드 및 호출			
속성	데이터 구조	필드	전화로 문의
애플리케이션 ID 데이터	MQMD	ApplIdentityData	
애플리케이션 원본 데이터	MQMD	ApplOriginData	
백아웃 수	MQMD	BackoutCount	
문자 세트	MQMD	CodedCharSetId	
encoding	MQMD	Encoding	
expiry	MQMD	만기	
형식	MQMD	형식	
메시지 플래그	MQMD	MsgFlags	
메시지 유형	MQMD	MsgType	
오프셋	MQMD	오프셋	

표 856. 속성, 데이터 구조, 필드 및 호출 (계속)			
속성	데이터 구조	필드	전화로 문의
원래 길이	MQMD	OriginalLength	
persistence	MQMD	Persistence	
priority	MQMD	Priority	
Put 애플리케이션 이름	MQMD	PutApplName	
Put 애플리케이션 유형	MQMD	PutApplType	
Put 날짜	MQMD	PutDate	
Put 시간	MQMD	PutTime	
응답 대상 큐 관리자 이름	MQMD	ReplyToQMgr	
응답 대상 큐 이름	MQMD	ReplyToQ	
보고	MQMD	보고서	
순서 번호	MQMD	MsgSeqNumber	
전체 메시지 길이		DataLength	MQGET
사용자 ID	MQMD	UserIdentifier	

### ImqMessageTracker 상호 참조

ImqMessageTracker C++ 클래스에 대한 속성, 데이터 구조 및 필드의 상호 참조입니다.

표 857. 속성, 데이터 구조 및 필드의 �핑		
속성	데이터 구조	필드
계정 토큰	MQMD	AccountingToken
상관 ID	MQMD	CorrelId
관리	MQMD	Feedback
그룹 ID	MQMD	GroupId
메시지 ID	MQMD	MsgId

### ImqNamelist 상호 참조

ImqNamelist C++ 클래스에 대한 속성, 조회 및 호출의 상호 참조입니다.

표 858. 속성, 조회 및 호출		
속성	조회	전화로 문의
이름 수	MQIA_NAME_COUNT	MQINQ
이름 목록 이름	MQCA_NAMELIST_NAME	MQINQ

### ImqObject 상호 참조

ImqObject C++ 클래스에 대한 속성, 데이터 구조, 필드, 조회 및 호출의 상호 참조입니다.

표 859. 속성, 데이터 구조, 필드, 조회 및 호출				
속성	데이터 구조	필드	조회	전화로 문의
대체 날짜			MQCA_ALTERATION_DATE	MQINQ
대체 시간			MQCA_ALTERATION_TIME	MQINQ
대체 사용자 ID	MQOD	AlternateUserId		
대체 보안 ID				
닫기 옵션				MQCLOSE
설명			MQCA_Q_DESC, MQCA_Q_MGR_DESC, MQCA_PROCESS_DESC	MQINQ
이름	MQOD	ObjectName	MQCA_Q_MGR_NAME, MQCQ_Q_NAME, MQCA_PROCESS_NAME	MQINQ
열기 옵션				MQOPEN
열기 상태				MQOPEN, MQCLOSE
큐 관리자 ID	큐 관리자 ID		MQCA_Q_MGR_IDENTIFIER	MQINQ

### ImqProcess 상호 참조

ImqAuthenticationRecord C++ 클래스에 대한 속성, 조회 및 호출의 상호 참조입니다.

표 860. 속성, 조회 및 호출		
속성	조회	전화로 문의
애플리케이션 ID	MQCA_APPL_ID	MQINQ
애플리케이션 유형	MQIA_APPL_TYPE	MQINQ
환경 데이터	MQCA_ENV_DATA	MQINQ
사용자 데이터	MQCA_USER_DATA	MQINQ

### ImqPutMessageOptions 상호 참조

ImqAuthenticationRecord C++ 클래스에 대한 필드 및 데이터 구조, 속성의 상호 참조입니다.

표 861. 속성, 데이터 구조 및 필드의 맵핑		
속성	데이터 구조	필드
컨텍스트 참조	MQPMO	컨텍스트
	MQPMO	InvalidDestCount
	MQPMO	KnownDestCount
옵션	MQPMO	옵션
레코드 필드	MQPMO	PutMsgRecFields
해석된 큐 관리자 이름	MQPMO	ResolvedQMgrName
해석된 큐 이름	MQPMO	ResolvedQName

표 861. 속성, 데이터 구조 및 필드의 맵핑 (계속)		
속성	데이터 구조	필드
	MQPMO	제한시간
	MQPMO	UnknownDestCount
동기점 참여	MQPMO	옵션

## ImqQueue 상호 참조

ImqQueue C++ 클래스에 대한 속성, 데이터 구조, 필드, 조회 및 호출의 상호 참조입니다.

표 862. ImqQueue 상호 참조				
속성	데이터 구조	필드	조회	전화로 문의
백아웃 리큐 이름			MQCA_BACKOUT_REQ_Q_NAME	MQINQ
백아웃 임계값			MQIA_BACKOUT_THRESHOLD	MQINQ
기본 큐 이름			MQCA_BASE_Q_NAME	MQINQ
클러스터 이름			MQCA_CLUSTER_NAME	MQINQ
클러스터 이름 목록 이름			MQCA_CLUSTER_NAMELIST	MQINQ
클러스터 워크로드 순위			MQIA_CLWL_Q_RANK	MQINQ
클러스터 워크로드 우선순위			MQIA_CLWL_Q_PRIORITY	MQINQ
클러스터 워크로드 사용 큐			MQIA_CLWL_USEQ	MQINQ
작성 날짜			MQCA_CREATION_DATE	MQINQ
작성 시간			MQCA_CREATION_TIME	MQINQ
현재 용량			MQIA_CURRENT_Q_DEPTH	MQINQ
기본 바인드			MQIA_DEF_BIND	MQINQ
기본 입력 열기 옵션			MQIA_DEF_INPUT_OPEN_OPTION	MQINQ
기본 지속성			MQIA_DEF_PERSISTENCE	MQINQ
기본 우선순위			MQIA_DEF_PRIORITY	MQINQ
정의 유형			MQIA_DEFINITION_TYPE	MQINQ
용량 상한 이벤트			MQIA_Q_DEPTH_HIGH_EVENT	MQINQ
용량 상한			MQIA_Q_DEPTH_HIGH_LIMIT	MQINQ
용량 하한 이벤트			MQIA_Q_DEPTH_LOW_EVENT	MQINQ
큐 용량 하한			MQIA_Q_DEPTH_LOW_LIMIT	MQINQ
용량 최대 이벤트			MQIA_Q_DEPTH_MAX_LIMIT	MQINQ
분배 목록			MQIA_DIST_LISTS	MQINQ, MQSET
동적 큐 이름	MQOD	DynamicQName		



표 862. <i>MsgQueue</i> 상호 참조 (계속)				
속성	데이터 구조	필드	조회	전화로 문의
백아웃 횟수 기록			MQIA_HARDEN_GET_BACKOUT	MQINQ
색인 유형			MQIA_INDEX_TYPE	MQINQ
Get 금지			MQIA_INHIBIT_GET	MQINQ, MQSET
Put 금지			MQIA_INHIBIT_PUT	MQINQ, MQSET
이니시에이션 큐 이름			MQCA_INITIATION_Q_NAME	MQINQ
최대 용량			MQIA_MAX_Q_DEPTH	MQINQ
최대 메시지 길이			MQIA_MAX_MSG_LENGTH	MQINQ
메시지 전달 순서			MQIA_MSG_DELIVERY_SEQUENCE	MQINQ
다음 분산 큐				
비지속 메시지 클래스			MQIA_NPM_CLASS	MQINQ
열기 입력 수			MQIA_OPEN_INPUT_COUNT	MQINQ
열기 출력 수			MQIA_OPEN_OUTPUT_COUNT	MQINQ
이전 분산 큐				
프로세스 이름			MQCA_PROCESS_NAME	MQINQ
큐 계정			MQIA_ACCOUNTING_Q	MQINQ
큐 관리자 이름	MQOD	ObjectQMgrName		
큐 모니터링			MQIA_MONITORING_Q	MQINQ
큐 통계			MQIA_STATISTICS_Q	MQINQ
큐 유형			MQIA_Q_TYPE	MQINQ
리모트 큐 관리자 이름			MQCA_REMOTE_Q_MGR_NAME	MQINQ
리모트 큐 이름			MQCA_REMOTE_Q_NAME	MQINQ
해석된 큐 관리자 이름	MQOD	ResolvedQMgrName		
해석된 큐 이름	MQOD	ResolvedQName		
보유 간격			MQIA_RETENTION_INTERVAL	MQINQ
scope			MQIA_SCOPE	MQINQ
서비스 간격(service interval)			MQIA_Q_SERVICE_INTERVAL	MQINQ
서비스 간격 이벤트 (service interval event)			MQIA_Q_SERVICE_INTERVAL_EVENT	MQINQ
공유 가용성			MQIA_SHAREABILITY	MQINQ
스토리지 클래스			MQCA_STORAGE_CLASS	MQINQ
전송 큐 이름			MQCA_XMIT_Q_NAME	MQINQ

표 862. <i>ImqQueue</i> 상호 참조 (계속)				
속성	데이터 구조	필드	조회	전화로 문의
트리거 제어			MQIA_TRIGGER_CONTROL	MQINQ, MQSET
트리거 데이터			MQCA_TRIGGER_DATA	MQINQ, MQSET
트리거 용량			MQIA_TRIGGER_DEPTH	MQINQ, MQSET
트리거 메시지 우선순위			MQIA_TRIGGER_MSG_PRIORITY	MQINQ, MQSET
트리거 유형			MQIA_TRIGGER_TYPE	MQINQ, MQSET
사용법			MQIA_USAGE	MQINQ

### ImqQueueManager 상호 참조

ImqQueueManager C++ 클래스에 대한 속성, 데이터 구조, 필드, 조회 및 호출의 상호 참조입니다.

표 863. 속성, 데이터 구조, 필드, 조회 및 호출				
속성	데이터 구조	필드	조회	전화로 문의
계정 연결 대체			MQIA_ACCOUNTING_CONN_OVERRIDE	MQINQ
계정 간격			MQIA_ACCOUNTING_INTERVAL	MQINQ
활동 기록			MQIA_ACTIVITY_RECORDING	MQINQ
새 mca 검사 채택			MQIA_ADOPTNEWMCA_CHECK	MQINQ
새 mca 유형 채택			MQIA_ADOPTNEWMCA_TYPE	MQINQ
인증 유형	MQCSP	AuthenticationType		MQCONN
권한 이벤트			MQIA_AUTHORITY_EVENT	MQINQ
시작 옵션	MQBO	옵션		MQBEGIN
브릿지 이벤트			MQIA_BRIDGE_EVENT	MQINQ
채널 자동 정의			MQIA_CHANNEL_AUTO_DEF	MQINQ
채널 자동 정의 이벤트			MQIA_CHANNEL_AUTO_EVENT	MQIA
채널 자동 정의 엑시트			MQIA_CHANNEL_AUTO_EXIT	MQIA
채널 이벤트 (channel event)			MQIA_CHANNEL_EVENT	MQINQ
채널 시작기 어댑터			MQIA_CHINIT_ADAPTERS	MQINQ
채널 시작기 제어			MQIA_CHINIT_CONTROL	MQINQ
채널 시작기 디스패처			MQIA_CHINIT_DISPATCHERS	MQINQ

표 863. 속성, 데이터 구조, 필드, 조회 및 호출 (계속)				
속성	데이터 구조	필드	조회	전화로 문의
채널 시작기 추적 자동 시작			MQIA_CHINIT_TRACE_AUTO_START	MQINQ
채널 시작기 추적 테이블 크기			MQIA_CHINIT_TRACE_TABLE_SIZE	MQINQ
채널 모니터링			MQIA_MONITORING_CHANNEL	MQINQ
채널 참조	MQCD	ChannelType		MQCONN, MQCONNX
채널 통계			MQIA_STATISTICS_CHANNEL	MQINQ
문자 세트			MQIA_CODED_CHAR_SET_ID	MQINQ
클러스터 송신자 모니터링			MQIA_MONITORING_AUTO_CLUSSDR	MQINQ
클러스터 송신자 통계			MQIA_STATISTICS_AUTO_CLUSSDR	MQINQ
클러스터 워크로드 데이터			MQCA_CLUSTER_WORKLOAD_DATA	MQINQ
클러스터 워크로드 엑시트			MQCA_CLUSTER_WORKLOAD_EXIT	MQINQ
클러스터 워크로드 길이			MQIA_CLUSTER_WORKLOAD_LENGTH	MQINQ
클러스터 워크로드 mru			MQIA_CLWL_MRU_CHANNELS	MQINQ
클러스터 워크로드 사용 큐			MQIA_CLWL_USEQ	MQINQ
명령 이벤트 (command event)			MQIA_COMMAND_EVENT	MQINQ
명령 입력 큐 이름			MQCA_COMMAND_INPUT_Q_NAME	MQINQ
명령 레벨			MQIA_COMMAND_LEVEL	MQINQ
명령 서버 제어			MQIA_CMD_SERVER_CONTROL	MQINQ
연결 옵션	MQCNO	옵션		MQCONN, MQCONNX
연결 ID	MQCNO	ConnectionId		MQCONNX
연결 상태				MQCONN, MQCONNX, MQDISC
연결 태그	MQCD	ConnTag		MQCONNX
암호화 하드웨어	MQSCO	CryptoHardware		MQCONNX
데드-레터 큐 이름			MQCA_DEAD_LETTER_Q_NAME	MQINQ
기본 전송 큐 이름			MQCA_DEF_XMIT_Q_NAME	MQINQ
분배 목록			MQIA_DIST_LISTS	MQINQ

표 863. 속성, 데이터 구조, 필드, 조회 및 호출 (계속)				
속성	데이터 구조	필드	조회	전화로 문의
dns 그룹			MQCA_DNS_GROUP	MQINQ
dns wlm			MQIA_DNS_WLM	MQINQ
첫 번째 인증 레코드	MQSCO	AuthInfoRecOffset		MQCONN
	MQSCO	AuthInfoRecPtr		MQCONN
금지 이벤트			MQIA_INHIBIT_EVENT	MQINQ
ip 주소 버전			MQIA_IP_ADDRESS_VERSION	MQINQ
키 저장소(key repository)	MQSCO	KeyRepository		MQCONN
키 재설정 수	MQSCO	KeyResetCount		MQCONN
리스너 타이머			MQIA_LISTENER_TIMER	MQINQ
로컬 이벤트			MQIA_LOCAL_EVENT	MQINQ
로거 이벤트			MQIA_LOGGER_EVENT	MQINQ
lu 그룹 이름			MQCA_LU_GROUP_NAME	MQINQ
lu 이름			MQCA_LU_NAME	MQINQ
lu62 arm 접미부			MQCA_LU62_ARM_SUFFIX	MQINQ
lu62 채널			MQIA_LU62_CHANNELS	MQINQ
최대 활성 채널			MQIA_ACTIVE_CHANNELS	MQINQ
최대 채널			MQIA_MAX_CHANNELS	MQINQ
최대 핸들			MQIA_MAX_HANDLES	MQINQ
최대 메시지 길이			MQIA_MAX_MSG_LENGTH	MQINQ
최대 우선순위			MQIA_MAX_PRIORITY	MQINQ
최대 커밋되지 않은 메시지			MQIA_MAX_UNCOMMITTED_MSGS	MQINQ
mqi 계정			MQIA_ACCOUNTING_MQI	MQINQ
mqi 통계			MQIA_STATISTICS_MQI	MQINQ
아웃바운드 포트 최대			MQIA_OUTBOUND_PORT_MAX	MQINQ
아웃바운드 포트 최소			MQIA_OUTBOUND_PORT_MIN	MQINQ
비밀번호	MQCSP	CSPPasswordPtr		MQCONN
	MQCSP	CSPPasswordOffset		MQCONN
	MQCSP	CSPPasswordLength		MQCONN
성능 이벤트 (performance event)			MQIA_PERFORMANCE_EVENT	MQINQ

표 863. 속성, 데이터 구조, 필드, 조회 및 호출 (계속)				
속성	데이터 구조	필드	조회	전화로 문의
플랫폼			MQIA_PLATFORM	MQINQ
큐 계정			MQIA_ACCOUNTING_Q	MQINQ
큐 모니터링			MQIA_MONITORING_Q	MQINQ
큐 통계			MQIA_STATISTICS_Q	MQINQ
수신 제한시간			MQIA_RECEIVE_TIMEOUT	MQINQ
수신 제한시간 최소			MQIA_RECEIVE_TIMEOUT_MIN	MQINQ
수신 제한시간 유형			MQIA_RECEIVE_TIMEOUT_TYPE	MQINQ
리모트 이벤트			MQIA_REMOTE_EVENT	MQINQ
저장소 이름			MQCA_REPOSITORY_NAME	MQINQ
저장소 이름 목록			MQCA_REPOSITORY_NAMELIST	MQINQ
공유 큐 큐 관리자 이름			MQIA_SHARED_Q_Q_MGR_NAME	MQINQ
ssl 이벤트			MQIA_SSL_EVENT	MQINQ
ssl fips			MQIA_SSL_FIPS_REQUIRED	MQINQ
ssl 키 재설정 수			MQIA_SSL_RESET_COUNT	MQINQ
시작-중지 이벤트			MQIA_START_STOP_EVENT	MQINQ
통계 간격			MQIA_STATISTICS_INTERVAL	MQINQ
동기점 가용성			MQIA_SYNCPOINT	MQINQ
tcp 채널			MQIA_TCP_CHANNELS	MQINQ
tcp 활성 유지			MQIA_TCP_KEEP_ALIVE	MQINQ
tcp 이름			MQCA_TCP_NAME	MQINQ
tcp 스택 유형			MQIA_TCP_STACK_TYPE	MQINQ
라우트 추적 기록			MQIA_TRACE_ROUTE_RECORDING	MQINQ
트리거 간격			MQIA_TRIGGER_INTERVAL	MQINQ
사용자 ID	MQCSP	CSPUserIdPtr		MQCONN
	MQCSP	CSPUserIdOffset		MQCONN
	MQCSP	CSPUserIdLength		MQCONN

### ImqReferenceHeader 상호 참조

ImqAuthenticationRecord C++ 클래스에 대한 필드 및 데이터 구조, 속성의 상호 참조입니다.

표 864. 속성, 데이터 구조 및 필드의 맵핑		
속성	데이터 구조	필드
목적지 환경	MQRMH	DestEnvLength, DestEnvOffset
목적지 이름	MQRMH	DestNameLength, DestNameOffset

표 864. 속성, 데이터 구조 및 필드의 맵핑 (계속)		
속성	데이터 구조	필드
인스턴스 ID	MQRMH	ObjectInstanceId
논리 길이	MQRMH	DataLogicalLength
논리 오프셋	MQRMH	DataLogicalOffset
논리 오프셋 2	MQRMH	DataLogicalOffset2
참조 유형	MQRMH	ObjectType
소스 환경	MQRMH	SrcEnvLength, SrcEnvOffset
소스 이름	MQRMH	SrcNameLength, SrcNameOffset

### ImqTrigger 상호 참조

ImqAuthenticationRecord C++ 클래스에 대한 필드 및 데이터 구조, 속성의 상호 참조입니다.

표 865. 속성, 데이터 구조 및 필드의 맵핑		
속성	데이터 구조	필드
애플리케이션 ID	MQTM	ApplId
애플리케이션 유형	MQTM	ApplType
환경 데이터	MQTM	EnvData
프로세스 이름	MQTM	ProcessName
큐 이름	MQTM	QName
트리거 데이터	MQTM	TriggerData
사용자 데이터	MQTM	UserData

### ImqWorkHeader 상호 참조

ImqAuthenticationRecord C++ 클래스에 대한 필드 및 데이터 구조, 속성의 상호 참조입니다.

표 866. 속성, 데이터 구조 및 필드의 맵핑		
속성	데이터 구조	필드
메시지 토큰(message token)	MQWIH	MessageToken
서비스 이름	MQWIH	ServiceName
서비스 단계	MQWIH	ServiceStep

## ImqAuthenticationRecord C++ 클래스

이 클래스는 사용자 정의 TLS 클라이언트 연결에 대해 ImqQueueManager::connect 메소드를 실행하는 동안 사용할 인증 정보 레코드(MQAIR)를 캡슐화합니다.

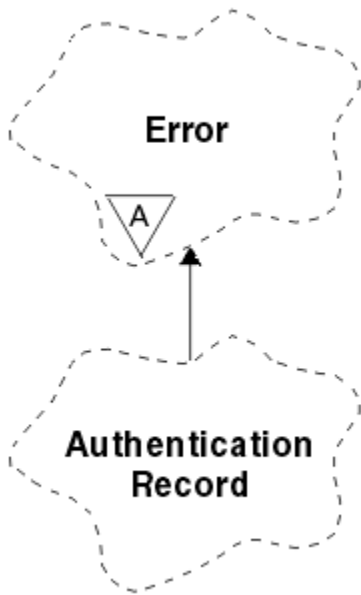


그림 14. *ImqAuthenticationRecord* 클래스

자세한 정보는 *ImqQueueManager::connect* 메소드에 대한 설명을 참조하십시오. z/OS 플랫폼에서는 이 클래스를 사용할 수 없습니다.

- [1723 페이지의 『오브젝트 속성』](#)
- [1723 페이지의 『구성자』](#)
- [1724 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1724 페이지의 『오브젝트 메소드\(보호됨\)』](#)

## 오브젝트 속성

### 연결 이름

LDAP CRL 서버에 대한 연결의 이름. 이 이름은 괄호로 묶은 포트 번호가 선택적으로 뒤에 오는 IP 주소 또는 DNS 이름입니다.

### 연결 참조

(로컬) 큐 관리자에 대한 필수 연결을 제공하는 *ImqQueueManager* 오브젝트의 참조입니다. 초기값은 0입니다. 이 참조를 이름 지정된 큐의 큐 관리자(리모트)를 식별하는 큐 관리자 이름과 혼동하지 마십시오.

### 다음 인증 레코드

특별한 순서 없이 이 오브젝트와 동일한 **연결 참조**를 갖는 이 클래스의 다음 오브젝트. 초기값은 0입니다.

### 비밀번호

LDAP CRL 서버 연결을 인증하기 위해 제공하는 비밀번호.

### 이전 인증 레코드

특별한 순서 없이 이 오브젝트와 동일한 **연결 참조**를 갖는 이 클래스의 이전 오브젝트. 초기값은 0입니다.

### 유형

레코드에 포함된 인증 정보의 유형.

### 사용자 이름

LDAP CRL 서버에 권한을 부여하기 위해 제공하는 사용자 ID.

## 구성자

### **ImqAuthenticationRecord();**

기본 구성자입니다.

## 오브젝트 메소드(공용)

**void operator = ( const ImqAuthenticationRecord & air );**

*air*에서 인스턴스 데이터를 복사하고, 기존 인스턴스 데이터를 대체합니다.

**const ImqString & connectionName ( ) const ;**

연결 이름을 리턴합니다.

**void setConnectionName ( const ImqString & name );**

연결 이름을 설정합니다.

**void setConnectionName ( const char \* name = 0 );**

연결 이름을 설정합니다.

**ImqQueueManager \* connectionReference ( ) const ;**

연결 참조를 리턴합니다.

**void setConnectionReference ( ImqQueueManager & manager );**

연결 참조를 설정합니다.

**void setConnectionReference ( ImqQueueManager \* manager = 0 );**

연결 참조를 설정합니다.

**void copyOut ( MQAIR \* pAir );**

*pAir*에서 인스턴스 데이터를 복사하고, 기존 인스턴스 데이터를 대체합니다. 종속 스토리지 할당이 여기에 포함될 수 있습니다.

**void clear ( MQAIR \* pAir );**

구조를 지우고 *pAir*에 참조되는 종속 스토리지를 해제합니다.

**ImqAuthenticationRecord \* nextAuthenticationRecord ( ) const ;**

다음 인증 레코드를 리턴합니다.

**const ImqString & password ( ) const ;**

비밀번호를 리턴합니다.

**void setPassword ( const ImqString & password );**

비밀번호를 설정합니다.

**void setPassword ( const char \* password = 0 );**

비밀번호를 설정합니다.

**ImqAuthenticationRecord \* previousAuthenticationRecord ( ) const ;**

이전 인증 레코드를 리턴합니다.

**MQLONG type ( ) const ;**

유형을 리턴합니다.

**void setType ( const MQLONG type );**

유형을 설정합니다.

**const ImqString & userName ( ) const ;**

사용자 이름을 리턴합니다.

**void setUserName ( const ImqString & name );**

사용자 이름을 설정합니다.

**void setUserName ( const char \* name = 0 );**

사용자 이름을 설정합니다.

## 오브젝트 메소드(보호됨)

**void setNextAuthenticationRecord ( ImqAuthenticationRecord \* pAir = 0 );**

다음 인증 레코드를 설정합니다.

주의: 인증 레코드 목록이 손상되지 않는 것이 확실한 경우에만 이 기능을 사용하십시오.

**void setPreviousAuthenticationRecord ( ImqAuthenticationRecord \* pAir = 0 );**

이전 인증 레코드를 설정합니다.

주의: 인증 레코드 목록이 손상되지 않는 것이 확실한 경우에만 이 기능을 사용하십시오.



## ImqBinary C++ 클래스

이 클래스는 ImqMessage 계정 토큰, 상관 ID 및 메시지 ID 값에 사용할 수 있는 2진 바이트 어레이를 캡슐화합니다. 쉬운 지정과 복사, 비교가 가능합니다.

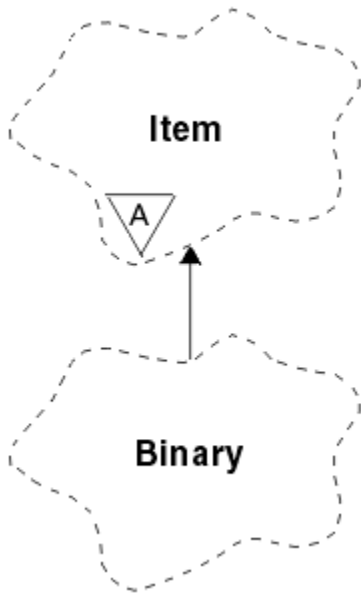


그림 15. ImqBinary 클래스

- [1725 페이지의 『오브젝트 속성』](#)
- [1725 페이지의 『구성자』](#)
- [1725 페이지의 『과부하된 ImqItem 메소드』](#)
- [1726 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1726 페이지의 『오브젝트 메소드\(보호됨\)』](#)
- [1726 페이지의 『이유 코드』](#)

### 오브젝트 속성

#### 데이터

2진 데이터 바이트의 어레이. 초기값은 널입니다.

#### 데이터 길이

바이트 수. 초기값은 0입니다.

#### 데이터 포인터

데이터의 첫 번째 바이트의 주소. 초기값은 0입니다.

### 구성자

#### **ImqBinary();**

기본 구성자입니다.

#### **ImqBinary( const ImqBinary & binary );**

복사 구성자입니다.

#### **ImqBinary( const void \* data, const size\_t length );**

*data*에서 *length*바이트를 복사합니다.

### 과부하된 ImqItem 메소드

#### **virtual ImqBoolean copyOut ( ImqMessage & msg );**

문자를 메시지 버퍼에 복사하고, 기존 콘텐츠를 대체합니다. *msg* 형식을 MQFMT\_NONE으로 설정합니다.

자세한 정보는 ImqItem 클래스 메소드 설명을 참조하십시오.

#### **virtual ImqBoolean pasteIn ( ImqMessage & msg );**

메시지 버퍼에서 나머지 데이터를 전송하여 데이터를 설정하고, 기존 데이터를 대체합니다.

성공적인 작업을 위해 ImqMessage 형식이 MQFMT\_NONE이어야 합니다.

자세한 정보는 ImqItem 클래스 메소드 설명을 참조하십시오.

### **오브젝트 메소드(공용)**

#### **void operator = ( const ImqBinary & binary );**

*binary*에서 바이트를 복사합니다.

#### **ImqBoolean operator == ( const ImqBinary & binary );**

이 오브젝트를 *binary*와 비교합니다. 같지 않으면 FALSE를 리턴하고, 그 외의 경우에는 TRUE를 리턴합니다. 데이터 길이가 동일하고 바이트가 일치하는 경우 오브젝트가 같습니다.

#### **ImqBoolean copyOut ( void \* buffer, const size\_t length, const char pad = 0 );**

데이터 포인터에서 *buffer*로 최대 *length*바이트를 복사합니다. 데이터 길이가 충분하지 않으면 *buffer*의 나머지 공간이 *pad* 바이트로 채워집니다. 또한 *length*가 0이면 *buffer*가 0이 될 수 있습니다. *length*는 음수가 아니어야 합니다. 성공하면 TRUE를 리턴합니다.

#### **size\_t dataLength ( ) const ;**

데이터 길이를 리턴합니다.

#### **ImqBoolean setDataLength ( const size\_t length );**

데이터 길이를 설정합니다. 이 메소드의 결과로 데이터 길이가 변경되면 오브젝트의 데이터가 초기화되지 않습니다. 성공하면 TRUE를 리턴합니다.

#### **void \* dataPointer ( ) const ;**

데이터 포인터를 리턴합니다.

#### **ImqBoolean isNull ( ) const ;**

데이터 길이가 0이거나 모든 데이터 바이트가 0이면 TRUE를 리턴합니다. 그렇지 않으면 FALSE를 리턴합니다.

#### **ImqBoolean set ( const void \* buffer, const size\_t length );**

*buffer*에서 *length*바이트를 복사합니다. 성공하면 TRUE를 리턴합니다.

### **오브젝트 메소드(보호됨)**

#### **void clear ( );**

데이터 길이를 0으로 줄입니다.

### **이유 코드**

- MQRC\_NO\_BUFFER
- MQRC\_STORAGE\_NOT\_AVAILABLE
- MQRC\_INCONSISTENT\_FORMAT

## **ImqCache C++ 클래스**

이 클래스는 데이터를 메모리에 보관하거나 모으는 데 사용합니다.

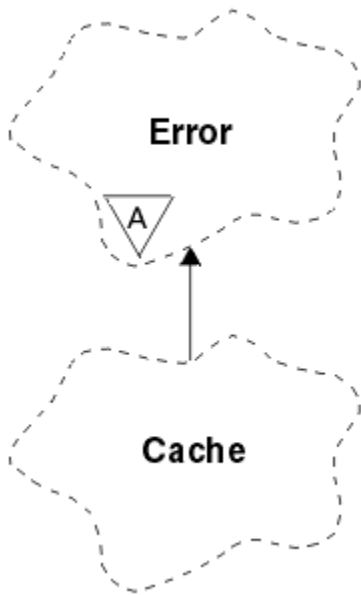


그림 16. *ImqCache* 클래스

이 클래스는 데이터를 메모리에 보관하거나 모으는 데 사용됩니다. 사용자가 고정 크기 메모리의 버퍼를 지정하거나, 시스템에서 유연한 크기의 메모리를 자동으로 제공할 수 있습니다. 이 클래스는 1709 페이지의 『[ImqCache 상호 참조](#)』에 나열된 MQI 호출과 관련이 있습니다.

- [1727 페이지의 『오브젝트 속성』](#)
- [1728 페이지의 『구성자』](#)
- [1728 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1729 페이지의 『이유 코드』](#)

## 오브젝트 속성

### 자동 버퍼

버퍼 메모리가 시스템에 의해 자동으로 관리되는지(TRUE), 사용자가 제공하는지(FALSE) 여부를 표시합니다. 처음에는 TRUE로 설정되어 있습니다.

이 속성은 직접적으로 설정되지 않습니다. **useEmptyBuffer** 또는 **useFullBuffer** 메소드를 사용하여 간접적으로 설정됩니다.

사용자 스토리지가 제공되면 이 속성이 FALSE이고, 버퍼 메모리가 증가하고, 버퍼 오버플로우 오류가 발생할 수 있습니다. 버퍼의 주소와 길이는 일정하게 유지됩니다.

사용자 스토리지가 제공되지 않으면 이 속성이 TRUE이고, 임의의 메시지 데이터 양을 수용할 수 있도록 버퍼 메모리가 증분식으로 커질 수 있습니다. 그러나, 버퍼가 커지면 버퍼의 주소가 변경될 수 있으므로 **버퍼 포인터** 및 **데이터 포인터**를 사용할 때는 주의를 기울이십시오.

### 버퍼 길이

버퍼에 있는 메모리의 바이트 수. 초기값은 0입니다.

### 버퍼 포인터

버퍼 메모리의 주소. 초기값은 널입니다.

### 데이터 길이

데이터 포인터의 뒤에 오는 바이트 수. 이는 **메시지 길이**보다 같거나 작습니다. 초기값은 0입니다.

### 데이터 오프셋

데이터 포인터의 앞에 오는 바이트 수. 이는 **메시지 길이**보다 같거나 작습니다. 초기값은 0입니다.

### 데이터 포인터

다음에 쓰거나 읽어오는 버퍼 부분의 주소. 초기값은 널입니다.

## 메시지 길이

버퍼에 있는 중요한 데이터의 바이트 수. 초기값은 0입니다.

## 구성자

### **ImqCache();**

기본 구성자입니다.

### **ImqCache( const ImqCache & cache );**

복사 구성자입니다.

## 오브젝트 메소드(공용)

### **void operator = ( const ImqCache & cache );**

*cache* 오브젝트에서 해당 오브젝트로 최대 **메시지 길이** 바이트의 데이터를 복사합니다. 자동 버퍼가 FALSE 이면 **버퍼 길이**는 이미 복사된 데이터를 수용하기에 충분합니다.

### **ImqBoolean automaticBuffer ( ) const ;**

자동 버퍼 값을 리턴합니다.

### **size\_t bufferSize ( ) const ;**

버퍼 길이를 리턴합니다.

### **char \* bufferPointer ( ) const ;**

버퍼 포인터를 리턴합니다.

### **void clearMessage ( );**

메시지 길이 및 데이터 오프셋을 0으로 설정합니다.

### **size\_t dataLength ( ) const ;**

데이터 길이를 리턴합니다.

### **size\_t dataOffset ( ) const ;**

데이터 오프셋을 리턴합니다.

### **ImqBoolean setDataOffset ( const size\_t offset );**

데이터 오프셋을 설정합니다. 데이터 오프셋보다 크거나 같도록 필요하면 **메시지 길이**가 늘어납니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

### **char \* dataPointer ( ) const ;**

데이터 포인터의 사본을 리턴합니다.

### **size\_t messageLength ( ) const ;**

메시지 길이를 리턴합니다.

### **ImqBoolean setMessageLength ( const size\_t length );**

메시지 길이를 설정합니다. **메시지 길이**가 **버퍼 길이**를 초과하지 않도록 필요하면 **버퍼 길이**를 늘립니다. **메시지 길이**를 초과하지 않도록 필요하면 **데이터 오프셋**을 줄입니다. 성공하면 TRUE를 리턴합니다.

### **ImqBoolean moreBytes ( const size\_t bytes-required );**

데이터 포인터와 버퍼 끝 간에 더 많은 *bytes-required* 바이트를 (쓰기에) 사용할 수 있는지 확인합니다. 성공하면 TRUE를 리턴합니다.

자동 버퍼가 TRUE이면 더 많은 메모리가 필수로 확보되고, 그렇지 않으면 **버퍼 길이**가 이미 적절해야 합니다.

### **ImqBoolean read ( const size\_t length, char \* & external-buffer );**

데이터 포인터 위치에서 시작하는 버퍼에서 *external-buffer*로 *length* 바이트를 복사합니다. 데이터를 복사한 후 **데이터 오프셋**이 *length*만큼 늘어납니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

### **ImqBoolean resizeBuffer ( const size\_t length );**

자동 버퍼가 TRUE이면 **버퍼 길이**를 변경합니다. 버퍼 메모리를 재할당하면 됩니다. 최대 **메시지 길이** 바이트의 데이터가 기존 버퍼에서 새 버퍼로 복사됩니다. 복사된 최대 수는 *length* 바이트입니다. **버퍼 포인터**는 변경됩니다. **메시지 길이** 및 **데이터 오프셋**은 새 버퍼의 범위 안에서 최대한 유사하게 보존됩니다. 성공하면 TRUE를 리턴하고, 자동 버퍼가 FALSE이면 FALSE를 리턴합니다.

**참고:** 시스템 자원에 문제가 있는 경우, MQRC\_STORAGE\_NOT\_AVAILABLE을 설정하면 이 메소드가 실패할 수 있습니다.

### **ImqBoolean useEmptyBuffer ( const char \* external-buffer, const size\_t length );**

빈 사용자 버퍼를 식별하고, 버퍼 포인터를 *external-buffer*로, 버퍼 길이를 *length*로, 메시지 길이를 0으로 설정합니다. **clearMessage**를 수행합니다. 버퍼가 데이터에 대해 완전히 준비되어 있으면 **useFullBuffer** 메소드를 대신 사용하십시오. 버퍼가 데이터에 대해 부분적으로 준비되어 있으면 **setMessageLength** 메소드를 사용하여 올바른 양을 나타내십시오. 이 메소드는 성공하면 TRUE를 리턴합니다.

이 메소드는 이전의 설명대로 고정 크기의 메모리를 식별하는 데 사용할 수 있으며(*external-buffer*가 널이 아니고 *length*가 0이 아님), 이 경우 **자동 버퍼**가 FALSE로 설정됩니다. 또는 시스템에서 관리하는 유연한 메모리로 되돌리는 데 사용할 수 있으며(*external-buffer*가 널이고 *length*가 0임), 이 경우 **자동 버퍼**가 TRUE로 설정됩니다.

### **ImqBoolean useFullBuffer ( const char \* externalBuffer, const size\_t length );**

메시지 길이가 *length*로 설정되는 점을 제외하고, **useEmptyBuffer**와 같습니다. 성공하면 TRUE를 리턴합니다.

### **ImqBoolean write ( const size\_t length, const char \* external-buffer );**

*external-buffer*에서 데이터 포인터 위치에서 시작하는 버퍼로 *length* 바이트를 복사합니다. 데이터를 복사한 후 데이터 오프셋이 *length*만큼 늘어나고, 메시지 길이는 새 데이터 오프셋 값보다 크거나 같도록 필요하면 늘어납니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

자동 버퍼가 TRUE이면 적절한 크기의 메모리가 보장되고, 그렇지 않으면 최종 데이터 오프셋이 버퍼 길이를 초과하지 않아야 합니다.

## 이유 코드

- MQRC\_BUFFER\_NOT\_AUTOMATIC
- MQRC\_DATA\_TRUNCATED
- MQRC\_INSUFFICIENT\_BUFFER
- MQRC\_INSUFFICIENT\_DATA
- MQRC\_NULL\_POINTER
- MQRC\_STORAGE\_NOT\_AVAILABLE
- MQRC\_ZERO\_LENGTH

## ImqChannel C++ 클래스

이 클래스는 사용자 정의 클라이언트 연결에 대해 `Manager::connect` 메소드를 실행하는 동안 사용할 채널 정의 (MQCD)를 캡슐화합니다.

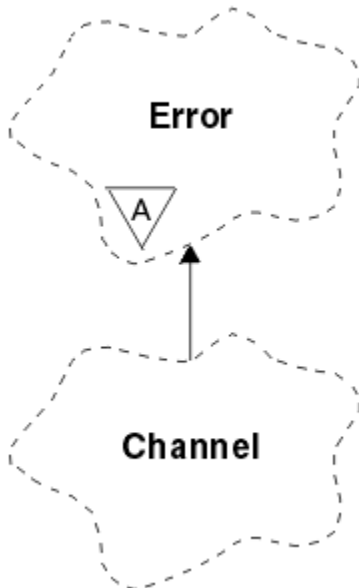


그림 17. *ImqChannel* 클래스

자세한 정보는 `Manager::connect` 메소드에 대한 설명과 [샘플 프로그램 HELLO WORLD \(imqwrld.cpp\)](#)를 참조하십시오.

나열된 모든 메소드를 모든 플랫폼에 적용할 수 있는 것은 아닙니다. 자세한 정보는 `DEFINE CHANNEL` 및 `ALTER CHANNEL` 명령에 대한 설명을 참조하십시오.

`ImqChannel` 클래스는 z/OS에서 지원되지 않습니다.

- [1730 페이지의 『오브젝트 속성』](#)
- [1731 페이지의 『구성자』](#)
- [1731 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1734 페이지의 『이유 코드』](#)

## 오브젝트 속성

### 배치 하트비트

리모트 채널이 활성화인지 검사하는 간격(밀리초)입니다. 초기값은 0입니다.

### 채널 이름

채널의 이름. 초기값은 널입니다.

### 연결 이름

연결의 이름. 예를 들어, 호스트 컴퓨터의 IP 주소입니다. 초기값은 널입니다.

### 헤더 압축

채널에서 지원하는 헤더 데이터 압축 기술에 대한 목록입니다. 초기값은 모두 `MQCOMPRESS_NOT_AVAILABLE`로 설정됩니다.

### 하트비트 간격

연결이 여전히 작동하고 있는지 검사하는 간격(초)입니다. 초기값은 300입니다.

### 활성 상태 지속 간격

채널에 대해 활성 유지 타이밍을 지정하는 통신 스택으로 전달되는 값(초)입니다. 초기값은 `MQKAI_AUTO`입니다.

### 로컬 주소

채널의 로컬 통신 주소입니다.

### 최대 메시지 길이

단일 통신에서 채널이 지원하는 메시지의 최대 길이. 초기값은 4 194 304입니다.

### 메시지 압축

채널에서 지원하는 메시지 데이터 압축 기술에 대한 목록. 초기값은 모두 `MQCOMPRESS_NOT_AVAILABLE`로 설정됩니다.

### 모드 이름

모드의 이름. 초기값은 널입니다.

### 비밀번호

연결 인증을 위해 제공하는 비밀번호. 초기값은 널입니다.

### 수신 엑시트 수

수신 엑시트의 수. 초기값은 0입니다. 이 속성은 읽기 전용입니다.

### 수신 엑시트 이름

수신 엑시트의 이름.

### 수신 사용자 데이터

수신 엑시트와 연관된 데이터.

### 보안 엑시트 이름

연결의 서버 측에 호출되는 보안 엑시트의 이름. 초기값은 널입니다.

### 보안 사용자 데이터

보안 엑시트에 전달될 데이터. 초기값은 널입니다.

### 송신 엑시트 수

송신 엑시트의 수. 초기값은 0입니다. 이 속성은 읽기 전용입니다.

### 송신 엑시트 이름

전송 엑시트의 이름.

### 송신 사용자 데이터

전송 엑시트와 연관된 데이터.

### SSL CipherSpec

TLS와 함께 사용할 CipherSpec.

### SSL 클라이언트 인증 유형

TLS와 함께 사용할 클라이언트 인증 유형.

### SSL 피어 이름

TLS와 함께 사용할 피어 이름.

### 트랜잭션 프로그램 이름

트랜잭션 프로그램의 이름. 초기값은 널입니다.

### 전송 유형

연결의 전송 유형. 초기값은 MQXPT\_LU62입니다.

### 사용자 ID

권한 부여를 위해 제공하는 사용자 ID. 초기값은 널입니다.

## 구성자

### ImqChannel();

기본 구성자입니다.

### ImqChannel( const ImqChannel & channel );

복사 구성자입니다.

## 오브젝트 메소드(공용)

### void operator = ( const ImqChannel & channel );

channel에서 인스턴스 데이터를 복사하고, 기존 인스턴스 데이터를 대체합니다.

### MQLONG batchHeartBeat() const ;

배치 하트비트를 리턴합니다.

### ImqBoolean setBatchHeartBeat( const MQLONG heartbeat = 0L );

배치 하트비트를 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

### ImqString channelName() const ;

채널 이름을 리턴합니다.

### ImqBoolean setChannelName( const char \* name = 0 );

채널 이름을 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

### ImqString connectionName() const ;

연결 이름을 리턴합니다.

### ImqBoolean setConnectionName( const char \* name = 0 );

연결 이름을 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

### size\_t headerCompressionCount() const ;

지원되는 헤더 데이터 압축 기술 수를 리턴합니다.

### ImqBoolean headerCompression( const size\_t count, MQLONG compress [ ] ) const ;

압축에서 지원되는 헤더 데이터 압축 기술의 사본을 리턴합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

### ImqBoolean setHeaderCompression( const size\_t count, const MQLONG compress [ ] );

압축에서 지원되는 헤더 데이터 압축 기술을 설정합니다.

지원되는 헤더 데이터 압축 기술 수를 수(count)로 설정합니다.

이 메소드는 성공하면 TRUE를 리턴합니다.

### MQLONG heartBeatInterval() const ;

하트비트 간격을 리턴합니다.

**ImqBoolean setHeartBeatInterval( const MQLONG interval = 300L );**  
 하트비트 간격을 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**MQLONG keepAliveInterval( ) const ;**  
 활성 유지 간격을 리턴합니다.

**ImqBoolean setKeepAliveInterval( const MQLONG interval = MQKAI\_AUTO );**  
 활성 유지 간격을 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqString localAddress( ) const ;**  
 로컬 주소를 리턴합니다.

**ImqBoolean setLocalAddress ( const char \* address = 0 );**  
 로컬 주소를 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**MQLONG maximumMessageLength( ) const ;**  
 최대 메시지 길이를 리턴합니다.

**ImqBoolean setMaximumMessageLength( const MQLONG length = 4194304L );**  
 최대 메시지 길이를 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**size\_t messageCompressionCount( ) const ;**  
 지원되는 메시지 데이터 압축 기술 수를 리턴합니다.

**ImqBoolean messageCompression( const size\_t count, MQLONG compress [ ] ) const ;**  
 압축에서 지원되는 메시지 데이터 압축 기술의 사본을 리턴합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqBoolean setMessageCompression( const size\_t count, const MQLONG compress [ ] );**  
 지원되는 메시지 데이터 압축 기술을 압축(compress)으로 설정합니다.  
 지원되는 메시지 데이터 압축 기술 수를 수(count)로 설정합니다.  
 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqString modeName( ) const ;**  
 모드 이름을 리턴합니다.

**ImqBoolean setModeName( const char \* name = 0 );**  
 모드 이름을 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqString password( ) const ;**  
 비밀번호를 리턴합니다.

**ImqBoolean setPassword( const char \* password = 0 );**  
 비밀번호를 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**size\_t receiveExitCount( ) const ;**  
 수신 엑시트 수를 리턴합니다.

**ImqString receiveExitName( );**  
 수신 엑시트 이름의 첫 번째 항목을 리턴합니다(있는 경우). 수신 엑시트 수가 0이면 빈 문자열을 리턴합니다.

**ImqBoolean receiveExitNames( const size\_t count, ImqString \* names [ ] );**  
 names의 수신 엑시트 이름의 사본을 리턴합니다. 수신 엑시트 수를 초과하는 names은 널 문자열로 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqBoolean setReceiveExitName( const char \* name = 0 );**  
 수신 엑시트 이름을 단일 name으로 설정합니다. name은 공백이거나 널일 수 있습니다. 수신 엑시트 수를 1 또는 0으로 설정합니다. 수신 사용자 데이터를 지웁니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqBoolean setReceiveExitNames( const size\_t count, const char \* names [ ] );**  
 수신 엑시트 이름을 names으로 설정합니다. 개별 names 값은 공백 또는 널이 아니어야 합니다. 수신 엑시트 수를 count로 설정합니다. 수신 사용자 데이터를 지웁니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqBoolean setReceiveExitNames( const size\_t count, const ImqString \* names [ ] );**  
 수신 엑시트 이름을 names으로 설정합니다. 개별 names 값은 공백 또는 널이 아니어야 합니다. 수신 엑시트 수를 count로 설정합니다. 수신 사용자 데이터를 지웁니다. 이 메소드는 성공하면 TRUE를 리턴합니다.



**ImqString receiveUserData();**

수신 사용자 데이터 항목의 첫 번째 항목을 리턴합니다(있는 경우). 수신 엑시트 수가 0이면 빈 문자열을 리턴합니다.

**ImqBoolean receiveUserData( const size\_t count, ImqString \* data [] );**

*data*의 수신 사용자 데이터 항목의 사본을 리턴합니다. 수신 엑시트 수를 초과하는 *data*는 널 문자열로 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqBoolean setReceiveUserData( const char \* data = 0 );**

수신 사용자 데이터를 단일 항목 *data*로 설정합니다. 데이터가 널이 아니면 수신 엑시트 개수는 1 이상이어야 합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqBoolean setReceiveUserData( const size\_t count, const char \* data [] );**

수신 사용자 데이터를 *data*로 설정합니다. *count*는 수신 엑시트 수를 초과하지 않아야 합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqBoolean setReceiveUserData( const size\_t count, const ImqString \* data [] );**

수신 사용자 데이터를 *data*로 설정합니다. *count*는 수신 엑시트 수를 초과하지 않아야 합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqString securityExitName() const ;**

보안 엑시트 이름을 리턴합니다.

**ImqBoolean setSecurityExitName( const char \* name = 0 );**

보안 엑시트 이름을 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqString securityUserData() const ;**

보안 사용자 데이터를 리턴합니다.

**ImqBoolean setSecurityUserData( const char \* data = 0 );**

보안 사용자 데이터를 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**size\_t sendExitCount() const ;**

송신 엑시트 수를 리턴합니다.

**ImqString sendExitName();**

송신 엑시트 이름의 첫 번째 항목을 리턴합니다(있는 경우). 송신 엑시트 수가 0이면 빈 문자열을 리턴합니다.

**ImqBoolean sendExitNames( const size\_t count, ImqString \* names [] );**

*names*의 송신 엑시트 이름의 사본을 리턴합니다. 송신 엑시트 수를 초과하는 *names*은 널 문자열로 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqBoolean setSendExitName( const char \* name = 0 );**

송신 엑시트 이름을 단일 *name*으로 설정합니다. *name*은 공백이거나 널일 수 있습니다. 송신 엑시트 수를 1 또는 0으로 설정합니다. 송신 사용자 데이터를 지웁니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqBoolean setSendExitNames( const size\_t count, const char \* names [] );**

송신 엑시트 이름을 *names*으로 설정합니다. 개별 *names* 값은 공백 또는 널이 아니어야 합니다. 송신 엑시트 수를 *count*로 설정합니다. 송신 사용자 데이터를 지웁니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqBoolean setSendExitNames( const size\_t count, const ImqString \* names [] );**

송신 엑시트 이름을 *names*으로 설정합니다. 개별 *names* 값은 공백 또는 널이 아니어야 합니다. 송신 엑시트 수를 *count*로 설정합니다. 송신 사용자 데이터를 지웁니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqString sendUserData();**

송신 사용자 데이터 항목 중 첫 번째 항목(있는 경우)을 리턴합니다. 송신 엑시트 수가 0이면 빈 문자열을 리턴합니다.

**ImqBoolean sendUserData( const size\_t count, ImqString \* data [] );**

*data*의 송신 사용자 데이터 항목의 사본을 리턴합니다. 송신 엑시트 수를 초과하는 *data*는 널 문자열로 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqBoolean setSendUserData( const char \* data = 0 );**

송신 사용자 데이터를 단일 항목 *data*로 설정합니다. 데이터가 널이 아니면 송신 엑시트 개수는 1 이상이어야 합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqBoolean setSendUserData( const size\_t count, const char \* data [] );**

송신 사용자 데이터를 *data*로 설정합니다. *count*는 송신 엑시트 수를 초과하지 않아야 합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqBoolean setSendUserData( const size\_t count, const ImqString \* data [ ] );**

송신 사용자 데이터를 *data*로 설정합니다. *count*는 송신 엑시트 수를 초과하지 않아야 합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqString sslCipherSpecification( ) const ;**

TLS 암호 스펙을 리턴합니다.

**ImqBoolean setSslCipherSpecification( const char \* name = 0 );**

TLS 암호 스펙을 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**MQLONG sslClientAuthentication( ) const ;**

TLS 클라이언트 인증 유형을 리턴합니다.

**ImqBoolean setSslClientAuthentication( const MQLONG auth = MQSCA\_REQUIRED );**

TLS 클라이언트 인증 유형을 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqString sslPeerName( ) const ;**

TLS 피어 이름을 리턴합니다.

**ImqBoolean setSslPeerName( const char \* name = 0 );**

TLS 피어 이름을 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqString transactionProgramName( ) const ;**

트랜잭션 프로그램 이름을 리턴합니다.

**ImqBoolean setTransactionProgramName( const char \* name = 0 );**

트랜잭션 프로그램 이름을 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**MQLONG transportType( ) const ;**

전송 유형을 리턴합니다.

**ImqBoolean setTransportType( const MQLONG type = MQXPT\_LU62 );**

전송 유형을 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqString userId( ) const ;**

사용자 ID를 리턴합니다.

**ImqBoolean setUserId( const char \* id = 0 );**

사용자 ID를 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

## 이유 코드

- MQRC\_DATA\_LENGTH\_ERROR
- MQRC\_ITEM\_COUNT\_ERROR
- MQRC\_NULL\_POINTER
- MQRC\_SOURCE\_BUFFER\_ERROR

## ImqCICSBridgeHeader C++ 클래스

이 클래스는 MQCIH 데이터 구조의 특정 기능을 캡슐화합니다.

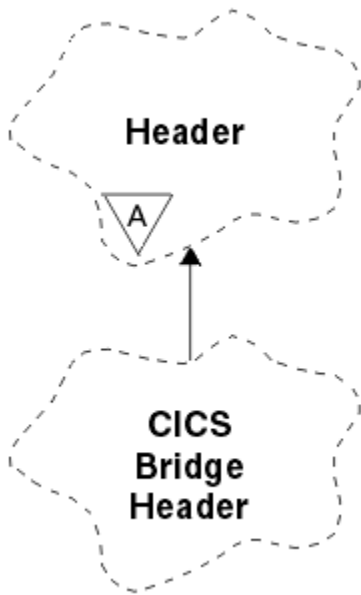


그림 18. *ImqCICSBridgeHeader* 클래스

이 클래스의 오브젝트는 IBM MQ for z/OS를 통해 CICS bridge 에 메시지를 보내는 애플리케이션에서 사용됩니다.

- [1735 페이지의 『오브젝트 속성』](#)
- [1737 페이지의 『구성자』](#)
- [1737 페이지의 『과부하된 ImqItem 메소드』](#)
- [1738 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1740 페이지의 『오브젝트 데이터\(보호됨\)』](#)
- [1740 페이지의 『이유 코드』](#)
- [1740 페이지의 『리턴 코드』](#)

## 오브젝트 속성

### ADS 디스크립터

ADS 디스크립터를 송신/수신합니다. 이 항목은 MQCADSD\_NONE을 사용하여 설정합니다. 초기값은 MQCADSD\_NONE입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQCADSD\_NONE
- MQCADSD\_SEND
- MQCADSD\_RECV
- MQCADSD\_MSGFORMAT

### 주의 ID

AID 키. 이 필드의 길이는 MQ\_ATTENTION\_ID\_LENGTH이어야 합니다.

### 인증자

RACF 비밀번호 또는 패스티켓. 초기값은 MQ\_AUTHENTICATOR\_LENGTH 길이의 공백이 포함되어 있습니다.

### 브릿지 이상종료 코드

MQ\_ABEND\_CODE\_LENGTH 길이의 브릿지 이상종료 코드. 초기값은 4자의 공백 문자입니다. 이 필드에서 리턴된 값은 리턴 코드에 종속적입니다. 자세한 내용은 [1740 페이지의 표 867](#)의 내용을 참조하십시오.

### 브릿지 취소 코드

브릿지 이상종료 트랜잭션 코드. 이 필드는 예약되고, MQ\_CANCEL\_CODE\_LENGTH 길이이며 공백이 포함되어 있어야 합니다.

### 브릿지 완료 코드

IBM MQ 완료 코드 또는 CICS EIBRESP 값을 포함할 수 있는 완료 코드. 이 필드의 초기값은 MQCC\_OK입니다. 이 필드에서 리턴된 값은 리턴 코드에 종속적입니다. 자세한 내용은 [1740 페이지의 표 867](#)의 내용을 참조하십시오.

### 브릿지 오류 오프셋

브릿지 오류 오프셋. 초기값은 0입니다. 이 속성은 읽기 전용입니다.

### 브릿지 이유 코드

이유 코드입니다. 이 필드는 IBM MQ 이유 또는 CICS EIBRESP2 값을 포함할 수 있습니다. 이 필드의 초기값은 MQRC\_NONE입니다. 이 필드에서 리턴된 값은 리턴 코드에 종속적입니다. 자세한 내용은 [1740 페이지의 표 867](#)의 내용을 참조하십시오.

### 브릿지 리턴 코드

CICS bridge에서 가져온 리턴 코드. 초기값은 MQCRC\_OK입니다.

### 대화식 태스크

태스크가 대화식이 될 수 있는지 여부. 초기값은 MQCCT\_NO입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQCCT\_YES
- MQCCT\_NO

### 커서 위치

커서 위치. 초기값은 0입니다.

### 기능 보관 시간

CICS bridge 기능 릴리스 시간.

### 기능 유사

터미널 에뮬레이트된 속성. 이 필드의 길이는 MQ\_FACILITY\_LIKE\_LENGTH이어야 합니다.

### 기능 토큰

BVT 토큰 값. 이 필드의 길이는 MQ\_FACILITY\_LENGTH이어야 합니다. 초기값은 MQCFAC\_NONE입니다.

### 함수

IBM MQ 호출 이름 또는 CICS EIBFN 함수를 포함할 수 있는 함수입니다. 이 필드의 초기값은 MQ\_FUNCTION\_LENGTH 길이의 MQCFUNC\_NONE입니다. 이 필드에서 리턴된 값은 리턴 코드에 종속적입니다. 자세한 내용은 [1740 페이지의 표 867](#)의 내용을 참조하십시오.

함수에 IBM MQ 호출 이름이 포함되어 있으면 다음과 같은 추가 값을 사용할 수 있습니다.

- MQCFUNC\_MQCONN
- MQCFUNC\_MQGET
- MQCFUNC\_MQINQ
- MQCFUNC\_NONE
- MQCFUNC\_MQOPEN
- MQCFUNC\_PUT
- MQCFUNC\_MQPUT1

### Get 대기 간격

CICS bridge 태스크에 의해 실행된 MQGET 호출의 대기 간격. 초기값은 MQCGWI\_DEFAULT입니다. 이 필드는 **uow 제어**의 값이 MQCUOWC\_FIRST인 경우에만 적용됩니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQCGWI\_DEFAULT
- MQWI\_UNLIMITED

### 링크 유형

링크 유형. 초기값은 MQCLT\_PROGRAM입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQCLT\_PROGRAM
- MQCLT\_TRANSACTION

### 다음 트랜잭션 ID

첨부할 다음 트랜잭션의 ID. 이 필드는 길이가 MQ\_TRANSACTION\_ID\_LENGTH여야 합니다.

### 출력 데이터 길이

COMMAREA 데이터 길이. 초기값은 MQCODL\_AS\_INPUT입니다.

### 회신 형식

응답 메시지의 형식 이름. 초기값은 길이가 MQ\_FORMAT\_LENGTH인 MQFMT\_NONE입니다.

### 시작 코드

트랜잭션 시작 코드. 이 필드는 길이가 MQ\_START\_CODE\_LENGTH여야 합니다. 초기값은 MQCSC\_NONE입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQCSC\_START
- MQCSC\_STARTDATA
- MQCSC\_TERMINPUT
- MQCSC\_NONE

### 태스크 종료 상태

태스크 종료 상태. 초기값은 MQCTES\_NOSYNC입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQCTES\_COMMIT
- MQCTES\_BACKOUT
- MQCTES\_ENDTASK
- MQCTES\_NOSYNC

### 트랜잭션 ID

첨부할 트랜잭션의 ID. 초기값은 MQ\_TRANSACTION\_ID\_LENGTH 길이의 공백을 포함해야 합니다. 이 필드는 **uow 제어**의 값이 MQCUOWC\_FIRST 또는 MQCUOWC\_ONLY인 경우에만 적용됩니다.

### UOW 제어

UOW 제어. 초기값은 MQCUOWC\_ONLY입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQCUOWC\_FIRST
- MQCUOWC\_MIDDLE
- MQCUOWC\_LAST
- MQCUOWC\_ONLY
- MQCUOWC\_COMMIT
- MQCUOWC\_BACKOUT
- MQCUOWC\_CONTINUE

### 버전

MQCIH 버전 번호. 초기값은 MQCIH\_VERSION\_2입니다. 다른 지원되는 유일한 값은 MQCIH\_VERSION\_1입니다.

## 구성자

### **ImqCICSBridgeHeader();**

기본 구성자입니다.

### **ImqCICSBridgeHeader( const ImqCICSBridgeHeader & header );**

복사 구성자입니다.

## 과부하된 ImqItem 메소드

### **virtual ImqBoolean copyOut( ImqMessage & msg );**

기존 메시지 데이터를 더 이동시켜서 MQCIH 데이터 구조를 메시지 버퍼의 처음에 삽입하고 메시지 형식을 MQFMT\_CICS로 설정합니다.

자세한 정보는 상위 클래스 메소드 설명을 참조하십시오.

**virtual ImqBoolean pasteIn( ImqMessage & msg );**

메시지 버퍼에서 MQCIH 데이터 구조를 읽어옵니다. 성공적인 작업을 위해 *msg* 오브젝트의 인코딩이 MQENC\_NATIVE이어야 합니다. MQGMO\_CONVERT ~ MQENC\_NATIVE로 메시지를 검색하십시오. 성공하려면 ImqMessage 형식이 MQFMT\_CICS여야 합니다.

자세한 정보는 상위 클래스 메소드 설명을 참조하십시오.

## 오브젝트 메소드(공용)

**void operator = ( const ImqCICSBridgeHeader & header );**

*header*에서 인스턴스 데이터를 복사하고, 기존 인스턴스 데이터를 대체합니다.

**MQLONG ADSDescriptor( ) const;**

ADS 디스크립터의 사본을 리턴합니다.

**void setADSDescriptor( const MQLONG descriptor = MQCADSD\_NONE );**

ADS 디스크립터를 설정합니다.

**ImqString attentionIdentifier( ) const;**

MQ\_ATTENTION\_ID\_LENGTH 길이가 되도록 후미 공백으로 채워진 주의 ID의 사본을 리턴합니다.

**void setAttentionIdentifier( const char \* data = 0 );**

MQ\_ATTENTION\_ID\_LENGTH 길이가 되도록 후미 공백으로 채워진 주의 ID를 설정합니다. *data*가 제공되지 않으면, 주의 ID를 초기값으로 재설정합니다.

**ImqString authenticator( ) const;**

MQ\_AUTHENTICATOR\_LENGTH 길이가 되도록 후미 공백으로 채워진 인증자의 사본을 리턴합니다.

**void setAuthenticator( const char \* data = 0 );**

MQ\_AUTHENTICATOR\_LENGTH 길이가 되도록 후미 공백으로 채워진 인증자를 설정합니다. *data*가 제공되지 않으면, 인증자를 초기값으로 재설정합니다.

**ImqString bridgeAbendCode( ) const;**

MQ\_ABEND\_CODE\_LENGTH 길이가 되도록 후미 공백으로 채워진 브릿지 이상종료 코드의 사본을 리턴합니다.

**ImqString bridgeCancelCode( ) const;**

MQ\_CANCEL\_CODE\_LENGTH 길이가 되도록 후미 공백으로 채워진 브릿지 취소 코드의 사본을 리턴합니다.

**void setBridgeCancelCode( const char \* data = 0 );**

MQ\_CANCEL\_CODE\_LENGTH 길이가 되도록 후미 공백으로 채워진 브릿지 취소 코드를 설정합니다. *data*가 제공되지 않으면, 브릿지 취소 코드를 초기값으로 재설정합니다.

**MQLONG bridgeCompletionCode( ) const;**

브릿지 완료 코드의 사본을 리턴합니다.

**MQLONG bridgeErrorOffset( ) const ;**

브릿지 오류 오프셋의 사본을 리턴합니다.

**MQLONG bridgeReasonCode( ) const;**

브릿지 이유 코드의 사본을 리턴합니다.

**MQLONG bridgeReturnCode( ) const;**

브릿지 리턴 코드를 리턴합니다.

**MQLONG conversationalTask( ) const;**

대화식 태스크의 사본을 리턴합니다.

**void setConversationalTask( const MQLONG task = MQCCT\_NO );**

대화식 태스크를 설정합니다.

**MQLONG cursorPosition( ) const ;**

커서 위치의 사본을 리턴합니다.

**void setCursorPosition( const MQLONG position = 0 );**

커서 위치를 설정합니다.

**MQLONG facilityKeepTime( ) const;**

기능 보관 시간의 사본을 리턴합니다.

**void setFacilityKeepTime( const MQLONG time = 0 );**

기능 보관 시간을 설정합니다.

**ImqString facilityLike( ) const;**

MQ\_FACILITY\_LIKE\_LENGTH 길이가 되도록 후미 공백으로 채워진 기능 유사 사본을 리턴합니다.

**void setFacilityLike( const char \* name = 0 );**

MQ\_FACILITY\_LIKE\_LENGTH 길이가 되도록 후미 공백으로 채워진 기능 유사를 설정합니다. *name*이 제공되지 않으면, 기능 유사를 초기값으로 재설정합니다.

**ImqBinary facilityToken( ) const;**

기능 토큰의 사본을 리턴합니다.

**ImqBoolean setFacilityToken( const ImqBinary & token );**

기능 토큰을 설정합니다. *token*의 데이터 길이는 0 또는 MQ\_FACILITY\_LENGTH이어야 합니다. 성공하면 TRUE를 리턴합니다.

**void setFacilityToken( const MQBYTE8 token = 0 );**

기능 토큰을 설정합니다. *token*은 0일 수 있으며, 이는 MQCFAC\_NONE을 지정하는 것과 같습니다. *token*이 0이 아니면 MQ\_FACILITY\_LENGTH 바이트의 2진 데이터를 처리해야 합니다. MQCFAC\_NONE과 같은 사전 정의된 값을 사용하는 경우, 서명이 일치하도록 캐스트를 만들어야 할 수 있습니다. 예: (MQBYTE \*)MQCFAC\_NONE.

**ImqString function( ) const;**

MQ\_FUNCTION\_LENGTH 길이가 되도록 후미 공백으로 채워진 함수의 사본을 리턴합니다.

**MQLONG getWaitInterval( ) const;**

Get 대기 간격의 사본을 리턴합니다.

**void setGetWaitInterval( const MQLONG interval = MQCGWI\_DEFA**

Get 대기 간격을 설정합니다.

**MQLONG linkType( ) const;**

링크 유형의 사본을 리턴합니다.

**void setLinkType( const MQLONG type = MQCLT\_PROGRAM );**

링크 유형을 설정합니다.

**ImqString nextTransactionIdentifier( ) const ;**

MQ\_TRANSACTION\_ID\_LENGTH 길이가 되도록 후미 공백으로 채워진 다음 트랜잭션 ID 데이터의 사본을 리턴합니다.

**MQLONG outputDataLength( ) const;**

출력 데이터 길이의 사본을 리턴합니다.

**void setOutputDataLength( const MQLONG length = MQCODL\_AS\_INPUT );**

출력 데이터 길이를 설정합니다.

**ImqString replyToFormat( ) const;**

MQ\_FORMAT\_LENGTH 길이가 되도록 후미 공백으로 채워진 회신 형식 이름의 사본을 리턴합니다.

**void setReplyToFormat( const char \* name = 0 );**

MQ\_FORMAT\_LENGTH 길이가 되도록 후미 공백으로 채워진 회신 형식을 설정합니다. *name*이 제공되지 않으면, 회신 형식을 초기값으로 재설정합니다.

**ImqString startCode( ) const;**

MQ\_START\_CODE\_LENGTH 길이가 되도록 후미 공백으로 채워진 시작 코드의 사본을 리턴합니다.

**void setStartCode( const char \* data = 0 );**

MQ\_START\_CODE\_LENGTH 길이가 되도록 후미 공백으로 채워진 시작 코드 데이터를 설정합니다. *data*가 제공되지 않으면, 시작 코드를 초기값으로 재설정합니다.

**MQLONG taskEndStatus( ) const;**

태스크 종료 상태의 사본을 리턴합니다.

**ImqString transactionIdentifier( ) const;**

MQ\_TRANSACTION\_ID\_LENGTH 길이가 되도록 후미 공백으로 채워진 트랜잭션 ID 데이터의 사본을 리턴합니다.

**void setTransactionIdentifier( const char \* data = 0 );**

MQ\_TRANSACTION\_ID\_LENGTH 길이가 되도록 후미 공백으로 채워진 트랜잭션 ID를 설정합니다. data가 제공되지 않으면, 트랜잭션 ID를 초기값으로 재설정합니다.

**MQLONG UOWControl( ) const;**

UOW 제어의 사본을 리턴합니다.

**void setUOWControl( const MQLONG control = MQCUOWC\_ONLY );**

UOW 제어를 설정합니다.

**MQLONG version( ) const;**

버전 번호를 리턴합니다.

**ImqBoolean setVersion( const MQLONG version = MQCIH\_VERSION\_2 );**

버전 번호를 설정합니다. 성공하면 TRUE를 리턴합니다.

## 오브젝트 데이터(보호됨)

**MQLONG olVersion**

opcih에 할당된 스토리지에 수용할 수 있는 최대 MQCIH 버전 번호.

**PMQCIH opcih**

MQCIH 데이터 구조의 주소. 할당된 스토리지의 크기는 olVersion로 표시됩니다.

## 이유 코드

- MQRC\_BINARY\_DATA\_LENGTH\_ERROR
- MQRC\_WRONG\_VERSION

## 리턴 코드

표 867. ImqCICSBridgeHeader 클래스 리턴 코드				
리턴 코드	Function	CompCode	원인	이상종료 코드
MQCRC_OK				
MQCRC_BRIDGE_ERROR			MQFB_CICS	
MQCRC_MQ_API_ERROR	IBM MQ 호출 이 름	IBM MQ CompCode	IBM MQ 이유	
MQCRC_BRIDGE_TIMEOUT	IBM MQ 호출 이 름	IBM MQ CompCode	IBM MQ 이유	
MQCRC_CICS_EXEC_ERROR	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	
MQCRC_SECURITY_ERROR	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	
MQCRC_PROGRAM_NOT_AVAILABLE	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	
MQCRC_TRANSID_NOT_AVAILABLE	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	
MQCRC_BRIDGE_ABEND				CICS ABCODE
MQCRC_APPLICATION_ABEND				CICS ABCODE

## ImqDeadLetterHeader C++ 클래스

이 클래스는 MQDLH 데이터 구조의 기능을 캡슐화합니다.



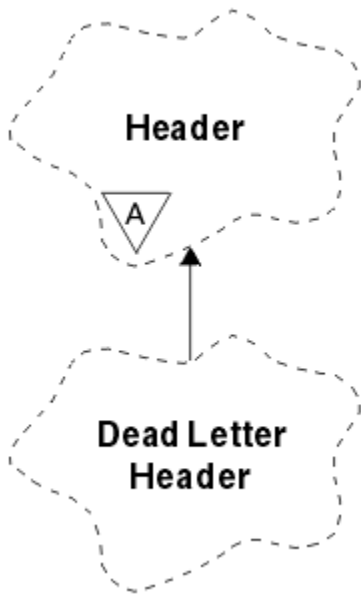


그림 19. `ImqDeadLetterHeader` 클래스

이 클래스의 오브젝트는 일반적으로 처리할 수 없는 메시지가 발생하는 애플리케이션에 사용됩니다. 데드 레터 헤더와 메시지 컨텐츠로 구성된 새 메시지는 데드-레터 큐에 놓이고, 해당 메시지는 제거됩니다.

- [1741 페이지의 『오브젝트 속성』](#)
- [1742 페이지의 『구성자』](#)
- [1742 페이지의 『과부하된 `ImqItem` 메소드』](#)
- [1742 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1743 페이지의 『오브젝트 데이터\(보호됨\)』](#)
- [1743 페이지의 『이유 코드』](#)

## 오브젝트 속성

### 데드 레터 이유 코드

메시지가 데드-레터 큐에 도착한 이유. 초기값은 `MQRC_NONE`입니다.

### 목적지 큐 관리자 이름

원래 목적지 큐 관리자의 이름. 이 이름은 `MQ_Q_MGR_NAME_LENGTH` 길이의 문자열입니다. 초기값은 널입니다.

### 목적지 큐 이름

원래 목적지 큐의 이름. 이 이름은 `MQ_Q_NAME_LENGTH` 길이의 문자열입니다. 초기값은 널입니다.

### Put 애플리케이션 이름

데드-레터 큐에 메시지를 넣는 애플리케이션의 이름. 이 이름은 `MQ_PUT_APPL_NAME_LENGTH` 길이의 문자열입니다. 초기값은 널입니다.

### Put 애플리케이션 유형

데드-레터 큐에 메시지를 넣는 애플리케이션의 유형. 초기값은 0입니다.

### Put 날짜

데드-레터 큐에 메시지가 놓인 날짜. 이 날짜는 `MQ_PUT_DATE_LENGTH` 길이의 문자열입니다. 초기값은 널 문자열입니다.

### Put 시간

데드-레터 큐에 메시지가 놓인 시간. 이 시간은 `MQ_PUT_TIME_LENGTH` 길이의 문자열입니다. 초기값은 널 문자열입니다.

## 구성자

**ImqDeadLetterHeader( );**

기본 구성자입니다.

**ImqDeadLetterHeader( const ImqDeadLetterHeader & header );**

복사 구성자입니다.

## 과부하된 ImqItem 메소드

**virtual ImqBoolean copyOut( ImqMessage & msg );**

시작 시 MQDLH 데이터 구조를 메시지 버퍼에 삽입하고, 그에 따라 기존 메시지 데이터를 이동합니다. msg 형식을 MQFMT\_DEAD\_LETTER\_HEADER로 설정합니다.

자세한 정보는 [1748 페이지의 『ImqHeader C++ 클래스』](#) 페이지의 ImqHeader 클래스 메소드 설명을 참조하십시오.

**virtual ImqBoolean pasteIn( ImqMessage & msg );**

메시지 버퍼에서 MQDLH 데이터 구조를 읽어옵니다.

성공적인 작업을 위해 ImqMessage 형식이 MQFMT\_DEAD\_LETTER\_HEADER여야 합니다.

자세한 정보는 [1748 페이지의 『ImqHeader C++ 클래스』](#) 페이지의 ImqHeader 클래스 메소드 설명을 참조하십시오.

## 오브젝트 메소드(공용)

**void operator = ( const ImqDeadLetterHeader & header );**

header에서 인스턴스 데이터를 복사하고, 기존 인스턴스 데이터를 대체합니다.

**MQLONG deadLetterReasonCode ( ) const ;**

데드 레터 이유 코드를 리턴합니다.

**void setDeadLetterReasonCode ( const MQLONG reason );**

데드 레터 이유 코드를 설정합니다.

**ImqString destinationQueueManagerName ( ) const ;**

후미 공백을 제거한 목적지 큐 관리자 이름을 리턴합니다.

**void setDestinationQueueManagerName ( const char \* name );**

목적지 큐 관리자 이름을 설정합니다. MQ\_Q\_MGR\_NAME\_LENGTH(48자)보다 긴 데이터를 자릅니다.

**ImqString destinationQueueName ( ) const ;**

후미 공백을 제거한 목적지 큐 이름의 사본을 리턴합니다.

**void setDestinationQueueName ( const char \* name );**

목적지 큐 이름을 설정합니다. MQ\_Q\_NAME\_LENGTH(48자)보다 긴 데이터를 자릅니다.

**ImqString putApplicationName ( ) const ;**

후미 공백을 제거한 Put 애플리케이션 이름의 사본을 리턴합니다.

**void setPutApplicationName ( const char \* name = 0 );**

Put 애플리케이션 이름을 설정합니다. MQ\_PUT\_APPL\_NAME\_LENGTH(28자)보다 긴 데이터를 자릅니다.

**MQLONG putApplicationType ( ) const ;**

Put 애플리케이션 유형을 리턴합니다.

**void setPutApplicationType ( const MQLONG type = MQAT\_NO\_CONTEXT );**

Put 애플리케이션 유형을 설정합니다.

**ImqString putDate ( ) const ;**

후미 공백을 제거한 Put 날짜의 사본을 리턴합니다.

**void setPutDate ( const char \* date = 0 );**

Put 날짜를 설정합니다. MQ\_PUT\_DATE\_LENGTH(8자)보다 긴 데이터를 자릅니다.

**ImqString putTime ( ) const ;**

후미 공백을 제거한 Put 시간의 사본을 리턴합니다.

**void setPutTime ( const char \* time = 0 );**

Put 시간을 설정합니다. MQ\_PUT\_TIME\_LENGTH(8자)보다 긴 데이터를 자릅니다.

## 오브젝트 데이터(보호됨)

**MQDLH omqdlh**

MQDLH 데이터 구조.

## 이유 코드

- MQRC\_INCONSISTENT\_FORMAT
- MQRC\_STRUC\_ID\_ERROR
- MQRC\_ENCODING\_ERROR

## ImqDistributionList C++ 클래스

이 클래스는 메시지를 여러 목적지로 송신하기 위해 하나 이상의 큐를 참조하는 동적 분배 목록을 캡슐화합니다.

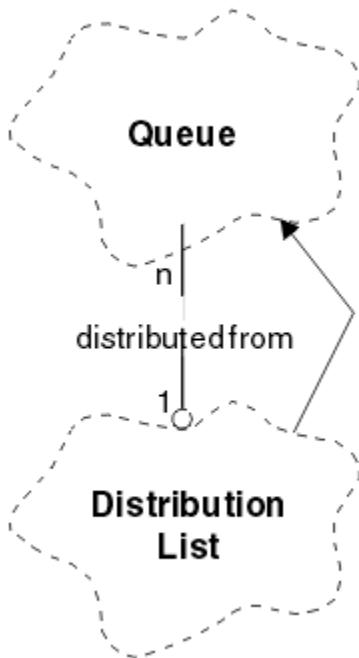


그림 20. *ImqDistributionList* 클래스

- [1743 페이지의 『오브젝트 속성』](#)
- [1744 페이지의 『구성자』](#)
- [1744 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1744 페이지의 『오브젝트 메소드\(보호됨\)』](#)

## 오브젝트 속성

### 첫 번째 분산 큐

하나 이상의 클래스 오브젝트 중 첫 번째 오브젝트로, **분배 목록 참조**는 특별한 순서 없이 이 오브젝트를 처리합니다.

처음에는 이러한 오브젝트가 없습니다. *ImqDistributionList*를 성공적으로 열려면 이러한 오브젝트가 하나 이상 있어야 합니다.

**참고:** *ImqDistributionList* 오브젝트를 열면 해당 오브젝트를 참조하는 열린 오브젝트가 자동으로 닫힙니다.

## 구성자

**ImqDistributionList();**

기본 구성자입니다.

**ImqDistributionList( const ImqDistributionList & list );**

복사 구성자입니다.

## 오브젝트 메소드(공용)

**void operator = ( const ImqDistributionList & list );**

이 오브젝트에 참조하는 모든 오브젝트는 복사하기 전에 참조 해제됩니다. 이 메소드를 호출한 후에는 이 오브젝트를 참조하는 오브젝트가 없습니다.

**\* firstDistributedQueue ( ) const ;**

첫 번째 분산 큐를 리턴합니다.

## 오브젝트 메소드(보호됨)

**void setFirstDistributedQueue ( \* queue = 0 );**

첫 번째 분산 큐를 설정합니다.

## ImqError C++ 클래스

이 추상 클래스는 오브젝트와 연관된 오류에 대한 정보를 제공합니다.

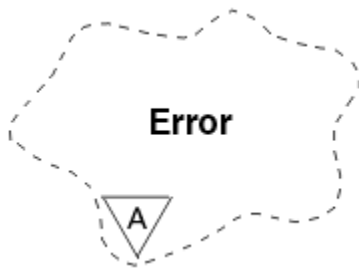


그림 21. *ImqError* 클래스

- [1744 페이지의 『오브젝트 속성』](#)
- [1744 페이지의 『구성자』](#)
- [1745 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1745 페이지의 『오브젝트 메소드\(보호됨\)』](#)
- [1745 페이지의 『이유 코드』](#)

## 오브젝트 속성

### 완료 코드

가장 최신 완료 코드. 초기값은 0입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQCC\_OK
- MQCC\_WARNING
- MQCC\_FAILED

### 이유 코드

가장 최신 이유 코드. 초기값은 0입니다.

## 구성자

**ImqError();**

기본 구성자입니다.

**ImqError( const ImqError & error );**

복사 구성자입니다.

### 오브젝트 메소드(공용)

**void operator = ( const ImqError & error );**

*error*에서 인스턴스 데이터를 복사하고, 기존 인스턴스 데이터를 대체합니다.

**void clearErrorCodes ( );**

완료 코드와 이유 코드를 둘 다 0으로 설정합니다.

**MQLONG completionCode ( ) const ;**

완료 코드를 리턴합니다.

**MQLONG reasonCode ( ) const ;**

이유 코드를 리턴합니다.

### 오브젝트 메소드(보호됨)

**ImqBoolean checkReadPointer ( const void \* *pointer*, const size\_t *length* );**

포인터와 길이의 조합이 읽기 전용 액세스에 대해서만 유효한지 확인하고, 성공하면 TRUE를 리턴합니다.

**ImqBoolean checkWritePointer ( const void \* *pointer*, const size\_t *length* );**

포인터와 길이의 조합이 읽기-쓰기 액세스에 대해서만 유효한지 확인하고, 성공하면 TRUE를 리턴합니다.

**void setCompletionCode ( const MQLONG *code* = 0 );**

완료 코드를 설정합니다.

**void setReasonCode ( const MQLONG *code* = 0 );**

이유 코드를 설정합니다.

### 이유 코드

- MQRC\_BUFFER\_ERROR

## ImqGetMessageOptions C++ 클래스

이 클래스는 MQGMO 데이터 구조를 캡슐화합니다.

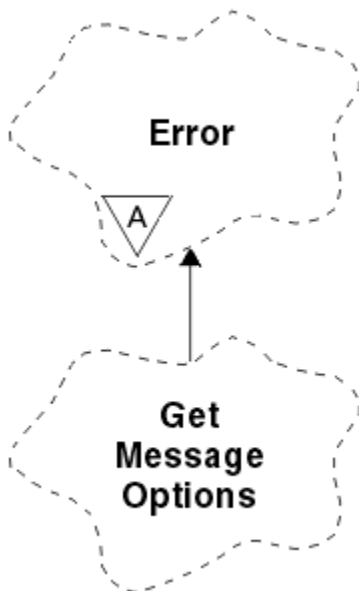


그림 22. *ImqGetMessageOptions* 클래스

- 1746 페이지의 『오브젝트 속성』
- 1747 페이지의 『구성자』

- [1747 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1748 페이지의 『오브젝트 메소드\(보호됨\)』](#)
- [1748 페이지의 『오브젝트 데이터\(보호됨\)』](#)
- [1748 페이지의 『이유 코드』](#)

## 오브젝트 속성

### 그룹 상태

메시지 그룹의 메시지 상태. 초기값은 MQGS\_NOT\_IN\_GROUP입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQGS\_MSG\_IN\_GROUP
- MQGS\_LAST\_MSG\_IN\_GROUP

### 일치 옵션

수신 메시지를 선택하기 위한 옵션. 초기값은 MQMO\_MATCH\_MSG\_ID | MQMO\_MATCH\_CORREL\_ID입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQMO\_GROUP\_ID
- MQMO\_MATCH\_MSG\_SEQ\_NUMBER
- MQMO\_MATCH\_OFFSET
- MQMO\_MSG\_TOKEN
- MQMO\_NONE

### 메시지 토큰(message token)

메시지 토큰. MQ\_MSG\_TOKEN\_LENGTH 길이의 2진 값(MQBYTE16)입니다. 초기값은 MQMTOK\_NONE입니다.

### 옵션

메시지에 적용할 수 있는 옵션. 초기값은 MQGMO\_NO\_WAIT입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQGMO\_WAIT
- MQGMO\_SYNCPOINT
- MQGMO\_SYNCPOINT\_IF\_PERSISTENT
- MQGMO\_NO\_SYNCPOINT
- MQGMO\_MARK\_SKIP\_BACKOUT
- MQGMO\_BROWSE\_FIRST
- MQGMO\_BROWSE\_NEXT
- MQGMO\_BROWSE\_MSG\_UNDER\_CURSOR
- MQGMO\_MSG\_UNDER\_CURSOR
- MQGMO\_LOCK
- MQGMO\_UNLOCK
- MQGMO\_ACCEPT\_TRUNCATED\_MSG
- MQGMO\_SET\_SIGNAL
- MQGMO\_FAIL\_IF QUIESCING
- MQGMO\_CONVERT
- MQGMO\_LOGICAL\_ORDER
- MQGMO\_COMPLETE\_MSG
- MQGMO\_ALL\_MSGS\_AVAILABLE
- MQGMO\_ALL\_SEGMENTS\_AVAILABLE
- MQGMO\_NONE

## 해석된 큐 이름

해석된 큐 이름. 이 속성은 읽기 전용입니다. 이름이 48자를 초과하지 않고 해당 길이가 되도록 널로 채울 수 있습니다. 초기값은 널 문자열입니다.

## 리턴된 길이

리턴된 길이. 초기값은 MQRL\_UNDEFINED입니다. 이 속성은 읽기 전용입니다.

## 세그먼트화(segmentation)

메시지를 세분화하는 기능. 초기값은 MQSEG\_INHIBITED입니다. 추가 값으로 MQSEG\_ALLOWED를 사용할 수 있습니다.

## 세그먼트 상태

메시지의 세그먼트화 상태. 초기값은 MQSS\_NOT\_A\_SEGMENT입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQSS\_SEGMENT
- MQSS\_LAST\_SEGMENT

## 동기점 참여

동기점 제어에서 메시지가 검색되는 경우 TRUE입니다.

## 대기 간격

적절한 메시지의 도착을 대기하는 동안 아직 사용할 수 있는 항목이 없으면 클래스 get 메소드가 일시정지하는 시간입니다. 초기값은 0으로, 무한 대기에 영향을 미칩니다. 추가 값으로 MQWI\_UNLIMITED를 사용할 수 있습니다. 옵션에 MQGMO\_WAIT이 포함되지 않는 경우 이 속성이 무시됩니다.

## 구성자

### **ImqGetMessageOptions( );**

기본 구성자입니다.

### **ImqGetMessageOptions( const ImqGetMessageOptions & gmo );**

복사 구성자입니다.

## 오브젝트 메소드(공용)

### **void operator = ( const ImqGetMessageOptions & gmo );**

*gmo*에서 인스턴스 데이터를 복사하고, 기존 인스턴스 데이터를 대체합니다.

### **MQCHAR groupStatus ( ) const ;**

그룹 상태를 리턴합니다.

### **void setGroupStatus ( const MQCHAR status );**

그룹 상태를 설정합니다.

### **MQLONG matchOptions ( ) const ;**

일치 옵션을 리턴합니다.

### **void setMatchOptions ( const MQLONG options );**

일치 옵션을 설정합니다.

### **ImqBinary messageToken( ) const;**

메시지 토큰을 리턴합니다.

### **ImqBoolean setMessageToken( const ImqBinary & token );**

메시지 토큰을 설정합니다. *token*의 데이터 길이는 0 또는 MQ\_MSG\_TOKEN\_LENGTH이어야 합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

### **void setMessageToken( const MQBYTE16 token = 0 );**

메시지 토큰을 설정합니다. *token*은 0일 수 있으며, 이는 MQMTOK\_NONE을 지정하는 것과 같습니다. *token*이 0이 아니면 MQ\_MSG\_TOKEN\_LENGTH 바이트의 2진 데이터를 처리해야 합니다.

MQMTOK\_NONE과 같은 사전정의된 값을 사용하는 경우, 서명이 일치하도록 캐스트를 만들 필요가 없습니다. 예: (MQBYTE \*)MQMTOK\_NONE.

### **MQLONG options ( ) const ;**

옵션을 리턴합니다.

**void setOptions ( const MQLONG options );**

동기점 참여 값을 포함하여 옵션을 설정합니다.

**ImqString resolvedQueueName ( ) const ;**

해석된 큐 이름의 사본을 리턴합니다.

**MQLONG returnedLength( ) const;**

리턴된 길이를 리턴합니다.

**MQCHAR segmentation ( ) const ;**

세그먼트화를 리턴합니다.

**void setSegmentation ( const MQCHAR value );**

세그먼트화를 설정합니다.

**MQCHAR segmentStatus ( ) const ;**

세그먼트 상태를 리턴합니다.

**void setSegmentStatus ( const MQCHAR status );**

세그먼트 상태를 설정합니다.

**ImqBoolean syncPointParticipation ( ) const ;**

동기점 참여 값을 리턴합니다. 옵션에 MQGMO\_SYNCPOINT 또는 MQGMO\_SYNCPOINT\_IF\_PERSISTENT가 포함되면 TRUE입니다.

**void setSyncPointParticipation ( const ImqBoolean sync );**

동기점 참여 값을 설정합니다. *sync*가 TRUE이면 MQGMO\_SYNCPOINT를 포함하고 MQGMO\_NO\_SYNCPOINT 및 MQGMO\_SYNCPOINT\_IF\_PERSISTENT는 둘 다 제외하도록 옵션을 대체합니다. *sync*가 FALSE이면 MQGMO\_NO\_SYNCPOINT를 포함하고 MQGMO\_SYNCPOINT 및 MQGMO\_SYNCPOINT\_IF\_PERSISTENT는 둘 다 제외하도록 옵션을 대체합니다.

**MQLONG waitInterval ( ) const ;**

대기 간격을 리턴합니다.

**void setWaitInterval ( const MQLONG interval );**

대기 간격을 설정합니다.

## 오브젝트 메소드(보호됨)

**static void setVersionSupported ( const MQLONG );**

MQGMO 버전을 설정합니다. 기본값은 MQGMO\_VERSION\_3입니다.

## 오브젝트 데이터(보호됨)

**MQGMO omqgmo**

MQGMO 버전 2 데이터 구조. MQGMO\_VERSION\_2의 경우에만 MQGMO 필드에 액세스할 수 있습니다.

**PMQGMO opgmo**

MQGMO 데이터 구조의 주소. 이 주소의 버전 번호는 *olVersion*에 표시됩니다. MQGMO 필드에 액세스하기 전에 해당 필드가 있는지 확인하려면 버전 번호를 조사하십시오.

**MQLONG olVersion**

*opgmo*에 의해 처리된 MQGMO 데이터 구조의 버전 번호.

## 이유 코드

- MQRC\_BINARY\_DATA\_LENGTH\_ERROR

## ImqHeader C++ 클래스

이 추상 클래스는 MQDLH 데이터 구조의 공통 기능을 캡슐화합니다.



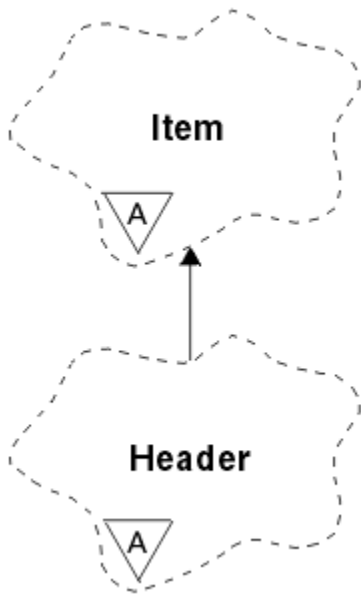


그림 23. *ImqHeader* 클래스

- [1749 페이지의 『오브젝트 속성』](#)
- [1749 페이지의 『구성자』](#)
- [1749 페이지의 『오브젝트 메소드\(공용\)』](#)

## 오브젝트 속성

### 문자 세트

원본 코드화 문자 세트 ID. 초기값은 MQCCSI\_Q\_MGR입니다.

### encoding

원본 인코딩. 초기값은 MQENC\_NATIVE입니다.

### 형식

원본 형식. 초기값은 MQFMT\_NONE입니다.

### 헤더 플래그

초기값은 다음과 같습니다.

- *ImqDeadLetterHeader* 클래스 오브젝트의 경우 0
- *ImqIMSBridgeHeader* 클래스 오브젝트의 경우 MQIIH\_NONE
- *ImqReferenceHeader* 클래스 오브젝트의 경우 MQRMHF\_LAST
- *ImqCICSBridgeHeader* 클래스 오브젝트의 경우 MQCIH\_NONE
- *ImqWorkHeader* 클래스 오브젝트의 경우 MQWIH\_NONE

## 구성자

### ***ImqHeader*( );**

기본 구성자입니다.

### ***ImqHeader*( const *ImqHeader* & *header* );**

복사 구성자입니다.

## 오브젝트 메소드(공용)

### **void operator = ( const *ImqHeader* & *header* );**

*header*에서 인스턴스 데이터를 복사하고 기존 인스턴스 데이터를 대체합니다.

**virtual MQLONG characterSet ( ) const ;**  
 문자 세트를 리턴합니다.

**virtual void setCharacterSet ( const MQLONG ccsid = MQCCSI\_Q\_MGR );**  
 문자 세트를 설정합니다.

**virtual MQLONG encoding ( ) const ;**  
 인코딩을 리턴합니다.

**virtual void setEncoding ( const MQLONG encoding = MQENC\_NATIVE );**  
 인코딩을 설정합니다.

**virtual ImqString format ( ) const ;**  
 후미 공백을 포함하여 형식의 사본을 리턴합니다.

**virtual void setFormat ( const char \* name = 0 );**  
 8자가 되도록 후미 공백으로 채운 형식을 설정합니다.

**virtual MQLONG headerFlags ( ) const ;**  
 헤더 플래그를 리턴합니다.

**virtual void setHeaderFlags ( const MQLONG flags = 0 );**  
 헤더 플래그를 설정합니다.

## ImqIMSBridgeHeader C++ 클래스

이 클래스는 MQIIH 데이터 구조의 기능을 캡슐화합니다.

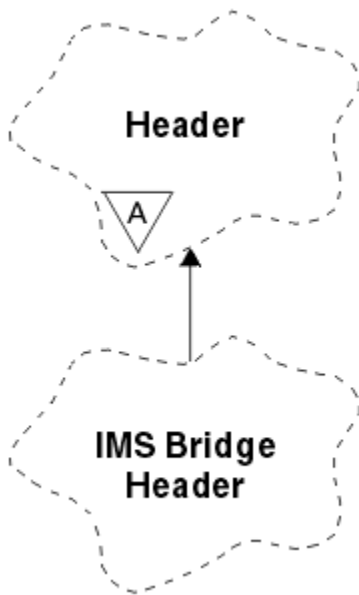


그림 24. *ImqIMSBridgeHeader* 클래스

이 클래스의 오브젝트는 IBM MQ for z/OS를 통해 IMS 브릿지로 메시지를 송신하는 애플리케이션에 사용됩니다.

**참고:** *ImqHeader* 문자 세트 및 인코딩은 기본값을 가져야 하며 그 외 다른 값으로 설정하지 않아야 합니다.

- [1751 페이지의 『오브젝트 속성』](#)
- [1751 페이지의 『구성자』](#)
- [1751 페이지의 『과부하된 ImqItem 메소드』](#)
- [1751 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1752 페이지의 『오브젝트 데이터\(보호됨\)』](#)
- [1752 페이지의 『이유 코드』](#)

## 오브젝트 속성

### 인증자

MQ\_AUTHENTICATOR\_LENGTH 길이의 RACF 비밀번호 또는 패스워드. 초기값은 MQIAUT\_NONE입니다.

### 커밋 모드

커밋 모드. IMS 커밋 모드에 대한 자세한 정보는 OTMA 사용자 안내서를 참조하십시오. 초기값은 MQICM\_COMMIT\_THEN\_SEND입니다. 추가 값으로 MQICM\_SEND\_THEN\_COMMIT를 사용할 수 있습니다.

### 논리 터미널 대체

길이 MQ\_LTERM\_OVERRIDE\_LENGTH의 논리 터미널 대체. 초기값은 널 문자열입니다.

### 메시지 형식 서비스 맵 이름

MQ\_MFS\_MAP\_NAME\_LENGTH 길이의 MFS 맵 이름. 초기값은 널 문자열입니다.

### 회신 형식

MQ\_FORMAT\_LENGTH 길이의 응답 형식. 초기값은 MQFMT\_NONE입니다.

### 보안 범위

IMS 보안 처리의 범위. 초기값은 MQISS\_CHECK입니다. 추가 값으로 MQISS\_FULL을 사용할 수 있습니다.

### 트랜잭션 인스턴스 ID

MQ\_TRAN\_INSTANCE\_ID\_LENGTH 길이의 2진(MQBYTE16) 값인 트랜잭션 인스턴스 ID. 초기값은 MQITII\_NONE입니다.

### 트랜잭션 상태

IMS 대화의 상태. 초기값은 MQITS\_NOT\_IN\_CONVERSATION입니다. 추가 값으로 MQITS\_IN\_CONVERSATION을 사용할 수 있습니다.

## 구성자

### **ImqIMSBridgeHeader();**

기본 구성자입니다.

### **ImqIMSBridgeHeader( const ImqIMSBridgeHeader & header );**

복사 구성자입니다.

## 과부하된 ImqItem 메소드

### **virtual ImqBoolean copyOut( ImqMessage & msg );**

시작 시 MQIIH 데이터 구조를 메시지 버퍼에 삽입하고, 그에 따라 기존 메시지 데이터를 이동합니다. msg 형식을 MQFMT\_IMS로 설정합니다.

추가적인 세부사항은 상위 클래스 메소드 설명을 참조하십시오.

### **virtual ImqBoolean pasteIn( ImqMessage & msg );**

메시지 버퍼에서 MQIIH 데이터 구조를 읽어옵니다.

성공적인 작업을 위해 msg 오브젝트의 인코딩이 MQENC\_NATIVE이어야 합니다. MQGMO\_CONVERT ~ MQENC\_NATIVE로 메시지를 검색하십시오.

성공적인 작업을 위해 ImqMessage 형식이 MQFMT\_IMS여야 합니다.

추가적인 세부사항은 상위 클래스 메소드 설명을 참조하십시오.

## 오브젝트 메소드(공용)

### **void operator = ( const ImqIMSBridgeHeader & header );**

header에서 인스턴스 데이터를 복사하고 기존 인스턴스 데이터를 대체합니다.

### **ImqString authenticator( ) const;**

MQ\_AUTHENTICATOR\_LENGTH 길이가 되도록 후미 공백으로 채워진 인증자의 사본을 리턴합니다.

### **void setAuthenticator ( const char \* name );**

인증자를 설정합니다.

**MQCHAR commitMode ( ) const ;**

커미트 모드를 리턴합니다.

**void setCommitMode ( const MQCHAR mode );**

커미트 모드를 설정합니다.

**ImqString logicalTerminalOverride ( ) const ;**

논리 터미널 대체의 사본을 리턴합니다.

**void setLogicalTerminalOverride ( const char \* override );**

논리 터미널 대체를 설정합니다.

**ImqString messageFormatServicesMapName ( ) const ;**

메시지 형식 서비스 맵 이름의 사본을 리턴합니다.

**void setMessageFormatServicesMapName ( const char \* name );**

메시지 형식 서비스 맵 이름을 설정합니다.

**ImqString replyToFormat( ) const;**

MQ\_FORMAT\_LENGTH 길이가 되도록 후미 공백으로 채워진 회신 형식의 사본을 리턴합니다.

**void setReplyToFormat ( const char \* format );**

MQ\_FORMAT\_LENGTH 길이가 되도록 후미 공백으로 채워진 회신 형식을 설정합니다.

**MQCHAR securityScope ( ) const ;**

보안 범위를 리턴합니다.

**void setSecurityScope ( const MQCHAR scope );**

보안 범위를 설정합니다.

**ImqBinary transactionInstanceId ( ) const ;**

트랜잭션 인스턴스 ID의 사본을 리턴합니다.

**ImqBoolean setTransactionInstanceId ( const ImqBinary & id );**

트랜잭션 인스턴스 ID를 설정합니다. *token*의 데이터 길이는 0 또는 MQ\_TRAN\_INSTANCE\_ID\_LENGTH여야 합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**void setTransactionInstanceId ( const MQBYTE16 id = 0 );**

트랜잭션 인스턴스 ID를 설정합니다. *id*가 0일 수 있으며, 이는 MQITII\_NONE을 지정하는 것과 같습니다. *id*가 0이 아니면 MQ\_TRAN\_INSTANCE\_ID\_LENGTH 바이트의 2진 데이터를 처리해야 합니다.

MQITII\_NONE과 같은 사전정의된 값을 사용하는 경우, 서명이 일치하도록 캐스트를 만들어야 할 수 있습니다. 예: (MQBYTE \*)MQITII\_NONE.

**MQCHAR transactionState ( ) const ;**

트랜잭션 상태를 리턴합니다.

**void setTransactionState ( const MQCHAR state );**

트랜잭션 상태를 설정합니다.

## 오브젝트 데이터(보호됨)

**MQIIH omqiih**

MQIIH 데이터 구조.

## 이유 코드

- MQRC\_BINARY\_DATA\_LENGTH\_ERROR
- MQRC\_INCONSISTENT\_FORMAT
- MQRC\_ENCODING\_ERROR
- MQRC\_STRUC\_ID\_ERROR

## ImqItem C++ 클래스

이 추상 클래스는 메시지 안에 있는 몇 가지 중 하나인 항목을 나타냅니다.

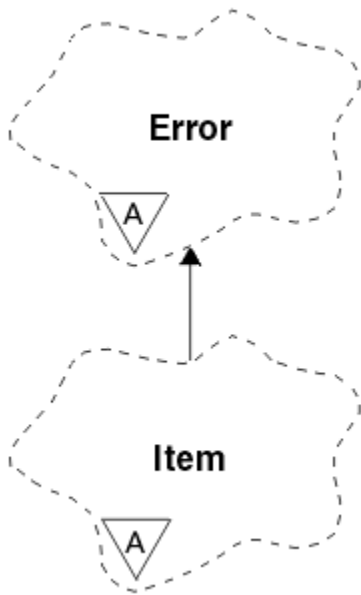


그림 25. *ImqItem* 클래스

항목들은 메시지 버퍼에서 함께 연결되어 있습니다. 각 특수화는 구조 ID로 시작하는 특정 데이터 구조와 연관됩니다.

이 추상 클래스의 다형적 메소드를 통해 항목을 메시지로(부터) 복사할 수 있습니다. *ImqMessage* 클래스 **readItem** 및 **writeItem** 메소드는 애플리케이션 프로그램에 한층 자연스러운 이러한 다형적 메소드를 호출하는 또 다른 방법을 제공합니다.

- [1753 페이지의 『오브젝트 속성』](#)
- [1753 페이지의 『구성자』](#)
- [1753 페이지의 『클래스 메소드\(공용\)』](#)
- [1754 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1754 페이지의 『이유 코드』](#)

## 오브젝트 속성

### 구조 ID

데이터 구조의 시작 부분에 있는 4자 문자열. 이 속성은 읽기 전용입니다. 도출된 클래스의 경우 이 속성을 고려하십시오. 속성이 자동으로 포함되지 않습니다.

### 구성자

#### **ImqItem();**

기본 구성자입니다.

#### **ImqItem( const ImqItem & item );**

복사 구성자입니다.

### 클래스 메소드(공용)

#### **static ImqBoolean structureIdIs ( const char \* structure-id-to-test, const ImqMessage & msg );**

수신 *msg*에서 다음 *ImqItem*의 구조 ID가 *structure-id-to-test*와 동일하면 TRUE를 리턴합니다. 다음 항목은 현재 *ImqCache* 데이터 포인터에 의해 처리되는 메시지 버퍼의 부분으로 식별됩니다. 이 메소드는 구조 ID를 사용하므로 모든 *ImqItem* 파생 클래스에 작동하는 것은 아닙니다.

## 오브젝트 메소드(공용)

**void operator = ( const ImqItem & item );**

*item*에서 인스턴스 데이터를 복사하고, 기존 인스턴스 데이터를 대체합니다.

**virtual ImqBoolean copyOut ( ImqMessage & msg ) = 0 ;**

보내는 메시지 버퍼에서 이 오브젝트를 다음 항목으로 작성하고, 이를 기존 항목에 추가합니다. 쓰기 조작에 성공하면 ImqCache 데이터 길이를 늘립니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

특정 서브클래스에 사용할 수 있도록 이 메소드를 대체하십시오.

**virtual ImqBoolean pasteIn ( ImqMessage & msg ) = 0 ;**

수신되는 메시지 버퍼에서 이 오브젝트를 파괴적으로 읽어들입니다. 이 읽기는 ImqCache 데이터 포인터가 이동한다는 점에서 파괴적입니다. 그러나 버퍼 콘텐츠는 동일하게 유지되므로 ImqCache 데이터 포인터를 재설정하면 데이터를 다시 읽을 수 있습니다.

이 오브젝트의 (서브)클래스는 *msg* 오브젝트의 메시지 버퍼에서 다음에 있는 구조 ID와 일치해야 합니다.

*msg* 오브젝트의 인코딩이 MQENC\_NATIVE이어야 합니다. 메시지는 MQENC\_NATIVE로 설정된 ImqMessage 인코딩 및 MQGMO\_CONVERT를 포함한 ImqGetMessageOptions 옵션을 사용하여 검색하는 것이 좋습니다.

읽기 조작에 성공하면 ImqCache 데이터 길이가 줄어듭니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

특정 서브클래스에 사용할 수 있도록 이 메소드를 대체하십시오.

## 이유 코드

- MQRC\_ENCODING\_ERROR
- MQRC\_STRUC\_ID\_ERROR
- MQRC\_INCONSISTENT\_FORMAT
- MQRC\_INSUFFICIENT\_BUFFER
- MQRC\_INSUFFICIENT\_DATA

## ImqMessage C++ 클래스

이 클래스는 MQMD 데이터 구조를 캡슐화하고 메시지 데이터의 구성과 재구성을 처리합니다.

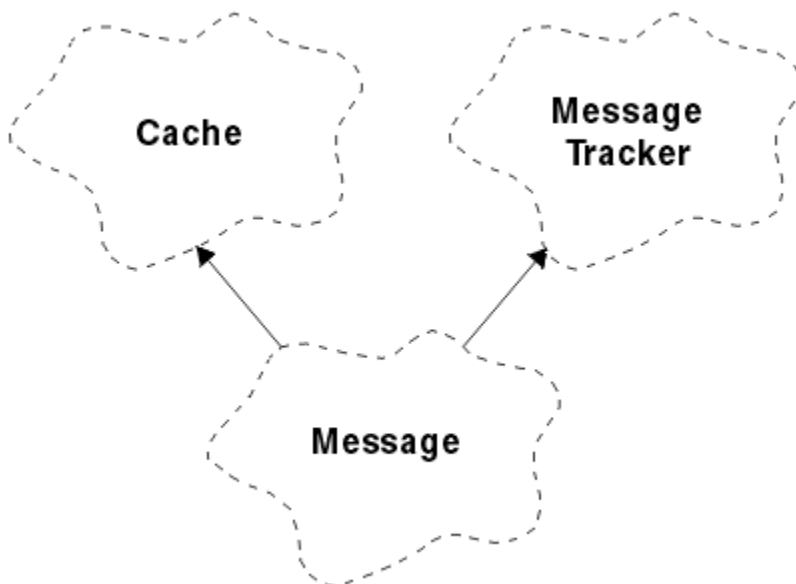


그림 26. ImqMessage 클래스

- [1755 페이지의 『오브젝트 속성』](#)
- [1758 페이지의 『구성자』](#)

- [1758 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1760 페이지의 『오브젝트 메소드\(보호됨\)』](#)
- [1760 페이지의 『오브젝트 데이터\(보호됨\)』](#)

## 오브젝트 속성

### 애플리케이션 ID 데이터

메시지와 연관된 ID 정보. 초기값은 널 문자열입니다.

### 애플리케이션 원본 데이터

메시지와 연관된 원본 정보. 초기값은 널 문자열입니다.

### 백아웃 수

메시지를 조심스럽게 검색하고 후속으로 백아웃한 횟수. 초기값은 0입니다. 이 속성은 읽기 전용입니다.

### 문자 세트

코드화 문자 세트 ID. 초기값은 MQCCSI\_Q\_MGR입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQCCSI\_INHERIT
- MQCCSI\_EMBEDDED

선택한 코드화 문자 세트 ID를 사용할 수도 있습니다. 이에 대한 자세한 내용은 [898 페이지의 『코드 페이지 변환』](#)를 참조하십시오.

### encoding

메시지 데이터의 시스템 인코딩. 초기값은 MQENC\_NATIVE입니다.

### expiry

IBM MQ에서 검색되지 않은 메시지를 제거 전에 보유하는 기간을 제어하는 시간 종속 수. 초기값은 MQEI\_UNLIMITED입니다.

### 형식

버퍼의 데이터 레이아웃을 설명하는 형식(템플릿)의 이름. 8자를 초과하는 이름은 8자로 잘립니다. 이름은 항상 8자가 되도록 공백으로 채워집니다. 초기 상수 값은 MQFMT\_NONE입니다. 다음 추가 상수를 사용할 수 있습니다.

- MQFMT\_ADMIN
- MQFMT\_CICS
- MQFMT\_COMMAND\_1
- MQFMT\_COMMAND\_2
- MQFMT\_DEAD\_LETTER\_HEADER
- MQFMT\_DIST\_HEADER
- MQFMT\_EVENT
- MQFMT\_IMS
- MQFMT\_IMS\_VAR\_STRING
- MQFMT\_MD\_EXTENSION
- MQFMT\_PCF
- MQFMT\_REF\_MSG\_HEADER
- MQFMT\_RF\_HEADER
- MQFMT\_STRING
- MQFMT\_TRIGGER
- MQFMT\_WORK\_INFO\_HEADER
- MQFMT\_XMIT\_Q\_HEADER

선택한 애플리케이션 고유 문자열을 사용할 수도 있습니다. 이에 대한 자세한 정보는 메시지 디스크립터 (MQMD)의 [435 페이지의 『MQMD의 형식 \(MQCHAR8\)』](#) 필드를 참조하십시오.

## 메시지 플래그

세그먼트화 제어 정보. 초기값은 MQMF\_SEGMENTATION\_INHIBITED입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQMF\_SEGMENTATION\_ALLOWED
- MQMF\_MSG\_IN\_GROUP
- MQMF\_LAST\_MSG\_IN\_GROUP
- MQMF\_SEGMENT
- MQMF\_LAST\_SEGMENT
- MQMF\_NONE

## 메시지 유형

메시지의 광범위한 분류. 초기값은 MQMT\_DATAGRAM입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQMT\_SYSTEM\_FIRST
- MQMT\_SYSTEM\_LAST
- MQMT\_DATAGRAM
- MQMT\_REQUEST
- MQMT\_REPLY
- MQMT\_REPORT
- MQMT\_APPL\_FIRST
- MQMT\_APPL\_LAST

선택한 애플리케이션 고유 값을 사용할 수도 있습니다. 이에 대한 자세한 정보는 메시지 디스크립터(MQMD)의 426 페이지의 『MQMD의 경우 MsgType (MQLONG)』 필드를 참조하십시오.

## 오프셋

오프셋 정보. 초기값은 0입니다.

## 원래 길이

세그먼트화된 메시지의 원본 길이. 초기값은 MQOL\_UNDEFINED입니다.

## persistence

메시지는 중요하므로 항상 지속적 스토리지를 사용하여 백업되어야 함을 나타냅니다. 이 옵션은 성능 저하를 의미합니다. 초기값은 MQPER\_PERSISTENCE\_AS\_Q\_DEF입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQPER\_PERSISTENT
- MQPER\_NOT\_PERSISTENT

## priority

전송과 전달의 상대적 우선순위. 우선순위가 같은 메시지는 일반적으로 제공된 순서와 동일한 순서로 전달됩니다(단, 이를 보충하기 위해서는 몇 가지 기준을 충족해야 함). 초기값은 MQPRI\_PRIORITY\_AS\_Q\_DEF입니다.

## 특성 유효성 검증

메시지 특성을 설정할 때 특성의 유효성을 검증하는지 여부를 지정합니다. 초기값은 **MQCMHO\_DEFAULT\_VALIDATION**입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQCMHO\_VALIDATE
- MQCMHO\_NO\_VALIDATION

다음 메소드는 특성 유효성 검증에 따라 작동합니다.

**MQLONG propertyValidation() const ;**

특성 유효성 검증 옵션을 리턴합니다.

**void setPropertyValidation( const MQLONG option );**

특성 유효성 검증 옵션을 설정합니다.



**Put 애플리케이션 이름**

메시지를 넣은 애플리케이션의 이름. 초기값은 널 문자열입니다.

**Put 애플리케이션 유형**

메시지를 넣는 애플리케이션의 유형. 초기값은 MQAT\_NO\_CONTEXT입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQAT\_AIX
- MQAT\_CICS
- MQAT\_CICS\_BRIDGE
- MQAT\_DOS
- MQAT\_IMS
- MQAT\_IMS\_BRIDGE
- MQAT\_MVS
- MQAT\_NOTES\_AGENT
- MQAT\_OS2
- MQAT\_OS390
- MQAT\_OS400
- MQAT\_QMGR
- MQAT\_UNIX
- MQAT\_WINDOWS
- MQAT\_WINDOWS\_NT
- MQAT\_XCF
- MQAT\_DEFAULT
- MQAT\_UNKNOWN
- MQAT\_USER\_FIRST
- MQAT\_USER\_LAST

선택한 애플리케이션 고유 문자열을 사용할 수도 있습니다. 이에 대한 자세한 정보는 메시지 디스크립터 (MQMD)의 448 페이지의 『PutApplMQMD의 유형 (MQLONG)』 필드를 참조하십시오.

**Put 날짜**

메시지가 놓여진 날짜. 초기값은 널 문자열입니다.

**Put 시간**

메시지가 놓여진 시간. 초기값은 널 문자열입니다.

**응답 대상 큐 관리자 이름**

응답이 송신되어야 하는 큐 관리자의 이름. 초기값은 널 문자열입니다.

**응답 대상 큐 이름**

응답이 송신되어야 하는 큐의 이름. 초기값은 널 문자열입니다.

**보고**

메시지와 연관된 피드백 정보. 초기값은 MQRO\_NONE입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQRO\_EXCEPTION
- MQRO\_EXCEPTION\_WITH\_DATA
- MQRO\_EXCEPTION\_WITH\_FULL\_DATA \*
- MQRO\_EXPIRATION
- MQRO\_EXPIRATION\_WITH\_DATA
- MQRO\_EXPIRATION\_WITH\_FULL\_DATA \*
- MQRO\_COA
- MQRO\_COA\_WITH\_DATA

- MQRO\_COA\_WITH\_FULL\_DATA \*
- MQRO\_COD
- MQRO\_COD\_WITH\_DATA
- MQRO\_COD\_WITH\_FULL\_DATA \*
- MQRO\_PAN
- MQRO\_NAN
- MQRO\_NEW\_MSG\_ID
- MQRO\_NEW\_CORREL\_ID
- MQRO\_COPY\_MSG\_ID\_TO\_CORREL\_ID
- MQRO\_PASS\_CORREL\_ID
- MQRO\_DEAD\_LETTER\_Q
- MQRO\_DISCARD\_MSG

여기서 \*는 IBM MQ for z/OS에서 지원되지 않는 값을 표시합니다.

#### 순서 번호

그룹 내에서 메시지를 식별하는 순서 정보. 초기값은 1입니다.

#### 전체 메시지 길이

가장 최근에 메시지를 읽으려고 시도하는 동안 사용 가능했던 바이트 수. 마지막 메시지가 잘렸거나 잘림 때문에 마지막 메시지를 읽지 못한 경우 이 숫자가 ImqCache 메시지 길이를 초과합니다. 이 속성은 읽기 전용입니다. 초기값은 0입니다.

이 속성은 잘린 메시지를 포함한 모든 상황에서 유용할 수 있습니다.

#### 사용자 ID

메시지와 연관된 사용자 ID. 초기값은 널 문자열입니다.

## 구성자

#### **ImqMessage( );**

기본 구성자입니다.

#### **ImqMessage( const ImqMessage & msg );**

복사 구성자입니다. 자세한 정보는 **operator =** 메소드를 참조하십시오.

## 오브젝트 메소드(공용)

#### **void operator = ( const ImqMessage & msg );**

*msg*에서 MQMD 및 메시지 데이터를 복사합니다. 사용자가 이 오브젝트에 대한 버퍼를 제공한 경우, 복사되는 데이터의 양이 사용 가능한 버퍼 크기로 제한됩니다. 그렇지 않으면, 복사된 데이터에 적절한 크기의 버퍼를 사용할 수 있는지 시스템에서 확인합니다.

#### **ImqString applicationIdData ( ) const ;**

애플리케이션 ID 데이터의 사본을 리턴합니다.

#### **void setApplicationIdData ( const char \* data = 0 );**

애플리케이션 ID 데이터를 설정합니다.

#### **ImqString applicationOriginData ( ) const ;**

애플리케이션 원본 데이터의 사본을 리턴합니다.

#### **void setApplicationOriginData ( const char \* data = 0 );**

애플리케이션 원본 데이터를 설정합니다.

#### **MQLONG backoutCount ( ) const ;**

백아웃 수를 리턴합니다.

#### **MQLONG characterSet ( ) const ;**

문자 세트를 리턴합니다.

**void setCharacterSet ( const MQLONG ccsid = MQCCSI\_Q\_MGR );**  
문자 세트를 설정합니다.

**MQLONG encoding ( ) const ;**  
인코딩을 리턴합니다.

**void setEncoding ( const MQLONG encoding = MQENC\_NATIVE );**  
인코딩을 설정합니다.

**MQLONG expiry ( ) const ;**  
만기를 리턴합니다.

**void setExpiry ( const MQLONG expiry );**  
만기를 설정합니다.

**ImqString format ( ) const ;**  
후미 공백을 포함하여 형식의 사본을 리턴합니다.

**ImqBoolean formatIs ( const char \* format-to-test ) const ;**  
형식이 *format-to-test*와 같으면 TRUE를 리턴합니다.

**void setFormat ( const char \* name = 0 );**  
8자가 되도록 후미 공백으로 채운 형식을 설정합니다.

**MQLONG messageFlags ( ) const ;**  
메시지 플래그를 리턴합니다.

**void setMessageFlags ( const MQLONG flags );**  
메시지 플래그를 설정합니다.

**MQLONG messageType ( ) const ;**  
메시지 유형을 리턴합니다.

**void setMessageType ( const MQLONG type );**  
메시지 유형을 설정합니다.

**MQLONG offset ( ) const ;**  
오프셋을 리턴합니다.

**void setOffset ( const MQLONG offset );**  
오프셋을 설정합니다.

**MQLONG originalLength ( ) const ;**  
원본 길이를 리턴합니다.

**void setOriginalLength ( const MQLONG length );**  
원본 길이를 설정합니다.

**MQLONG persistence ( ) const ;**  
지속을 리턴합니다.

**void setPersistence ( const MQLONG persistence );**  
지속을 설정합니다.

**MQLONG priority ( ) const ;**  
우선순위를 리턴합니다.

**void setPriority ( const MQLONG priority );**  
우선순위를 설정합니다.

**ImqString putApplicationName ( ) const ;**  
Put 애플리케이션 이름의 사본을 리턴합니다.

**void setPutApplicationName ( const char \* name = 0 );**  
Put 애플리케이션 이름을 설정합니다.

**MQLONG putApplicationType ( ) const ;**  
Put 애플리케이션 유형을 리턴합니다.

**void setPutApplicationType ( const MQLONG type = MQAT\_NO\_CONTEXT );**  
Put 애플리케이션 유형을 설정합니다.

**ImqString putDate ( ) const ;**  
Put 날짜의 사본을 리턴합니다.

**void setPutDate ( const char \* date = 0 );**

Put 날짜를 설정합니다.

**ImqString putTime ( ) const ;**

Put 시간의 사본을 리턴합니다.

**void setPutTime ( const char \* time = 0 );**

Put 시간을 설정합니다.

**ImqBoolean readItem ( ImqItem & item );**

ImqItem **pasteIn** 메소드를 사용하여 메시지 버퍼에서 *item* 오브젝트로 읽어옵니다. 성공하면 TRUE를 리턴합니다.

**ImqString replyToQueueManagerName ( ) const ;**

응답 대상 큐 관리자 이름의 사본을 리턴합니다.

**void setReplyToQueueManagerName ( const char \* name = 0 );**

응답 대상 큐 관리자 이름을 설정합니다.

**ImqString replyToQueueName ( ) const ;**

응답 대상 큐 이름의 사본을 리턴합니다.

**void setReplyToQueueName ( const char \* name = 0 );**

응답 대상 큐 이름을 설정합니다.

**MQLONG report ( ) const ;**

보고서를 리턴합니다.

**void setReport ( const MQLONG report );**

보고서를 설정합니다.

**MQLONG sequenceNumber ( ) const ;**

순서 번호를 리턴합니다.

**void setSequenceNumber ( const MQLONG number );**

순서 번호를 설정합니다.

**size\_t totalMessageLength ( ) const ;**

총 메시지 길이를 리턴합니다.

**ImqString userId ( ) const ;**

사용자 ID의 사본을 리턴합니다.

**void setUserId ( const char \* id = 0 );**

사용자 ID를 설정합니다.

**ImqBoolean writeItem ( ImqItem & item );**

ImqItem **copyOut** 메소드를 사용하여 *item* 오브젝트를 메시지 버퍼에 씁니다. 쓰기는 삽입, 대체 또는 추가의 형식을 취할 수 있으며, 이는 *item* 오브젝트의 클래스에 따라 다릅니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

## 오브젝트 메소드(보호됨)

**static void setVersionSupported ( const MQLONG );**

MQMD 버전을 설정합니다. 기본값은 MQMD\_VERSION\_2입니다.

## 오브젝트 데이터(보호됨)

 **MQMD1 omqmd**

z/OS의 MQMD 데이터 구조입니다.

 **MQMD2 omqmd**

멀티플랫폼의 MQMD 데이터 구조입니다.

## ImqMessageTracker C++ 클래스

이 클래스는 오브젝트와 연관될 수 있는 ImqMessage 또는 ImqQueue 오브젝트의 해당 속성을 캡슐화합니다.

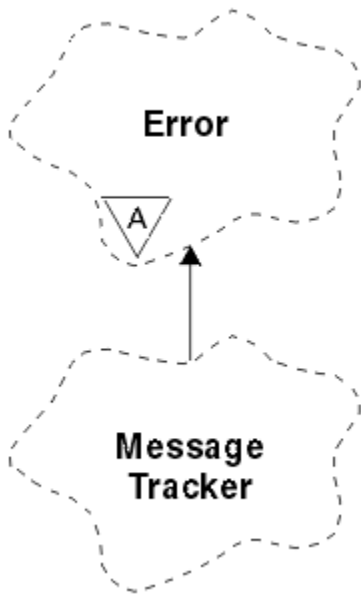


그림 27. *ImqMessageTracker* 클래스

이 클래스는 1714 페이지의 『[ImqMessageTracker 상호 참조](#)』에 나열된 MQI 호출과 관련이 있습니다.

- [1761 페이지의 『오브젝트 속성』](#)
- [1762 페이지의 『구성자』](#)
- [1762 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1763 페이지의 『이유 코드』](#)

## 오브젝트 속성

### 계정 토큰

MQ\_ACCOUNTING\_TOKEN\_LENGTH 길이의 2진 값(MQBYTE32)입니다. 초기값은 MQACT\_NONE입니다.

### 상관 ID

메시지와 상관 관계를 갖도록 지정하는 MQ\_CORREL\_ID\_LENGTH 길이의 2진 값(MQBYTE24)입니다. 초기값은 MQCI\_NONE입니다. 추가 값으로 MQCI\_NEW\_SESSION을 사용할 수 있습니다.

### 관리

메시지와 송신할 피드백 정보. 초기값은 MQFB\_NONE입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQFB\_SYSTEM\_FIRST
- MQFB\_SYSTEM\_LAST
- MQFB\_APPL\_FIRST
- MQFB\_APPL\_LAST
- MQFB\_COA
- MQFB\_COD
- MQFB\_EXPIRATION
- MQFB\_PAN
- MQFB\_NAN
- MQFB\_QUIT
- MQFB\_DATA\_LENGTH\_ZERO
- MQFB\_DATA\_LENGTH\_NEGATIVE
- MQFB\_DATA\_LENGTH\_TOO\_BIG
- MQFB\_BUFFER\_OVERFLOW

- MQFB\_LENGTH\_OFF\_BY\_ONE
- MQFB\_IIH\_ERROR
- MQFB\_NOT\_AUTHORIZED\_FOR\_IMS
- MQFB\_IMS\_ERROR
- MQFB\_IMS\_FIRST
- MQFB\_IMS\_LAST
- MQFB\_CICS\_APPL\_ABENDED
- MQFB\_CICS\_APPL\_NOT\_STARTED
- MQFB\_CICS\_BRIDGE\_FAILURE
- MQFB\_CICS\_CCSID\_ERROR
- MQFB\_CICS\_CIH\_ERROR
- MQFB\_CICS\_COMMAREA\_ERROR
- MQFB\_CICS\_CORREL\_ID\_ERROR
- MQFB\_CICS\_DLQ\_ERROR
- MQFB\_CICS\_ENCODING\_ERROR
- MQFB\_CICS\_INTERNAL\_ERROR
- MQFB\_CICS\_NOT\_AUTHORIZED
- MQFB\_CICS\_UOW\_BACKED\_OUT
- MQFB\_CICS\_UOW\_ERROR

선택한 애플리케이션 고유 문자열을 사용할 수도 있습니다. 이에 대한 자세한 정보는 메시지 디스크립터 (MQMD)의 430 페이지의 『MQMD에 대한 피드백 (MQLONG)』 필드를 참조하십시오.

#### 그룹 ID

큐 안에서 고유한 MQ\_GROUP\_ID\_LENGTH 길이의 2진 값(MQBYTE24)입니다. 초기값은 MQGI\_NONE입니다.

#### 메시지 ID

큐 안에서 고유한 MQ\_MSG\_ID\_LENGTH 길이의 2진 값(MQBYTE24)입니다. 초기값은 MQMI\_NONE입니다.

## 구성자

### **ImqMessageTracker();**

기본 구성자입니다.

### **ImqMessageTracker( const ImqMessageTracker & tracker );**

복사 구성자입니다. 자세한 정보는 **operator =** 메소드를 참조하십시오.

## 오브젝트 메소드(공용)

### **void operator = ( const ImqMessageTracker & tracker );**

*tracker*에서 인스턴스 데이터를 복사하고, 기존 인스턴스 데이터를 대체합니다.

### **ImqBinary accountingToken ( ) const ;**

계정 토큰의 사본을 리턴합니다.

### **ImqBoolean setAccountingToken ( const ImqBinary & token );**

계정 토큰을 설정합니다. *token*의 데이터 길이는 0 또는 MQ\_ACCOUNTING\_TOKEN\_LENGTH이어야 합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

### **void setAccountingToken ( const MQBYTE32 token = 0 );**

계정 토큰을 설정합니다. *token*은 0일 수 있으며, 이는 MQACT\_NONE을 지정하는 것과 같습니다. *token*이 0이 아니면 MQ\_ACCOUNTING\_TOKEN\_LENGTH 바이트의 2진 데이터를 처리해야 합니다. MQACT\_NONE과 같은 사전정의된 값을 사용하는 경우, 서명이 일치하도록 캐스트를 만들어야 할 수 있습니다. 예: (MQBYTE \*)MQACT\_NONE.

**ImqBinary correlationId ( ) const ;**

상관 ID의 사본을 리턴합니다.

**ImqBoolean setCorrelationId ( const ImqBinary & token );**

상관 ID를 설정합니다. *token*의 데이터 길이는 0 또는 MQ\_CORREL\_ID\_LENGTH이어야 합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**void setCorrelationId ( const MQBYTE24 id = 0 );**

상관 ID를 설정합니다. *id*가 0일 수 있으며, 이는 MQCI\_NONE을 지정하는 것과 같습니다. *id*가 0이 아니면 MQ\_CORREL\_ID\_LENGTH 바이트의 2진 데이터를 처리해야 합니다. MQCI\_NONE과 같은 사전정의된 값을 사용하는 경우, 서명이 일치하도록 캐스트를 만들어야 할 수 있습니다. 예: (MQBYTE \*)MQCI\_NONE.

**MQLONG feedback ( ) const ;**

피드백을 리턴합니다.

**void setFeedback ( const MQLONG feedback );**

피드백을 설정합니다.

**ImqBinary groupId ( ) const ;**

그룹 ID의 사본을 리턴합니다.

**ImqBoolean setGroupId ( const ImqBinary & token );**

그룹 ID를 설정합니다. *token*의 데이터 길이는 0 또는 MQ\_GROUP\_ID\_LENGTH이어야 합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**void setGroupId ( const MQBYTE24 id = 0 );**

그룹 ID를 설정합니다. *id*가 0일 수 있으며, 이는 MQGI\_NONE을 지정하는 것과 같습니다. *id*가 0이 아니면 MQ\_GROUP\_ID\_LENGTH 바이트의 2진 데이터를 처리해야 합니다. MQGI\_NONE과 같은 사전정의된 값을 사용하는 경우, 서명이 일치하도록 캐스트를 만들어야 할 수 있습니다. 예: (MQBYTE \*)MQGI\_NONE.

**ImqBinary messageId ( ) const ;**

메시지 ID의 사본을 리턴합니다.

**ImqBoolean setMessageId ( const ImqBinary & token );**

메시지 ID를 설정합니다. *token*의 데이터 길이는 0 또는 MQ\_MSG\_ID\_LENGTH이어야 합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**void setMessageId ( const MQBYTE24 id = 0 );**

메시지 ID를 설정합니다. *id*가 0일 수 있으며, 이는 MQMI\_NONE을 지정하는 것과 같습니다. *id*가 0이 아니면 MQ\_MSG\_ID\_LENGTH 바이트의 2진 데이터를 처리해야 합니다. MQMI\_NONE과 같은 사전정의된 값을 사용하는 경우, 서명이 일치하도록 캐스트를 만들어야 할 수 있습니다. 예: (MQBYTE \*)MQMI\_NONE.

## 이유 코드

- MQRC\_BINARY\_DATA\_LENGTH\_ERROR

## ImqNamelist C++ 클래스

이 클래스는 이름 목록을 캡슐화합니다.

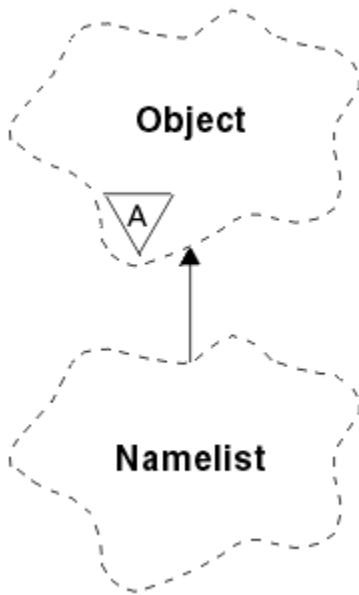


그림 28. *ImqNamelist* 클래스

이 클래스는 1714 페이지의 『[ImqNamelist 상호 참조](#)』에 나열된 MQI 호출과 관련이 있습니다.

- [1764 페이지의 『오브젝트 속성』](#)
- [1764 페이지의 『구성자』](#)
- [1764 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1765 페이지의 『이유 코드』](#)

## 오브젝트 속성

### 이름 수

이름 목록 이름에 있는 오브젝트 이름의 수. 이 속성은 읽기 전용입니다.

### 이름 목록 이름

오브젝트 이름으로, 이름 수로 표시된 숫자입니다. 이 속성은 읽기 전용입니다.

## 구성자

### **ImqNamelist();**

기본 구성자입니다.

### **ImqNamelist( const ImqNamelist & list );**

복사 구성자입니다. ImqObject 열림 상태는 false입니다.

### **ImqNamelist( const char \* name );**

ImqObject 이름을 이름(name)으로 설정합니다.

## 오브젝트 메소드(공용)

### **void operator = ( const ImqNamelist & list );**

*list*에서 인스턴스 데이터를 복사하고, 기존 인스턴스 데이터를 대체합니다. ImqObject 열림 상태는 false입니다.

### **ImqBoolean nameCount( MQLONG & count );**

이름 수의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

### **MQLONG nameCount ( );**

오류 가능성에 대한 표시 없이 이름 수를 리턴합니다.

### **ImqBoolean namelistName ( const MQLONG index, ImqString & name );**

0 기반 인덱스에 따라 이름 목록 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.



### ImqString namelistName ( const MQLONG index );

오류 가능성에 대한 표시 없이 0 기반 인덱스에 따라 이름 목록 이름 중 하나를 리턴합니다.

### 이유 코드

- MQRC\_INDEX\_ERROR
- MQRC\_INDEX\_NOT\_PRESENT

## ImqObject C++ 클래스

이 클래스는 추상입니다. 이 클래스의 오브젝트를 영구 삭제하면 자동으로 닫히고 해당 ImqQueueManager 연결이 끊깁니다.

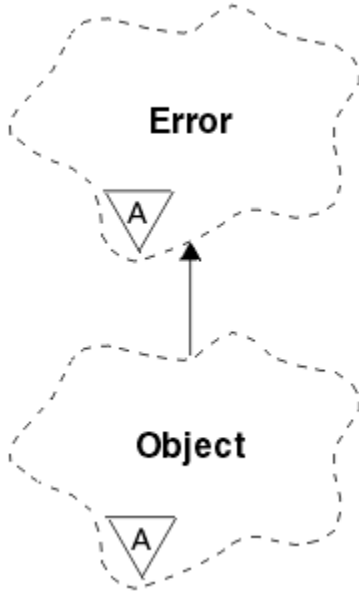


그림 29. ImqObject 클래스

이 클래스는 1714 페이지의 『ImqObject 상호 참조』에 나열된 MQI 호출과 관련이 있습니다.

- 1765 페이지의 『클래스 속성』
- 1766 페이지의 『오브젝트 속성』
- 1767 페이지의 『구성자』
- 1767 페이지의 『클래스 메소드(공용)』
- 1767 페이지의 『오브젝트 메소드(공용)』
- 1769 페이지의 『오브젝트 메소드(보호됨)』
- 1769 페이지의 『오브젝트 데이터(보호됨)』
- 1770 페이지의 『이유 코드』
- 

### 클래스 속성

#### 작동(behavior)

암시적 열기의 동작을 제어합니다.

#### IMQ\_IMPL\_OPEN (8L)

암시적 열기가 허용됩니다. 기본값입니다.

## 오브젝트 속성

### 대체 날짜

대체 날짜. 이 속성은 읽기 전용입니다.

### 대체 시간

대체 시간. 이 속성은 읽기 전용입니다.

### 대체 사용자 ID

최대 MQ\_USER\_ID\_LENGTH자의 대체 사용자 ID. 초기값은 널 문자열입니다.

### 대체 보안 ID

대체 보안 ID. MQ\_SECURITY\_ID\_LENGTH 길이의 2진 값(MQBYTE40)입니다. 초기값은 MQSID\_NONE입니다.

### 닫기 옵션

오브젝트를 닫을 때 적용되는 옵션. 초기값은 MQCO\_NONE입니다. 암시적 다시 열기 조작 중에는 이 속성이 무시되고, 이 경우 MQCO\_NONE의 값이 항상 사용됩니다.

### 연결 참조

(로컬) 큐 관리자에 대한 필수 연결을 제공하는 ImqQueueManager 오브젝트의 참조입니다. ImqQueueManager 오브젝트의 경우, 오브젝트 자체입니다. 초기값은 0입니다.

**참고:** 이 참조를 이름 지정된 큐의 큐 관리자(리모트)를 식별하는 큐 관리자 이름과 혼동하지 마십시오.

### 설명

큐 관리자, 큐, 이름 목록 또는 프로세스로 구성된 설명적 이름입니다(최대 64자). 이 속성은 읽기 전용입니다.

### 이름

큐 관리자, 큐, 이름 목록 또는 프로세스로 구성된 이름입니다(최대 48자). 초기값은 널 문자열입니다. 모델 큐의 이름은 **open** 다음에 결과 동적 큐의 이름으로 변경됩니다.

**참고:** ImqQueueManager는 기본 큐 관리자를 나타내는 널 이름을 가질 수 있습니다. 성공적인 열기 이후 이름이 실제 큐 관리자로 변경됩니다. ImqDistributionList는 동적이고 널 이름이 있어야 합니다.

### 다음 관리 오브젝트

특별한 순서 없이 이 오브젝트와 동일한 연결 참조를 갖는 이 클래스의 다음 오브젝트입니다. 초기값은 0입니다.

### 열기 옵션

오브젝트를 열 때 적용되는 옵션. 초기값은 MQOO\_INQUIRE입니다. 적절한 값을 설정하는 두 가지 방법이 있습니다.

1. 열기 옵션을 설정하지 않고 열기 메소드를 사용하지 마십시오. IBM MQ는 열기 옵션을 자동으로 조정하고, 필요에 따라 오브젝트를 자동으로 열고, 다시 열고, 닫습니다. 이렇게 하면 IBM MQ에서 openFor 메소드를 사용하기 때문에 불필요한 다시 열기 조작으로 이어지고, 열기 옵션을 증분식으로만 추가합니다.
2. MQI 호출로 이어지는 메소드를 사용하기 전에 열기 옵션을 설정하십시오(1708 페이지의 『C++ 및 MQI 상호 참조』 참조). 이렇게 하면 불필요한 다시 열기 조작이 발생하지 않습니다. 잠재된 다시 열기 문제가 발생할 가능성이 높으면 열기 옵션을 명시적으로 설정하십시오(다시 열기 참조).

열기 메소드를 사용하는 경우, 먼저 열기 옵션이 적절한지 확인해야 합니다. 그러나, 열기 메소드 사용이 필수는 아닙니다. IBM MQ에 사례 1과 같은 동작이 계속 나타나지만, 이 환경에서는 해당 동작이 효율적입니다.

0은 올바른 값이 아닙니다. 오브젝트를 열기 전에 적절한 값을 설정하십시오. 이 작업은 **setOpenOptions** (*IOpenOptions*)과 **open** ()을 차례로 사용하거나 **openFor** (*IRequiredOpenOption*)를 사용하여 완료할 수 있습니다.

### 참고:

1. MQOO\_OUTPUT is substituted for MQOO\_INQUIRE during the **open** method for a distribution list, as MQOO\_OUTPUT is the only valid **open option** at this time. 그러나 **open** 메소드를 사용하는 애플리케이션 프로그램에서 항상 MQOO\_OUTPUT을 명시적으로 설정하는 것이 좋습니다.
2. 클래스의 **resolved queue manager name** 및 **resolved queue name** 속성을 사용하려면 MQOO\_RESOLVE\_NAMES를 지정하십시오.

## 열기 상태

오브젝트가 열려 있는지(TRUE) 또는 닫혀 있는지(FALSE) 여부. 초기값은 FALSE입니다. 이 속성은 읽기 전용입니다.

## 이전 관리 오브젝트

특별한 순서 없이 이 오브젝트와 동일한 연결 참조를 갖는 이 클래스의 이전 오브젝트. 초기값은 0입니다.

## queue-manager-identifier

큐 관리자 ID. 이 속성은 읽기 전용입니다.

## 구성자

### ImqObject();

기본 구성자입니다.

### ImqObject( const ImqObject & object );

복사 구성자입니다. 열기 상태는 FALSE가 됩니다.

## 클래스 메소드(공용)

### static MQLONG behavior();

작동을 리턴합니다.

### void setBehavior( const MQLONG behavior = 0 );

작동을 설정합니다.

## 오브젝트 메소드(공용)

### void operator = ( const ImqObject & object );

필요에 따라 닫기를 수행하고 *object*에서 인스턴스 데이터를 복사합니다. 열기 상태는 FALSE가 됩니다.

### ImqBoolean alterationDate( ImqString & date );

대체 날짜의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

### ImqString alterationDate();

오류 가능성에 대한 표시 없이 대체 날짜를 리턴합니다.

### ImqBoolean alterationTime( ImqString & time );

대체 시간의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

### ImqString alterationTime();

오류 가능성에 대한 표시 없이 대체 시간을 리턴합니다.

### ImqString alternateUserId ( ) const ;

대체 사용자 ID의 사본을 리턴합니다.

### ImqBoolean setAlternateUserId ( const char \* id );

대체 사용자 ID를 설정합니다. 대체 사용자 ID는 열기 상태가 FALSE인 동안에만 설정할 수 있습니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

### ImqBinary alternateSecurityId() const ;

대체 보안 ID의 사본을 리턴합니다.

### ImqBoolean setAlternateSecurityId( const ImqBinary & token );

대체 보안 ID를 설정합니다. 대체 보안 ID는 열기 상태가 FALSE인 동안에만 설정할 수 있습니다. *token*의 데이터 길이는 0 또는 MQ\_SECURITY\_ID\_LENGTH이어야 합니다. 성공하면 TRUE를 리턴합니다.

### ImqBoolean setAlternateSecurityId( const MQBYTE\* token = 0);

대체 보안 ID를 설정합니다. *token*은 0일 수 있으며, 이는 MQSID\_NONE을 지정하는 것과 같습니다. *token*이 0이 아니면 MQ\_SECURITY\_ID\_LENGTH 바이트의 2진 데이터를 처리해야 합니다. MQSID\_NONE과 같은 사전정의된 값을 사용하는 경우, 서명이 일치하도록 캐스트를 만들어야 할 수 있습니다. 예: (MQBYTE\*)MQSID\_NONE.

대체 보안 ID는 열기 상태가 TRUE인 동안에만 설정할 수 있습니다. 성공하면 TRUE를 리턴합니다.

### ImqBoolean setAlternateSecurityId( const unsigned char \* id = 0);

대체 보안 ID를 설정합니다.

**ImqBoolean close ();**

열기 상태를 FALSE로 설정합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG closeOptions () const ;**

닫기 옵션을 리턴합니다.

**void setCloseOptions ( const MQLONG options );**

닫기 옵션을 설정합니다.

**ImqQueueManager \* connectionReference () const ;**

연결 참조를 리턴합니다.

**void setConnectionReference ( ImqQueueManager & manager );**

연결 참조를 설정합니다.

**void setConnectionReference ( ImqQueueManager \* manager = 0 );**

연결 참조를 설정합니다.

**virtual ImqBoolean description ( ImqString & description ) = 0 ;**

설명의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString description ();**

오류 가능성에 대한 표시 없이 설명의 사본을 리턴합니다.

**virtual ImqBoolean name ( ImqString & name );**

이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString name ();**

오류 가능성에 대한 표시 없이 이름의 사본을 리턴합니다.

**ImqBoolean setName ( const char \* name = 0 );**

이름을 설정합니다. 이름은 열기 상태가 FALSE인 동안에만 그리고 ImqQueueManager의 경우에는 연결 상태가 FALSE인 동안에만 설정할 수 있습니다. 성공하면 TRUE를 리턴합니다.

**ImqObject \* nextManagedObject () const ;**

다음 관리 오브젝트를 리턴합니다.

**ImqBoolean open ();**

다른 속성 중에서 열기 옵션 및 이름을 사용하여 오브젝트를 필요에 따라 여는 방법으로 열기 상태를 TRUE로 변경합니다. 이 메소드는 ImqQueueManager 연결 상태가 TRUE인지 확인하기 위해 필요하면 연결 참조 정보 및 ImqQueueManager 연결 메소드를 사용합니다. 열기 상태를 리턴합니다.

**ImqBoolean openFor ( const MQLONG required-options = 0 );**

열기 옵션 또는 *required-options* 매개변수 값이 의미하는 동작을 보장하는 열기 옵션을 사용하여 오브젝트를 열었는지 확인하려고 합니다.

*required-options*가 0이면 입력이 필수이고 어떤 입력 옵션이라도 충분합니다. 따라서 열기 옵션에 이미 다음 중 하나가 있는 경우:

- MQOO\_INPUT\_AS\_Q\_DEF
- MQOO\_INPUT\_SHARED
- MQOO\_INPUT\_EXCLUSIVE

열기 옵션은 이미 만족스러우며 변경되지 않습니다. 열기 옵션에 이들 옵션 중 하나라도 아직 없으면 MQOO\_INPUT\_AS\_Q\_DEF가 열기 옵션에 설정됩니다.

*required-options*가 0이 아니면 필수 옵션이 열기 옵션에 추가됩니다. *required-options*이 이들 옵션 중 하나이면 다른 항목이 재설정됩니다.

열기 옵션 중 하나라도 변경되고 오브젝트가 이미 열려 있으면 오브젝트가 일시적으로 닫혔다가 열기 옵션을 조정하기 위해 다시 열립니다.

성공하면 TRUE를 리턴합니다. 성공은 오브젝트가 적절한 옵션과 함께 열리는 것을 의미합니다.

**MQLONG openOptions () const ;**

열기 옵션을 리턴합니다.

**ImqBoolean setOpenOptions ( const MQLONG options );**

열기 옵션을 설정합니다. 열기 옵션은 열기 상태가 FALSE인 동안에만 설정할 수 있습니다. 성공하면 TRUE를 리턴합니다.

**ImqBoolean openStatus ( ) const ;**

열기 상태를 리턴합니다.

**ImqObject \* previousManagedObject ( ) const ;**

이전 관리 오브젝트를 리턴합니다.

**ImqBoolean queueManagerIdentifier( ImqString & id );**

큐 관리자 ID의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString queueManagerIdentifier();**

오류 가능성에 대한 표시 없이 큐 관리자 ID를 리턴합니다.

## 오브젝트 메소드(보호됨)

**virtual ImqBoolean closeTemporarily ( );**

다시 열기 전에 오브젝트를 안전하게 닫습니다. 성공하면 TRUE를 리턴합니다. 이 메소드는 열기 상태가 TRUE라고 가정합니다.

**MQHCONN connectionHandle ( ) const ;**

연결 참조와 연관된 MQHCONN을 리턴합니다. 연결 참조가 없거나 관리자가 연결되어 있지 않으면 이 값은 0입니다.

**ImqBoolean inquire ( const MQLONG int-attr, MQLONG & value );**

색인이 MQIA\_\* 값인 정수 값을 리턴합니다. 오류가 발생하면 값이 MQIAV\_UNDEFINED로 설정됩니다.

**ImqBoolean inquire ( const MQLONG char-attr, char \* & buffer, const size\_t length );**

색인이 MQCA\_\* 값인 문자열을 리턴합니다.

**참고:** 이 메소드는 둘 다 하나의 속성 값만 리턴합니다. 둘 이상의 값에 스냅샷이 필요하고 해당 값이 잠시 서로 일치하는 경우, IBM MQ C++은 이 기능을 제공하지 않으며 적절한 매개변수를 사용하여 MQINQ 호출을 사용해야 합니다.

**virtual void openInformationDisperse ( );**

MQOPEN 호출 이후 바로 MQOD 데이터 구조의 변수 섹션에서 정보를 분산시킵니다.

**virtual ImqBoolean openInformationPrepare ( );**

MQOPEN 호출 직전에 MQOD 데이터 구조의 변수 섹션에 대한 정보를 준비하고, 성공하면 TRUE를 리턴합니다.

**ImqBoolean set ( const MQLONG int-attr, const MQLONG value );**

IBM MQ 정수 속성을 설정합니다.

**ImqBoolean set ( const MQLONG char-attr, const char \* buffer, const size\_t required-length );**

IBM MQ 문자 속성을 설정합니다.

**void setNextManagedObject ( const ImqObject \* object = 0 );**

다음 관리 오브젝트를 설정합니다.

주의: 관리 오브젝트 목록이 손상되지 않는 것이 확실한 경우에만 이 함수를 사용하십시오.

**void setPreviousManagedObject ( const ImqObject \* object = 0 );**

이전 관리 오브젝트를 설정합니다.

주의: 관리 오브젝트 목록이 손상되지 않는 것이 확실한 경우에만 이 함수를 사용하십시오.

## 오브젝트 데이터(보호됨)

**MQHOBJ ohobj**

IBM MQ 오브젝트 핸들(열기 상태가 TRUE인 경우에만 유효함).

**MQOD omqod**

임베드된 MQOD 데이터 구조. 이 데이터 구조에 할당된 스토리지 양은 MQOD 버전 2에 필요한 양입니다. 버전 번호 (*omqod.Version*)를 검사하고 다음과 같이 다른 필드에 액세스하십시오.

**MQOD\_VERSION\_1**

*omqod*의 다른 모든 필드에 액세스할 수 있습니다.

**MQOD\_VERSION\_2**

*omqod*의 다른 모든 필드에 액세스할 수 있습니다.

### **MQOD\_VERSION\_3**

*omqod.pmqod*는 동적으로 할당되고 크기가 큰 MQOD에 대한 포인터입니다. *omqod*의 다른 필드에는 액세스할 수 없습니다. *omqod.pmqod*에 의해 처리되는 모든 필드에 액세스할 수 있습니다.

**참고:** *omqod.pmqod.Version*은 *omqod.Version* 미만이며, 이는 IBM MQ MQI client에 IBM MQ 서버보다 더 많은 기능이 있음을 의미합니다.

### **이유 코드**

- MQRC\_ATTRIBUTE\_LOCKED
- MQRC\_INCONSISTENT\_OBJECT\_STATE
- MQRC\_NO\_CONNECTION\_REFERENCE
- MQRC\_STORAGE\_NOT\_AVAILABLE
- MQRC\_REOPEN\_SAVED\_CONTEXT\_ERR
- (MQCLOSE의 이유 코드)
- (MQCONN의 이유 코드)
- (MQINQ의 이유 코드)
- (MQOPEN의 이유 코드)
- (MQSET의 이유 코드)

### **ImqProcess C++ 클래스**

이 클래스는 트리거 모니터에 의해 트리거할 수 있는 애플리케이션 프로세스(MQOT\_PROCESS 유형의 IBM MQ 오브젝트)를 캡슐화합니다.

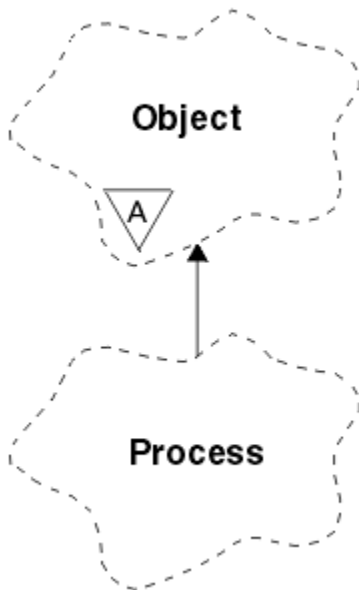


그림 30. *ImqProcess* 클래스

- [1770 페이지의 『오브젝트 속성』](#)
- [1771 페이지의 『구성자』](#)
- [1771 페이지의 『오브젝트 메소드\(공용\)』](#)

### **오브젝트 속성**

#### **애플리케이션 ID**

애플리케이션 프로세스의 ID. 이 속성은 읽기 전용입니다.

## 애플리케이션 유형

애플리케이션 프로세스의 유형. 이 속성은 읽기 전용입니다.

## 환경 데이터

프로세스의 환경 정보. 이 속성은 읽기 전용입니다.

## 사용자 데이터

프로세스에 대한 사용자 데이터. 이 속성은 읽기 전용입니다.

## 구성자

### **ImqProcess();**

기본 구성자입니다.

### **ImqProcess( const ImqProcess & process );**

복사 구성자입니다. ImqObject 열기 상태는 FALSE입니다.

### **ImqProcess( const char \* name );**

ImqObject 이름을 설정합니다.

## 오브젝트 메소드(공용)

### **void operator = ( const ImqProcess & process );**

필요에 따라 닫기를 수행하고, *process*에서 인스턴스 데이터를 복사합니다. ImqObject 열기 상태는 FALSE가 됩니다.

### **ImqBoolean applicationId ( ImqString & id );**

애플리케이션 ID의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

### **ImqString applicationId ( );**

오류 가능성에 대한 표시 없이 애플리케이션 ID를 리턴합니다.

### **ImqBoolean applicationType ( MQLONG & type );**

애플리케이션 유형의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

### **MQLONG applicationType ( );**

오류 가능성에 대한 표시 없이 애플리케이션 유형을 리턴합니다.

### **ImqBoolean environmentData ( ImqString & data );**

환경 데이터의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

### **ImqString environmentData ( );**

오류 가능성에 대한 표시 없이 환경 데이터를 리턴합니다.

### **ImqBoolean userData ( ImqString & data );**

사용자 데이터의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

### **ImqString userData ( );**

오류 가능성에 대한 표시 없이 사용자 데이터를 리턴합니다.

## ImqPutMessageOptions C++ 클래스

이 클래스는 MQPMO 데이터 구조를 캡슐화합니다.

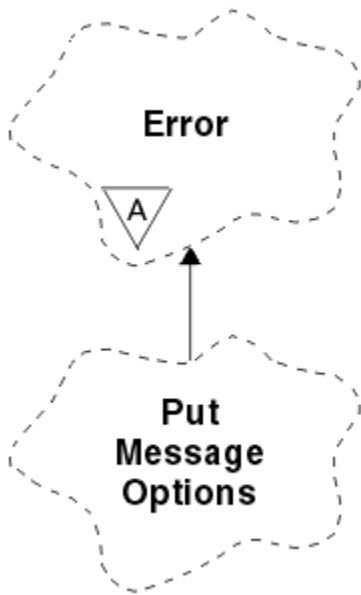


그림 31. *ImqPutMessageOptions* 클래스

- [1772 페이지의 『오브젝트 속성』](#)
- [1773 페이지의 『구성자』](#)
- [1773 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1774 페이지의 『오브젝트 데이터\(보호됨\)』](#)
- [1774 페이지의 『이유 코드』](#)

## 오브젝트 속성

### 컨텍스트 참조

메시지 컨텍스트를 제공하는 *ImqQueue*. 처음에는 참조가 없습니다.

### 옵션

메시지 넣기 옵션입니다. 초기값은 *MQPMO\_NONE*입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- *MQPMO\_SYNCPOINT*
- *MQPMO\_NO\_SYNCPOINT*
- *MQPMO\_NEW\_MSG\_ID*
- *MQPMO\_NEW\_CORREL\_ID*
- *MQPMO\_LOGICAL\_ORDER*
- *MQPMO\_NO\_CONTEXT*
- *MQPMO\_DEFAULT\_CONTEXT*
- *MQPMO\_PASS\_IDENTITY\_CONTEXT*
- *MQPMO\_PASS\_ALL\_CONTEXT*
- *MQPMO\_SET\_IDENTITY\_CONTEXT*
- *MQPMO\_SET\_ALL\_CONTEXT*
- *MQPMO\_ALTERNATE\_USER\_AUTHORITY*
- *MQPMO\_FAIL\_IF QUIESCING*

### 레코드 필드

메시지를 넣을 때 메시지 넣기 레코드의 포함을 제어하는 플래그. 초기값은 *MQPMRF\_NONE*입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- *MQPMRF\_MSG\_ID*



- MQPMRF\_CORREL\_ID
- MQPMRF\_GROUP\_ID
- MQPMRF\_FEEDBACK
- MQPMRF\_ACCOUNTING\_TOKEN

ImqMessageTracker 속성은 지정된 필드의 오브젝트에서 가져옵니다. ImqMessageTracker 속성은 지정되지 않은 필드의 ImqMessage 오브젝트에서 가져옵니다.

#### 해석된 큐 관리자 이름

넣는 동안 판별된 목적지 큐 관리자의 이름. 초기값은 널입니다. 이 속성은 읽기 전용입니다.

#### 해석된 큐 이름

넣는 동안 판별된 목적지 큐의 이름. 초기값은 널입니다. 이 속성은 읽기 전용입니다.

#### 동기점 참여

동기점 제어에서 메시지를 넣는 경우 TRUE입니다.

## 구성자

### ImqPutMessageOptions();

기본 구성자입니다.

### ImqPutMessageOptions( const ImqPutMessageOptions & pmo );

복사 구성자입니다.

## 오브젝트 메소드(공용)

### void operator = ( const ImqPutMessageOptions & pmo );

pmo에서 인스턴스 데이터를 복사하고, 기존 인스턴스 데이터를 대체합니다.

### ImqQueue \* contextReference ( ) const ;

컨텍스트 참조를 리턴합니다.

### void setContextReference ( const ImqQueue & queue );

컨텍스트 참조를 설정합니다.

### void setContextReference ( const ImqQueue \* queue = 0 );

컨텍스트 참조를 설정합니다.

### MQLONG options ( ) const ;

옵션을 리턴합니다.

### void setOptions ( const MQLONG options );

동기점 참여 값을 포함하여 옵션을 설정합니다.

### MQLONG recordFields ( ) const ;

레코드 필드를 리턴합니다.

### void setRecordFields ( const MQLONG fields );

레코드 필드를 설정합니다.

### ImqString resolvedQueueManagerName ( ) const ;

해석된 큐 관리자 이름의 사본을 리턴합니다.

### ImqString resolvedQueueName ( ) const ;

해석된 큐 이름의 사본을 리턴합니다.

### ImqBoolean syncPointParticipation ( ) const ;

동기점 참여 값을 리턴합니다. 옵션에 MQPMO\_SYNCPOINT가 포함되면 TRUE입니다.

### void setSyncPointParticipation ( const ImqBoolean sync );

동기점 참여 값을 설정합니다. sync가 TRUE이면 MQPMO\_SYNCPOINT를 포함하고 MQPMO\_NO\_SYNCPOINT를 제외하도록 옵션이 대체됩니다. sync가 FALSE이면 MQPMO\_NO\_SYNCPOINT를 포함하고 MQPMO\_SYNCPOINT를 제외하도록 옵션이 대체됩니다.

## 오브젝트 데이터(보호됨)

### MQPMO omqpmo

MQPMO 데이터 구조.

### 이유 코드

- MQRC\_STORAGE\_NOT\_AVAILABLE

## ImqQueue C++ 클래스

이 클래스는 메시지 큐(MQOT\_Q 유형의 IBM MQ 오브젝트)를 캡슐화합니다.

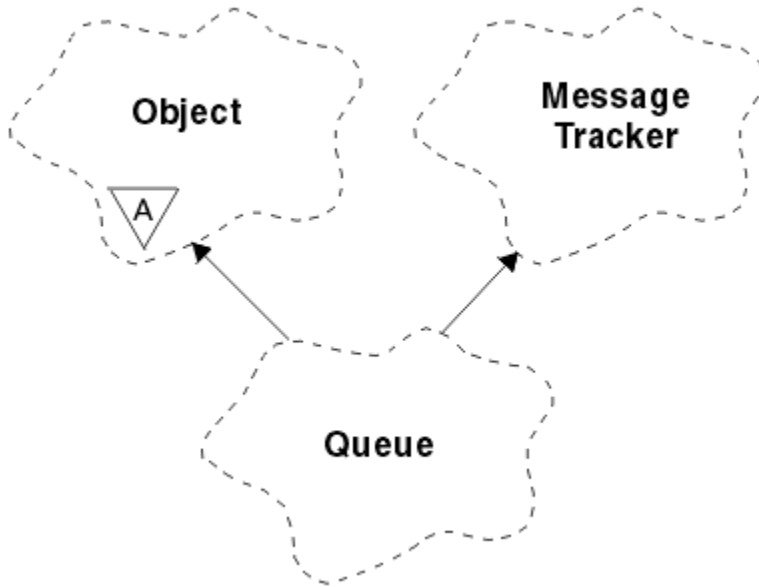


그림 32. ImqQueue 클래스

이 클래스는 1716 페이지의 표 862에 나열된 MQI 호출과 관련이 있습니다.

- [1774 페이지의 『오브젝트 속성』](#)
- [1777 페이지의 『구성자』](#)
- [1777 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1783 페이지의 『오브젝트 메소드\(보호됨\)』](#)
- [1783 페이지의 『이유 코드』](#)

### 오브젝트 속성

#### 백아웃 리큐 이름

초과 백아웃 리큐 이름. 이 속성은 읽기 전용입니다.

#### 백아웃 임계값

백아웃 임계값. 이 속성은 읽기 전용입니다.

#### 기본 큐 이름

알리어스가 해석되는 큐의 이름. 이 속성은 읽기 전용입니다.

#### 클러스터 이름

클러스터 이름. 이 속성은 읽기 전용입니다.

#### 클러스터 이름 목록 이름

클러스터 이름 목록 이름. 이 속성은 읽기 전용입니다.

#### 클러스터 워크로드 순위

클러스터 워크로드 순위입니다. 이 속성은 읽기 전용입니다.

**클러스터 워크로드 우선순위**

클러스터 워크로드 우선순위입니다. 이 속성은 읽기 전용입니다.

**클러스터 워크로드 사용 큐**

클러스터 워크로드 사용 큐 값. 이 속성은 읽기 전용입니다.

**작성 날짜**

큐 작성 날짜. 이 속성은 읽기 전용입니다.

**작성 시간**

큐 작성 시간. 이 속성은 읽기 전용입니다.

**현재 용량**

큐에 있는 메시지의 수. 이 속성은 읽기 전용입니다.

**기본 바인드**

기본 바인드. 이 속성은 읽기 전용입니다.

**기본 입력 열기 옵션**

기본 입력 열기 옵션. 이 속성은 읽기 전용입니다.

**기본 지속성**

기본 메시지 지속성. 이 속성은 읽기 전용입니다.

**기본 우선순위**

기본 메시지 우선순위입니다. 이 속성은 읽기 전용입니다.

**정의 유형**

큐 정의 유형입니다. 이 속성은 읽기 전용입니다.

**용량 상한 이벤트**

큐 용량 상한 이벤트의 제어 속성. 이 속성은 읽기 전용입니다.

**용량 상한**

큐 용량의 상한. 이 속성은 읽기 전용입니다.

**용량 하한 이벤트**

큐 용량 하한 이벤트의 제어 속성. 이 속성은 읽기 전용입니다.

**큐 용량 하한**

큐 용량의 하한. 이 속성은 읽기 전용입니다.

**용량 최대 이벤트**

큐 용량 최대 이벤트의 제어 속성. 이 속성은 읽기 전용입니다.

**분배 목록 참조**

이 항목을 포함하여 메시지를 둘 이상의 큐에 분배하는 데 사용할 수 있는 ImqDistributionList에 대한 선택적 참조. 초기값은 널입니다.

**참고:** ImqQueue 오브젝트를 열면 해당 오브젝트가 참조하는 열린 ImqDistributionList 오브젝트가 자동으로 닫힙니다.

**분배 목록**

분배 목록을 지원하기 위한 전송 큐의 기능. 이 속성은 읽기 전용입니다.

**동적 큐 이름**

동적 큐 이름. 초기값은 AMQ.\*입니다. 모든 AIX, Linux, and Windows 플랫폼의 경우.

**백아웃 횟수 기록**

백아웃 횟수 기록 여부. 이 속성은 읽기 전용입니다.

**색인 유형**

색인 유형. 이 속성은 읽기 전용입니다.

**Get 금지**

가져오기 조작이 허용되는지 여부. 초기값은 큐 정의에 따라 다릅니다. 이 속성은 알리어스 또는 로컬 큐에 대해서만 유효합니다.

**Put 금지**

넣기 조작이 허용되는지 여부. 초기값은 큐 정의에 따라 다릅니다.

**이니시에이션 큐 이름**

이니시에이션 큐의 이름. 이 속성은 읽기 전용입니다.

**최대 용량**

큐에 허용된 최대 메시지 수. 이 속성은 읽기 전용입니다.

**최대 메시지 길이**

이 큐에 있는 메시지의 최대 길이로, 연관된 큐 관리자가 관리하는 큐의 최대값보다 작을 수 있습니다. 이 속성은 읽기 전용입니다.

**메시지 전달 순서**

메시지 우선순위가 적절한지 여부. 이 속성은 읽기 전용입니다.

**다음 분산 큐**

특별한 순서 없이 이 오브젝트와 동일한 **분배 목록 참조**를 갖는 이 클래스의 다음 오브젝트. 초기값은 0입니다.

체인의 오브젝트가 삭제되면 해당 분산 큐 링크가 더 이상 삭제된 오브젝트를 가리키지 않도록 이전 오브젝트와 다음 오브젝트가 업데이트됩니다.

**비지속 메시지 클래스**

이 큐에 넣은 비지속 메시지의 신뢰성 레벨. 이 속성은 읽기 전용입니다.

**열기 입력 수**

입력을 위해 연 ImqQueue 오브젝트의 수. 이 속성은 읽기 전용입니다.

**열기 출력 수**

출력을 위해 연 ImqQueue 오브젝트의 수. 이 속성은 읽기 전용입니다.

**이전 분산 큐**

특별한 순서 없이 이 오브젝트와 동일한 **분배 목록 참조**를 갖는 이 클래스의 이전 오브젝트. 초기값은 0입니다.

체인의 오브젝트가 삭제되면 해당 분산 큐 링크가 더 이상 삭제된 오브젝트를 가리키지 않도록 이전 오브젝트와 다음 오브젝트가 업데이트됩니다.

**프로세스 이름**

프로세스 정의의 이름. 이 속성은 읽기 전용입니다.

**큐 계정**

큐에 대한 계정 정보의 레벨. 이 속성은 읽기 전용입니다.

**큐 관리자 이름**

큐가 상주하는 큐 관리자(리모트)의 이름. 여기서 이름을 지정한 큐 관리자와, 연결을 제공하는 (로컬) 큐 관리자를 참조하는 ImqObject **연결 참조**를 혼동하지 마십시오. 초기값은 널입니다.

**큐 모니터링**

큐에 대한 모니터링 데이터 컬렉션의 레벨. 이 속성은 읽기 전용입니다.

**큐 통계**

큐에 대한 통계 데이터의 레벨. 이 속성은 읽기 전용입니다.

**큐 유형**

큐 유형. 이 속성은 읽기 전용입니다.

**리모트 큐 관리자 이름**

리모트 큐 관리자의 이름. 이 속성은 읽기 전용입니다.

**리모트 큐 이름**

리모트 큐 관리자에 인식되는 리모트 큐의 이름. 이 속성은 읽기 전용입니다.

**해석된 큐 관리자 이름**

해석된 큐 관리자 이름. 이 속성은 읽기 전용입니다.

**해석된 큐 이름**

해석된 큐 이름. 이 속성은 읽기 전용입니다.

**보유 간격**

큐 보유 간격. 이 속성은 읽기 전용입니다.

**scope**

큐 정의의 범위. 이 속성은 읽기 전용입니다.

**서비스 간격(service interval)**

서비스 간격. 이 속성은 읽기 전용입니다.

### 서비스 간격 이벤트(service interval event)

서비스 간격 이벤트의 제어 속성. 이 속성은 읽기 전용입니다.

### 공유 가용성

큐를 공유할 수 있는지 여부를 지정합니다. 이 속성은 읽기 전용입니다.

### 스토리지 클래스

스토리지 클래스. 이 속성은 읽기 전용입니다.

### 전송 큐 이름

전송 큐의 이름. 이 속성은 읽기 전용입니다.

### 트리거 제어

트리거 제어. 초기값은 큐 정의에 따라 다릅니다. 이 속성은 로컬 큐에 대해서만 유효합니다.

### 트리거 데이터

트리거 데이터. 초기값은 큐 정의에 따라 다릅니다. 이 속성은 로컬 큐에 대해서만 유효합니다.

### 트리거 용량

트리거 용량. 초기값은 큐 정의에 따라 다릅니다. 이 속성은 로컬 큐에 대해서만 유효합니다.

### 트리거 메시지 우선순위

트리거에 대한 메시지 우선순위 임계값입니다. 초기값은 큐 정의에 따라 다릅니다. 이 속성은 로컬 큐에 대해서만 유효합니다.

### 트리거 유형

트리거 유형. 초기값은 큐 정의에 따라 다릅니다. 이 속성은 로컬 큐에 대해서만 유효합니다.

### 사용법

사용법. 이 속성은 읽기 전용입니다.

## 구성자

### **ImqQueue();**

기본 구성자입니다.

### **ImqQueue( const ImqQueue & queue );**

복사 구성자입니다. ImqObject 열기 상태는 FALSE가 됩니다.

### **ImqQueue( const char \* name );**

ImqObject 이름을 설정합니다.

## 오브젝트 메소드(공용)

### **void operator = ( const ImqQueue & queue );**

필요에 따라 닫기를 수행하고, queue에서 인스턴스 데이터를 복사합니다. ImqObject 열기 상태는 FALSE가 됩니다.

### **ImqBoolean backoutRequeueName ( ImqString & name );**

백아웃 리큐 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

### **ImqString backoutRequeueName ( );**

오류 가능성에 대한 표시 없이 백아웃 리큐 이름을 리턴합니다.

### **ImqBoolean backoutThreshold ( MQLONG & threshold );**

백아웃 임계값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

### **MQLONG backoutThreshold ( );**

오류 가능성에 대한 표시 없이 백아웃 임계값 값을 리턴합니다.

### **ImqBoolean baseQueueName ( ImqString & name );**

기본 큐 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

### **ImqString baseQueueName ( );**

오류 가능성에 대한 표시 없이 기본 큐 이름을 리턴합니다.

### **ImqBoolean clusterName( ImqString & name );**

클러스터 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString clusterName();**  
오류 가능성에 대한 표시 없이 클러스터 이름을 리턴합니다.

**ImqBoolean clusterNameIstName( ImqString & name );**  
클러스터 이름 목록 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString clusterNameIstName();**  
오류 가능성에 대한 표시 없이 클러스터 이름 목록 이름을 리턴합니다.

**ImqBoolean clusterWorkLoadPriority ( MQLONG & priority );**  
클러스터 워크로드 우선순위 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG clusterWorkLoadPriority ();**  
오류 가능성에 대한 표시 없이 클러스터 워크로드 우선순위 값을 리턴합니다.

**ImqBoolean clusterWorkLoadRank ( MQLONG & rank );**  
클러스터 워크로드 순위 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG clusterWorkLoadRank ();**  
오류 가능성에 대한 표시 없이 클러스터 워크로드 순위 값을 리턴합니다.

**ImqBoolean clusterWorkLoadUseQ ( MQLONG & useq );**  
클러스터 워크로드 사용 큐 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG clusterWorkLoadUseQ ();**  
오류 가능성에 대한 표시 없이 클러스터 워크로드 사용 큐 값을 리턴합니다.

**ImqBoolean creationDate ( ImqString & date );**  
작성 날짜의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString creationDate ();**  
오류 가능성에 대한 표시 없이 작성 날짜를 리턴합니다.

**ImqBoolean creationTime ( ImqString & time );**  
작성 시간의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString creationTime ();**  
오류 가능성에 대한 표시 없이 작성 시간을 리턴합니다.

**ImqBoolean currentDepth ( MQLONG & depth );**  
현재 용량의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG currentDepth ();**  
오류 가능성에 대한 표시 없이 현재 용량을 리턴합니다.

**ImqBoolean defaultInputOpenOption ( MQLONG & option );**  
기본 입력 열기 옵션의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG defaultInputOpenOption ();**  
오류 가능성에 대한 표시 없이 기본 입력 열기 옵션을 리턴합니다.

**ImqBoolean defaultPersistence( MQLONG & persistence );**  
기본 지속성의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG defaultPersistence ();**  
오류 가능성에 대한 표시 없이 기본 지속성을 리턴합니다.

**ImqBoolean defaultPriority( MQLONG & priority );**  
기본 우선순위의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG defaultPriority ();**  
오류 가능성에 대한 표시 없이 기본 우선순위를 리턴합니다.

**ImqBoolean defaultBind( MQLONG & bind );**  
기본 바인드의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG defaultBind ();**  
오류 가능성에 대한 표시 없이 기본 바인드를 리턴합니다.

**ImqBoolean definitionType( MQLONG & type );**  
정의 유형의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG definitionType ();**  
오류 가능성에 대한 표시 없이 정의 유형을 리턴합니다.

**ImqBoolean depthHighEvent( MQLONG & event );**

용량 상한 이벤트의 인에이블먼트 상태의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG depthHighEvent ( );**

오류 가능성에 대한 표시 없이 용량 상한 이벤트의 인에이블먼트 상태를 리턴합니다.

**ImqBoolean depthHighLimit( MQLONG & limit );**

용량 상한의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG depthHighLimit ( );**

오류 가능성에 대한 표시 없이 용량 상한 값을 리턴합니다.

**ImqBoolean depthLowEvent( MQLONG & event );**

용량 하한 이벤트의 인에이블먼트 상태의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG depthLowEvent ( );**

오류 가능성에 대한 표시 없이 용량 하한 이벤트의 인에이블먼트 상태를 리턴합니다.

**ImqBoolean depthLowLimit( MQLONG & limit );**

용량 하한의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG depthLowLimit ( );**

오류 가능성에 대한 표시 없이 용량 하한 값을 리턴합니다.

**ImqBoolean depthMaximumEvent( MQLONG & event );**

용량 최대 이벤트의 인에이블먼트 상태의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG depthMaximumEvent ( );**

오류 가능성에 대한 표시 없이 용량 최대 이벤트의 인에이블먼트 상태를 리턴합니다.

**ImqDistributionList \* distributionListReference ( ) const ;**

분배 목록 참조를 리턴합니다.

**void setDistributionListReference( ImqDistributionList & list );**

분배 목록 참조를 설정합니다.

**void setDistributionListReference ( ImqDistributionList \* list = 0 );**

분배 목록 참조를 설정합니다.

**ImqBoolean distributionLists( MQLONG & support );**

분배 목록 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG distributionLists ( );**

오류 가능성에 대한 표시 없이 분배 목록 값을 리턴합니다.

**ImqBoolean setDistributionLists ( const MQLONG support );**

분배 목록 값을 설정합니다. 성공하면 TRUE를 리턴합니다.

**ImqString dynamicQueueName ( ) const ;**

동적 큐 이름의 사본을 리턴합니다.

**ImqBoolean setDynamicQueueName ( const char \* name );**

동적 큐 이름을 설정합니다. 동적 큐 이름은 열기 상태가 FALSE인 동안에만 설정할 수 있습니다. 성공하면 TRUE를 리턴합니다.

**ImqBoolean get( ImqMessage & msg, ImqGetMessageOptions & options );**

지정된 *options*를 사용하여 큐에서 메시지를 검색합니다. ImqObject 열기 옵션이 *options*에 따라 MQOO\_INPUT\_\* 값 또는 MQOO\_BROWSE 값 중 하나를 포함하도록 필요하면 ImqObject **openFor**를 호출합니다. *msg* 오브젝트에 ImqCache 자동 버퍼가 있으면 검색되는 메시지를 수용할 수 있도록 버퍼가 커집니다. **clearMessage** 메소드는 검색 전에 *msg* 오브젝트에 대해 호출됩니다.

이 메소드는 성공하면 TRUE를 리턴합니다.

**참고:** 이 이유 코드가 경고로 분류된 경우에도 ImqObject 이유 코드가 MQRC\_TRUNCATED\_MSG\_FAILED 이면 메소드 호출 결과가 FALSE입니다. 잘린 메시지가 허용되는 경우, ImqCache 메시지 길이는 잘린 길이를 반영합니다. 두 이벤트에서 ImqMessage 총 메시지 길이는 사용 가능했던 바이트 수를 표시합니다.

**ImqBoolean get( ImqMessage & msg );**

기본 메시지 가져오기 옵션을 사용하는 점을 제외하고, 이전 메소드와 같습니다.

**ImqBoolean get( ImqMessage & msg, ImqGetMessageOptions & options, const size\_t buffer-size );**  
 대체 *buffer-size*가 표시되는 점을 제외하고, 이전 두 개 메소드와 같습니다. *msg* 오브젝트가 *ImqCache* 자동 버퍼를 사용하는 경우, 메시지 검색 전에 **resizeBuffer** 메소드가 *msg* 오브젝트에서 호출되고 대용량 메시지를 수용할 수 있도록 버퍼가 커지지 않습니다.

**ImqBoolean get( ImqMessage & msg, const size\_t buffer-size );**  
 기본 메시지 가져오기 옵션을 사용하는 점을 제외하고, 이전 메소드와 같습니다.

**ImqBoolean hardenGetBackout( MQLONG & harden );**  
 백아웃 횟수 기록 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG hardenGetBackout ( );**  
 오류 가능성에 대한 표시 없이 백아웃 횟수 기록 값을 리턴합니다.

**ImqBoolean indexType( MQLONG & type );**  
 색인 유형의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG indexType( );**  
 오류 가능성에 대한 표시 없이 색인 유형을 리턴합니다.

**ImqBoolean inhibitGet( MQLONG & inhibit );**  
**Get** 금지 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG inhibitGet ( );**  
 오류 가능성에 대한 표시 없이 **Get** 금지 값을 리턴합니다.

**ImqBoolean setInhibitGet ( const MQLONG inhibit );**  
**Get** 금지 값을 설정합니다. 성공하면 TRUE를 리턴합니다.

**ImqBoolean inhibitPut( MQLONG & inhibit );**  
**Put** 금지 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG inhibitPut ( );**  
 오류 가능성에 대한 표시 없이 **Put** 금지 값을 리턴합니다.

**ImqBoolean setInhibitPut ( const MQLONG inhibit );**  
**Put** 금지 값을 설정합니다. 성공하면 TRUE를 리턴합니다.

**ImqBoolean initiationQueueName( ImqString & name );**  
 이니시에이션 큐 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString initiationQueueName ( );**  
 오류 가능성에 대한 표시 없이 이니시에이션 큐 이름을 리턴합니다.

**ImqBoolean maximumDepth( MQLONG & depth );**  
 최대 용량의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG maximumDepth ( );**  
 오류 가능성에 대한 표시 없이 최대 용량을 리턴합니다.

**ImqBoolean maximumMessageLength( MQLONG & length );**  
 최대 메시지 길이의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG maximumMessageLength ( );**  
 오류 가능성에 대한 표시 없이 최대 메시지 길이를 리턴합니다.

**ImqBoolean messageDeliverySequence( MQLONG & sequence );**  
 메시지 전달 순서의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG messageDeliverySequence ( );**  
 오류 가능성에 대한 표시 없이 메시지 전달 순서 값을 리턴합니다.

**ImqQueue \* nextDistributedQueue ( ) const ;**  
 다음 분산 큐를 리턴합니다.

**ImqBoolean nonPersistentMessageClass ( MQLONG & monq );**  
 비지속 메시지 클래스 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG nonPersistentMessageClass ( );**  
 오류 가능성에 대한 표시 없이 비지속 메시지 클래스 값을 리턴합니다.

**ImqBoolean openInputCount( MQLONG & count );**  
 열기 입력 수의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.



**MQLONG openInputCount ( );**

오류 가능성에 대한 표시 없이 열기 입력 수를 리턴합니다.

**ImqBoolean openOutputCount( MQLONG & count );**

열기 출력 수의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG openOutputCount ( );**

오류 가능성에 대한 표시 없이 열기 출력 수를 리턴합니다.

**ImqQueue \* previousDistributedQueue ( ) const ;**

첫 번째 분산 큐를 리턴합니다.

**ImqBoolean processName( ImqString & name );**

프로세스 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString processName ( );**

오류 가능성에 대한 표시 없이 프로세스 이름을 리턴합니다.

**ImqBoolean put( ImqMessage & msg );**

기본 메시지 넣기 옵션을 사용하여 메시지를 큐에 넣습니다. ImqObject 열기 옵션이 MQOO\_OUTPUT을 포함하도록 필요하다면 ImqObject **openFor** 메소드를 사용하십시오.

이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqBoolean put( ImqMessage & msg, ImqPutMessageOptions & pmo );**

지정된 *pmo*를 사용하여 메시지를 큐에 넣습니다. ImqObject 열기 옵션이 MQOO\_OUTPUT 및 (*pmo options*이 MQPMO\_PASS\_IDENTITY\_CONTEXT, MQPMO\_PASS\_ALL\_CONTEXT, MQPMO\_SET\_IDENTITY\_CONTEXT 또는 MQPMO\_SET\_ALL\_CONTEXT 중 하나를 포함하는 경우) 해당 MQOO\_\*\_CONTEXT 값을 포함하도록 필요에 따라 ImqObject **openFor** 메소드를 사용하십시오.

이 메소드는 성공하면 TRUE를 리턴합니다.

참고: *pmo*가 컨텍스트 참조를 포함하는 경우, 참조된 오브젝트가 필요에 따라 컨텍스트를 제공하기 위해 열립니다.

**ImqBoolean queueAccounting ( MQLONG & acctq );**

큐 계정 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG queueAccounting ( );**

오류 가능성에 대한 표시 없이 큐 계정 값을 리턴합니다.

**ImqString queueManagerName ( ) const ;**

큐 관리자 이름을 리턴합니다.

**ImqBoolean setQueueManagerName ( const char \* name );**

큐 관리자 이름을 설정합니다. 큐 관리자 이름은 열기 상태가 FALSE인 동안에만 설정할 수 있습니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqBoolean queueMonitoring ( MQLONG & monq );**

큐 모니터링 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG queueMonitoring ( );**

오류 가능성에 대한 표시 없이 큐 모니터링 값을 리턴합니다.

**ImqBoolean queueStatistics ( MQLONG & statq );**

큐 통계 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG queueStatistics ( );**

오류 가능성에 대한 표시 없이 큐 통계 값을 리턴합니다.

**ImqBoolean queueType( MQLONG & type );**

큐 유형 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG queueType ( );**

오류 가능성에 대한 표시 없이 큐 유형을 리턴합니다.

**ImqBoolean remoteQueueManagerName( ImqString & name );**

리모트 큐 관리자 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString remoteQueueManagerName ( );**

오류 가능성에 대한 표시 없이 리모트 큐 관리자 이름을 리턴합니다.

**ImqBoolean remoteQueueName( ImqString & name );**

리모트 큐 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString remoteQueueName ( );**

오류 가능성에 대한 표시 없이 리모트 큐 이름을 리턴합니다.

**ImqBoolean resolvedQueueManagerName( ImqString & name );**

해석된 큐 관리자 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

참고: 이 메소드는 MQOO\_RESOLVE\_NAMES가 ImqObject 열기 옵션에 속하는 경우를 제외하고 실패합니다.

**ImqString resolvedQueueManagerName( );**

오류 가능성에 대한 표시 없이 해석된 큐 관리자 이름을 리턴합니다.

**ImqBoolean resolvedQueueName( ImqString & name );**

해석된 큐 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

참고: 이 메소드는 MQOO\_RESOLVE\_NAMES가 ImqObject 열기 옵션에 속하는 경우를 제외하고 실패합니다.

**ImqString resolvedQueueName( );**

오류 가능성에 대한 표시 없이 해석된 큐 이름을 리턴합니다.

**ImqBoolean retentionInterval( MQLONG & interval );**

보유 간격의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG retentionInterval ( );**

오류 가능성에 대한 표시 없이 보유 간격을 리턴합니다.

**ImqBoolean scope( MQLONG & scope );**

범위의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG scope ( );**

오류 가능성에 대한 표시 없이 범위를 리턴합니다.

**ImqBoolean serviceInterval( MQLONG & interval );**

서비스 간격의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG serviceInterval ( );**

오류 가능성에 대한 표시 없이 서비스 간격을 리턴합니다.

**ImqBoolean serviceIntervalEvent( MQLONG & event );**

서비스 간격 이벤트의 인에이블먼트 상태의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG serviceIntervalEvent ( );**

오류 가능성에 대한 표시 없이 서비스 간격 이벤트의 인에이블먼트 상태를 리턴합니다.

**ImqBoolean shareability( MQLONG & shareability );**

공유가능성 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG shareability ( );**

오류 가능성에 대한 표시 없이 공유가능성 값을 리턴합니다.

**ImqBoolean storageClass( ImqString & class );**

스토리지 클래스의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString storageClass( );**

오류 가능성에 대한 표시 없이 스토리지 클래스를 리턴합니다.

**ImqBoolean transmissionQueueName( ImqString & name );**

전송 큐 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString transmissionQueueName ( );**

오류 가능성에 대한 표시 없이 전송 큐 이름을 리턴합니다.

**ImqBoolean triggerControl( MQLONG & control );**

트리거 제어 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG triggerControl ( );**

오류 가능성에 대한 표시 없이 트리거 제어 값을 리턴합니다.

**ImqBoolean setTriggerControl ( const MQLONG control );**

트리거 제어 값을 설정합니다. 성공하면 TRUE를 리턴합니다.

**ImqBoolean triggerData( ImqString & data );**

트리거 데이터의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString triggerData ( );**

오류 가능성에 대한 표시 없이 트리거 데이터의 사본을 리턴합니다.

**ImqBoolean setTriggerData ( const char \* data );**

트리거 데이터를 설정합니다. 성공하면 TRUE를 리턴합니다.

**ImqBoolean triggerDepth( MQLONG & depth );**

트리거 용량의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG triggerDepth ( );**

오류 가능성에 대한 표시 없이 트리거 용량을 리턴합니다.

**ImqBoolean setTriggerDepth ( const MQLONG depth );**

트리거 용량을 설정합니다. 성공하면 TRUE를 리턴합니다.

**ImqBoolean triggerMessagePriority( MQLONG & priority );**

트리거 메시지 우선순위의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG triggerMessagePriority ( );**

오류 가능성에 대한 표시 없이 트리거 메시지 우선순위를 리턴합니다.

**ImqBoolean setTriggerMessagePriority ( const MQLONG priority );**

트리거 메시지 우선순위를 설정합니다. 성공하면 TRUE를 리턴합니다.

**ImqBoolean triggerType( MQLONG & type );**

트리거 유형의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG triggerType ( );**

오류 가능성에 대한 표시 없이 트리거 유형을 리턴합니다.

**ImqBoolean setTriggerType ( const MQLONG type );**

트리거 유형을 설정합니다. 성공하면 TRUE를 리턴합니다.

**ImqBoolean usage( MQLONG & usage );**

사용법 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG usage ( );**

오류 가능성에 대한 표시 없이 사용법 값을 리턴합니다.

## 오브젝트 메소드(보호됨)

**void setNextDistributedQueue ( ImqQueue \* queue = 0 );**

다음 분산 큐를 설정합니다.

주의: 분산 큐 목록이 손상되지 않는 것이 확실한 경우에만 이 함수를 사용하십시오.

**void setPreviousDistributedQueue ( ImqQueue \* queue = 0 );**

이전 분산 큐를 설정합니다.

주의: 분산 큐 목록이 손상되지 않는 것이 확실한 경우에만 이 함수를 사용하십시오.

## 이유 코드

- MQRC\_ATTRIBUTE\_LOCKED
- MQRC\_CONTEXT\_OBJECT\_NOT\_VALID
- MQRC\_CONTEXT\_OPEN\_ERROR
- MQRC\_CURSOR\_NOT\_VALID
- MQRC\_NO\_BUFFER
- MQRC\_REOPEN\_EXCL\_INPUT\_ERROR
- MQRC\_REOPEN\_INQUIRE\_ERROR

- MQRC\_REOPEN\_TEMPORARY\_Q\_ERROR
- (MQGET의 이유 코드)
- (MQPUT의 이유 코드)

## ImqQueueManager C++ 클래스

이 클래스는 큐 관리자(MQOT\_Q\_MGR 유형의 IBM MQ 오브젝트)를 캡슐화합니다.

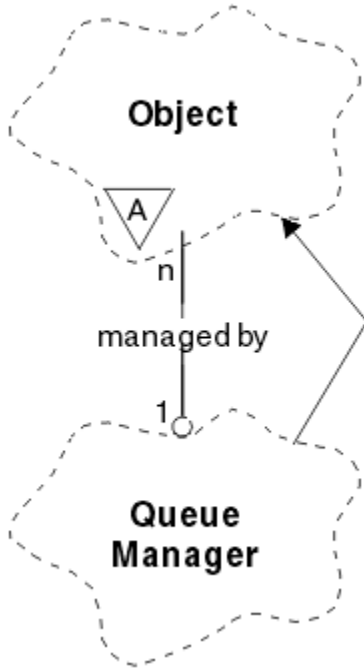


그림 33. *ImqQueueManager* 클래스

이 클래스는 1718 페이지의 『[ImqQueueManager 상호 참조](#)』에 나열된 MQI 호출과 관련이 있습니다. 나열된 모든 메소드를 모든 플랫폼에 적용할 수 있는 것은 아닙니다. 자세한 정보는 [ALTER QMGR](#)을 참조하십시오.

- [1784 페이지의 『클래스 속성』](#)
- [1785 페이지의 『오브젝트 속성』](#)
- [1790 페이지의 『구성자』](#)
- [1790 페이지의 『소멸자』](#)
- [1790 페이지의 『클래스 메소드\(공용\)』](#)
- [1790 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1798 페이지의 『오브젝트 메소드\(보호됨\)』](#)
- [1798 페이지의 『오브젝트 데이터\(보호됨\)』](#)
- [1798 페이지의 『이유 코드』](#)

### 클래스 속성

#### 작동(behavior)

암시적 연결과 연결 끊기의 동작을 제어합니다.

#### **IMQ\_EXPL\_DISC\_BACKOUT (0L)**

연결 끊기 방법에 대한 명시적 호출이 백아웃을 의미합니다. 이 속성은 IMQ\_EXPL\_DISC\_COMMIT와 상호 배타적입니다.

#### **IMQ\_EXPL\_DISC\_COMMIT (1L)**

연결 끊기 방법에 대한 명시적 호출이 커밋(기본값)를 의미합니다. 이 속성은 IMQ\_EXPL\_DISC\_BACKOUT과 상호 배타적입니다.

### **IMQ\_IMPL\_CONN (2L)**

암시적 연결이 허용됩니다(기본값).

### **IMQ\_IMPL\_DISC\_BACKOUT (0L)**

오브젝트 삭제 중에 발생할 수 있는 연결 끊기 방법에 대한 암시적 호출이 백아웃을 의미합니다. 이 속성은 IMQ\_IMPL\_DISC\_COMMIT와 상호 배타적입니다.

### **IMQ\_IMPL\_DISC\_COMMIT (4L)**

오브젝트 삭제 중에 발생할 수 있는 연결 끊기 방법에 대한 암시적 호출이 커밋(기본값)를 의미합니다. 이 속성은 IMQ\_IMPL\_DISC\_BACKOUT과 상호 배타적입니다.

IBM MQ V7.0 이상에서 암시적 연결을 사용하는 C++ 애플리케이션은 `ImqQueueManager` 클래스 오브젝트의 `setBehavior()` 메소드에 제공되는 다른 옵션과 함께 `IMQ_IMPL_CONN`을 지정해야 합니다. 애플리케이션이 `setBehavior()` 메소드를 사용하여 동작 옵션을 명시적으로 설정하지 않는 경우, 예를 들면 다음과 같습니다.

```
ImqQueueManager_object.setBehavior(IMQ_IMPL_DISC_COMMIT)
```

`MQ_IMPL_CONN`이 기본적으로 사용 가능하므로 이렇게 변경해도 아무런 영향이 없습니다.

애플리케이션이 동작 옵션을 명시적으로 설정하는 경우, 예를 들면 다음과 같습니다.

```
ImqQueueManager_object.setBehavior(IMQ_IMPL_DISC_COMMIT)
```

다음과 같이 `setBehavior()` 메소드에 `IMQ_IMPL_CONN`을 포함하여 애플리케이션이 암시적 연결을 완료할 수 있도록 해야 합니다.

```
ImqQueueManager_object.setBehavior(IMQ_IMPL_CONN | IMQ_IMPL_DISC_COMMIT)
```

## **오브젝트 속성**

### **계정 연결 대체**

애플리케이션이 MQI 계정 및 큐 계정 값의 설정을 대체하도록 허용합니다. 이 속성은 읽기 전용입니다.

### **계정 간격**

중간 계정 레코드가 작성되기까지 얼마나 걸렸습니까(초)? 이 속성은 읽기 전용입니다.

### **활동 기록**

활동 보고서의 생성을 제어합니다. 이 속성은 읽기 전용입니다.

### **새 mca 검사 채택**

이미 활성 상태인 MCA와 동일한 이름의 새 인바운드 채널이 감지될 때 MCA를 채택해야 하는지를 판별하기 위해 검사된 요소. 이 속성은 읽기 전용입니다.

### **새 mca 유형 채택**

'새 mca 검사 채택' 매개변수와 일치하는 새 인바운드 채널 요청이 감지된 경우 특정 채널 유형의 Orphan MCA 인스턴스를 자동으로 재시작해야 하는지 여부. 이 속성은 읽기 전용입니다.

### **인증 유형**

수행되는 인증의 유형을 표시합니다.

### **권한 이벤트**

권한 이벤트를 제어합니다. 이 속성은 읽기 전용입니다.

### **시작 옵션**

시작 메소드에 적용되는 옵션. 초기값은 `MQBO_NONE`입니다.

### **브릿지 이벤트**

IMS 브릿지 이벤트의 생성 여부. 이 속성은 읽기 전용입니다.

### **채널 자동 정의**

채널 자동 정의 값. 이 속성은 읽기 전용입니다.

### **채널 자동 정의 이벤트**

채널 자동 정의 이벤트 값. 이 속성은 읽기 전용입니다.

### 채널 자동 정의 엑시트

채널 자동 정의 엑시트 이름. 이 속성은 읽기 전용입니다.

### 채널 이벤트(channel event)

채널 이벤트의 생성 여부. 이 속성은 읽기 전용입니다.

### 채널 시작기 어댑터

IBM MQ 호출 처리에 사용할 어댑터 하위 태스크의 수. 이 속성은 읽기 전용입니다.

### 채널 시작기 제어

큐 관리자를 시작할 때 채널 시작기의 자동 시작 여부. 이 속성은 읽기 전용입니다.

### 채널 시작기 디스패처

채널 시작기에 사용할 디스패처 수. 이 속성은 읽기 전용입니다.

### 채널 시작기 추적 자동시작

채널 시작기 추적의 자동 시작 여부. 이 속성은 읽기 전용입니다.

### 채널 시작기 추적 테이블 크기

채널 시작기 추적 데이터 공간의 크기(MB). 이 속성은 읽기 전용입니다.

### 채널 모니터링

채널에 대한 온라인 모니터링 데이터의 콜렉션을 제어합니다. 이 속성은 읽기 전용입니다.

### 채널 참조

클라이언트 연결 중에 사용하는 채널 정의에 대한 참조. 연결된 동안 이 속성을 널로 설정할 수 있으며, 그 외 값으로 변경할 수는 없습니다. 초기값은 널입니다.

### 채널 통계

채널에 대한 통계 데이터의 콜렉션을 제어합니다. 이 속성은 읽기 전용입니다.

### 문자 세트

코드화 문자 세트 ID(CCSID). 이 속성은 읽기 전용입니다.

### 클러스터 송신자 모니터링

자동 정의 클러스터 송신자 채널에 대한 온라인 모니터링 데이터 콜렉션을 제어합니다. 이 속성은 읽기 전용입니다.

### 클러스터 송신자 통계

자동 정의 클러스터 송신자 채널에 대한 통계 데이터 콜렉션을 제어합니다. 이 속성은 읽기 전용입니다.

### 클러스터 워크로드 데이터

CLWL 워크로드 엑시트 데이터. 이 속성은 읽기 전용입니다.

### 클러스터 워크로드 엑시트

클러스터 워크로드 엑시트 이름. 이 속성은 읽기 전용입니다.

### 클러스터 워크로드 길이

클러스터 워크로드 길이 이 속성은 읽기 전용입니다.

### 클러스터 워크로드 mru

클러스터 워크로드의 최근 사용된 채널 값. 이 속성은 읽기 전용입니다.

### 클러스터 워크로드 사용 큐

클러스터 워크로드 사용 큐 값. 이 속성은 읽기 전용입니다.

### 명령 이벤트(command event)

명령 이벤트의 생성 여부. 이 속성은 읽기 전용입니다.

### 명령 입력 큐 이름

시스템 명령 입력 큐 이름. 이 속성은 읽기 전용입니다.

### 명령 레벨

큐 관리자가 지원하는 명령 레벨. 이 속성은 읽기 전용입니다.

### 명령 서버 제어

큐 관리자를 시작할 때 명령 서버의 자동 시작 여부. 이 속성은 읽기 전용입니다.

### 연결 옵션

연결 메소드에 적용되는 옵션. 초기값은 MQCNO\_NONE입니다. 플랫폼에 따라 다음과 같은 추가 값을 사용할 수 있습니다.

- MQCNO\_STANDARD\_BINDING

- MQCNO\_FASTPATH\_BINDING
- MQCNO\_HANDLE\_SHARE\_NONE
- MQCNO\_HANDLE\_SHARE\_BLOCK
- MQCNO\_HANDLE\_SHARE\_NO\_BLOCK
- MQCNO\_SERIALIZE\_CONN\_TAG\_Q\_MGR
- MQCNO\_SERIALIZE\_CONN\_TAG\_QSG
- MQCNO\_RESTRICT\_CONN\_TAG\_Q\_MGR
- MQCNO\_RESTRICT\_CONN\_TAG\_QSG

#### 연결 ID

MQ가 애플리케이션을 안정적으로 식별할 수 있는 고유 ID.

#### 연결 상태

큐 관리자에 연결된 경우 TRUE입니다. 이 속성은 읽기 전용입니다.

#### 연결 태그

연결과 연관된 태그. 이 속성은 연결되지 않은 경우에만 설정할 수 있습니다. 초기값은 널입니다.

#### 암호화 하드웨어

암호화 하드웨어에 대한 구성 세부사항. MQ MQI 클라이언트 연결에 해당합니다.

#### 데드-레터 큐 이름

데드-레터 큐의 이름. 이 속성은 읽기 전용입니다.

#### 기본 전송 큐 이름

기본 전송 큐 이름. 이 속성은 읽기 전용입니다.

#### 분배 목록

분배 목록을 지원하기 위한 큐 관리자의 기능.

#### dns 그룹

큐 공유 그룹의 인바운드 전송을 처리하는 TCP 리스너가 워크로드 관리자 동적 도메인 이름 서비스 지원을 사용할 때 결합해야 하는 그룹의 이름. 이 속성은 읽기 전용입니다.

#### dns wlm

큐 공유 그룹에 대한 인바운드 전송을 처리하는 TCP 리스너가 동적 도메인 이름 서비스를 위한 워크로드 관리자에 등록해야 하는지 여부. 이 속성은 읽기 전용입니다.

#### 첫 번째 인증 레코드

하나 이상의 ImqAuthenticationRecord 클래스 오브젝트 중 첫 번째 오브젝트로, ImqAuthenticationRecord 연결 참조는 특별한 순서 없이 이 오브젝트를 처리합니다. MQ MQI 클라이언트 연결에 해당합니다.

#### 첫 번째 관리 오브젝트

하나 이상의 ImqObject 클래스 오브젝트 중 첫 번째 오브젝트로, ImqObject 연결 참조는 특별한 순서 없이 이 오브젝트를 처리합니다. 초기값은 0입니다.

#### 금지 이벤트

금지 이벤트를 제어합니다. 이 속성은 읽기 전용입니다.

#### ip 주소 버전

채널 연결에 사용할 IP 프로토콜(IPv4 또는 IPv6). 이 속성은 읽기 전용입니다.

#### 키 저장소(key repository)

키와 인증서가 저장되는 키 데이터베이스 파일의 위치. IBM MQ MQI client 연결에 해당합니다.

#### 키 재설정 수

비밀 키를 재협상하기 전에 SSL 대화 안에서 송신 및 수신된 암호화되지 않은 바이트 수. 이 속성은 MQCONNX를 사용하는 클라이언트 연결에만 적용됩니다. [ssl 키 재설정 수도](#) 참조하십시오.

#### 리스너 타이머

APPC 또는 TCP/IP 실패가 발생한 경우 IBM MQ에서 리스너를 재시작하려고 시도하는 시간 간격(초). 이 속성은 읽기 전용입니다.

#### 로컬 이벤트

로컬 이벤트를 제어합니다. 이 속성은 읽기 전용입니다.

## 로거 이벤트

복구 로그 이벤트의 생성 여부를 제어합니다. 이 속성은 읽기 전용입니다.

## lu 그룹 이름

큐 공유 그룹에 대한 인바운드 전송을 처리하는 LU 6.2 리스너가 사용해야 하는 일반 LU 이름. 이 속성은 읽기 전용입니다.

## lu 이름

아웃바운드 LU 6.2 전송에 사용할 LU 이름. 이 속성은 읽기 전용입니다.

## lu62 arm 접미부

이 채널 시작기의 LUADD를 지정하는 SYS1.PARMLIB 멤버 APPCPMxx의 접미부. 이 속성은 읽기 전용입니다.

## lu62 채널

LU 6.2 전송 프로토콜을 사용하는 현재 실행할 수 있는 채널 또는 연결 가능한 클라이언트의 최대 수. 이 속성은 읽기 전용입니다.

## 최대 활성 채널

언제든지 활성화될 수 있는 최대 채널 수. 이 속성은 읽기 전용입니다.

## 최대 채널

현재 실행될 수 있는 채널의 최대 수(연결된 클라이언트가 있는 서버 연결 채널 포함). 이 속성은 읽기 전용입니다.

## 최대 핸들

최대 핸들 수. 이 속성은 읽기 전용입니다.

## 최대 메시지 길이

이 큐 관리자에 의해 관리되는 큐에 있는 메시지의 최대 가능 길이. 이 속성은 읽기 전용입니다.

## 최대 우선순위

최대 메시지 우선순위. 이 속성은 읽기 전용입니다.

## 최대 커밋되지 않은 메시지

작업 단위 내에서 커밋되지 않은 메시지의 최대 수. 이 속성은 읽기 전용입니다.

## mqi 계정

MQI 데이터에 대한 계정 정보 컬렉션을 제어합니다. 이 속성은 읽기 전용입니다.

## mqi 통계

큐 관리자에 대한 통계 모니터링 정보의 컬렉션을 제어합니다. 이 속성은 읽기 전용입니다.

## 아웃바운드 포트 최대

발신 채널을 바인딩할 때 사용되는 포트 번호 범위에서 최대 값. 이 속성은 읽기 전용입니다.

## 아웃바운드 포트 최소

발신 채널을 바인딩할 때 사용되는 포트 번호 범위에서 최소 값. 이 속성은 읽기 전용입니다.

## 비밀번호

사용자 ID와 연관된 비밀번호

## 성능 이벤트(performance event)

성능 이벤트를 제어합니다. 이 속성은 읽기 전용입니다.

## 플랫폼

큐 관리자가 상주하는 플랫폼. 이 속성은 읽기 전용입니다.

## 큐 계정

큐에 대한 계정 정보 컬렉션을 제어합니다. 이 속성은 읽기 전용입니다.

## 큐 모니터링

큐에 대한 온라인 모니터링 데이터의 컬렉션을 제어합니다. 이 속성은 읽기 전용입니다.

## 큐 통계

큐에 대한 통계 데이터 컬렉션을 제어합니다. 이 속성은 읽기 전용입니다.

## 수신 제한시간

비활성 상태로 돌아가기 전에 TCP/IP 메시지 채널이 파트너로부터 하트비트를 포함한 데이터를 수신하기 위해 기다리는 대략적인 시간. 이 속성은 읽기 전용입니다.



### 수신 제한시간 최소

비활성 상태로 돌아가기 전에 TCP/IP 채널이 파트너로부터 하트비트를 포함한 데이터를 수신하기 위해 기다리는 최소 시간. 이 속성은 읽기 전용입니다.

### 수신 제한시간 유형

수신 제한시간에 적용된 규정자. 이 속성은 읽기 전용입니다.

### 리모트 이벤트

리모트 이벤트를 제어합니다. 이 속성은 읽기 전용입니다.

### 저장소 이름

저장소 이름. 이 속성은 읽기 전용입니다.

### 저장소 이름 목록

저장소 이름 목록 이름. 이 속성은 읽기 전용입니다.

### 공유 큐 관리자 이름

ObjectQMgrName이 큐 공유 그룹의 다른 큐 관리자인 공유 큐의 MQOPEN가 로컬 큐 관리자에서 공유 큐의 열림으로 해석되어야 하는지 여부. 이 속성은 읽기 전용입니다.

### ssl 이벤트

SSL 이벤트의 생성 여부. 이 속성은 읽기 전용입니다.

### ssl FIPS 필요

암호화가 IBM MQ 소프트웨어에서 실행되는 경우 FIPS 인증 알고리즘만 사용해야 하는지 여부. 이 속성은 읽기 전용입니다.

### ssl 키 재설정 수

비밀 키를 재협상하기 전에 SSL 대화 안에서 송신 및 수신된 암호화되지 않은 바이트 수. 이 속성은 읽기 전용입니다.

### 시작-중지 이벤트


시작-중지 이벤트를 제어합니다. 이 속성은 읽기 전용입니다.

### 통계 간격

통계 모니터링 데이터를 모니터링 큐에 기록하는 빈도. 이 속성은 읽기 전용입니다.

### 동기점 가용성

동기점 참여의 가용성. 이 속성은 읽기 전용입니다.

**참고:** 큐 관리자가 조정하는 글로벌 작업 단위는 IBM i 플랫폼에서 지원되지 않습니다.  `_Rcommit` 및 `_Rback` 기본 시스템 호출을 사용하여 IBM i에 의해 외부에서 조정되는 작업 단위를 프로그래밍할 수 있습니다. `STRCMTCTL` 명령을 사용하여 작업 레벨 커밋 제어 아래에서 IBM MQ 애플리케이션을 시작하여 이 유형의 작업 단위를 시작하십시오. 자세한 정보는 [IBM i 외부 동기점 관리자에 대한 인터페이스](#)를 참조하십시오. 큐 관리자가 조정하는 로컬 작업 단위에 대한 백아웃 및 커밋은 IBM i 플랫폼에서 지원됩니다.

### tcp 채널

TCP/IP 전송 프로토콜을 사용하는 현재 채널 또는 연결할 수 있는 클라이언트의 최대 수. 이 속성은 읽기 전용입니다.

### tcp 활성 유지(keepalive)

연결의 다른 끝이 여전히 사용 가능한지 검사하기 위해 KEEPALIVE 기능을 사용하는지 여부. 이 속성은 읽기 전용입니다.

### tcp 이름

tcp 스택 유형의 값에 따라 사용할 유일한 또는 기본 TCP/IP 시스템의 이름. 이 속성은 읽기 전용입니다.

### tcp 스택 유형

채널 시작기가 tcp 이름에 지정된 TCP/IP 주소 공간만 사용하도록 허용되는지, 또는 선택된 모든 TCP/IP 주소에 바인딩할 수 있는지 여부. 이 속성은 읽기 전용입니다.

### 라우트 추적 기록

라우트 추적 정보의 기록을 제어합니다. 이 속성은 읽기 전용입니다.

### 트리거 간격

트리거 간격. 이 속성은 읽기 전용입니다.

## 사용자 ID

AIX and Linux 플랫폼에서는 애플리케이션의 실제 사용자 ID. Windows 플랫폼에서는 애플리케이션의 사용자 ID.

## 구성자

**ImqQueueManager();**

기본 구성자입니다.

**ImqQueueManager( const ImqQueueManager & manager );**

복사 구성자입니다. 연결 상태는 FALSE가 됩니다.

**ImqQueueManager( const char \* name );**

ImqObject 이름을 이름(*name*)으로 설정합니다.

## 소멸자

ImqQueueManager 오브젝트를 영구 삭제하면 자동으로 연결이 끊깁니다.

## 클래스 메소드(공용)

**static MQLONG behavior();**

작동을 리턴합니다.

**void setBehavior( const MQLONG behavior = 0 );**

작동을 설정합니다.

## 오브젝트 메소드(공용)

**void operator = ( const ImqQueueManager & mgr );**

필요한 경우 연결을 끊고 *mgr*에서 인스턴스 데이터를 복사합니다. 연결 상태는 FALSE입니다.

**ImqBoolean accountingConnOverride ( MQLONG & statint );**

계정 연결 대체 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG accountingConnOverride ( );**

오류 가능성에 대한 표시 없이 계정 연결 대체 값을 리턴합니다.

**ImqBoolean accountingInterval ( MQLONG & statint );**

계정 간격 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG accountingInterval ( );**

오류 가능성에 대한 표시 없이 계정 간격 값을 리턴합니다.

**ImqBoolean activityRecording ( MQLONG & rec );**

활동 기록 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG activityRecording ( );**

오류 가능성에 대한 표시 없이 활동 기록 값을 리턴합니다.

**ImqBoolean adoptNewMCACheck ( MQLONG & check );**

새 MCA 검사 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG adoptNewMCACheck ( );**

오류 가능성에 대한 표시 없이 새 MCA 검사 값 채택을 리턴합니다.

**ImqBoolean adoptNewMCAType ( MQLONG & type );**

새 MCA 유형 채택의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG adoptNewMCAType ( );**

오류 가능성에 대한 표시 없이 새 MCA 유형 채택을 리턴합니다.

**QLONG authenticationType ( ) const;**

인증 유형을 리턴합니다.

**void setAuthenticationType ( const MQLONG type = MQCSP\_AUTH\_NONE );**

인증 유형을 설정합니다.

**ImqBoolean authorityEvent( MQLONG & event );**

권한 이벤트의 인에이블먼트 상태의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG authorityEvent( );**

오류 가능성에 대한 표시 없이 권한 이벤트의 인에이블먼트 상태를 리턴합니다.

**ImqBoolean backout( );**

커밋되지 않은 변경사항을 백아웃합니다. 성공하면 TRUE를 리턴합니다.

**ImqBoolean begin( );**

작업 단위를 시작합니다. 시작 옵션은 이 메소드의 동작에 영향을 줍니다. 성공하면 TRUE를 리턴하지만, 기본 MQBEGIN 호출이 MQRC\_NO\_EXTERNAL\_PARTICIPANTS 또는 MQRC\_PARTICIPANT\_NOT\_AVAILABLE(둘 다 MQCC\_WARNING과 연관되어 있음)을 리턴하는 경우에도 TRUE를 리턴합니다.

**MQLONG beginOptions( ) const ;**

시작 옵션을 리턴합니다.

**void setBeginOptions( const MQLONG options = MQBO\_NONE );**

시작 옵션을 설정합니다.

**ImqBoolean bridgeEvent ( MQLONG & event);**

브릿지 이벤트 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG bridgeEvent ( );**

오류 가능성에 대한 표시 없이 브릿지 이벤트 값을 리턴합니다.

**ImqBoolean channelAutoDefinition( MQLONG & value );**

채널 자동 정의 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG channelAutoDefinition( );**

오류 가능성에 대한 표시 없이 채널 자동 정의 값을 리턴합니다.

**ImqBoolean channelAutoDefinitionEvent( MQLONG & value );**

채널 자동 정의 이벤트 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG channelAutoDefinitionEvent( );**

오류 가능성에 대한 표시 없이 채널 자동 정의 이벤트 값을 리턴합니다.

**ImqBoolean channelAutoDefinitionExit( ImqString & name );**

채널 자동 정의 엑시트 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString channelAutoDefinitionExit( );**

오류 가능성에 대한 표시 없이 채널 자동 정의 엑시트 이름을 리턴합니다.

**ImqBoolean channelEvent ( MQLONG & event);**

채널 이벤트 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG channelEvent( );**

오류 가능성에 대한 표시 없이 채널 이벤트 값을 리턴합니다.

**MQLONG channelInitiatorAdapters ( );**

오류 가능성에 대한 표시 없이 채널 시작기 어댑터 값을 리턴합니다.

**ImqBoolean channelInitiatorAdapters ( MQLONG & adapters );**

채널 시작기 어댑터 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG channelInitiatorControl ( );**

오류 가능성에 대한 표시 없이 채널 시작기 시동 값을 리턴합니다.

**ImqBoolean channelInitiatorControl ( MQLONG & init );**

채널 시작기 제어 시동 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG channelInitiatorDispatchers ( );**

오류 가능성에 대한 표시 없이 채널 시작기 디스패처 값을 리턴합니다.

**ImqBoolean channelInitiatorDispatchers ( MQLONG & dispatchers );**

채널 시작기 디스패처 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG channelInitiatorTraceAutoStart ( );**

오류 가능성에 대한 표시 없이 채널 시작기 추적 자동 시작 값을 리턴합니다.

**ImqBoolean channelInitiatorTraceAutoStart ( MQLONG & auto);**

채널 시작기 추적 자동 시작 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG channelInitiatorTraceTableSize ( );**

오류 가능성에 대한 표시 없이 채널 시작기 추적 테이블 크기 값을 리턴합니다.

**ImqBoolean channelInitiatorTraceTableSize ( MQLONG & size);**

채널 시작기 추적 테이블 크기 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqBoolean channelMonitoring ( MQLONG & monchl );**

채널 모니터링 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG channelMonitoring ( );**

오류 가능성에 대한 표시 없이 채널 모니터링 값을 리턴합니다.

**ImqBoolean channelReference( ImqChannel \* & pchannel );**

채널 참조의 사본을 제공합니다. 채널 참조가 올바르지 않으면 *pchannel*을 널로 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqChannel \* channelReference( );**

오류 가능성에 대한 표시 없이 채널 참조를 리턴합니다.

**ImqBoolean setChannelReference( ImqChannel & channel );**

채널 참조를 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqBoolean setChannelReference( ImqChannel \* channel = 0 );**

채널 참조를 설정하거나 재설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqBoolean channelStatistics ( MQLONG & statchl );**

채널 통계 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG channelStatistics ( );**

오류 가능성에 대한 표시 없이 채널 통계 값을 리턴합니다.

**ImqBoolean characterSet( MQLONG & ccsid );**

문자 세트의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG characterSet( );**

오류 가능성에 대한 표시 없이 문자 세트의 사본을 리턴합니다.

**MQLONG clientSslKeyResetCount ( ) const;**

클라이언트 연결에 사용된 SSL 키 재설정 수 값을 리턴합니다.

**void setClientSslKeyResetCount( const MQLONG count );**

클라이언트 연결에 사용된 SSL 키 재설정 수를 설정합니다.

**ImqBoolean clusterSenderMonitoring ( MQLONG & monacsl );**

클러스터 송신자 모니터링 기본값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG clusterSenderMonitoring ( );**

오류 가능성에 대한 표시 없이 클러스터 송신자 모니터링 기본값을 리턴합니다.

**ImqBoolean clusterSenderStatistics ( MQLONG & statacsl );**

클러스터 송신자 통계 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG clusterSenderStatistics ( );**

오류 가능성에 대한 표시 없이 클러스터 송신자 통계 값을 리턴합니다.

**ImqBoolean clusterWorkloadData( ImqString & data );**

CLWL 엑시트 데이터의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString clusterWorkloadData( );**

오류 가능성에 대한 표시 없이 CLWL 엑시트 데이터를 리턴합니다.

**ImqBoolean clusterWorkloadExit( ImqString & name );**

CLWL 엑시트 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString clusterWorkloadExit( );**

오류 가능성에 대한 표시 없이 CLWL 엑시트 이름을 리턴합니다.

**ImqBoolean clusterWorkloadLength( MQLONG & length );**

클러스터 워크로드 길이의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG clusterWorkloadLength();**

오류 가능성에 대한 표시 없이 클러스터 워크로드 길이를 리턴합니다.

**ImqBoolean clusterWorkLoadMRU ( MQLONG & mru );**

클러스터 워크로드의 가장 최근에 사용된 채널 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG clusterWorkLoadMRU ();**

오류 가능성에 대한 표시 없이 클러스터 워크로드의 가장 최근에 사용된 채널 값을 리턴합니다.

**ImqBoolean clusterWorkLoadUseQ ( MQLONG & useq );**

클러스터 워크로드 사용 큐 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG clusterWorkLoadUseQ ();**

오류 가능성에 대한 표시 없이 클러스터 워크로드 사용 큐 값을 리턴합니다.

**ImqBoolean commandEvent ( MQLONG & event );**

명령 이벤트 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG commandEvent ();**

오류 가능성에 대한 표시 없이 명령 이벤트 값을 리턴합니다.

**ImqBoolean commandInputQueueName( ImqString & name );**

명령 입력 큐 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString commandInputQueueName ();**

오류 가능성에 대한 표시 없이 명령 입력 큐 이름을 리턴합니다.

**ImqBoolean commandLevel( MQLONG & level );**

명령 레벨의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG commandLevel();**

오류 가능성에 대한 표시 없이 명령 레벨을 리턴합니다.

**MQLONG commandServerControl ();**

오류 가능성에 대한 표시 없이 명령 서버 구동 값을 리턴합니다.

**ImqBoolean commandServerControl ( MQLONG & server );**

명령 서버 제어 구동 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqBoolean commit();**

커미트되지 않은 변경사항을 커미트합니다. 성공하면 TRUE를 리턴합니다.

**ImqBoolean connect();**

주어진 ImqObject 이름을 갖는 큐 관리자에 연결하며, 기본값은 로컬 큐 관리자입니다. 특정 큐 관리자에 연결하려는 경우, 연결 전에 ImqObject setName 메소드를 사용하십시오. 채널 참조가 있으면, MQCD의 MQCONN에 채널 정의 정보를 전달하는 데 사용됩니다. MQCD의 ChannelType은 MQCHT\_CLNTCONN으로 설정됩니다. 클라이언트 연결에서만 의미가 있는 채널 참조 정보는 서버 연결에서는 무시됩니다. 연결 옵션은 이 메소드의 동작에 영향을 줍니다. 성공하면 이 메소드가 연결 상태를 TRUE로 설정합니다. 새 연결 상태를 리턴합니다.

첫 번째 인증 레코드가 있는 경우, 보안 클라이언트 채널에 대한 디지털 인증서를 인증하는 데 인증 레코드 체인이 사용됩니다.

하나 이상의 ImqQueueManager 오브젝트를 동일한 큐 관리자에 연결할 수 있습니다. 모든 항목이 동일한 MQHCONN 연결 핸들을 사용하고 스레드와 연관된 연결에 대해 UOW 기능을 공유합니다. 연결할 첫 번째 ImqQueueManager는 MQHCONN 핸들을 가져옵니다. 연결을 끊을 마지막 ImqQueueManager는 MQDISC를 수행합니다.

멀티스레드 프로그램의 경우 각 스레드에 별도의 ImqQueueManager 오브젝트를 사용하는 것이 좋습니다.

**ImqBinary connectionId () const ;**

연결 ID를 리턴합니다.

**ImqBinary connectionTag () const ;**

연결 태그를 리턴합니다.

**ImqBoolean setConnectionTag ( const MQBYTE128 tag = 0 );**

연결 태그를 설정합니다. tag가 0이면 연결 태그를 지웁니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqBoolean setConnectionTag ( const ImqBinary & tag );**  
 연결 태그를 설정합니다. *tag*의 데이터 길이는 0(연결 태그를 지우는 경우) 또는 MQ\_CONN\_TAG\_LENGTH입니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**MQLONG connectOptions() const ;**  
 연결 옵션을 리턴합니다.

**void setConnectOptions( const MQLONG options = MQCNO\_NONE );**  
 연결 옵션을 설정합니다.

**ImqBoolean connectionStatus() const ;**  
 연결 상태를 리턴합니다.

**ImqString cryptographicHardware ();**  
 암호화 하드웨어를 리턴합니다.

**ImqBoolean setCryptographicHardware ( const char \* hardware = 0 );**  
 암호화 하드웨어를 설정합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**ImqBoolean deadLetterQueueName( ImqString & name );**  
 데드-레터 큐 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString deadLetterQueueName();**  
 오류 가능성에 대한 표시 없이 데드-레터 큐 이름의 사본을 리턴합니다.

**ImqBoolean defaultTransmissionQueueName( ImqString & name );**  
 기본 전송 큐 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString defaultTransmissionQueueName();**  
 오류 가능성에 대한 표시 없이 기본 전송 큐 이름을 리턴합니다.

**ImqBoolean disconnect();**  
 큐 관리자와의 연결을 끊고 연결 상태를 FALSE로 설정합니다. 이 오브젝트와 연관된 모든 ImqProcess 및 ImqQueue 오브젝트를 닫고, 연결을 끊기 전에 연결 참조를 제공합니다. 둘 이상의 ImqQueueManager 오브젝트가 동일한 큐 관리자에 연결되어 있는 경우, 연결을 끊는 마지막 항목만 실제 연결 끊기를 수행하고 다른 항목은 논리적 연결 끊기를 수행합니다. 커밋되지 않은 변경은 물리적 연결 끊기에서만 커밋됩니다.  
 이 메소드는 성공하면 TRUE를 리턴합니다. 기존의 연결이 없을 때 호출되는 경우에도 리턴 코드가 true입니다.

**ImqBoolean distributionLists( MQLONG & support );**  
 분배 목록 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG distributionLists();**  
 오류 가능성에 대한 표시 없이 분배 목록 값을 리턴합니다.

**ImqBoolean dnsGroup ( ImqString & group );**  
 DNS 그룹 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString dnsGroup ();**  
 오류 가능성에 대한 표시 없이 DNS 그룹 이름을 리턴합니다.

**ImqBoolean dnsWlm ( MQLONG & wlm );**  
 DNS WLM 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG dnsWlm ();**  
 오류 가능성에 대한 표시 없이 DNS WLM 값을 리턴합니다.

**ImqAuthenticationRecord \* firstAuthenticationRecord ( ) const ;**  
 첫 번째 인증 레코드를 리턴합니다.

**void setFirstAuthenticationRecord ( const ImqAuthenticationRecord \* air = 0 );**  
 첫 번째 인증 레코드를 설정합니다.

**ImqObject \* firstManagedObject() const ;**  
 첫 번째 관리 오브젝트를 리턴합니다.

**ImqBoolean inhibitEvent( MQLONG & event );**  
 금지 이벤트의 인에이블먼트 상태의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG inhibitEvent();**  
 오류 가능성에 대한 표시 없이 금지 이벤트의 인에이블먼트 상태를 리턴합니다.

**ImqBoolean ipAddressVersion ( MQLONG & version );**  
IP 주소 버전 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG ipAddressVersion ( );**  
오류 가능성에 대한 표시 없이 IP 주소 버전 값을 리턴합니다.

**ImqBoolean keepAlive ( MQLONG & keepalive );**  
활성 유지(keepalive) 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG keepAlive ( );**  
오류 가능성에 대한 표시 없이 활성 유지(keepalive) 값을 리턴합니다.

**ImqString keyRepository ( );**  
키 저장소를 리턴합니다.

**ImqBoolean setKeyRepository ( const char \* repository = 0 );**  
키 저장소를 설정합니다. 성공하면 TRUE를 리턴합니다.

**ImqBoolean listenerTimer ( MQLONG & timer );**  
리스너 타이머 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG listenerTimer ( );**  
오류 가능성에 대한 표시 없이 리스너 타이머 값을 리턴합니다.

**ImqBoolean localEvent( MQLONG & event );**  
로컬 이벤트의 인에이블먼트 상태의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG localEvent( );**  
오류 가능성에 대한 표시 없이 로컬 이벤트의 인에이블먼트 상태를 리턴합니다.

**ImqBoolean loggerEvent ( MQLONG & count );**  
로거 이벤트 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG loggerEvent ( );**  
오류 가능성에 대한 표시 없이 로거 이벤트 값을 리턴합니다.

**ImqBoolean luGroupName ( ImqString & name );**  
LU 그룹 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString luGroupName ( );**  
오류 가능성에 대한 표시 없이 LU 그룹 이름을 리턴합니다.

**ImqBoolean lu62ARMSuffix ( ImqString & suffix );**  
LU62 ARM 접미부의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString lu62ARMSuffix ( );**  
오류 가능성에 대한 표시 없이 LU62 ARM 접미부를 리턴합니다.

**ImqBoolean luName ( ImqString & name );**  
LU 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString luName ( );**  
오류 가능성에 대한 표시 없이 LU 이름을 리턴합니다.

**ImqBoolean maximumActiveChannels ( MQLONG & channels);**  
최대 활성 채널 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG maximumActiveChannels ( );**  
오류 가능성에 대한 표시 없이 최대 활성 채널 값을 리턴합니다.

**ImqBoolean maximumCurrentChannels ( MQLONG & channels );**  
최대 현재 채널 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG maximumCurrentChannels ( );**  
오류 가능성에 대한 표시 없이 최대 현재 채널 값을 리턴합니다.

**ImqBoolean maximumHandles( MQLONG & number );**  
최대 핸들의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG maximumHandles( );**  
오류 가능성에 대한 표시 없이 최대 핸들을 리턴합니다.

**ImqBoolean maximumLu62Channels ( MQLONG & channels );**  
최대 LU62 채널 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG maximumLu62Channels ();**

오류 가능성에 대한 표시 없이 최대 LU62 채널 값을 리턴합니다.

**ImqBoolean maximumMessageLength( MQLONG & length );**

최대 메시지 길이의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG maximumMessageLength();**

오류 가능성에 대한 표시 없이 최대 메시지 길이를 리턴합니다.

**ImqBoolean maximumPriority( MQLONG & priority );**

최대 우선순위의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG maximumPriority();**

오류 가능성에 대한 표시 없이 최대 우선순위의 사본을 리턴합니다.

**ImqBoolean maximumTcpChannels ( MQLONG & channels );**

최대 TCP 채널 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG maximumTcpChannels ();**

오류 가능성에 대한 표시 없이 최대 TCP 채널 값을 리턴합니다.

**ImqBoolean maximumUncommittedMessages( MQLONG & number );**

최대 커밋되지 않은 메시지의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG maximumUncommittedMessages();**

오류 가능성에 대한 표시 없이 최대 커밋되지 않은 메시지를 리턴합니다.

**ImqBoolean mqAccounting ( MQLONG & statint );**

MQI 계정 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG mqAccounting ();**

오류 가능성에 대한 표시 없이 MQI 계정 값을 리턴합니다.

**ImqBoolean mqStatistics ( MQLONG & statmq );**

MQI 통계 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG mqStatistics ();**

오류 가능성에 대한 표시 없이 MQI 통계 값을 리턴합니다.

**ImqBoolean outboundPortMax ( MQLONG & max );**

최대 아웃바운드 포트 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG outboundPortMax ();**

오류 가능성에 대한 표시 없이 최대 아웃바운드 포트 값을 리턴합니다.

**ImqBoolean outboundPortMin ( MQLONG & min );**

최소 아웃바운드 포트 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG outboundPortMin ();**

오류 가능성에 대한 표시 없이 최소 아웃바운드 포트 값을 리턴합니다.

**ImqBinary password ( ) const;**

클라이언트 연결에 사용된 비밀번호를 리턴합니다.

**ImqBoolean setPassword ( const ImqString & password );**

클라이언트 연결에 사용되는 비밀번호를 설정합니다.

**ImqBoolean setPassword ( const char \* = 0 password );**

클라이언트 연결에 사용되는 비밀번호를 설정합니다.

**ImqBoolean setPassword ( const ImqBinary & password );**

클라이언트 연결에 사용되는 비밀번호를 설정합니다.

**ImqBoolean performanceEvent( MQLONG & event );**

성능 이벤트의 인에이블먼트 상태의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG performanceEvent();**

오류 가능성에 대한 표시 없이 성능 이벤트의 인에이블먼트 상태를 리턴합니다.

**ImqBoolean platform( MQLONG & platform );**

플랫폼의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG platform();**

오류 가능성에 대한 표시 없이 플랫폼을 리턴합니다.



**ImqBoolean queueAccounting ( MQLONG & acctq );**  
 큐 계정 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG queueAccounting ( );**  
 오류 가능성에 대한 표시 없이 큐 계정 값을 리턴합니다.

**ImqBoolean queueMonitoring ( MQLONG & monq );**  
 큐 모니터링 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG queueMonitoring ( );**  
 오류 가능성에 대한 표시 없이 큐 모니터링 값을 리턴합니다.

**ImqBoolean queueStatistics ( MQLONG & statq );**  
 큐 통계 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG queueStatistics ( );**  
 오류 가능성에 대한 표시 없이 큐 통계 값을 리턴합니다.

**ImqBoolean receiveTimeout ( MQLONG & timeout );**  
 수신 제한시간 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG receiveTimeout ( );**  
 오류 가능성에 대한 표시 없이 수신 제한시간 값을 리턴합니다.

**ImqBoolean receiveTimeoutMin ( MQLONG & min );**  
 최소 수신 제한시간 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG receiveTimeoutMin ( );**  
 오류 가능성에 대한 표시 없이 최소 수신 제한시간 값을 리턴합니다.

**ImqBoolean receiveTimeoutType ( MQLONG & type );**  
 수신 제한시간 유형의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG receiveTimeoutType ( );**  
 오류 가능성에 대한 표시 없이 수신 제한시간 유형을 리턴합니다.

**ImqBoolean remoteEvent( MQLONG & event );**  
 리모트 이벤트의 인에이블먼트 상태의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG remoteEvent( );**  
 오류 가능성에 대한 표시 없이 리모트 이벤트의 인에이블먼트 상태를 리턴합니다.

**ImqBoolean repositoryName( ImqString & name );**  
 저장소 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString repositoryName( );**  
 오류 가능성에 대한 표시 없이 저장소 이름을 리턴합니다.

**ImqBoolean repositoryNamelistName( ImqString & name );**  
 저장소 이름 목록 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString repositoryNamelistName( );**  
 오류 가능성에 대한 표시 없이 저장소 이름 목록 이름의 사본을 리턴합니다.

**ImqBoolean sharedQueueQueueManagerName ( MQLONG & name );**  
 공유 큐의 큐 관리자 이름 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG sharedQueueQueueManagerName ( );**  
 오류 가능성에 대한 표시 없이 공유 큐의 큐 관리자 이름 값을 리턴합니다.

**ImqBoolean sslEvent ( MQLONG & event );**  
 SSL 이벤트 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG sslEvent ( );**  
 오류 가능성에 대한 표시 없이 SSL 이벤트 값을 리턴합니다.

**ImqBoolean sslFips ( MQLONG & sslfips );**  
 SSL FIPS 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG sslFips ( );**  
 오류 가능성에 대한 표시 없이 SSL FIPS 값을 리턴합니다.

**ImqBoolean sslKeyResetCount ( MQLONG & count );**  
 SSL 키 재설정 수 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG sslKeyResetCount ( );**

오류 가능성에 대한 표시 없이 SSL 키 재설정 수 값을 리턴합니다.

**ImqBoolean startStopEvent( MQLONG & event );**

시작-중지 이벤트의 인에이블먼트 상태의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG startStopEvent( );**

오류 가능성에 대한 표시 없이 시작-중지 이벤트의 인에이블먼트 상태를 리턴합니다.

**ImqBoolean statisticsInterval ( MQLONG & statint );**

통계 간격 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG statisticsInterval ( );**

오류 가능성에 대한 표시 없이 통계 간격 값을 리턴합니다.

**ImqBoolean syncPointAvailability( MQLONG & sync );**

동기점 가용성 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG syncPointAvailability( );**

오류 가능성에 대한 표시 없이 동기점 가용성 값의 사본을 리턴합니다.

**ImqBoolean tcpName ( ImqString & name );**

TCP 시스템 이름의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**ImqString tcpName ( );**

오류 가능성에 대한 표시 없이 TCP 시스템 이름을 리턴합니다.

**ImqBoolean tcpStackType ( MQLONG & type );**

TCP 스택 유형의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG tcpStackType ( );**

오류 가능성에 대한 표시 없이 TCP 스택 유형을 리턴합니다.

**ImqBoolean traceRouteRecording ( MQLONG & routerec );**

라우트 추적 기록 값의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG traceRouteRecording ( );**

오류 가능성에 대한 표시 없이 라우트 추적 기록 값을 리턴합니다.

**ImqBoolean triggerInterval( MQLONG & interval );**

트리거 간격의 사본을 제공합니다. 성공하면 TRUE를 리턴합니다.

**MQLONG triggerInterval( );**

오류 가능성에 대한 표시 없이 트리거 간격을 리턴합니다.

**ImqBinary userId ( ) const;**

클라이언트 연결에 사용된 사용자 ID를 리턴합니다.

**ImqBoolean setUserId ( const ImqString & id );**

클라이언트 연결에 사용되는 사용자 ID를 설정합니다.

**ImqBoolean setUserId ( const char \* = 0 id );**

클라이언트 연결에 사용되는 사용자 ID를 설정합니다.

**ImqBoolean setUserId ( const ImqBinary & id );**

클라이언트 연결에 사용되는 사용자 ID를 설정합니다.

**오브젝트 메소드(보호됨)****void setFirstManagedObject ( const ImqObject \* object = 0 );**

첫 번째 관리 오브젝트를 설정합니다.

**오브젝트 데이터(보호됨)****MQHCONN ohconn**

IBM MQ 연결 핸들(연결 상태가 TRUE인 동안에만 의미가 있음)

**이유 코드**

- MQRC\_ATTRIBUTE\_LOCKED

- MQRC\_ENVIRONMENT\_ERROR
- MQRC\_FUNCTION\_NOT\_SUPPORTED
- MQRC\_REFERENCE\_ERROR
- (MQBACK에 대한 이유 코드)
- (MQBEGIN에 대한 이유 코드)
- (MQCMIT에 대한 이유 코드)
- (MQCONN에 대한 이유 코드)
- (MQDISC에 대한 이유 코드)
- (MQCONN에 대한 이유 코드)

## ImqReferenceHeader C++ 클래스

이 클래스는 MQRMH 데이터 구조의 기능을 캡슐화합니다.

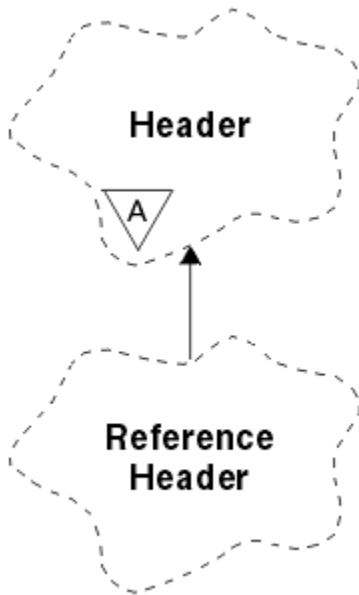


그림 34. *ImqReferenceHeader* 클래스

이 클래스는 1721 페이지의 『[ImqReferenceHeader 상호 참조](#)』에 나열된 MQI 호출과 관련이 있습니다.

- [1799 페이지의 『오브젝트 속성』](#)
- [1800 페이지의 『구성자』](#)
- [1800 페이지의 『과부하된 ImqItem 메소드』](#)
- [1800 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1801 페이지의 『오브젝트 데이터\(보호됨\)』](#)
- [1801 페이지의 『이유 코드』](#)

### 오브젝트 속성

#### 목적지 환경

대상의 환경. 초기값은 널 문자열입니다.

#### 목적지 이름

데이터 대상의 이름. 초기값은 널 문자열입니다.

#### 인스턴스 ID

인스턴스 ID. MQ\_OBJECT\_INSTANCE\_ID\_LENGTH 길이의 2진 값(MQBYTE24)입니다. 초기값은 MQOII\_NONE입니다.

## 논리 길이

이 헤더 뒤에 오는 메시지 데이터의 논리 길이 또는 의도한 길이. 초기값은 0입니다.

## 논리 오프셋

최종 목적지의 데이터 컨텍스트에서 전반적으로 해석되도록, 뒤에 오는 메시지 데이터에 대한 논리 오프셋. 초기값은 0입니다.

## 논리 오프셋 2

논리 오프셋에 대한 높은 순서의 확장. 초기값은 0입니다.

## 참조 유형

참조 유형입니다. 초기값은 널 문자열입니다.

## 소스 환경

소스의 환경입니다. 초기값은 널 문자열입니다.

## 소스 이름

데이터 소스의 이름입니다. 초기값은 널 문자열입니다.

## 구성자

### **ImqReferenceHeader();**

기본 구성자입니다.

### **ImqReferenceHeader( const ImqReferenceHeader & header );**

복사 구성자입니다.

## 과부하된 ImqItem 메소드

### **virtual ImqBoolean copyOut( ImqMessage & msg );**

시작 시 MQRMH 데이터 구조를 메시지 버퍼에 삽입하고, 그에 따라 기존 메시지 데이터를 이동하며, *msg* 형식을 MQFMT\_REF\_MSG\_HEADER로 설정합니다.

자세한 정보는 1748 페이지의 『ImqHeader C++ 클래스』의 ImqHeader 클래스 메소드 설명을 참조하십시오.

### **virtual ImqBoolean pasteIn( ImqMessage & msg );**

메시지 버퍼에서 MQRMH 데이터 구조를 읽어옵니다.

성공적인 작업을 위해 ImqMessage 형식이 MQFMT\_REF\_MSG\_HEADER여야 합니다.

자세한 정보는 1748 페이지의 『ImqHeader C++ 클래스』의 ImqHeader 클래스 메소드 설명을 참조하십시오.

## 오브젝트 메소드(공용)

### **void operator = ( const ImqReferenceHeader & header );**

*header*에서 인스턴스 데이터를 복사하고 기존 인스턴스 데이터를 대체합니다.

### **ImqString destinationEnvironment ( ) const ;**

대상 환경의 사본을 리턴합니다.

### **void setDestinationEnvironment ( const char \* environment = 0 );**

대상 환경을 설정합니다.

### **ImqString destinationName ( ) const ;**

대상 이름의 사본을 리턴합니다.

### **void setDestinationName ( const char \* name = 0 );**

대상 이름을 설정합니다.

### **ImqBinary instanceId ( ) const ;**

인스턴스 ID의 사본을 리턴합니다.

### **ImqBoolean setInstanceId ( const ImqBinary & id );**

인스턴스 ID를 설정합니다. *token*의 데이터 길이는 0 또는 MQ\_OBJECT\_INSTANCE\_ID\_LENGTH여야 합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

**void setId ( const MQBYTE24 id = 0 );**

인스턴스 ID를 설정합니다. *id*가 0일 수 있으며, 이는 MQOII\_NONE을 지정하는 것과 같습니다. *id*가 0이 아니면 MQ\_OBJECT\_INSTANCE\_ID\_LENGTH 바이트의 2진 데이터를 처리해야 합니다. MQOII\_NONE과 같은 사전정의된 값을 사용하는 경우, 서명이 일치하도록 캐스트를 만들어야 할 수 있습니다. 예: (MQBYTE \*)MQOII\_NONE.

**MQLONG logicalLength ( ) const ;**

논리 길이를 리턴합니다.

**void setLogicalLength ( const MQLONG length );**

논리 길이를 설정합니다.

**MQLONG logicalOffset ( ) const ;**

논리 오프셋을 리턴합니다.

**void setLogicalOffset ( const MQLONG offset );**

논리 오프셋을 설정합니다.

**MQLONG logicalOffset2 ( ) const ;**

논리 오프셋 2를 리턴합니다.

**void setLogicalOffset2 ( const MQLONG offset );**

논리 오프셋 2를 설정합니다.

**ImqString referenceType ( ) const ;**

참조 유형의 사본을 리턴합니다.

**void setReferenceType ( const char \* name = 0 );**

참조 유형을 설정합니다.

**ImqString sourceEnvironment ( ) const ;**

소스 환경의 사본을 리턴합니다.

**void setSourceEnvironment ( const char \* environment = 0 );**

소스 환경을 설정합니다.

**ImqString sourceName ( ) const ;**

소스 이름의 사본을 리턴합니다.

**void setSourceName ( const char \* name = 0 );**

소스 이름을 설정합니다.

## 오브젝트 데이터(보호됨)

**MQRMH omqrmh**

MQRMH 데이터 구조입니다.

## 이유 코드

- MQRC\_BINARY\_DATA\_LENGTH\_ERROR
- MQRC\_STRUC\_LENGTH\_ERROR
- MQRC\_STRUC\_ID\_ERROR
- MQRC\_INSUFFICIENT\_DATA
- MQRC\_INCONSISTENT\_FORMAT
- MQRC\_ENCODING\_ERROR

## ImqString C++ 클래스

이 클래스는 널(Null) 종료 문자열에 대한 문자열 스토리지 및 조작을 제공합니다.

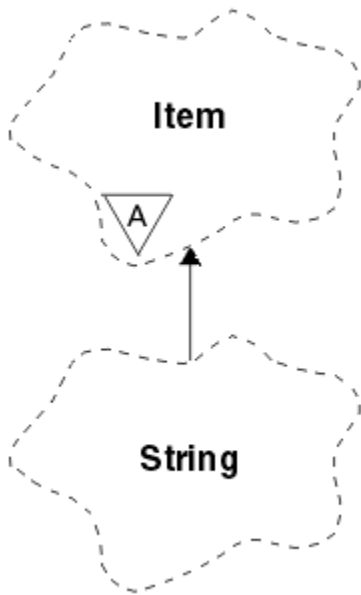


그림 35. *ImqString* 클래스

매개변수가 **char \***를 요청하는 대부분의 상황에서 **char \*** 대신 *ImqString*을 사용하십시오.

- [1802 페이지의 『오브젝트 속성』](#)
- [1802 페이지의 『구성자』](#)
- [1803 페이지의 『클래스 메소드\(공용\)』](#)
- [1803 페이지의 『과부하된 \*ImqItem\* 메소드』](#)
- [1803 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1806 페이지의 『오브젝트 메소드\(보호됨\)』](#)
- [1806 페이지의 『이유 코드』](#)

## 오브젝트 속성

### 문자

후미 널 앞에 오는 스토리지의 문자입니다.

### length

문자의 바이트 수입니다. 스토리지가 없으면 길이가 0입니다. 초기값은 0입니다.

### 스토리지

임의 바이트 크기의 일시적 배열입니다. 문자의 끝을 감지할 수 있도록 후미 널이 항상 문자 뒤의 스토리지에 있어야 합니다. 메소드가 이 상황을 유지해 주지만, 배열에 직접 바이트를 설정하는 경우에는 후미 널이 수정 후에 있도록 합니다. 처음에는 스토리지 속성이 없습니다.

## 구성자

### **ImqString();**

기본 구성자입니다.

### **ImqString( const ImqString & string );**

복사 구성자입니다.

### **ImqString( const char c );**

문자가 *c*로 구성됩니다.

### **ImqString( const char \* text );**

문자가 *text*에서 복사됩니다.

### **ImqString( const void \* *buffer*, const size\_t *length* );**

*buffer*에서부터 *length* 바이트를 복사하고 이를 문자에 지정합니다. 복사된 널 문자에 대해서는 대체가 수행됩니다. 대체 문자는 마침표(.)입니다. 인쇄할 수 없거나 표시할 수 없는 문자를 복사할 때 특별한 고려사항이 없습니다.

### **클래스 메소드(공용)**

#### **static ImqBoolean copy( char \* *destination-buffer*, const size\_t *length*, const char \* *source-buffer*, const char *pad* = 0 );**

*source-buffer*에서 *destination-buffer*로 최대 *length* 바이트를 복사합니다. *source-buffer*에서 문자 수가 충분하지 않으면 *destination-buffer*의 나머지 공간에 *pad* 문자를 채웁니다. *source-buffer*는 0일 수 있습니다. 또한 *length*가 0이면 *destination-buffer*가 0이 될 수 있습니다. 오류 코드가 유실됩니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

#### **static ImqBoolean copy ( char \* *destination-buffer*, const size\_t *length*, const char \* *source-buffer*, ImqError & *error-object*, const char *pad* = 0 );**

*source-buffer*에서 *destination-buffer*로 최대 *length* 바이트를 복사합니다. *source-buffer*에서 문자 수가 충분하지 않으면 *destination-buffer*의 나머지 공간에 *pad* 문자를 채웁니다. *source-buffer*는 0일 수 있습니다. 또한 *length*가 0이면 *destination-buffer*가 0이 될 수 있습니다. 오류 코드가 *error-object*에 설정됩니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

### **과부하된 ImqItem 메소드**

#### **virtual ImqBoolean copyOut ( ImqMessage & *msg* );**

기존 콘텐츠를 대체하고 문자를 메시지 버퍼에 복사합니다. *msg* 형식을 MQFMT\_STRING으로 설정합니다. 추가적인 세부사항은 상위 클래스 메소드 설명을 참조하십시오.

#### **virtual ImqBoolean pasteIn ( ImqMessage & *msg* );**

메시지 버퍼에서 나머지 데이터를 전송하여 문자를 설정하고, 기존 문자를 대체합니다.

성공적인 작업을 위해 *msg* 오브젝트의 인코딩이 MQENC\_NATIVE이어야 합니다. MQGMO\_CONVERT ~ MQENC\_NATIVE로 메시지를 검색하십시오.

성공적인 작업을 위해 ImqMessage 형식이 MQFMT\_STRING이어야 합니다.

추가적인 세부사항은 상위 클래스 메소드 설명을 참조하십시오.

### **오브젝트 메소드(공용)**

#### **char & operator [ ] ( const size\_t *offset* ) const ;**

스토리지의 오프셋 *offset*에 있는 문자를 참조합니다. 관련 바이트가 존재하고 처리 가능한지 확인하십시오.

#### **ImqString operator ( ) ( const size\_t *offset*, const size\_t *length* = 1 ) const ;**

*offset*에서 시작하는 문자에서 바이트를 복사하여 하위 문자열을 리턴합니다. *length*가 0이면 문자의 나머지를 리턴합니다. *offset* 및 *length*의 조합이 문자 내에서 참조를 생성하지 않는 경우, 빈 ImqString을 리턴합니다.

#### **void operator = ( const ImqString & *string* );**

*string*에서 인스턴스 데이터를 복사하고, 기존 인스턴스 데이터를 대체합니다.

#### **ImqString operator + ( const char *c* ) const ;**

*c*를 문자에 추가한 결과를 리턴합니다.

#### **ImqString operator + ( const char \* *text* ) const ;**

*text*를 문자에 추가한 결과를 리턴합니다. 결과가 뒤바뀔 수도 있습니다. 예를 들면, 다음과 같습니다.

```
strOne + "string two" ;  
"string one" + strTwo ;
```

참고: 대부분의 컴파일러가 **strOne + "string two;"**를 승인하지만, Microsoft Visual C++에는 **strOne + (char \*)"string two" ;**가 필요합니다.

**ImqString operator + ( const ImqString & string1 ) const ;**

*string1*을 문자에 추가한 결과를 리턴합니다.

**ImqString operator + ( const double number ) const ;**

텍스트로 변환한 후 *number*를 문자에 추가한 결과를 리턴합니다.

**ImqString operator + ( const long number ) const ;**

텍스트로 변환한 후 *number*를 문자에 추가한 결과를 리턴합니다.

**void operator += ( const char c );**

*c*를 문자에 추가합니다.

**void operator += ( const char \* text );**

*text*를 문자에 추가합니다.

**void operator += ( const ImqString & string );**

*string*을 문자에 추가합니다.

**void operator += ( const double number );**

텍스트로 변환한 후 *number*를 문자에 추가합니다.

**void operator += ( const long number );**

텍스트로 변환한 후 *number*를 문자에 추가합니다.

**operator char \* ( ) const ;**

스토리지에 있는 첫 번째 바이트의 주소를 리턴합니다. 이 값은 0일 수 있으며, 일시적입니다. 이 메소드는 읽기 전용 용도로만 사용하십시오.

**ImqBoolean operator < ( const ImqString & string ) const ;**

**compare** 메소드를 사용하여 문자를 *string*의 문자와 비교합니다. '보다 작음'이면 결과가 TRUE이고 '보다 큼'이면 FALSE입니다.

**ImqBoolean operator > ( const ImqString & string ) const ;**

**compare** 메소드를 사용하여 문자를 *string*의 문자와 비교합니다. '초과'이면 결과가 TRUE이고 '이하'이면 FALSE입니다.

**ImqBoolean operator <= ( const ImqString & string ) const ;**

**compare** 메소드를 사용하여 문자를 *string*의 문자와 비교합니다. '이하'이면 결과가 TRUE이고 '초과'이면 FALSE입니다.

**ImqBoolean operator >= ( const ImqString & string ) const ;**

**compare** 메소드를 사용하여 문자를 *string*의 문자와 비교합니다. '이상'이면 결과가 TRUE이고 '미만'이면 FALSE입니다.

**ImqBoolean operator == ( const ImqString & string ) const ;**

**compare** 메소드를 사용하여 문자를 *string*의 문자와 비교합니다. TRUE 또는 FALSE를 리턴합니다.

**ImqBoolean operator != ( const ImqString & string ) const ;**

**compare** 메소드를 사용하여 문자를 *string*의 문자와 비교합니다. TRUE 또는 FALSE를 리턴합니다.

**short compare( const ImqString & string ) const ;**

문자를 *string*의 문자와 비교합니다. 문자가 같으면 결과가 0이고, '보다 작음'이면 음수이고, '보다 큼'이면 양수입니다. 비교는 대소문자를 구분합니다. 널 ImqString은 널이 아닌 ImqString보다 작은 것으로 간주됩니다.

**ImqBoolean copyOut( char \* buffer, const size\_t length, const char pad = 0 );**

문자에서 *buffer*로 최대 *length*바이트를 복사합니다. 문자의 수가 충분하지 않으면 *buffer*의 나머지 공간을 *pad* 문자로 채웁니다. 또한 *length*가 0이면 *buffer*가 0이 될 수 있습니다. 성공하면 TRUE를 리턴합니다.

**size\_t copyOut( long & number ) const ;**

텍스트로 변환한 후 문자에서 *number*를 설정하고, 변환에 포함된 문자의 수를 리턴합니다. 이 값이 0이면 변환이 수행되지 않고 *number*가 설정되지 않습니다. 변환 가능한 문자 시퀀스는 다음 값으로 시작해야 합니다.

```
<blank(s)>  
<+|->  
digit(s)
```



### **size\_t copyOut( ImqString & token, const char c = '' ) const ;**

문자에 *c*가 아닌 문자가 하나 이상 포함되어 있으면 해당 문자의 첫 번째 연속 시퀀스로 토큰을 식별합니다. 이 경우, *token*이 해당 시퀀스로 설정되고 리턴된 값은 선두 문자 *c*의 수와 시퀀스에 있는 바이트 수의 합계입니다. 그렇지 않으면 0을 리턴하고 *token*을 설정하지 않습니다.

### **size\_t cutOut( long & number );**

**copy** 메소드에 대해 *number*를 설정하고, 문자에서 리턴 값에 표시된 바이트 수를 제거합니다. 예를 들어, 다음 예에 표시된 문자열은 **cutOut** (*number*)을 세 번 사용하여 세 개의 숫자로 자를 수 있습니다.

```
strNumbers = "-1 0 +55 "
while ( strNumbers.cutOut( number ) );
number becomes -1, then 0, then 55
leaving strNumbers == " "
```

### **size\_t cutOut( ImqString & token, const char c = '' )**

**copyOut** 메소드에 대해 *token*을 설정하고, 문자에서 *strToken* 문자와 *token* 문자 앞에 오는 *c* 문자를 제거합니다. *c*가 공백이면 *token* 문자 바로 뒤에 오는 *c* 문자를 제거합니다. 제거된 문자의 수를 리턴합니다. 예를 들어, 다음 예에 표시된 문자열은 **cutOut** (*token*)을 세 번 사용하여 세 개의 토큰으로 자를 수 있습니다.

```
strText = " Program Version 1.1 "
while ( strText.cutOut( token ) );
// token becomes "Program", then "Version",
// then "1.1" leaving strText == " "
```

다음 예는 DOS 경로 이름을 구문 분석하는 방법을 보여줍니다.

```
strPath = "C:\OS2\BITMAP\OS2LOGO.BMP"
strPath.cutOut( strDrive, ':' );
strPath.stripLeading( ':' );
while ( strPath.cutOut( strFile, '\\' ) );
// strDrive becomes "C".
// strFile becomes "OS2", then "BITMAP",
// then "OS2LOGO.BMP" leaving strPath empty.
```

### **ImqBoolean find( const ImqString & string );**

문자 내 어디에서든 *string*의 정확한 일치 항목을 검색합니다. 일치하는 항목이 없으면 FALSE를 리턴합니다. 그렇지 않으면, TRUE를 리턴합니다. *string*이 널이면 TRUE를 리턴합니다.

### **ImqBoolean find( const ImqString & string, size\_t & offset );**

이후 오프셋 *offset*의 문자 내 어딘가에서 *string*의 정확한 일치 항목을 검색합니다. *string*이 널이면 *offset*을 업데이트하지 않고 TRUE를 리턴합니다. 일치하는 항목이 없으면 FALSE를 리턴합니다(*offset*의 값이 증가했을 수 있음). 일치하는 항목이 있으면 TRUE를 리턴하고 *offset*을 문자 내 문자열의 오프셋으로 업데이트합니다.

### **size\_t length() const ;**

길이를 리턴합니다.

### **ImqBoolean pasteIn( const double number, const char \* format = "%f" );**

텍스트로 변환한 후 *number*를 문자에 추가합니다. 성공하면 TRUE를 리턴합니다.

스펙 *format*은 부동 소수점 변환을 형식화하는 데 사용합니다. 지정하는 경우, 이 값은 **printf**와 부동 소수점 숫자와 함께 사용하기에 적절한 값이어야 합니다(예: **%3f**).

### **ImqBoolean pasteIn( const long number );**

텍스트로 변환한 후 *number*를 문자에 추가합니다. 성공하면 TRUE를 리턴합니다.

### **ImqBoolean pasteIn( const void \* buffer, const size\_t length );**

*buffer*의 *length* 바이트를 문자에 추가하고, 마지막 후미 널을 추가합니다. 복사되는 널 문자를 대체합니다. 대체 문자는 마침표(.)입니다. 인쇄할 수 없거나 표시할 수 없는 문자를 복사할 때 특별한 고려사항이 없습니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

### **ImqBoolean set( const char \* *buffer*, const size\_t *length* );**

널이 있을 수 있는 고정 길이 문자 필드에서 문자를 설정합니다. 필요한 경우, 고정 길이 필드의 문자에 널을 추가합니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

### **ImqBoolean setStorage( const size\_t *length* );**

스토리를 할당(또는 재할당)합니다. 문자를 위한 공간이 여전히 있지만 추가 스토리를 초기화하지 않은 경우, 후미 널을 포함한 원본 문자를 보존합니다.

이 메소드는 성공하면 TRUE를 리턴합니다.

### **size\_t storage( ) const ;**

스토리의 바이트 수를 리턴합니다.

### **size\_t stripLeading( const char *c* = ' ' );**

문자에서 선두 문자 *c*를 제거하고, 제거된 수를 리턴합니다.

### **size\_t stripTrailing( const char *c* = ' ' );**

문자에서 후미 문자 *c*를 제거하고, 제거된 수를 리턴합니다.

### **ImqString upperCase( ) const ;**

문자의 대문자 사본을 리턴합니다.

## **오브젝트 메소드(보호됨)**

### **ImqBoolean assign ( const ImqString & *string* );**

동일한 **operator =** 메소드와 같은 기능을 수행하지만, 가상이 아닙니다. 성공하면 TRUE를 리턴합니다.

## **이유 코드**

- MQRC\_DATA\_TRUNCATED
- MQRC\_NULL\_POINTER
- MQRC\_STORAGE\_NOT\_AVAILABLE
- MQRC\_BUFFER\_ERROR
- MQRC\_INCONSISTENT\_FORMAT

## **ImqTrigger C++ 클래스**

이 클래스는 MQTM(트리거 메시지) 데이터 구조를 캡슐화합니다.

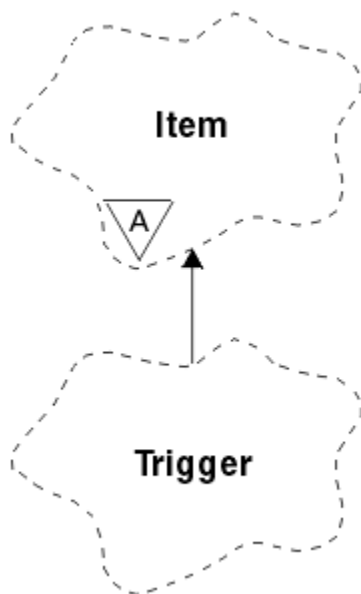


그림 36. *ImqTrigger* 클래스

이 클래스의 오브젝트는 일반적으로 트리거 모니터 프로그램에 사용됩니다. 트리거 모니터 프로그램의 태스크는 이러한 특정 메시지를 대기하고 해당 메시지에 따라 조치를 취해 메시지가 대기 중일 때 다른 IBM MQ 애플리케이션이 시작되도록 하는 것입니다.

사용법 예는 IMQSTRG 샘플 프로그램을 참조하십시오.

- [1807 페이지의 『오브젝트 속성』](#)
- [1807 페이지의 『구성자』](#)
- [1808 페이지의 『과부하된 ImqItem 메소드』](#)
- [1808 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1809 페이지의 『오브젝트 데이터\(보호됨\)』](#)
- [1809 페이지의 『이유 코드』](#)

## 오브젝트 속성

### 애플리케이션 ID

메시지를 보낸 애플리케이션의 ID. 초기값은 널 문자열입니다.

### 애플리케이션 유형

메시지를 보낸 애플리케이션의 유형. 초기값은 0입니다. 다음과 같은 추가 값을 사용할 수 있습니다.

- MQAT\_AIX
- MQAT\_CICS
- MQAT\_DOS
- MQAT\_IMS
- MQAT\_MVS
- MQAT\_NOTES\_AGENT
- MQAT\_OS2
- MQAT\_OS390
- MQAT\_OS400
- MQAT\_UNIX
- MQAT\_WINDOWS
- MQAT\_WINDOWS\_NT
- MQAT\_USER\_FIRST
- MQAT\_USER\_LAST

### 환경 데이터

프로세스에 대한 환경 데이터. 초기값은 널 문자열입니다.

### 프로세스 이름

프로세스 이름입니다. 초기값은 널 문자열입니다.

### 큐 이름

시작할 큐의 이름. 초기값은 널 문자열입니다.

### 트리거 데이터

프로세스에 대한 트리거 데이터. 초기값은 널 문자열입니다.

### 사용자 데이터

프로세스에 대한 사용자 데이터. 초기값은 널 문자열입니다.

## 구성자

### **ImqTrigger();**

기본 구성자입니다.

### **ImqTrigger( const ImqTrigger & trigger );**

복사 구성자입니다.

## 과부하된 ImqItem 메소드

### **virtual ImqBoolean copyOut( ImqMessage & msg );**

MQTM 데이터 구조를 메시지 버퍼에 작성하고, 기존 콘텐츠를 대체합니다. *msg* 형식을 MQFMT\_TRIGGER 로 설정합니다.

자세한 정보는 1752 페이지의 『ImqItem C++ 클래스』의 ImqItem 클래스 메소드 설명을 참조하십시오.

### **virtual ImqBoolean pasteIn( ImqMessage & msg );**

메시지 버퍼에서 MQTM 데이터 구조를 읽어옵니다.

성공적인 작업을 위해 ImqMessage 형식이 MQFMT\_TRIGGER여야 합니다.

자세한 정보는 1752 페이지의 『ImqItem C++ 클래스』의 ImqItem 클래스 메소드 설명을 참조하십시오.

## 오브젝트 메소드(공용)

### **void operator = ( const ImqTrigger & trigger );**

*trigger*에서 인스턴스 데이터를 복사하고 기존 인스턴스 데이터를 대체합니다.

### **ImqString applicationId ( ) const ;**

애플리케이션 ID의 사본을 리턴합니다.

### **void setApplicationId ( const char \* id );**

애플리케이션 ID를 설정합니다.

### **MQLONG applicationType ( ) const ;**

애플리케이션 유형을 리턴합니다.

### **void setApplicationType ( const MQLONG type );**

애플리케이션 유형을 설정합니다.

### **ImqBoolean copyOut ( MQTMC2 \* ptmc2 );**

이니시에이션 큐에 수신된 MQTM 데이터 구조를 캡슐화합니다. 호출자가 제공하는 동일한 MQTMC2 데이터 구조를 채우고, QMgrName 필드(MQTM 데이터 구조에 표시되지 않음)를 모두 공백으로 설정합니다.

MQTMC2 데이터 구조는 예전부터 트리거 모니터에 의해 시작된 애플리케이션에 대한 매개변수로 사용됩니다. 이 메소드는 성공하면 TRUE를 리턴합니다.

### **ImqString environmentData ( ) const ;**

환경 데이터의 사본을 리턴합니다.

### **void setEnvironmentData ( const char \* data );**

환경 데이터를 설정합니다.

### **ImqString processName ( ) const ;**

프로세스 이름의 사본을 리턴합니다.

### **void setProcessName ( const char \* name );**

공백으로 채워진 프로세스 이름을 48자로 설정합니다.

### **ImqString queueName ( ) const ;**

큐 이름의 사본을 리턴합니다.

### **void setQueueName ( const char \* name );**

공백으로 채워진 큐 이름을 48자로 설정합니다.

### **ImqString triggerData ( ) const ;**

트리거 데이터의 사본을 리턴합니다.

### **void setTriggerData ( const char \* data );**

트리거 데이터를 설정합니다.

### **ImqString userData ( ) const ;**

사용자 데이터의 사본을 리턴합니다.

### **void setUserData ( const char \* data );**

사용자 데이터를 설정합니다.

## 오브젝트 데이터(보호됨)

### MQTM *omqtm*

MQTM 데이터 구조.

### 이유 코드

- MQRC\_NULL\_POINTER
- MQRC\_INCONSISTENT\_FORMAT
- MQRC\_ENCODING\_ERROR
- MQRC\_STRUC\_ID\_ERROR

## ImqWorkHeader C++ 클래스

이 클래스는 MQWIH 데이터 구조의 특정 기능을 캡슐화합니다.

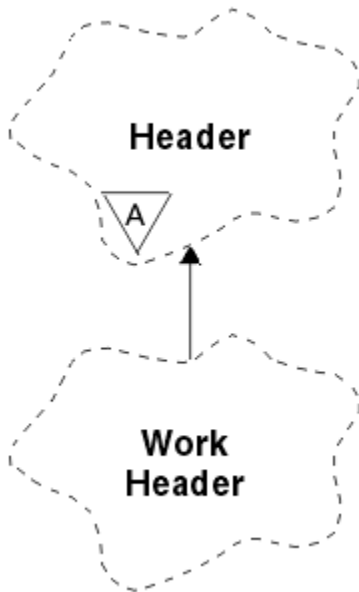


그림 37. *ImqWorkHeader* 클래스

이 클래스의 오브젝트는 z/OS 워크로드 관리자를 통해 관리되는 큐에 메시지를 넣는 애플리케이션에 사용됩니다.

- [1809 페이지의 『오브젝트 속성』](#)
- [1810 페이지의 『구성자』](#)
- [1810 페이지의 『과부하된 ImqItem 메소드』](#)
- [1810 페이지의 『오브젝트 메소드\(공용\)』](#)
- [1810 페이지의 『오브젝트 데이터\(보호됨\)』](#)
- [1810 페이지의 『이유 코드』](#)

## 오브젝트 속성

### 메시지 토큰(message token)

MQ\_MSG\_TOKEN\_LENGTH 길이의 z/OS 워크로드 관리자에 대한 메시지 토큰. 초기값은 MQMTOK\_NONE입니다.

### 서비스 이름

32자의 프로세스 이름. 이 이름은 처음에는 공백입니다.

### 서비스 단계

프로세스 내에 있는 단계의 8자 이름. 이 이름은 처음에는 공백입니다.

## 구성자

**ImqWorkHeader();**

기본 구성자입니다.

**ImqWorkHeader( const ImqWorkHeader & header );**

복사 구성자입니다.

## 과부하된 ImqItem 메소드

**virtual ImqBoolean copyOut( ImqMessage & msg );**

메시지 버퍼의 시작 부분에 MQWIH 데이터 구조를 삽입하고, 그에 따라 기존 메시지 데이터를 이동하며, *msg* 형식을 MQFMT\_WORK\_INFO\_HEADER로 설정합니다.

자세한 정보는 상위 클래스 메소드 설명을 참조하십시오.

**virtual ImqBoolean pasteIn( ImqMessage & msg );**

메시지 버퍼에서 MQWIH 데이터 구조를 읽어옵니다.

성공적인 작업을 위해 *msg* 오브젝트의 인코딩이 MQENC\_NATIVE이어야 합니다. MQGMO\_CONVERT ~ MQENC\_NATIVE로 메시지를 검색하십시오.

ImqMessage 형식이 MQFMT\_WORK\_INFO\_HEADER여야 합니다.

자세한 정보는 상위 클래스 메소드 설명을 참조하십시오.

## 오브젝트 메소드(공용)

**void operator = ( const ImqWorkHeader & header );**

*header*에서 인스턴스 데이터를 복사하고 기존 인스턴스 데이터를 대체합니다.

**ImqBinary messageToken ( ) const;**

메시지 토큰을 리턴합니다.

**ImqBoolean setMessageToken( const ImqBinary & token );**

메시지 토큰을 설정합니다. *token*의 데이터 길이는 0 또는 MQ\_MSG\_TOKEN\_LENGTH이어야 합니다. 성공하면 TRUE를 리턴합니다.

**void setMessageToken( const MQBYTE16 token = 0 );**

메시지 토큰을 설정합니다. *token*은 0일 수 있으며, 이는 MQMTOK\_NONE을 지정하는 것과 같습니다. *token*이 0이 아니면 MQ\_MSG\_TOKEN\_LENGTH 바이트의 2진 데이터를 처리해야 합니다.

MQMTOK\_NONE과 같은 사전정의된 값을 사용하는 경우, 서명이 일치하도록 캐스트를 만들어야 할 수 있습니다. 예: (MQBYTE \*)MQMTOK\_NONE.

**ImqString serviceName ( ) const;**

후미 공백을 포함하여 서비스 이름을 리턴합니다.

**void setServiceName( const char \* name );**

서비스 이름을 설정합니다.

**ImqString serviceStep ( ) const;**

후미 공백을 포함하여 서비스 단계를 리턴합니다.

**void setServiceStep( const char \* step );**

서비스 단계를 설정합니다.

## 오브젝트 데이터(보호됨)

**MQWIH omqwih**

MQWIH 데이터 구조.

## 이유 코드

- MQRC\_BINARY\_DATA\_LENGTH\_ERROR

## IBM MQ classes for JMS 오브젝트의 특성

IBM MQ classes for JMS의 모든 오브젝트는 특성을 갖습니다. 다양한 오브젝트 유형에 각기 다른 특성이 적용됩니다. 특성은 저마다 다른 허용 가능 값을 가지며, 기호 특성 값은 관리 도구와 프로그램 코드에서 서로 다릅니다.

IBM MQ classes for JMS에서는 IBM MQ JMS 관리 도구, IBM MQ 탐색기를 사용하거나 애플리케이션 안에서 오브젝트의 특성을 설정하고 조회하는 기능을 제공합니다. 이들 특성의 대부분은 오브젝트 유형의 특정 서브세트와만 관련이 있습니다.

IBM MQ JMS 관리 도구의 사용 방법에 대한 자세한 정보는 [관리 도구를 사용하여 JMS 오브젝트 구성을 참조하십시오](#).

1811 페이지의 표 868에서는 각 특성에 대한 간단한 설명과, 각 특성이 어떤 오브젝트 유형에 적용되는지를 보여줍니다. 오브젝트 유형은 키워드를 사용하여 식별됩니다. 이 오브젝트에 대한 설명은 [관리 도구를 사용하여 JMS 오브젝트 구성을 참조하십시오](#).

숫자는 테이블 끝에 나온 참고를 의미합니다. 1814 페이지의 『IBM MQ classes for JMS 오브젝트 특성의 종속성』도 참조하십시오.

특성은 다음 형식의 이름-값 쌍으로 구성됩니다.

```
PROPERTY_NAME(property_value)
```

이 절의 주제에서는 각 특성에 대한 특성 이름 및 간략한 설명을 나열하고 관리 도구에서 사용되는 올바른 특성 값을 표시합니다. 및 애플리케이션에서 특성의 값을 설정하는 데 사용되는 set 메소드를 포함합니다. 뿐만 아니라 각 특성의 올바른 특성 값 및 도구에 사용된 기호 특성 값과 프로그래밍 가능한 동일 특성 값 간의 매핑도 보여줍니다.

특성 이름은 대소문자를 구분하지 않고, 이 주제에 나와 있는 인식 이름 세트로 제한됩니다.

특성	축약형	오브젝트 유형							
		CF	QCF	TCF	Q	T	XACF	XAQCF	XATCF
<a href="#">1816 페이지의 『APPLICATIONNAME』</a>	APPNAME	Y	Y	Y			Y	Y	Y
<a href="#">1817 페이지의 『ASYNCEXCEPTION』</a>	AEX	Y	Y	Y			Y	Y	Y
<a href="#">1818 페이지의 『기본 옵션』</a>	옵션	Y	Y	Y			Y	Y	Y
<a href="#">1818 페이지의 『BALTYPE』</a>	유형	Y	Y	Y			Y	Y	Y
<a href="#">1819 페이지의 『BALTIMEOUT』</a>	TIMEOUT	Y	Y	Y			Y	Y	Y
<a href="#">1819 페이지의 『BROKERCCDURSUBQ』<sup>1</sup></a>	CCDSUB					Y			
<a href="#">1820 페이지의 『BROKERCCSUBQ』<sup>1</sup></a>	CCSUB	Y		Y			Y		Y
<a href="#">1820 페이지의 『BROKERCONQ』<sup>1</sup></a>	BCON	Y		Y			Y		Y
<a href="#">1821 페이지의 『BROKERDURSUBQ』<sup>1</sup></a>	BDSUB					Y			
<a href="#">1821 페이지의 『BROKERPUBQ』<sup>1</sup></a>	BPUB	Y		Y		Y	Y		Y
<a href="#">1821 페이지의 『BROKERPUBQMGR』<sup>1</sup></a>	BPQM					Y			
<a href="#">1822 페이지의 『BROKERQMGR』<sup>1</sup></a>	BQM	Y		Y			Y		Y
<a href="#">1822 페이지의 『BROKERSUBQ』<sup>1</sup></a>	BSUB	Y		Y			Y		Y

표 868. 특성 이름 및 적용 가능한 오브젝트 유형 (계속)

특성	축약형	오브젝트 유형								
		CF	QCF	TCF	Q	T	XACF	XAQCF	XATCF	
1823 페이지의 『BROKERVER』 <sup>1</sup>	BVER	Y <sup>2</sup>		Y <sup>2</sup>		Y	Y		Y	
1823 페이지의 『CCDTURL』 <sup>3</sup>	CCDT	Y	Y	Y			Y	Y	Y	
1824 페이지의 『CCSID』	CCS	Y	Y	Y	Y	Y	Y	Y	Y	
1825 페이지의 『CHANNEL』 <sup>3</sup>	CHAN	Y	Y	Y			Y	Y	Y	
1825 페이지의 『CLEANUP』 <sup>1</sup>	CL	Y		Y			Y		Y	
1826 페이지의 『CLEANUPINT』 <sup>1</sup>	CLINT	Y		Y			Y		Y	
1826 페이지의 『CONNECTIONNAMELIST』	CNLIST	Y	Y	Y						
1827 페이지의 『CLIENTRECONNECTOPTIONS』	CROPT	Y	Y	Y						
1827 페이지의 『CLIENTRECONNECTTIMEOUT』	CRT	Y	Y	Y						
1828 페이지의 『CLIENTID』	CID	Y <sup>2</sup>	Y	Y <sup>2</sup>			Y	Y	Y	
1828 페이지의 『CLONESUPP』	CLS	Y		Y			Y		Y	
1829 페이지의 『COMPHDR』	HC	Y		Y			Y		Y	
1829 페이지의 『COMPMSG』	MC	Y	Y	Y			Y	Y	Y	
1830 페이지의 『CONNOPT』	CNOPT	Y	Y	Y			Y	Y	Y	
1831 페이지의 『CONNTAG』	CNTAG	Y	Y	Y			Y	Y	Y	
1831 페이지의 『설명』	DESC	Y <sup>2</sup>	Y	Y <sup>2</sup>	Y	Y	Y	Y	Y	
1831 페이지의 『DIRECTAUTH』	DAUTH	Y <sup>2</sup>		Y <sup>2</sup>						
1832 페이지의 『ENCODING』	ENC				Y	Y				
1833 페이지의 『EXPIRY』	EXP				Y	Y				
1833 페이지의 『FAILIFQUIESCE』	FIQ	Y	Y	Y	Y	Y	Y	Y	Y	
1834 페이지의 『호스트 이름』	HOST	Y <sup>2</sup>	Y	Y <sup>2</sup>			Y	Y	Y	
1834 페이지의 『LOCALADDRESS』	LA	Y <sup>2</sup>	Y	Y <sup>2</sup>			Y	Y	Y	
1835 페이지의 『MAPNAMESTYLE』	MNST	Y	Y	Y			Y	Y	Y	
1836 페이지의 『MAXBUFFSIZE』	MBSZ	Y <sup>2</sup>		Y <sup>2</sup>						
1836 페이지의 『MDREAD』	MDR				Y	Y				
1837 페이지의 『MDWRITE』	MDW				Y	Y				
1837 페이지의 『MDMSGCTX』	MDCTX				Y	Y				
1838 페이지의 『MSGBATCHSZ』 <sup>1</sup>	MBS	Y	Y	Y			Y	Y	Y	
1838 페이지의 『MSGBODY』	MBODY				Y	Y				
1839 페이지의 『MSGRETENTION』	MRET	Y	Y				Y	Y		
1839 페이지의 『MSGSELECTION』 <sup>1</sup>	MSEL	Y		Y			Y		Y	



표 868. 특성 이름 및 적용 가능한 오브젝트 유형 (계속)									
특성	축약형	오브젝트 유형							
		CF	QCF	TCF	Q	T	XACF	XAQCF	XATCF
1840 페이지의 『MULTICAST』	MCAST	Y <sup>2</sup>		Y <sup>2</sup>		Y			
1840 페이지의 『OPTIMISTICPUBLICATION』 <sup>1</sup>	OPTPUB	Y		Y					
1841 페이지의 『OUTCOMENOTIFICATION』 <sup>1</sup>	NOTIFY	Y		Y					
1841 페이지의 『PERSISTENCE』	PER				Y	Y			
1842 페이지의 『POLLINGINT』 <sup>1</sup>	PINT	Y	Y	Y			Y	Y	Y
1842 페이지의 『포트』	포트	Y <sup>2</sup>	Y	Y <sup>2</sup>			Y	Y	Y
1843 페이지의 『PRIORITY』	PRI				Y	Y			
1843 페이지의 『PROCESSDURATION』 <sup>1</sup>	PROCDUR	Y		Y					
1844 페이지의 『PROVIDERVERSION』	PVER	Y	Y	Y			Y	Y	Y
1846 페이지의 『PROXYHOSTNAME』	PHOST	Y <sup>2</sup>		Y <sup>2</sup>					
1846 페이지의 『PROXYPORT』	PPORT	Y <sup>2</sup>		Y <sup>2</sup>					
1847 페이지의 『PUBACKINT』 <sup>1</sup>	PAI	Y		Y			Y		Y
1847 페이지의 『PUTASYNCALLOWED』	PAALD				Y	Y			
1848 페이지의 『QMANAGER』	QMGR	Y	Y	Y	Y		Y	Y	Y
1848 페이지의 『큐』	QU				Y				
1848 페이지의 『READAHEADALLOWED』	RAALD				Y	Y			
1849 페이지의 『READAHEADCLOSEPOLICY』	RACP				Y	Y			
1850 페이지의 『RECEIVECCSID』	RCCS				Y	Y			
1850 페이지의 『RECEIVECONVERSION』	RCNV				Y	Y			
1851 페이지의 『RECEIVEISOLATION』 <sup>1</sup>	RCVISOL	Y		Y					
1851 페이지의 『RECEXIT』	RCX	Y	Y	Y			Y	Y	Y
1851 페이지의 『RECEXITINIT』	RCXI	Y	Y	Y			Y	Y	Y
1852 페이지의 『REPLYTOSTYLE』	RTOST				Y	Y			
1853 페이지의 『RESCANINT』 <sup>1</sup>	RINT	Y	Y				Y	Y	
1853 페이지의 『SECEXIT』	SCX	Y	Y	Y			Y	Y	Y
1854 페이지의 『SECEXITINIT』	SCXI	Y	Y	Y			Y	Y	Y
1854 페이지의 『SENDCHECKCOUNT』	SCC	Y	Y	Y			Y	Y	Y

표 868. 특성 이름 및 적용 가능한 오브젝트 유형 (계속)

특성	축약형	오브젝트 유형								
		CF	QCF	TCF	Q	T	XACF	XAQCF	XATCF	
1854 페이지의 『SENDEXIT』	SDX	Y	Y	Y			Y	Y	Y	
1855 페이지의 『SENDEXITINIT』	SDXI	Y	Y	Y			Y	Y	Y	
1855 페이지의 『SHARECONVALLOWED』	SCALD	Y	Y	Y			Y	Y	Y	
1856 페이지의 『SPARSESUBS』 <sup>1</sup>	SSUBS	Y		Y						
1856 페이지의 『SSLCIPHERSUITE』	SCPHS	Y	Y	Y			Y	Y	Y	
1857 페이지의 『SSLCRL』	SCRL	Y	Y	Y			Y	Y	Y	
1857 페이지의 『SSLFIPSREQUIRED』	SFIPS	Y	Y	Y			Y	Y	Y	
1858 페이지의 『SSLPEERNAME』	SPEER	Y	Y	Y			Y	Y	Y	
1858 페이지의 『SSLRESETCOUNT』	SRC	Y	Y	Y			Y	Y	Y	
1859 페이지의 『STATREFRESHINT』 <sup>1</sup>	SRI	Y		Y			Y		Y	
1859 페이지의 『SUBSTORE』 <sup>1</sup>	SS	Y		Y			Y		Y	
1860 페이지의 『SYNCPOINTALLGETS』	SPAG	Y	Y	Y			Y	Y	Y	
1860 페이지의 『TARGCLIENT』	TC				Y	Y				
1861 페이지의 『TARGCLIENTMATCHING』	TCM	Y	Y				Y	Y		
1861 페이지의 『TEMPMODEL』	TM	Y	Y				Y	Y		
1862 페이지의 『TEMPQPREFIX』	TQP	Y	Y				Y	Y		
1862 페이지의 『TEMPTOPICPREFIX』	TTP	Y		Y			Y		Y	
1862 페이지의 『TOPIC』	TOP					Y				
1863 페이지의 『TRANSPORT』	TRAN	Y <sup>2</sup>	Y	Y <sup>2</sup>			Y	Y	Y	
1864 페이지의 『WILDCARDFORMAT』	WCFMT	Y		Y			Y		Y	

**참고:**

- 이 특성은 IBM MQ classes for JMS의 버전 70에 사용할 수 있지만, 연결 팩토리의 PROVIDERVERSION 특성이 7 미만의 버전 번호로 설정된 경우를 제외하고 IBM WebSphere MQ 7.0 큐 관리자에 연결된 애플리케이션에 대해서는 아무런 영향을 미치지 않습니다.
- 브로커에 대한 실시간 연결을 사용하는 경우 BROKERVER, CLIENTID, DESCRIPTION, DIRECTAUTH, HOSTNAME, LOCALADDRESS, MAXBUFFSIZE, MULTICAST, PORT, PROXYHOSTNAME, PROXYPORT 및 TRANSPORT 특성만 ConnectionFactory 또는 TopicConnectionFactory 오브젝트에 지원됩니다.
- 오브젝트의 CCDURL 및 CHANNEL 특성은 둘 다 동시에 설정하지 않아야 합니다.

**IBM MQ classes for JMS 오브젝트 특성의 종속성**

일부 특성의 검증이 다른 특성의 특정 값에 좌우됩니다.

이 종속성은 다음 특성 그룹에서 발생할 수 있습니다.

- 클라이언트 특성
- 브로커에 대한 실시간 연결 특성
- 엑시트 초기화 문자열

#### **클라이언트 특성**

큐 관리자에 대한 연결에서 다음 특성은 TRANSPORT의 값이 CLIENT인 경우에만 관련이 있습니다.

- 호스트 이름
- 포트
- CHANNEL
- LOCALADDRESS
- CCDTURL
- CCSID
- COMPHDR
- COMPMSG
- REEXIT
- REEXITINIT
- SEEXIT
- SEEXITINIT
- SENDEXIT
- SENDEXITINIT
- SHARECONVALLOWED
- SSLCIPHERSUITE
- SSLCRL
- SSLFIPSREQUIRED
- SSLPEERNAME
- SSLRESETCOUNT
- APPLICATIONNAME

TRANSPORT의 값이 BIND이면 관리 도구를 사용하여 이 특성의 값을 설정할 수 없습니다.

TRANSPORT의 값이 CLIENT이면 BROKERVER 특성의 기본값이 V1이고 PORT 특성의 기본값은 1414입니다. BROKERVER 또는 PORT의 값을 명시적으로 설정하는 경우, 나중에 TRANSPORT의 값으로 변경해도 선택사항이 대체되지 않습니다.

#### **브로커에 대한 실시간 연결 특성**

TRANSPORT의 값이 DIRECT 또는 DIRECTHTTP이면 다음 특성만 관련이 있습니다.

- BROKERVER
- CLIENTID
- 설명
- DIRECTAUTH
- 호스트 이름
- LOCALADDRESS
- MAXBUFFSIZE
- MULTICAST(DIRECT에만 지원됨)
- 포트

- PROXYHOSTNAME(DIRECT에만 지원됨)
- PROXYPORT(DIRECT에만 지원됨)

TRANSPORT의 값이 DIRECT 또는 DIRECTHTTP이면 BROKERVER 특성의 기본값이 V2이고 PORT 특성의 기본값은 1506입니다. BROKERVER 또는 PORT의 값을 명시적으로 설정하는 경우, 나중에 TRANSPORT의 값으로 변경해도 선택사항이 대체되지 않습니다.

#### 엑시트 초기화 문자열

해당 엑시트 이름을 제공하지 않은 채 엑시트 초기화 문자열을 설정하지 마십시오. 엑시트 초기화 특성은 다음과 같습니다.

- REEXITINIT
- SEEXITINIT
- SENDEXITINIT

예를 들어, REEXIT(some.exit.classname)를 지정하지 않은 채 REEXITINIT(myString)를 지정하면 오류가 발생합니다.

#### 관련 참조

1863 페이지의 『TRANSPORT』

큐 관리자 또는 브로커에 대한 연결의 네이처입니다.

## APPLICATIONNAME

애플리케이션은 큐 관리자에 대한 연결을 식별하는 이름을 설정할 수 있습니다. 이 애플리케이션 이름은 **DISPLAY CONN MQSC/PCF** 명령(여기서 필드는 **APPLTAG**라고 함)을 사용하여 표시하거나 IBM MQ 탐색기 애플리케이션 연결 표시(여기서 필드는 **App name**이라고 함)에 표시됩니다.

#### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: APPLICATIONNAME

JMS 관리 도구 짧은 이름: APPNAME

#### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setAppName()
- MQConnectionFactory.getAppname()

#### 값

유효한 문자열은 28자를 넘지 않습니다. 이름이 이보다 길면 필요에 따라 패키지 이름의 선두 문자를 제거하여 기준에 맞도록 조정합니다. 예를 들어 호출 클래스가 com.example.MainApp이면 전체 이름이 사용되지만 호출 클래스가 com.example.dictionaryAndThesaurus.multilingual.mainApp이면 multilingual.mainApp 이름이 사용됩니다. 사용 가능한 길이에 맞도록 조정된 맨 오른쪽 패키지 이름과 클래스 이름의 가장 긴 조합이기 때문입니다.

클래스 이름 자체가 28자를 초과하면 길이에 맞게 잘립니다. 예를 들어 com.example.mainApplicationForSecondTestCase는 mainApplicationForSecondTest가 됩니다.

 z/OS에서 APPNAME은 다음과 같습니다.

- 바인딩 모드는 설정된 경우 무시되며, 설정하는 경우 공백으로만 설정할 수 있습니다.
- 클라이언트 모드를 설정하고 사용할 수 있습니다.

## ASYNCEXCEPTION

이 특성은 연결이 끊어진 경우에만 IBM MQ classes for JMS 가 ExceptionListener 에 알리는지 또는 JMS API 호출에 비동기식으로 예외가 발생한 경우에만 알리는지 여부를 판별합니다. 이 특성은 등록된 ExceptionListener가 있고 이 ConnectionFactory에서 작성된 모든 연결에 적용됩니다.

### 적용 가능 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: ASYNCEXCEPTION

JMS 관리 도구 짧은 이름: AEX

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setAsyncExceptions()
- MQConnectionFactory.getAsyncExceptions()

### 값

#### ASYNCEXCEPTIONS\_ALL

동기 API 호출의 범위 밖에서 비동기적으로 발견된 예외와, 연결이 끊긴 모든 예외가 ExceptionListener로 전송됩니다.

표 869. 모두 비동기 예외: 환경 및 관련 상수 이름	
환경	값
JMS 관리 도구	모두
프로그램 방식	WMQCONSTANTS.ASYNCEXCEPTIONS_ALL = -1
IBM MQ Explorer	모두

#### ASYNCEXCEPTIONS\_CONNECTIONBROKEN

끊긴 연결을 나타내는 예외만 ExceptionListener로 전송됩니다. 비동기 처리 중에 발생한 다른 예외는 ExceptionListener에 보고되지 않으므로 애플리케이션에 이 예외 정보가 제공되지 않습니다. 이 값은 IBM MQ 8.0.0 Fix Pack 2의 기본값입니다. JMS: IBM MQ 8.0의 예외 리스너 변경사항을 참조하십시오.

표 870. 끊어진 연결을 표시하는 예외: 환경 및 관련 상수 이름	
환경	값
JMS 관리 도구	CONNECTIONBROKEN
프로그램 방식	WMQCONSTANTS.ASYNCEXCEPTIONS_CONNECTIONBROKEN = 1
IBM MQ Explorer	중단된 연결

다음 추가 상수가 정의됩니다.

- IBM MQ 8.0.0 Fix Pack 2에서: WMQCONSTANTS.ASYNCEXCEPTIONS\_DEFAULT = ASYNCEXCEPTIONS\_CONNECTIONBROKEN
- IBM MQ 8.0.0 Fix Pack 2 이전: WMQCONSTANTS.ASYNCEXCEPTIONS\_DEFAULT = ASYNCEXCEPTIONS\_ALL

### 관련 개념

IBM MQ classes for JMS의 예외

## V 9.4.0 기본 옵션

클라이언트 전송을 사용하는 IBM MQ classes for JMS 애플리케이션이 균등 클러스터에서 재조정되는 방법을 제어합니다.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory.

JMS 관리 도구 긴 이름: **BALOPTIONS**

JMS 관리 도구 짧은 이름: **OPTIONS**

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setBalancingOptions()
- MQConnectionFactory.getBalancingOptions()

### 값

#### 아이GNNONE

트랜잭션의 정상 처리가 적용되고 애플리케이션은 트랜잭션 중에 이동하도록 요청되지 않습니다.

이 값은 IBM MQ *BalancingOption* MQBNO\_OPTIONS\_NONE에 맵핑됩니다.

#### 전송

트랜잭션 중에 애플리케이션을 이동하도록 요청할 수 있습니다.

이 값은 IBM MQ *BalancingOption* MQBNO\_OPTIONS\_IGNORE\_TRANS에 맵핑됩니다.

## V 9.4.0 BALTYPE

클라이언트 전송을 사용하는 IBM MQ classes for JMS 애플리케이션이 균등 클러스터에서 재조정될 수 있는 방법을 제어합니다.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory.

JMS 관리 도구 긴 이름: **BALTYPE**

JMS 관리 도구 짧은 이름: **TYPE**

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setBalancingApplicationType()
- MQConnectionFactory.getBalancingApplicationType()

### 값

#### SIMPLE

균등 클러스터에서 애플리케이션의 기본 처리가 적용됩니다.

이 값은 IBM MQ *BalancingOption* MQBNO\_BALTYPE\_SIMPLE에 맵핑됩니다.

## 요청 응답

제한시간 기간이 경과하지 않으면 **MQPUT** 가 **MQGET**에 의해 밸런싱되지 않은 경우 애플리케이션은 다시 연결하도록 요청되지 않습니다.

이 값은 IBM MQ *BalancingOption* **MQBNO\_BALTYPE\_REQREP** 에 맵핑됩니다.

## V 9.4.0 **BALTIMEOUT**

클라이언트 전송을 사용하는 IBM MQ classes for JMS 애플리케이션이 균등 클러스터에서 재조정되는 방법을 제어합니다.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory.

JMS 관리 도구 긴 이름: **BALTIMEOUT**

JMS 관리 도구 짧은 이름: **TIMEOUT**

### 프로그래밍 방식 액세스

Setter/Getter

- `MQConnectionFactory.setBalancingTimeout()`
- `MQConnectionFactory.getBalancingTimeout()`

### 값

#### NEVER

균등 클러스터에서 재밸런싱을 위해 애플리케이션이 제한시간 초과되지 않습니다.

이 값은 IBM MQ *BalancingOption* **MQBNO\_TIMEOUT\_NEVER**에 맵핑됩니다.

#### IMMEDIATE

애플리케이션은 균등 클러스터에서 재조정을 위해 즉시 제한시간이 초과됩니다.

이 값은 IBM MQ *BalancingOption* **MQBNO\_TIMEOUT\_IMMEDIATE**에 맵핑됩니다.

#### DEFAULT

애플리케이션은 기본 기간인 10초후에 균등 클러스터에서 재밸런싱을 위해 제한시간이 초과됩니다.

이 값은 IBM MQ *BalancingOption* **MQBNO\_TIMEOUT\_AS\_DEFAULT**에 맵핑됩니다.

#### nn

nn 초 후에 균등 클러스터에서 재밸런싱을 위해 애플리케이션이 제한시간 초과되었습니다.

nn 은 1-9999999999사이일 수 있습니다.

## **BROKERCCDURSUBQ**

ConnectionConsumer에 대해 지속 가능 구독 메시지가 검색되는 큐의 이름입니다.

### 적용 가능 오브젝트

토픽

JMS 관리 도구 긴 이름: **BROKERCCDURSUBQ**

JMS 관리 도구 짧은 이름: **CCDSUB**

### 프로그래밍 방식 액세스

Setter/Getter

- MQTopic.setBrokerCCDurSubQueue()
- MQTopic.getBrokerCCDurSubQueue()

## 값

### **SYSTEM.JMS.D.CC.SUBSCRIBER.QUEUE**

이는 기본값입니다.

유효한 문자열

## **BROKERCCSUBQ**

ConnectionConsumer에 대해 지속 불가능한 구독 메시지가 검색되는 큐의 이름입니다.

### 적용 가능 오브젝트

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: BROKERCCSUBQ

JMS 관리 도구 짧은 이름: CCSUB

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setBrokerCCSubQueue()
- MQConnectionFactory.getBrokerCCSubQueue()

## 값

### **SYSTEM.JMS.ND.CC.SUBSCRIBER.QUEUE**

이는 기본값입니다.

유효한 문자열

## **BROKERCONQ**

브로커의 제어 큐 이름입니다.

### 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: BROKERCONQ

JMS 관리 도구 짧은 이름: BCON

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setBrokerControlQueue()
- MQConnectionFactory.getBrokerControlQueue()

## 값

### **SYSTEM.BROKER.CONTROL.QUEUE**

이는 기본값입니다.

유효한 문자열



## BROKERDURSUBQ

IBM MQ classes for JMS가 IBM MQ 메시징 제공자 마이그레이션 모드에서 사용 중이면 이 특성은 지속 가능 구독 메시지가 검색되는 큐의 이름을 지정합니다.

### 적용 가능 오브젝트

토픽

JMS 관리 도구 긴 이름: BROKERDURSUBQ

JMS 관리 도구 짧은 이름: BDSUB

### 프로그래밍 방식 액세스

Setter/Getter

- MQTopic.setBrokerDurSubQueue()
- MQTopic.getBrokerDurSubQueue()

### 값

**SYSTEM.JMS.D.SUBSCRIBER.QUEUE**

이는 기본값입니다.

유효한 문자열

SYSTEM.JMS.D로 시작

관련 태스크

JMS **PROVIDERVERSION** 특성 구성

## BROKERPUBQ

발행된 메시지가 송신되는 큐(스트림 큐)의 이름입니다.

### 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory, Topic, XAConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: BROKERPUBQ

JMS 관리 도구 짧은 이름: BPUB

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setBrokerPubQueue
- MQConnectionFactory.getBrokerPubQueue

### 값

**SYSTEM.BROKER.DEFAULT.STREAM**

이는 기본값입니다.

유효한 문자열

## BROKERPUBQMGR

토픽에서 발행된 메시지가 송신되는 큐를 소유하는 큐 관리자의 이름입니다.

## 적용 가능한 오브젝트

토픽

JMS 관리 도구 긴 이름: BROKERPUBQMGR

JMS 관리 도구 짧은 이름: BPQM

## 프로그래밍 방식 액세스

Setter/Getter

- MQTopic.setBrokerPubQueueManager()
- MQTopic.getBrokerPubQueueManager()

**값**

**널**

이는 기본값입니다.

유효한 문자열

## BROKERQMGR

브로커를 실행 중인 큐 관리자의 이름입니다.

## 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: BROKERQMGR

JMS 관리 도구 짧은 이름: BQM

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setBrokerQueueManager()
- MQConnectionFactory.getBrokerQueueManager()

**값**

**널**

이는 기본값입니다.

유효한 문자열

## BROKERSUBQ

IBM MQ classes for JMS가 IBM MQ 메시징 제공자 마이그레이션 모드에서 사용 중이면 이 특성은 지속 불가능한 구독 메시지가 검색되는 큐의 이름을 지정합니다.

## 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: BROKERSUBQ

JMS 관리 도구 짧은 이름: BSUB

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setBrokerSubQueue()
- MQConnectionFactory.getBrokerSubQueue()

## 값

### SYSTEM.JMS.ND.SUBSCRIBER.QUEUE

이는 기본값입니다.

유효한 문자열

SYSTEM.JMS.ND로 시작

관련 태스크

JMS PROVIDERVERSION 특성 구성

## BROKERVER

사용 중인 브로커의 버전입니다.

## 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory, Topic, XAConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: BROKERVER

JMS 관리 도구 짧은 이름: BVER

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setBrokerVersion()
- MQConnectionFactory.getBrokerVersion()

## 값

### V1

IBM MQ 발행/구독 브로커를 사용하거나, IBM MQ Integrator, WebSphere Event Broker, WebSphere Business Integration Event Broker 또는 WebSphere Business Integration Message Broker의 브로커를 호환 모드에서 사용합니다. TRANSPORT를 BIND 또는 CLIENT로 설정한 경우 이 값이 기본값입니다.

### V2

IBM MQ Integrator, WebSphere Event Broker, WebSphere Business Integration Event Broker 또는 WebSphere Business Integration Message Broker의 브로커를 고유 모드에서 사용합니다. TRANSPORT를 DIRECT 또는 DIRECTHTTP로 설정한 경우 기본값입니다.

지정되지 않음

브로커를 V6에서 V7로 마이그레이션한 후 RFH2 헤더가 더 이상 사용되지 않도록 이 특성을 설정하십시오. 마이그레이션하고 나면 이 특성은 더 이상 적절하지 않습니다.

## CCDTURL

클라이언트 채널 정의 테이블을 포함하여 파일의 이름과 위치를 식별하고 이 파일의 액세스 방법을 지정하는 URL(Uniform Resource Locator)입니다.

## 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: CCDTURL

JMS 관리 도구 짧은 이름: CCDT

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setCCDTURL()
- MQConnectionFactory.getCCDTURL()

### 값

#### 널

이는 기본값입니다.

### URL(Uniform Resource Locator)

## CCSID

연결 팩토리의 경우, 이 특성은 큐 관리자와 함께 내부 데이터 플로우에 사용할 코드화된 문자 세트 ID (CCSID) 를 지정합니다. 대상의 경우 특성은 해당 대상에 넣은 MapMessages, StreamMessages 및 TextMessages 에서 문자열 데이터를 인코딩하는 데 사용할 CCSID를 정의합니다.

**참고:** 일반적으로 연결 팩토리에 대해 이 특성을 변경할 필요가 없습니다.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, Queue, Topic, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: CCSID

JMS 관리 도구 짧은 이름: CCS

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setCCSID()
- MQConnectionFactory.getCCSID()

### 값

#### 819

연결 팩토리의 기본값입니다.

#### 1208

대상의 기본값입니다.

#### 양수

#### 관련 개념

[JMS 메시지 변환](#)

## **V 9.4.0** **V 9.4.0** CERTVALPO

TLS 연결에 사용할 인증서 유효성 검증 정책입니다.

### 적용 가능한 오브젝트

ConnectionFactory

JMS 관리 도구 긴 이름: CERTVALPO

JMS 관리 도구 짧은 이름: CVAP

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setCertificateValPolicy()
- MQConnectionFactory.getCertificateValPolicy()

### 값

#### 0/ANY

연결 팩토리의 기본값입니다.

#### 2/없음

이 정책은 인증서 유효성 검증을 포함하지 않습니다. 클라이언트 애플리케이션에서만 설정할 수 있습니다.

## CHANNEL

사용되는 클라이언트 연결 채널의 이름입니다.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: CHANNEL

JMS 관리 도구 짧은 이름: CHAN

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setChannel()
- MQConnectionFactory.getChannel()

### 값

#### SYSTEM.DEF.SVRCONN

이는 기본값입니다.

유효한 문자열

## CLEANUP

BROKER 또는 MIGRATE 구독 저장소의 정리 레벨입니다.

### 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: CLEANUP

JMS 관리 도구 짧은 이름: CL

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setCleanupLevel()
- MQConnectionFactory.getCleanupLevel()

## 값

### SAFE

안전한 정리를 사용합니다. 이는 기본값입니다.

### ASPROP

Java 명령행에 설정된 특성에 따라 안전하고 강력한 정리를 사용하거나 정리를 사용하지 않습니다.

### NONE

정리를 사용하지 않습니다.

### 강함

강력한 정리를 사용합니다.

## CLEANUPINT

발행/구독 정리 유틸리티의 백그라운드 실행 간격(밀리초)입니다.

### 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: CLEANUPINT

JMS 관리 도구 짧은 이름: CLINT

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setCleanupInterval()
- MQConnectionFactory.getCleanupInterval()

## 값

### 3600000

이는 기본값입니다.

### 양수

## CONNECTIONNAMELIST

TCP/IP 연결 이름의 목록입니다. 다시 연결을 재시도할 때마다 한 번씩 목록을 순서대로 시도합니다.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory

JMS 관리 도구 긴 이름: CONNECTIONNAMELIST

JMS 관리 도구 짧은 이름: CNLIST

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setconnectionNameList()
- MQConnectionFactory.getconnectionNameList()

## 값

HOSTNAME(PORT)의 쉼표로 구분된 목록. HOSTNAME은 DNS 이름 또는 IP 주소일 수 있습니다.

포트 기본값은 1414입니다.

## CLIENTRECONNECTOPTIONS

다시 연결 통제 옵션입니다.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory

JMS 관리 도구 긴 이름: CLIENTRECONNECTOPTIONS

JMS 관리 도구 짧은 이름: CROPT

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setClientReconnectOptions()
- MQConnectionFactory.getClientReconnectOptions()

### 값

#### QMGR

애플리케이션을 원래 연결된 동일한 큐 관리자에 다시 연결할 수 있습니다.

연결 이름 목록에 지정된 대로 애플리케이션이 연결하려고 하는 큐 관리자에 원래 연결되는 큐 관리자와 다른 QMID가 있는 경우 이유 코드가 MQRC\_RECONNECT\_QMID\_MISMATCH인 오류가 리턴됩니다.

애플리케이션이 다시 연결될 수 있지만 처음 연결을 설정한 큐 관리자와 IBM MQ classes for JMS 애플리케이션 사이에 연관관계가 있는 경우 이 값을 사용하십시오.

애플리케이션이 고가용성 큐 관리자의 대기 인스턴스에 자동으로 다시 연결되게 하려면 이 값을 선택하십시오.

이 값을 프로그래밍 방식으로 사용하려면 상수 WMQConstants.WMQ\_CLIENT\_RECONNECT\_Q\_MGR을 사용하십시오.

#### ANY

애플리케이션은 연결 이름 목록에 지정된 큐 관리자에 다시 연결할 수 있습니다.

JMS 애플리케이션의 IBM MQ 클래스와 처음 연결을 설정한 큐 관리자 사이에 연관관계가 없는 경우에만 다시 연결 옵션을 사용하십시오.

프로그램에서 이 값을 사용하려면 상수 WMQConstants.WMQ\_CLIENT\_RECONNECT를 사용하십시오.

#### DISABLED

애플리케이션이 다시 연결되지 않습니다.

이 값을 프로그래밍 방식으로 사용하려면 상수 WMQConstants.WMQ\_CLIENT\_RECONNECT\_DISABLED를 사용하십시오.

#### ASDEF

애플리케이션이 자동으로 다시 연결되는지 여부는 IBM MQ 채널 속성 DefReconnect의 값에 따라 다릅니다.

이는 기본값입니다.

프로그램에서 이 값을 사용하려면 상수 WMQConstants.WMQ\_CLIENT\_RECONNECT\_AS\_DEF를 사용하십시오.

## CLIENTRECONNECTTIMEOUT

다시 연결 재시도를 멈출 때까지 시간입니다.

## 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory

JMS 관리 도구 긴 이름: CLIENTRECONNECTTIMEOUT

JMS 관리 도구 짧은 이름: CRT

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setClientReconnectTimeout()
- MQConnectionFactory.setClientReconnectTimeout()

## 값

간격(초). 기본값은 1800(30분)입니다.

## CLIENTID

클라이언트 ID는 지속 가능 구독에 대한 애플리케이션 연결을 고유하게 식별하는 데 사용됩니다.

## 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: CLIENTID

JMS 관리 도구 짧은 이름: CID

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setClientId()
- MQConnectionFactory.getClientId()

## 값

널

이는 기본값입니다.

유효한 문자열

## CLONESUPP

동일한 지속 가능 토픽 구독자의 인스턴스를 동시에 두 개 이상 실행할 수 있는지 여부입니다.

## 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: CLONESUPP

JMS 관리 도구 짧은 이름: CLS

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setCloneSupport()



- MQConnectionFactory.getCloneSupport()

## 값

### DISABLED

한 번에 하나의 지속 가능 토픽 구독자 인스턴스만 실행할 수 있습니다. 이는 기본값입니다.

### ENABLED

동일한 지속 가능 토픽 구독자의 두 개 이상 인스턴스를 동시에 실행할 수 있지만, 각 인스턴스가 별도의 Java 가상 머신(JVM)에서 실행되어야 합니다.

## COMPHDR

연결에서 헤더 데이터를 압축하는 데 사용할 수 있는 기술 목록입니다.

### 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: COMPHDR

JMS 관리 도구 짧은 이름: HC

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setHdrCompList()
- MQConnectionFactory.getHdrCompList()

## 값

### NONE

이는 기본값입니다.

### SYSTEM

RLE 메시지 헤더 압축이 수행됩니다.

## COMPMSG

연결에서 메시지 데이터를 압축하는 데 사용할 수 있는 기술 목록입니다.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: COMPMSG

JMS 관리 도구 짧은 이름: MC

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setMsgCompList()
- MQConnectionFactory.getMsgCompList()

## 값

### NONE

이는 기본값입니다.

하나 이상의 다음 값이 공백 문자로 분리된 목록입니다.

RLE ZLIBFAST ZLIBHIGH **V 9.4.0** LZ4FAST LZ4HIGH

## CONNOPT

바인딩 전송을 사용하는 IBM MQ classes for JMS 애플리케이션이 큐 관리자에 연결하는 방식을 제어합니다.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory.

JMS 관리 도구 긴 이름: CONNOPT

JMS 관리 도구 짧은 이름: CNOPT

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setMQConnectionOptions()
- MQConnectionFactory.getMQConnectionOptions()

### 값

#### STANDARD

애플리케이션 및 큐 관리자 사이의 바인딩 네이처는 큐 관리자의 *DefaultBindType* 속성 값에 따라 다릅니다. STANDARD 값은 IBM MQ *ConnectOption* MQCNO\_STANDARD\_BINDING에 맵핑됩니다.

#### SHARED

애플리케이션 및 로컬 큐 관리자 에이전트가 별도의 실행 단위에 실행되지만, 일부 자원을 공유합니다. 이 값은 IBM MQ *ConnectOption* MQCNO\_SHARED\_BINDING에 맵핑됩니다.

#### ISOLATED

애플리케이션 및 로컬 큐 관리자 에이전트가 별도의 실행 단위에 실행되고 자원을 공유하지 않습니다. ISOLATED 값은 IBM MQ *ConnectOption* MQCNO\_ISOLATED\_BINDING에 맵핑됩니다.

#### FASTPATH

애플리케이션 및 로컬 큐 관리자 에이전트는 동일한 실행 단위에서 실행됩니다. 이 값은 IBM MQ *ConnectOption* MQCNO\_FASTPATH\_BINDING에 맵핑됩니다.

#### SERIALQM

애플리케이션이 큐 관리자 범위 내에서 연결 태그의 독점 사용을 요청합니다. 이 값은 IBM MQ *ConnectOption* MQCNO\_SERIALIZE\_CONN\_TAG\_Q\_MGR에 맵핑됩니다.

#### SERIALQSG

애플리케이션이 큐 관리자가 속한 큐 공유 그룹 범위 내에서 연결 태그의 독점 사용을 요청합니다. SERIALQSG 값이 IBM MQ *ConnectOption* MQCNO\_SERIALIZE\_CONN\_TAG\_QSG에 맵핑됩니다.

#### RESTRICTQM

애플리케이션이 연결 태그의 공유 사용을 요청하지만, 큐 관리자 범위 내에서 연결 태그의 공유 사용에는 제한이 있습니다. 이 값은 IBM MQ *ConnectOption* MQCNO\_RESTRICT\_CONN\_TAG\_Q\_MGR에 맵핑됩니다.

#### RESTRICTQSG

애플리케이션이 연결 태그의 공유 사용을 요청하지만, 큐 관리자가 속한 큐 공유 그룹 범위 내에서 연결 태그의 공유 사용에는 제한이 있습니다. 이 값은 IBM MQ *ConnectOption* MQCNO\_RESTRICT\_CONN\_TAG\_QSG에 맵핑됩니다.

IBM MQ 연결 옵션에 대한 자세한 정보는 [MQCONNX 호출을 사용하여 큐 관리자에 연결의 내용을 참조하십시오](#).

## CONNTAG

애플리케이션이 큐 관리자에 연결된 동안 큐 관리자가 작업 단위 내에서 애플리케이션이 업데이트한 자원과 연 관시키는 태그입니다.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: CONNTAG

JMS 관리 도구 짧은 이름: CNTAG

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setConnTag()
- MQConnectionFactory.getConnTag()

### 값

**128개** 요소의 바이트 배열이며, 여기서 각 요소는 **0**입니다.  
이는 기본값입니다.

입력의 문자열

128바이트보다 긴 경우 값이 잘립니다.

## 설명

저장 오브젝트에 대한 설명입니다.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, Queue, Topic, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: DESCRIPTION

JMS 관리 도구 짧은 이름: DESC

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setDescription()
- MQConnectionFactory.getDescription()

### 값

널

이는 기본값입니다.

유효한 문자열

## DIRECTAUTH

TLS 인증이 브로커에 대한 실시간 연결에 사용되는지 여부입니다.

## 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory

JMS 관리 도구 긴 이름: DIRECTAUTH

JMS 관리 도구 짧은 이름: DAUTH

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setDirectAuth()
- MQConnectionFactory.getDirectAuth()

## 값

### BASIC

인증, 사용자 이름 인증 또는 비밀번호 인증이 없습니다. 이는 기본값입니다.

### CERTIFICATE

공개 키 인증서 인증.

## ENCODING

메시지를 이 목적지로 송신할 때 메시지 본문의 숫자 데이터를 표시하는 방법입니다. 이 특성은 2진 정수, 팩형 10진수 정수 및 부동 소수점 숫자의 표시를 지정합니다.

## 적용 가능한 오브젝트

큐, 토픽

JMS 관리 도구 긴 이름: ENCODING

JMS 관리 도구 짧은 이름: ENC

## 프로그래밍 방식 액세스

Setter/Getter

- MQDestination.setEncoding()
- MQDestination.getEncoding()

## 값

### ENCODING property

ENCODING 특성에 사용하는 올바른 값은 3개의 하위 특성으로 구성됩니다.

#### 정수 인코딩

정상 또는 역방향입니다.

#### 10진수 인코딩

정상 또는 역방향입니다.

#### 부동 소수점 인코딩

IEEE 일반, IEEE 예약 또는 z/OS

ENCODING 특성은 다음 구문을 사용하여 3자 문자열로 표시됩니다.

```
{N|R}{N|R}{N|R|3}
```

이 문자열에서:

- N은 정상을 나타냅니다.
- R은 역방향을 나타냅니다.
- 3은 z/OS를 나타냅니다.
- 첫 번째 문자는 정수 인코딩을 나타냅니다.
- 두 번째 문자는 10진수 인코딩을 나타냅니다.
- 세 번째 문자는 부동 소수점 인코딩을 나타냅니다.

ENCODING 특성에 대해 12개의 가능한 값 세트를 제공합니다.

추가 값인 문자열 NATIVE도 있으며, 이 값은 Java 플랫폼에 적절한 인코딩 값을 설정합니다.

다음 예는 올바른 ENCODING 조합을 보여줍니다.

```
ENCODING(NNR)
ENCODING(NATIVE)
ENCODING(RR3)
```

## EXPIRY

목적지의 메시지가 만기되는 시간입니다.

### 적용 가능한 오브젝트

큐, 토픽

JMS 관리 도구 긴 이름: EXPIRY

JMS 관리 도구 짧은 이름: EXP

### 프로그래밍 방식 액세스

Setter/Getter

- MQDestination.setExpiry()
- MQDestination.getExpiry()

### 값

#### APP

만기를 JMS 애플리케이션에서 정의할 수 있습니다. 이는 기본값입니다.

#### UNLIM

만기가 발생하지 않습니다.

#### 0

만기가 발생하지 않습니다.

밀리초 단위로 만기를 나타내는 양의 정수.

## FAILIFQUIESCE

이 특성은 큐 관리자가 정지 상태에 있거나 애플리케이션이 CLIENT 전송을 사용하여 큐 관리자에 연결 중이고 애플리케이션이 사용 중인 채널이 정지 상태에 있는 경우 (예: **STOP CHANNEL** 또는 **STOP CHANNEL MODE(QUIESCE)**) MQSC 명령을 사용하여 특정 메소드에 대한 호출이 실패하는지 여부를 판별합니다.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, Queue, Topic, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: FAILIFQUIESCE

JMS 관리 도구 짧은 이름: FIQ

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setFailIfQuiesce()
- MQConnectionFactory.getFailIfQuiesce()

### 값

#### YES

큐 관리자가 정지 중 상태거나 큐 관리자에 연결하는 데 사용 중인 채널이 정지 중인 경우 특정 메소드에 대한 호출이 실패합니다. 애플리케이션이 이러한 연결 중 하나를 감지하면 애플리케이션은 즉각적인 태스크를 완료하고 연결을 닫아서 큐 관리자 또는 채널 인스턴스를 중지할 수 있도록 합니다. 이는 기본값입니다.

#### NO

큐 관리자 또는 큐 관리자에 연결하는 데 사용 중인 채널이 정지 중 상태이므로 메소드 호출이 실패하지 않습니다. 이 값을 지정하면 애플리케이션은 큐 관리자 또는 채널이 정지 중임을 발견할 수 없습니다. 애플리케이션이 큐 관리자에 대한 조작을 계속 수행할 수 있으므로 큐 관리자가 중지되지 않습니다.

## 호스트 이름

큐 관리자에 대한 연결에서는 큐 관리자가 실행되는 시스템의 호스트 이름 또는 IP 주소이고, 브로커에 대한 실시간 연결에서는 브로커가 실행되는 시스템의 호스트 이름 또는 IP 주소입니다.

## 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: HOSTNAME

JMS 관리 도구 짧은 이름: HOST

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setHostName()
- MQConnectionFactory.getHostName()

### 값

#### localhost

이는 기본값입니다.

유효한 문자열

## LOCALADDRESS

큐 관리자에 대한 연결에서 이 특성은 사용할 로컬 네트워크 인터페이스 또는 사용할 로컬 포트나 로컬 포트 범위를 지정합니다.

## 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: LOCALADDRESS

JMS 관리 도구 짧은 이름: LA

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setLocalAddress()
- MQConnectionFactory.getLocalAddress()

## 값

""(빈 문자열)

이는 기본값입니다.

**[ip-addr][(low-port[,high-port])] 형식의 문자열**

다음은 몇 가지 예입니다.

192.0.2.0

채널이 주소 192.0.2.0에 로컬로 바인드합니다.

192.0.2.0(1000)

채널이 주소 192.0.2.0에 로컬로 바인드하고 포트 1000을 사용합니다.

192.0.2.0(1000,2000)

채널이 주소 192.0.2.0에 로컬로 바인드하고 1000 - 2000 범위의 포트를 사용합니다.

(1000)

채널이 포트 1000에 로컬로 바인드합니다.

(1000,2000)

채널이 1000 - 2000 범위의 포트에 로컬로 바인드합니다.

IP 주소 대신에 호스트 이름을 지정할 수 있습니다. 브로커에 대한 실시간 연결에서 이 속성은 멀티캐스트를 사용하는 경우에만 관련이 있고, 특성 값에 포트 번호 또는 포트 번호 범위가 포함되지 않아야 합니다. 이 경우, 특성의 올바른 값은 널, IP 주소 또는 호스트 이름뿐입니다.

## MAPNAMESTYLE

MapMessage 요소 이름에 호환성 스타일을 사용할 수 있습니다.

## 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: MAPNAMESTYLE

JMS 관리 도구 짧은 이름: MNST

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setMapNameStyle()
- MQConnectionFactory.getMapNameStyle()

## 값

**STANDARD**

표준 com.ibm.jms.JMSMapMessage 요소 이름 지정 형식을 사용합니다. 기본값이며, 잘못된 Java ID를 요소 이름으로 사용할 수 있습니다.

## COMPATIBLE

이전 com.ibm.jms.JMSMapMessage 요소 이름 지정 형식을 사용합니다. 올바른 Java ID만 요소 이름으로 사용할 수 있습니다. 5.3보다 이전 버전의 IBM MQ classes for JMS를 사용하는 애플리케이션에 맵 메시지를 송신하는 경우에만 필요합니다.

## MAXBUFFSIZE

애플리케이션에서 처리할 때까지 대기하는 동안 내부 메시지 버퍼에 저장할 수 있는 수신 메시지의 최대 수입니다. 이 특성은 TRANSPORT의 값이 DIRECT 또는 DIRECTHTTP인 경우에만 적용됩니다.

### 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory

JMS 관리 도구 긴 이름: MAXBUFFSIZE

JMS 관리 도구 짧은 이름: MBSZ

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setMaxBufferSize()
- MQConnectionFactory.getMaxBufferSize()

### 값

**1000**

이는 기본값입니다.

양수

## MDREAD

이 특성은 JMS 애플리케이션이 MQMD 필드의 값을 추출할 수 있는지 여부를 판별합니다.

### 적용 가능한 오브젝트

JMS 관리 도구 긴 이름: MDREAD

JMS 관리 도구 짧은 이름: MDR

### 프로그래밍 방식 액세스

Setter/Getter

- MQDestination.setMQMDReadEnabled()
- MQDestination.getMQMDReadEnabled()

### 값

**NO**

메시지를 송신할 때 송신된 메시지의 JMS\_IBM\_MQMD\* 특성이 MQMD에서 업데이트된 필드 값을 반영하도록 업데이트되지 않습니다. 메시지를 수신할 때 어떤 JMS\_IBM\_MQMD\* 특성도 수신된 메시지에서 사용 가능하지 않습니다(송신자가 이 중에서 일부 또는 모두를 설정한 경우에도). 이 값은 관리 도구의 기본값입니다.

프로그램에는 False를 사용합니다.

### 예

메시지를 송신할 때 송신자가 명시적으로 설정하지 않은 특성을 포함하여 MQMD에서 업데이트된 필드 값을 반영하기 위해 송신된 메시지의 모든 JMS\_IBM\_MQMD\* 특성이 업데이트됩니다. 메시지를 수신할 때 송신



자가 명시적으로 설정하지 않은 특성을 포함하여 모든 JMS\_IBM\_MQMD\* 특성을 수신 메시지에서 사용할 수 있습니다.

프로그램에는 True를 사용합니다.

## MDWRITE

이 특성은 JMS 애플리케이션이 MQMD 필드의 값을 설정할 수 있는지 여부를 판별합니다.

### 적용 가능한 오브젝트

큐, 토픽

JMS 관리 도구 긴 이름: MDWRITE

JMS 관리 도구 짧은 이름: MDR

### 프로그래밍 방식 액세스

Setter/Getter

- MQDestination.setMQMDWriteEnabled()
- MQDestination.getMQMDWriteEnabled()

### 값

#### NO

모든 JMS\_IBM\_MQMD\* 특성이 무시되며, 해당 값이 기본 MQMD 구조로 복사되지 않습니다. 이 값은 관리 도구의 기본값입니다.

프로그램에는 False를 사용합니다.

#### YES

JMS\_IBM\_MQMD\* 특성이 처리됩니다. 해당 값이 기본 MQMD 구조로 복사됩니다.

프로그램에는 True를 사용합니다.

## MDMSGCTX

JMS 애플리케이션이 설정하는 메시지 컨텍스트의 레벨. 이 특성이 적용되려면 애플리케이션이 적합한 컨텍스트 권한으로 실행 중이어야 합니다.

### 적용 가능한 오브젝트

JMS 관리 도구 긴 이름: MDMSGCTX

JMS 관리 도구 짧은 이름: MDCTX

### 프로그래밍 방식 액세스

Setter/Getter

- MQDestination.setMQMDMessageContext()
- MQDestination.getMQMDMessageContext()

### 값

#### DEFAULT

MQOPEN API 호출 및 MQPMO 구조가 명시적 메시지 컨텍스트 옵션을 지정하지 않습니다. 이 값은 관리 도구의 기본값입니다.

프로그램에는 WMQ\_MDCTX\_DEFAULT를 사용합니다.

### **SET\_IDENTITY\_CONTEXT**

MQOPEN API 호출이 메시지 컨텍스트 옵션 MQOO\_SET\_IDENTITY\_CONTEXT를 지정하고 MQPMO 구조가 MQPMO\_SET\_IDENTITY\_CONTEXT를 지정합니다.

프로그램에는 WMQ\_MDCTX\_SET\_IDENTITY\_CONTEXT를 사용합니다.

### **SET\_ALL\_CONTEXT**

MQOPEN API 호출이 메시지 컨텍스트 옵션 MQOO\_SET\_ALL\_CONTEXT를 지정하고 MQPMO 구조가 MQPMO\_SET\_ALL\_CONTEXT를 지정합니다.

프로그램에는 WMQ\_MDCTX\_SET\_ALL\_CONTEXT를 사용합니다.

## **MSGBATCHSZ**

비동기 메시지 전달을 사용할 때 하나의 패킷으로 큐에서 가져올 최대 메시지 수입니다.

### **적용 가능한 오브젝트**

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: MAXBUFFSIZE

JMS 관리 도구 짧은 이름: MBSZ

### **프로그래밍 방식 액세스**

Setter/Getter

- MQConnectionFactory.setMsgBatchSize()
- MQConnectionFactory.getMsgBatchSize()

### **값**

**10**

이는 기본값입니다.

양수

## **MSGBODY**

JMS 애플리케이션이 메시지 페이로드의 일부로 IBM MQ 메시지의 MQRFH2에 액세스하는지 여부를 판별합니다.

### **적용 가능한 오브젝트**

큐, 토픽

JMS 관리 도구 긴 이름: WMQ\_MESSAGE\_BODY

JMS 관리 도구 짧은 이름: MBODY

### **프로그래밍 방식 액세스**

Setter/Getter

- MQConnectionFactory.setMessageBodyStyle()
- MQConnectionFactory.getMessageBodyStyle()

## 값

### UNSPECIFIED

송신 시에 IBM MQ classes for JMS가 WMQ\_TARGET\_CLIENT의 값에 따라 MQRFH2 헤더를 생성 및 포함하거나 생성 및 포함하지 않습니다. 수신할 때, JMS 값으로 작동합니다.

### JMS

송신 시에 IBM MQ classes for JMS가 MQRFH2 헤더를 자동 생성하며 이를 IBM MQ 메시지에 포함합니다.

수신 시에 IBM MQ classes for JMS가 MQRFH2(존재하는 경우)의 값에 따라 JMS 메시지 특성을 설정합니다. 이는 JMS 메시지 본문의 일부로서 MQRFH2를 제시하지 않습니다.

### MQ

송신 시에 IBM MQ classes for JMS가 MQRFH2를 생성하지 않습니다.

수신 시에 IBM MQ classes for JMS가 MQRFH2를 JMS 메시지 본문의 일부로서 제시합니다.

## MSGRETENTION

연결 이용자가 미배달 메시지를 입력 큐에 계속 유지하는지 여부입니다.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory,

JMS 관리 도구 긴 이름: MSGRETENTION

JMS 관리 도구 짧은 이름: MRET

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setMessageRetention()
- MQConnectionFactory.getMessageRetention()

## 값

### 예

미배달 메시지가 입력 큐에 남아 있습니다. 이는 기본값입니다.

### 아니오

해당 처리 옵션에 따라 미배달 메시지는 처리합니다.

## MSGSELECTION

IBM MQ classes for JMS 또는 브로커를 통해 메시지가 선택되는지 여부를 판별합니다. TRANSPORT의 값이 DIRECT이면 항상 브로커를 통해 메시지가 선택되고 값이 MSGSELECTION이면 무시됩니다. BROKERVER의 값이 V1이면 브로커가 메시지를 선택할 수 없습니다.

### 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: MSGSELECTION

JMS 관리 도구 짧은 이름: MSEL

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setMessageSelection()
- MQConnectionFactory.getMessageSelection()

## 값

### 클라이언트

IBM MQ classes for JMS를 통해 메시지가 선택됩니다. 이는 기본값입니다.

### BROKER

브로커를 통해 메시지가 선택됩니다.

## MULTICAST

브로커에 대한 실시간 연결에서 멀티캐스트를 사용하도록 설정할 수 있으며, 설정한 경우 멀티캐스트를 사용하여 브로커에서 메시지 이용자로 메시지를 전달하는 정확한 방법을 지정합니다. 이 특성은 메시지 작성자가 어떻게 브로커에 메시지를 송신하는지에는 영향을 주지 않습니다.

### 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory, Topic

JMS 관리 도구 긴 이름: MULTICAST

JMS 관리 도구 짧은 이름: MCAST

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setMulticast()
- MQConnectionFactory.getMulticast()

## 값

### DISABLED

메시지가 멀티캐스트 전송을 사용하여 메시지 이용자로 전달되지 않습니다. 이 값이 ConnectionFactory 및 TopicConnectionFactory 오브젝트의 기본값입니다.

### ASCF

메시지가 메시지 이용자와 연관된 연결 팩토리의 멀티캐스트 설정에 따라 메시지 이용자로 전달됩니다. 메시지 이용자를 작성할 때 연결 팩토리의 멀티캐스트 설정이 표시됩니다. 이 값은 토픽 오브젝트에서만 유효하고, 토픽 오브젝트의 기본값입니다.

### ENABLED

브로커에서 토픽에 멀티캐스트가 구성된 경우, 메시지가 멀티캐스트 전송을 사용하여 메시지 이용자로 전달됩니다. 토픽에 신뢰할 수 있는 멀티캐스트가 구성되면 신뢰할 수 있는 서비스 품질(QoS)이 사용됩니다.

### RELIABLE

브로커에서 토픽에 신뢰할 수 있는 멀티캐스트가 구성된 경우, 메시지가 신뢰할 수 있는 서비스 품질(QoS)의 멀티캐스트 전송을 사용하여 메시지 이용자로 전달됩니다. 토픽에 신뢰할 수 있는 멀티캐스트가 구성되지 않으면 토픽에 대한 메시지 이용자를 작성할 수 없습니다.

### NOTR

브로커에서 토픽에 멀티캐스트가 구성된 경우, 메시지가 멀티캐스트 전송을 사용하여 메시지 이용자로 전달됩니다. 토픽에 신뢰할 수 있는 멀티캐스트가 구성되더라도 신뢰할 수 있는 서비스 품질(QoS)이 사용되지 않습니다.

## OPTIMISTICPUBLICATION

이 특성은 IBM MQ classes for JMS에서 메시지를 발행한 발행자에게 즉시 제어를 리턴하는지, 또는 호출과 연관된 모든 처리를 완료하고 발행자에게 결과를 보고할 수 있게 된 후에만 제어를 리턴하는지 여부를 판별합니다.

### 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory

JMS 관리 도구 긴 이름: OPTIMISTICPUBLICATION

JMS 관리 도구 짧은 이름: OPTPUB

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setOptimisticPublication()
- MQConnectionFactory.getOptimisticPublication()

### 값

#### NO

발행자가 메시지를 발행하면, 호출과 연관된 모든 처리를 완료하고 발행자에게 결과를 보고할 수 있을 때까지 IBM MQ classes for JMS에서 발행자에게 제어를 리턴하지 않습니다. 이는 기본값입니다.

#### YES

발행자가 메시지를 발행하면, 호출과 연관된 모든 처리를 완료하고 발행자에게 결과를 보고할 수 있기 전에 IBM MQ classes for JMS에서 발행자에게 즉시 제어를 리턴합니다. IBM MQ classes for JMS는 발행자가 메시지를 커밋할 때에만 결과를 보고합니다.

## OUTCOMENOTIFICATION

이 특성은 IBM MQ classes for JMS에서 방금 메시지를 수신확인했거나 커밋한 구독자에게 즉시 제어를 리턴하는지, 또는 호출과 연관된 모든 처리를 완료한 후에만 제어를 리턴하는지 여부를 판별하고 구독자에게 결과를 보고할 수 있습니다.

### 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory

JMS 관리 도구 긴 이름: OUTCOMENOTIFICATION

JMS 관리 도구 짧은 이름: NOTIFY

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setOutcomeNotification()
- MQConnectionFactory.getOutcomeNotification()

### 값

#### YES

구독자가 메시지를 수신확인 또는 커밋하면, 호출과 연관된 모든 처리를 완료할 때까지 IBM MQ classes for JMS에서 구독자에게 제어를 리턴하지 않고 구독자에게 결과를 보고할 수 있습니다. 이는 기본값입니다.

#### NO

구독자가 메시지를 수신확인 또는 커밋하면, 호출과 연관된 모든 처리가 완료되기 전에 IBM MQ classes for JMS에서 구독자에게 제어를 즉시 리턴하고 구독자에게 결과를 보고할 수 있습니다.

## PERSISTENCE

목적지로 송신되는 메시지의 지속성입니다.

### 적용 가능한 오브젝트

큐, 토픽

JMS 관리 도구 긴 이름: PERSISTENCE

JMS 관리 도구 짧은 이름: PER

## 프로그래밍 방식 액세스

Setter/Getter

- MQDestination.setPersistence()
- MQDestination.getPersistence()

## 값

### APP

지속성을 JMS 애플리케이션에서 정의합니다. 이는 기본값입니다.

### QDEF

지속성에 큐 기본값이 적용됩니다.

### PERS

메시지가 지속됩니다.

### NON

메시지가 지속되지 않습니다.

### HIGH

이 값의 사용에 대한 자세한 정보는 [JMS 지속 메시지의 내용](#)을 참조하십시오.

## POLLINGINT

세션의 각 메시지 리스너에서 큐에 적절한 메시지가 없는 경우, 각 메시지 리스너가 큐에서 메시지를 다시 가져오려고 시도할 때까지의 최대 간격(밀리초)입니다. 세션의 메시지 리스너에 사용 가능한 적합한 메시지가 없는 경우가 자주 발생하면 이 특성 값을 높여 보십시오. 이 특성은 TRANSPORT의 값이 BIND이거나 CLIENT인 경우에만 해당합니다.

## 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: POLLINGINT

JMS 관리 도구 짧은 이름: PINT

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setPollingInterval()
- MQConnectionFactory.getPollingInterval()

## 값

### 5,000

이는 기본값입니다.

양수

## 포트

큐 관리자에 대한 연결에서는 큐 관리자가 대기하는 포트 번호이고, 브로커에 대한 실시간 연결에서는 브로커가 실시간 연결을 대기하는 포트 번호입니다.

## 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: PORT

JMS 관리 도구 짧은 이름: PORT

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setPort()
- MQConnectionFactory.getPort()

### 값

#### 1414

TRANSPORT를 CLIENT로 설정한 경우 기본값입니다.

#### 1506

TRANSPORT를 DIRECT 또는 DIRECTHTTP로 설정한 경우 기본값입니다.

양수

## PRIORITY

목적지로 송신되는 메시지의 우선순위입니다.

### 적용 가능한 오브젝트

큐, 토픽

JMS 관리 도구 긴 이름: PRIORITY

JMS 관리 도구 짧은 이름: PRI

### 프로그래밍 방식 액세스

Setter/Getter

- MQDestination.setPriority()
- MQDestination.getPriority()

### 값

#### APP

우선순위를 JMS 애플리케이션에서 정의합니다. 이는 기본값입니다.

#### QDEF

우선순위에 큐 기본값이 적용됩니다.

#### 0-9 범위의 정수

가장 낮은 값에서 가장 높은 값입니다.

## PROCESSDURATION

이 특성은 구독자가 제어를 IBM MQ classes for JMS로 리턴하기 전에 수신 메시지의 빠른 처리를 보장하는지 여부를 판별합니다.

### 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory

JMS 관리 도구 긴 이름: PROCESSDURATION

JMS 관리 도구 짧은 이름: PROCDUR

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setProcessDuration()
- MQConnectionFactory.getProcessDuration()

## 값

알 수 없음

구독자가 수신 메시지의 빠른 처리를 보장할 수 없습니다. 이는 기본값입니다.

**SHORT**

구독자가 제어를 IBM MQ classes for JMS로 리턴하기 전에 수신 메시지의 빠른 처리를 보장합니다.

## PROVIDERVERSION

이 특성은 IBM MQ 메시징 제공자 정상 모드, IBM MQ 메시징 제공자 정상 모드(제한 있음), IBM MQ 메시징 제공자 마이그레이션 모드 등 3가지 IBM MQ 메시징 조작 모드를 구분합니다.

IBM MQ 메시징 제공자 정상 모드는 IBM MQ 큐 관리자의 모든 기능을 사용하여 JMS를 구현합니다. 이 모드는 JMS 2.0 API 및 기능을 사용하도록 최적화되어 있습니다. IBM MQ 메시징 제공자 정상 모드(제한 있음)는 JMS 2.0 API를 사용하지만, 공유 구독, 지연 배달, 비동기 송신과 같은 새 기능은 사용하지 않습니다.

## 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: PROVIDERVERSION

JMS 관리 도구 짧은 이름: PVER

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setProviderVersion()
- MQConnectionFactory.getProviderVersion()

## 값

**PROVIDERVERSION** 특성을 8(일반 모드), 7(제한사항이 있는 정상 모드), 6(마이그레이션 모드) 또는 불특정화(기본값) 중 하나로 설정할 수 있습니다. **PROVIDERVERSION** 특성에 지정하는 값은 문자열이어야 합니다. 8, 7 또는 6 옵션을 지정하는 경우, 다음 형식으로 이를 수행할 수 있습니다.

- V.R.M.F
- V.R.M
- V.R
- V

여기서 V, R, M 및 F는 0 이상의 정수 값입니다. 추가 R, M 및 F 값은 선택적이며, 미세한 제어가 필요한 경우에 사용할 수 있습니다. 예를 들어 **PROVIDERVERSION** 레벨 7을 사용하려면 **PROVIDERVERSION**=7, 7.0, 7.0.0 또는 7.0.0.0으로 설정할 수 있습니다.

### 8 - 정상 모드

JMS 애플리케이션은 IBM MQ 메시징 제공자 정상 모드를 사용합니다. 정상 모드는 IBM MQ 큐 관리자의 모든 기능을 사용하여 JMS를 구현합니다. 이 모드는 JMS 2.0 API 및 기능을 사용하도록 최적화되어 있습니다.

명령 레벨이 800이상인 큐 관리자에 연결하는 경우, 비동기 전송, 지연된 전달 또는 공유 등록과 같은 모든 JMS 2.0 API 및 기능을 사용할 수 있습니다.



연결 팩토리 설정에 지정된 큐 관리자가 IBM MQ 8.0.0 이상의 큐 관리자가 아닌 경우 createConnection 메소드가 JMSFMQ0003 예외와 함께 실패합니다.

IBM MQ 메시징 제공자 정상 모드는 공유 대화 기능을 사용하며 공유할 수 있는 대화 수는 서버 연결 채널의 **SHARECNV()** 특성에 의해 제어됩니다. 이 특성이 0으로 설정되면, IBM MQ 메시징 제공자 정상 모드를 사용할 수 없고 createConnection 메소드는 JMSSC5007 예외와 함께 실패합니다.

## 7 - 제한이 있는 정상 모드

JMS 애플리케이션은 제한이 있는 IBM MQ 메시징 제공자 정상 모드를 사용합니다. 이 모드는 JMS 2.0 API 를 사용하지만 공유 구독, 지연 전달 또는 비동기 송신과 같은 새 기능은 사용하지 않습니다.

**PROVIDERVERSION**을(를) 7로 설정하면 제한 조작 모드로 정상적인 IBM MQ 메시징 제공자만 사용할 수 있습니다.

제한사항이 있는 정상 모드를 사용하여 연결하는 경우, 명령 레벨이 800미만인 큐 관리자에게는 JMS 2.0 API 를 사용할 수 있지만 비동기 전송, 지연된 전달 또는 공유 구독 기능은 사용할 수 없습니다.

제한사항이 있는 IBM MQ 메시징 제공자 정상 모드는 공유 대화 기능을 사용하며 공유할 수 있는 대화 수는 서버 연결 채널의 **SHARECNV()** 특성에 의해 제어됩니다. 이 특성이 0으로 설정되면 제한이 있는 IBM MQ 메시징 제공자 정상 모드를 사용할 수 없으며 createConnection 메소드가 실패하고 예외 코드는 JMSSC5007입니다.

## 6 - 마이그레이션 모드

JMS 애플리케이션이 IBM MQ 메시징 제공자 마이그레이션 모드를 사용합니다.

이 모드를 사용하여 IBM MQ 8.0 또는 나중의 큐 관리자에 연결할 수 있지만, IBM MQ classes for JMS 큐 관리자의 새 기능(예: 미리 읽기 또는 스트리밍)은 사용되지 않습니다.

IBM MQ 8.0 이상의 클라이언트가 IBM MQ 8.0 이상의 큐 관리자에 연결되어 있는 경우 클라이언트 시스템 대신 큐 관리자가 메시지 선택을 수행합니다.

IBM MQ 메시징 제공자 마이그레이션 모드가 지정되고 JMS 2.0 API 중 하나를 사용하려고 하면 API 메소드 호출이 JMSSC5007 예외로 실패합니다.

### unspecified(기본값)

**PROVIDERVERSION** 특성은 기본적으로 불특정화로 설정됩니다.

연결 팩토리가 새 버전의 IBM MQ classes for JMS에서 사용되는 경우 JNDI의 이전 IBM MQ classes for JMS 버전에서 작성된 연결 팩토리는 이 값을 사용합니다. 다음 알고리즘을 사용하여 어떤 조작 모드를 사용할 지 판별합니다. 이 알고리즘은 createConnection 메소드가 호출될 때 사용되며 연결 팩토리의 다른 측면을 사용하여, IBM MQ 메시징 제공자 정상 모드, 제한이 있는 정상 모드 또는 IBM MQ 메시징 제공자 마이그레이션 모드 중 필요한 모드를 판별합니다.

1. 먼저 IBM MQ 메시징 제공자 정상 모드를 사용합니다.
2. 연결된 큐 관리자가 IBM MQ 8.0 이상이 아닌 경우 제한이 있는 IBM MQ 메시징 제공자 정상 모드를 사용합니다.
3. 연결된 큐 관리자가 IBM WebSphere MQ 7.0.1 이상이 아닌 경우 연결이 차단되고 IBM MQ 메시징 제공자 마이그레이션 모드가 사용됩니다.
4. 서버 연결 채널의 **SHARECNV** 특성이 0으로 설정된 경우, 연결이 닫히고 대신 IBM MQ 메시징 제공자 이주 모드가 사용됩니다.
5. **BROKERVER** 이 V1 또는 기본 지정되지 않음 값으로 설정된 경우, IBM MQ 메시징 제공자 정상 모드가 계속 사용됩니다.

호환성에 대한 자세한 정보는 ALTER QMGR 명령의 PSMODE 매개변수 정보에서 [ALTER QMGR](#)을 참조하십시오.

6. **BROKERVER**이(가) V2로 설정된 경우 수행되는 조치는 **BROKERQMGR**의 값에 따라 다릅니다.

- **BROKERQMGR**이(가) 공백인 경우:

**BROKERCONQ** 특성에 지정된 큐를 출력을 위해 열 수 있고(즉, 출력을 위한 MQOPEN이 성공하는 경우) 큐 관리자의 **PSMODE**가 COMPAT 또는 DISABLED로 설정된 경우, IBM MQ 메시징 제공자 마이그레이션 모드가 사용됩니다.

- **BROKERCONQ** 특성으로 지정된 큐를 출력용으로 열 수 없거나 **PSMODE** 속성이 **ENABLED:**로 설정된 경우  
IBM MQ 메시징 제공자 정상 모드가 사용됩니다.
- **BROKERQMGR**이(가) 비어 있지 않은 인 경우:  
IBM MQ 메시징 제공자 마이그레이션 모드가 사용됩니다.

사용 중인 연결 팩토리를 변경할 수 없는 경우 `com.ibm.msg.client.wmq.overrideProviderVersion` 특성을 사용하여 연결 팩토리의 설정을 대체할 수 있습니다. 이 대체는 JVM의 모든 연결 팩토리에 적용되지만 실제 연결 팩토리 오브젝트는 수정되지 않습니다.

#### 관련 태스크

JMS **PROVIDERVERSION** 특성 구성

## PROXYHOSTNAME

프록시 서버를 통해 브로커에 대한 실시간 연결을 사용하여 프록시 서버가 실행되는 시스템의 호스트 이름 또는 IP 주소입니다.

### 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory

JMS 관리 도구 긴 이름: PROXYHOSTNAME

JMS 관리 도구 짧은 이름: PHOST

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setProxyHostName()
- MQConnectionFactory.getProxyHostName()

### 값

#### 널

프록시 서버의 호스트 이름. 이는 기본값입니다.

## PROXYPORT

프록시 서버를 통해 브로커에 대한 실시간 연결을 사용하여 프록시 서버가 대기하는 포트 번호입니다.

### 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory

JMS 관리 도구 긴 이름: PROXYPORT

JMS 관리 도구 짧은 이름: PPORT

### 프로그래밍 방식 액세스

Setter/Getter

MQConnectionFactory.setProxyPort()

MQConnectionFactory.getProxyPort()

## 값

443

프록시 서버의 포트 번호입니다. 이는 기본값입니다.

## PUBACKINT

IBM MQ classes for JMS에서 브로커의 수신확인을 요청하기 전에 발행자가 발행한 메시지 수입니다.

이 특성의 값을 낮추면 IBM MQ classes for JMS가 수신확인을 더 자주 요청하므로 발행자의 성능이 저하됩니다. 값을 높이면 IBM MQ classes for JMS에서 브로커 실패 시 예외를 전달하는 데 시간이 더 오래 소요됩니다. 이 특성은 TRANSPORT의 값이 BIND이거나 CLIENT인 경우에만 해당합니다.

### 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: PROXYPORT

JMS 관리 도구 짧은 이름: PPORT

### 프로그래밍 방식 액세스

Setter/Getter

MQConnectionFactory.setPubAckInterval()

MQConnectionFactory.getPubAckInterval()

## 값

25GB

임의의 양의 정수가 기본값이 될 수 있습니다.

## PUTASYNCALOWED

이 특성은 메시지 작성자가 비동기 Put을 사용하여 메시지를 이 목적지로 송신할 수 있는지 여부를 판별합니다.

### 적용 가능한 오브젝트

큐, 토픽

JMS 관리 도구 긴 이름: PUTASYNCALOWED

JMS 관리 도구 짧은 이름: PAALD

### 프로그래밍 방식 액세스

Setter/Getter

MQDestination.setPutAsyncAllowed()

MQDestination.getPutAsyncAllowed()

## 값

AS\_DEST

큐 정의 또는 토픽 정의를 참조하여 비동기 Put이 허용되는지 여부를 판별합니다. 이는 기본값입니다.

AS\_Q\_DEF

큐 정의를 참조하여 비동기 Put이 허용되는지 여부를 판별합니다.

AS\_TOPIC\_DEF

토픽 정의를 참조하여 비동기 Put이 허용되는지 여부를 판별합니다.

**NO**

비동기 Put이 허용되지 않습니다.

**YES**

비동기 Put이 허용됩니다.

## QMANAGER

연결할 큐 관리자의 이름입니다.

그러나, 애플리케이션이 클라이언트 채널 정의 테이블을 사용하여 큐 관리자에 연결하는 경우 [IBM MQ classes for JMS](#)에 클라이언트 채널 정의 테이블 사용을 참조하십시오.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, Queue, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: QMANAGER

JMS 관리 도구 짧은 이름: QMGR

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setQueueManager()
- MQConnectionFactory.getQueueManager()

### 값

""(빈 문자열)

임의의 문자열이 기본값일 수 있습니다.

## 큐

JMS 큐 목적지의 이름입니다. 이 이름은 큐 관리자에 사용되는 큐의 이름과 같습니다.

### 적용 가능한 오브젝트

큐

JMS 관리 도구 긴 이름: QUEUE

JMS 관리 도구 짧은 이름: QU

### 값

임의의 문자열

유효한 IBM MQ 큐 이름.

### 관련 참조

[IBM MQ 오브젝트의 이름 지정 규칙](#)>

## READAHEADALLOWED

이 특성은 메시지를 수신하기 전에 메시지 이용자와 큐 브라우저가 이 목적지에서 내부 버퍼로 비지속 메시지를 가져오기 위해 미리 읽기를 사용할 수 있는지 여부를 판별합니다.

### 적용 가능한 오브젝트

큐, 토픽

JMS 관리 도구 긴 이름: READAHEADALLOWED

JMS 관리 도구 짧은 이름: RAALD

## 프로그래밍 방식 액세스

Setter/Getter

- MQDestination.setReadAheadAllowed()
- MQDestination.getReadAheadAllowed()

### 값

#### AS\_DEST

큐 정의 또는 토픽 정의를 참조하여 미리 읽기가 허용되는지 여부를 판별합니다. 관리 도구에서 기본값입니다.

프로그램에서 WMQConstants.WMQ\_READ\_AHEAD\_ALLOWED\_AS\_DEST를 사용하십시오.

#### AS\_Q\_DEF

큐 정의를 참조하여 미리 읽기가 허용되는지 여부를 판별합니다.

프로그램에서 WMQConstants.WMQ\_READ\_AHEAD\_ALLOWED\_AS\_Q\_DEF를 사용하십시오.

#### AS\_TOPIC\_DEF

토픽 정의를 참조하여 미리 읽기가 허용되는지 여부를 판별합니다.

프로그램에서 WMQConstants.WMQ\_READ\_AHEAD\_ALLOWED\_AS\_TOPIC\_DEF를 사용하십시오.

#### NO

미리 읽기가 허용되지 않습니다.

프로그램에서 WMQConstants.WMQ\_READ\_AHEAD\_ALLOWED\_DISABLED를 사용하십시오.

#### YES

미리 읽기가 허용됩니다.

프로그램에서 WMQConstants.WMQ\_READ\_AHEAD\_ALLOWED\_ENABLED를 사용하십시오.

## READAHEADCLOSEPOLICY

비동기 메시지 리스너로 전달되는 메시지의 경우, 메시지 이용자를 닫을 때 내부 미리 읽기 버퍼에서 메시지에 일어나는 일입니다.

### 적용 가능한 오브젝트

큐, 토픽

JMS 관리 도구 긴 이름: READAHEADCLOSEPOLICY

JMS 관리 도구 짧은 이름: RACP

## 프로그래밍 방식 액세스

Setter/Getter

- MQDestination.setReadAheadClosePolicy()
- MQDestination.getReadAheadClosePolicy()

### 값

#### DELIVER\_ALL

리턴하기 전에 내부 미리 읽기 버퍼의 모든 메시지가 애플리케이션의 메시지 리스너로 전달됩니다. 관리 도구에서 기본값입니다.

프로그램에서 WMQConstants.WMQ\_READ\_AHEAD\_DELIVERALL을 사용하십시오.

## **DELIVER\_CURRENT**

리턴하기 전에 현재 메시지 리스너 호출만 완료하고, 내부 미리 읽기 버퍼에 메시지를 남길 수 있습니다. 이 메시지는 이후 제거됩니다.

프로그램에서 `WMQConstants.WMQ_READ_AHEAD_DELIVERCURRENT`를 사용하십시오.

## **RECEIVECCSID**

큐 관리자 메시지 변환용 대상 CCSID를 설정하는 목적지 특성입니다. `RECEIVECONVERSION`이 `WMQ_RECEIVE_CONVERSION_QMGR`로 설정되지 않으면 값이 무시됩니다.

### **적용 가능한 오브젝트**

큐, 토픽

JMS 관리 도구 긴 이름: `RECEIVECCSID`

JMS 관리 도구 짧은 이름: `RCCS`

### **프로그래밍 방식 액세스**

#### **Setter/Getter**

- `MQDestination.setReceiveCCSID`
- `MQDestination.getReceiveCCSID`

#### **값**

##### **`WMQConstants.WMQ_RECEIVE_CC_SID_JVM_DEFAULT`**

0 - JVM Charset.defaultCharset 사용

##### **1208**

UTF-8

##### **CCSID**

지원되는 코드화 문자 세트 ID.

## **RECEIVECONVERSION**

데이터 변환이 큐 관리자에 의해 수행되는지 여부를 판별하는 목적지 특성입니다.

### **적용 가능한 오브젝트**

큐, 토픽

JMS 관리 도구 긴 이름: `RECEIVECONVERSION`

JMS 관리 도구 짧은 이름: `RCNV`

### **프로그래밍 방식 액세스**

#### **Setter/Getter**

- `MQDestination.setReceiveConversion`
- `MQDestination.getReceiveConversion`

#### **값**

##### **`WMQConstants.WMQ_RECEIVE_CONVERSION_CLIENT_MSG`**

1 - JMS 클라이언트에서만 데이터 변환을 수행합니다. 기본값은 최대 V7.0부터이며, 7.0.1.5를 포함합니다.

## **WMQConstants.WMQ\_RECEIVE\_CONVERSION\_QMGR**

2 - 메시지를 클라이언트로 송신하기 전에 큐 관리자에서 데이터 변환을 수행합니다. APAR IC72897을 적용한 경우를 제외하고, 기본값(유일한 값)은 V7.0 - V7.0.1.4(해당 값 포함)입니다.

## **RECEIVEISOLATION**

이 특성은 구독자가 구독자 큐에서 커밋되지 않은 메시지를 수신할 수 있는지 여부를 판별합니다.

### **적용 가능한 오브젝트**

ConnectionFactory, TopicConnectionFactory

JMS 관리 도구 긴 이름: RECEIVEISOLATION

JMS 관리 도구 짧은 이름: RCVISOL

### **값**

#### **COMMITTED**

구독자가 구독자 큐에서 커밋된 메시지만 수신합니다. 관리 도구에서 기본값입니다.

프로그램에서 WMQConstants.WMQ\_RCVISOL\_COMMITTED를 사용하십시오.

#### **UNCOMMITTED**

구독자가 구독자 큐에서 커밋되지 않은 메시지를 수신할 수 있습니다.

프로그램에서 WMQConstants.WMQ\_RCVISOL\_UNCOMMITTED를 사용하십시오.

## **RECEXIT**

채널 수신 엑시트 또는 연속으로 실행할 일련의 수신 엑시트를 식별합니다.

IBM MQ classes for JMS에서 수신 엑시트를 찾으려 하는 데 추가 구성이 필요할 수 있습니다. 자세한 정보는 [JMS에서 채널 엑시트를 사용하도록 IBM MQ 클래스 구성을 참조하십시오](#).

### **적용 가능한 오브젝트**

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: RECEXIT

JMS 관리 도구 짧은 이름: RCX

### **프로그래밍 방식 액세스**

Setter/Getter

- MQConnectionFactory.setReceiveExit()
- MQConnectionFactory.getReceiveExit()

### **값**

- 널. 이는 기본값입니다.
- 하나 이상의 항목으로 구성되고 심볼로 구분된 문자열로, 각 항목은 다음과 같습니다.
  - WMQReceiveExit 인터페이스를 구현하는 클래스의 이름(Java에서 작성된 채널 수신 엑시트의 경우)
  - *libraryName(entryPointName)* 형식의 문자열(Java에서 작성되지 않은 채널 수신 엑시트의 경우).

## **RECEXITINIT**

호출 시 채널 수신 엑시트로 전달되는 사용자 데이터입니다.

## 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: RECEXITINIT

JMS 관리 도구 짧은 이름: RCXI

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setReceiveExitInit()
- MQConnectionFactory.getReceiveExitInit()

## 값

### 널

하나 이상의 사용자 데이터 항목으로 구성되고 심표로 구분된 문자열입니다. 이는 기본값입니다.

## REPLYTOSTYLE

수신 메시지의 JMSReplyTo 필드를 구성하는 방법을 판별합니다.

## 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: REPLYTOSTYLE

JMS 관리 도구 짧은 이름: RTOST

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setReplyToStyle()
- MQConnectionFactory.getReplyToStyle()

## 값

### DEFAULT

MQMD와 같습니다.

### RFH2

RFH2 헤더에 제공된 값을 사용합니다. 송신 애플리케이션에 JMSReplyTo 값을 설정한 경우, 해당 값을 사용합니다.

### MQMD

MQMD에서 제공된 값을 사용합니다. 이 동작은 IBM WebSphere MQ 6.0.2.4 및 6.0.2.5의 기본 동작과 같습니다.

송신 애플리케이션이 설정한 JMSReplyTo 값에 큐 관리자 이름이 없으면 수신 큐 관리자가 고유 이름을 MQMD에 삽입합니다. 이 매개변수를 MQMD로 설정하면 사용하는 응답 대상 큐가 수신 큐 관리자에 있습니다. 이 매개변수를 RFH2로 설정하면 사용하는 응답 대상 큐가 송신 애플리케이션에서 원래 설정한 대로 송신 메시지의 RFH2에 지정된 큐 관리자에 있습니다.

송신 애플리케이션이 설정한 JMSReplyTo 값에 큐 관리자 이름이 있으면 MQMD 및 RFH2의 값이 동일하기 때문에 이 매개변수의 값이 중요하지 않습니다.



## RESCANINT

포인트-투-포인트 도메인의 메시지 이용자가 메시지 선택자를 사용하여 수신할 메시지를 선택하는 경우, IBM MQ classes for JMS는 IBM MQ 큐에서 큐의 `MsgDeliverySequence` 속성으로 결정된 일련의 적절한 메시지를 검색합니다.

IBM MQ classes for JMS가 적절한 메시지를 찾아 이용자에 전달한 후 IBM MQ classes for JMS는 큐의 현재 위치에서 적절한 다음 메시지를 검색하는 작업을 재개합니다. IBM MQ classes for JMS는 큐의 끝에 도달하거나 이 특성 값으로 판별되는 시간 간격(밀리초)이 만기될 때까지 이런 식으로 큐를 계속 검색합니다. 이 경우, IBM MQ classes for JMS는 큐의 시작 부분으로 되돌아가 검색을 계속하고, 새로운 시간 간격이 시작됩니다.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory

JMS 관리 도구 긴 이름: RESCANINT

JMS 관리 도구 짧은 이름: RINT

### 프로그래밍 방식 액세스

Setter/Getter

- `MQConnectionFactory.setRescanInterval()`
- `MQConnectionFactory.getRescanInterval()`

### 값

**5,000**

임의의 양의 정수가 기본값이 될 수 있습니다.

## SECEXIT

채널 보안 엑시트를 식별합니다.

IBM MQ classes for JMS에서 보안 엑시트를 찾으려 하는 데 추가 구성이 필요할 수 있습니다. 자세한 정보는 [JMS에서 채널 엑시트를 사용하도록 IBM MQ 클래스 구성을 참조하십시오.](#)

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: SECEXIT

JMS 관리 도구 짧은 이름: SXC

### 프로그래밍 방식 액세스

Setter/Getter

- `MQConnectionFactory.setSecurityExit()`
- `MQConnectionFactory.getSecurityExit()`

### 값

- 널. 이는 기본값입니다.
- 하나 이상의 항목으로 구성되고 쉼표로 구분된 문자열로, 각 항목은 다음과 같습니다.
  - `WMQSecurityExit` 인터페이스를 구현하는 클래스의 이름(Java에서 작성된 채널 보안 엑시트의 경우)
  - `libraryName(entryPointName)` 형식의 문자열(Java에서 작성되지 않은 채널 보안 엑시트의 경우).

## SECEXITINIT

호출 시 채널 보안 엑시트로 전달되는 사용자 데이터입니다.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: SECEXITINIT

JMS 관리 도구 짧은 이름: SCXI

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setSecurityExitInit()
- MQConnectionFactory.getSecurityExitInit()

### 값

#### 널

임의의 문자열이 기본값일 수 있습니다.

## SENDCHECKCOUNT

변환되지 않은 하나의 JMS 세션에서 비동기 put 오류를 검사하는 사이에 허용할 수 있는 송신 호출 수입니다.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: SENDCHECKCOUNT

JMS 관리 도구 짧은 이름: SCC

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setSendCheckCount()
- MQConnectionFactory.getSendCheckCount()

### 값

#### 널

임의의 문자열이 기본값일 수 있습니다.

## SENDEXIT

채널 송신 엑시트 또는 연속으로 실행할 일련의 송신 엑시트를 식별합니다.

IBM MQ classes for JMS에서 송신 엑시트를 찾으려 하는 데 추가 구성이 필요할 수 있습니다. 자세한 정보는 [JMS에서 채널 엑시트를 사용하도록 IBM MQ 클래스 구성을 참조하십시오](#).

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: SENDEXIT

JMS 관리 도구 짧은 이름: SDX

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setSendExit()
- MQConnectionFactory.getSendExit()

### 값

- 널. 이는 기본값입니다.
- 하나 이상의 항목으로 구성되고 쉼표로 구분된 문자열로, 각 항목은 다음과 같습니다.
  - WMQSendExit 인터페이스를 구현하는 클래스의 이름(Java에서 작성된 채널 송신 엑시트의 경우)
  - *libraryName(entryPointName)* 형식의 문자열(Java에서 작성되지 않은 채널 송신 엑시트의 경우).

## SENDEXITINIT

호출 시 채널 송신 엑시트로 전달되는 사용자 데이터입니다.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: SENDEXITINIT

JMS 관리 도구 짧은 이름: SDXI

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setSendExitInit()
- MQConnectionFactory.getSendExitInit()

### 값

널

하나 이상의 사용자 데이터 항목으로 구성되고 쉼표로 구분된 문자열이 기본값이 될 수 있습니다.

## SHARECONVALLOWED

IBM MQ 메시징 제공자 정상 모드 또는 제한이 있는 정상 모드를 사용하는 애플리케이션의 경우, 이 특성은 공유 대화 기능이 연결 팩토리에서 작성된 JMS 연결, 세션 및 컨텍스트에 사용되는지 여부를 판별합니다.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: SHARECONVALLOWED

JMS 관리 도구 짧은 이름: SCALD

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setShareConvAllowed()
- MQConnectionFactory.getShareConvAllowed()

## 값

### YES

동일한 JVM내의 연결 팩토리에서 작성된 JMS 연결, 세션 및 컨텍스트는 적절한 경우 채널 인스턴스 (TCP/IP 연결에 매핑됨) 를 공유할 수 있습니다.

이 값은 관리 도구의 기본값입니다.

프로그램에는 WMQConstants.WMQ\_SHARE\_CONV\_ALLOWED\_YES를 사용하십시오.

### NO

연결 팩토리에서 작성된 모든 JMS 연결 및 해당 JMS연결에서 작성된 모든 JMS 세션에는 큐 관리자에 대한 자체 채널 인스턴스 (TCP/IP 연결) 가 있습니다.

JMS 컨텍스트의 경우, 연결 팩토리에서 작성되는 첫 번째 컨텍스트는 두 개의 채널 인스턴스 (TCP/IP 연결) 를 작성합니다. 첫 번째 컨텍스트에서 작성된 기타 JMS 컨텍스트에는 자체 채널 인스턴스 (TCP/IP 연결) 가 있습니다.

프로그램에는 WMQConstants.WMQ\_SHARE\_CONV\_ALLOWED\_NO를 사용하십시오.

## 관련 개념

[IBM MQ 메시징 제공자 조작 모드](#)

[JMS용 IBM MQ 클래스에서 TCP/IP 연결 공유](#)

## SPARSESUBS

TopicSubscriber 오브젝트의 메시지 검색 정책을 제어합니다.

## 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory

JMS 관리 도구 긴 이름: SPARSESUBS

JMS 관리 도구 짧은 이름: SSUBS

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setSparseSubscriptions()
- MQConnectionFactory.getSparseSubscriptions()

## 값

### NO

구독에서 자주 일치하는 메시지를 수신합니다. 이 값은 관리 도구의 기본값입니다.

프로그램에는 false를 사용하십시오.

### YES

구독에서 드물게 일치하는 메시지를 수신합니다. 이 값을 사용하려면 구독 큐를 읽기 전용으로 열 수 있어야 합니다.

프로그램에는 true를 사용하십시오.

## SSLCIPHERSUITE

TLS 연결에 사용할 CipherSuite입니다.

## 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: SSLCIPHERSUITE

JMS 관리 도구 짧은 이름: SCPHS

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setSSLCipherSuite()
- MQConnectionFactory.getSSLCipherSuite()

## 값 널

이는 기본값입니다. 자세한 정보는 [JMS 오브젝트의 TLS 특성을 참조하십시오](#).

## SSLCRL

TLS 인증서 폐기를 검사하는 CRL 서버입니다.

## 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: SSLCRL

JMS 관리 도구 짧은 이름: SCRL

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setSSLCertStores()
- MQConnectionFactory.getSSLCertStores()

## 값 널

간격으로 구분된 LDAP URL 목록. 이는 기본값입니다. 자세한 정보는 [JMS 오브젝트의 TLS 특성을 참조하십시오](#).

## SSLFIPSREQUIRED

이 특성은 TLS 연결이 IBM Java JSSE FIPS 제공자(IBMJSSEFIPS)에서 지원되는 CipherSuite를 사용해야 하는지 여부를 판별합니다.

**참고:** AIX, Linux, and Windows에서 IBM MQ 는 IBM Crypto for C (ICC) 암호화 모듈을 통해 FIPS 140-2준수를 제공합니다. 이 모듈의 인증서가 히스토리 상태로 이동되었습니다. 고객은 IBM Crypto for C (ICC) 인증서를 보고 NIST에서 제공하는 조언을 알고 있어야 합니다. 대체 FIPS 140-2모듈이 현재 진행 중이며 해당 상태는 프로세스 목록의 [NIST CMVP 모듈](#)에서 검색하여 볼 수 있습니다.

IBM MQ Operator 3.2.0 및 큐 관리자 컨테이너 이미지 9.4.0.0 이상은 UBI 9를 기반으로 합니다. FIPS 140-2 준수가 현재 보류 중이며 해당 상태는 [프로세스 목록의 NIST CMVP 모듈](#)에서 "Red Hat Enterprise Linux 9-OpenSSL FIPS 제공자"를 검색하여 볼 수 있습니다.

## 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: SSLFIPSREQUIRED

JMS 관리 도구 짧은 이름: SFIPS

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setSSLFipsRequired()
- MQConnectionFactory.getSSLFipsRequired()

## 값

### NO

TLS 연결이 IBM Java JSSE FIPS 제공자(IBMJSSEFIPS)에서 지원되지 않는 CipherSuite를 사용할 수 있습니다.

이는 기본값입니다. 프로그램에서 false를 사용하십시오.

### YES

TLS 연결이 IBMJSSEFIPS에서 지원되는 CipherSuite를 사용해야 합니다.

프로그램에서 true를 사용하십시오.

## SSLPEERNAME

TLS의 경우, 큐 관리자가 제공하는 것과 일치해야 하는 식별 이름 스켈레톤입니다.

## 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: SSLPEERNAME

JMS 관리 도구 짧은 이름: SPEER

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setSSLPeerName()
- MQConnectionFactory.getSSLPeerName()

## 값

### 널

이는 기본값입니다. 자세한 정보는 [JMS 오브젝트의 TLS 특성을 참조하십시오](#).

## SSLRESETCOUNT

TLS의 경우, 암호화에 사용되는 비밀 키가 재협상되기 전에 연결을 통해 송신 및 수신되는 바이트 합계입니다.

## 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: SSLRESETCOUNT

JMS 관리 도구 짧은 이름: SRC

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setSSLResetCount()
- MQConnectionFactory.getSSLResetCount()

### 값

0

0 또는 999, 999, 999 이하의 양의 정수. 이는 기본값입니다. 자세한 정보는 [JMS 오브젝트의 TLS 특성을 참조하십시오](#).

## STATREFRESHINT

구독자와 큐 관리자의 연결이 끊길 때 발견한 장기 실행 트랜잭션의 새로 고치기 간격(밀리초)입니다.

이 특성은 SUBSTORE의 값이 QUEUE인 경우에만 관련이 있습니다.

### 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: STATREFRESHINT

JMS 관리 도구 짧은 이름: SRI

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setStatusRefreshInterval()
- MQConnectionFactory.getStatusRefreshInterval()

### 값

60000

임의의 양의 정수가 기본값이 될 수 있습니다. 자세한 정보는 [JMS 오브젝트의 TLS 특성을 참조하십시오](#).

## SUBSTORE

IBM MQ classes for JMS에서 활성 구독과 관련된 지속 데이터를 저장하는 위치입니다.

### 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: SUBSTORE

JMS 관리 도구 짧은 이름: SS

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setSubscriptionStore()
- MQConnectionFactory.getSubscriptionStore()

## 값

### **BROKER**

구독 세부사항을 저장하기 위해 브로커 기반 구독 저장소를 사용합니다. 이 값은 관리 도구의 기본값입니다. 프로그램에는 WMQConstants.WMQ\_SUBSTORE\_BROKER를 사용하십시오.

### **MIGRATE**

큐 기반 구독 저장소에서 브로커 기반 구독 저장소로 구독 정보를 전송합니다. 프로그램에는 WMQConstants.WMQ\_SUBSTORE\_MIGRATE를 사용하십시오.

### **큐**

구독 세부사항을 저장하기 위해 큐 기반 구독 저장소를 사용합니다. 프로그램에는 WMQConstants.WMQ\_SUBSTORE\_QUEUE를 사용하십시오.

## **SYNCPOINTALLGETS**

이 특성은 모든 Get이 동기점에서 수행되는지 여부를 판별합니다.

### **적용 가능한 오브젝트**

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: SYNCPOINTALLGETS

JMS 관리 도구 짧은 이름: SPAG

### **프로그래밍 방식 액세스**

Setter/Getter

- MQConnectionFactory.setSyncpointAllGets()
- MQConnectionFactory.getSyncpointAllGets()

## 값

### **아니오**

이는 기본값입니다.

### **예**

## **TARGCLIENT**

이 특성은 대상 애플리케이션과 정보를 교환하는 데 IBM MQ RFH2 형식이 사용되는지 여부를 판별합니다.

### **적용 가능한 오브젝트**

큐, 토픽

JMS 관리 도구 긴 이름: TARGCLIENT

JMS 관리 도구 짧은 이름: TC

### **프로그래밍 방식 액세스**

Setter/Getter

- MQDestination.setTargetClient()
- MQDestination.getTargetClient()



## 값

### JMS

메시지 대상이 JMS 애플리케이션입니다. 이 값은 관리 도구의 기본값입니다.

프로그램에는 `WMQConstants.WMQ_CLIENT_JMS_COMPLIANT`를 사용하십시오.

### MQ

메시지 대상이 JMS IBM MQ 이외의 애플리케이션입니다.

프로그램에는 `WMQConstants.WMQ_CLIENT_NONJMS_MQ`를 사용하십시오.

## TARGCLIENTMATCHING

이 특성은 수신 메시지에 `MQRFH2` 헤더가 있는 경우에 한해 수신 메시지의 `JMSReplyTo` 헤더 필드로 식별된 큐에 송신된 응답 메시지에 `MQRFH2` 헤더가 있는지 여부를 판별합니다.

### 적용 가능한 오브젝트

`ConnectionFactory`, `QueueConnectionFactory`, `XAConnectionFactory`, `XAQueueConnectionFactory`

JMS 관리 도구 긴 이름: `TARGCLIENTMATCHING`

JMS 관리 도구 짧은 이름: `TCM`

### 프로그래밍 방식 액세스

Setter/Getter

- `MQConnectionFactory.setTargetClientMatching()`
- `MQConnectionFactory.getTargetClientMatching()`

## 값

### YES

수신 메시지에 `MQRFH2` 헤더가 없으면 메시지의 `JMSReplyTo` 헤더 필드에서 파생된 큐 오브젝트의 `TARGCLIENT` 특성이 `MQ`로 송신됩니다. 메시지에 `MQRFH2` 헤더가 있으면 `TARGCLIENT` 특성이 대신 `JMS`로 설정됩니다. 이 값은 관리 도구의 기본값입니다.

프로그램에는 `true`를 사용하십시오.

### NO

수신 메시지의 `JMSReplyTo` 헤더 필드에서 파생된 큐 오브젝트의 `TARGCLIENT` 특성이 항상 `JMS`로 설정됩니다.

프로그램에는 `false`를 사용하십시오.

## TEMPMODEL

JMS 임시 큐가 작성되는 모델 큐의 이름입니다.

### 적용 가능한 오브젝트

`ConnectionFactory`, `QueueConnectionFactory`, `XAConnectionFactory`, `XAQueueConnectionFactory`

JMS 관리 도구 긴 이름: `TEMPMODEL`

JMS 관리 도구 짧은 이름: `TM`

### 프로그래밍 방식 액세스

Setter/Getter

- `MQConnectionFactory.setTemporaryModel()`
- `MQConnectionFactory.getTemporaryModel()`

## 값

### SYSTEM.DEFAULT.MODEL.QUEUE

임의의 문자열이 기본값일 수 있습니다.

## TEMPQPREFIX

IBM MQ 동적 큐의 이름을 형성하는 데 사용하는 접두부.

### 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory

JMS 관리 도구 긴 이름: TEMPQPREFIX

JMS 관리 도구 짧은 이름: TQP

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setTempQPrefix()
- MQConnectionFactory.getTempQPrefix()

## 값

""(빈 문자열)

사용되는 접두부는 CSQ.\*(z/OS) 및 AMQ.\*(그 외 모든 플랫폼)입니다. 이 값이 기본값입니다.

### queue prefix

큐 접두부는 IBM MQ 오브젝트 디스크립터 (구조 MQOD) 에서 *DynamicQName* 필드의 콘텐츠를 형성하기 위한 규칙을 준수하는 문자열이지만 마지막 공백이 아닌 문자는 별표여야 합니다.

## TEMPTOPICPREFIX

임시 토픽을 작성할 때 JMS 는 " TEMP /TEMPTOPICPREFIX/unique\_id " 양식의 토픽 문자열을 생성하거나 이 특성이 기본값으로 남아 있는 경우 " TEMP /unique\_id " 만 생성합니다. 비어 있지 않은 TEMPTOPICPREFIX를 지정하면 이 연결 하에 작성된 임시 토픽의 구독자를 위한 관리 큐 작성을 위해 특정 모델 큐가 정의될 수 있습니다.

### 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: TEMPTOPICPREFIX

JMS 관리 도구 짧은 이름: TTP

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setTempTopicPrefix()
- MQConnectionFactory.getTempTopicPrefix()

## 값

IBM MQ 토픽 문자열에 유효한 문자만으로 구성된 널이 아닌 문자열입니다. 기본값은 ""(빈 문자열)입니다.

## TOPIC

JMS 토픽 목적지의 이름으로, 큐 관리자는 이 값을 발행 또는 구독의 토픽 문자열로 사용합니다.

## 적용 가능한 오브젝트

토픽

JMS 관리 도구 긴 이름: TOPIC

JMS 관리 도구 짧은 이름: TOP

## 값

### 임의의 문자열

올바른 IBM MQ 토픽 문자열을 만드는 문자열. IBM MQ를 WebSphere Application Server에서 메시징 제공자로 사용하는 경우, WebSphere Application Server 내에서 관리 목적으로 인식되는 토픽의 이름과 일치하는 값을 지정하십시오.

### 관련 참조

[토픽 문자열](#)

## TRANSPORT

큐 관리자 또는 브로커에 대한 연결의 네이처입니다.

## 적용 가능한 오브젝트

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: TRANSPORT

JMS 관리 도구 짧은 이름: TRAN

## 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setTransportType()
- MQConnectionFactory.getTransportType()

## 값

### BIND

바인딩 모드에서 큐 관리자에 대한 연결에 적용됩니다. 이 값은 관리 도구의 기본값입니다.

프로그램에는 WMQConstants.WMQ\_CM\_BINDINGS를 사용하십시오.

### 클라이언트

클라이언트 모드에서 큐 관리자에 대한 연결에 적용됩니다.

프로그램에는 WMQConstants.WMQ\_CM\_CLIENT를 사용하십시오.

### DIRECT

HTTP 터널링을 사용하지 않는 브로커에 대한 실시간 연결에 적용됩니다.

프로그램에는 WMQConstants.WMQ\_CM\_DIRECT\_TCPIP를 사용하십시오.

### DIRECTHTTP

HTTP 터널링을 사용하는 브로커에 대한 실시간 연결에 적용됩니다. HTTP 1.0만 지원됩니다.

프로그램에는 WMQConstants.WMQ\_CM\_DIRECT\_HTTP를 사용하십시오.

### 관련 개념

[1814 페이지의 『IBM MQ classes for JMS 오브젝트 특성의 종속성』](#)  
일부 특성의 검증이 다른 특성의 특정 값에 좌우됩니다.

## WILDCARDFORMAT

이 특성은 사용되는 와일드카드 구문 버전을 판별합니다.

### 적용 가능한 오브젝트

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XATopicConnectionFactory

JMS 관리 도구 긴 이름: WILDCARDFORMAT

JMS 관리 도구 짧은 이름: WCFMT

### 프로그래밍 방식 액세스

Setter/Getter

- MQConnectionFactory.setWildCardFormat()
- MQConnectionFactory.getWildCardFormat()

### 값

#### TOPIC\_ONLY

브로커 버전 2에 사용된 토픽 레벨 와일드카드만 인식합니다. 이 값은 관리 도구의 기본값입니다.

프로그램에는 WMQConstants.WMQ\_WILDCARD\_TOPIC\_ONLY를 사용하십시오.

#### CHAR\_ONLY

브로커 버전 1에 사용될 때 문자 와일드카드만 인식합니다.

프로그램에는 WMQConstants.WMQ\_WILDCARD\_CHAR\_ONLY를 사용하십시오.

## ENCODING 특성

ENCODING 특성은 12가지 조합이 가능한 3개의 하위 특성으로 구성됩니다.

ENCODING 특성에 사용하는 올바른 값은 3개의 하위 특성으로 구성됩니다.

#### 정수 인코딩

정상 또는 역방향입니다.

#### 10진수 인코딩

정상 또는 역방향입니다.

#### 부동 소수점 인코딩

IEEE 일반, IEEE 예약 또는 z/OS

ENCODING 특성은 다음 구문을 사용하여 3자 문자열로 표시됩니다.

```
{N|R}{N|R}{N|R}3
```

이 문자열에서:

- N은 정상을 나타냅니다.
- R은 역방향을 나타냅니다.
- 3은 z/OS를 나타냅니다.
- 첫 번째 문자는 정수 인코딩을 나타냅니다.
- 두 번째 문자는 10진수 인코딩을 나타냅니다.
- 세 번째 문자는 부동 소수점 인코딩을 나타냅니다.

ENCODING 특성에 대해 12개의 가능한 값 세트를 제공합니다.

추가 값인 문자열 NATIVE도 있으며, 이 값은 Java 플랫폼에 적절한 인코딩 값을 설정합니다.

다음 예는 올바른 ENCODING 조합을 보여줍니다.

```
ENCODING (NRR)
ENCODING (NATIVE)
ENCODING (RR3)
```

## JMS 오브젝트의 TLS 특성

SSLCIPHERSUITE 특성을 사용하여 TLS(Transport Layer Security) 암호화를 사용하도록 설정합니다. 그런 다음 몇 가지 다른 특성을 사용하여 TLS 암호화의 특성을 변경할 수 있습니다.

TRANSPORT(CLIENT)를 지정할 때 SSLCIPHERSUITE 특성을 사용하여 TLS 암호화 통신을 사용하도록 설정할 수 있습니다. 이 특성을 JSSE 제공자에서 제공하는 올바른 CipherSuite로 설정하십시오. 이 값이 CHANNEL 특성으로 지정된 SVRCONN 채널의 CipherSpec과 일치해야 합니다.

그러나 CipherSpec(SVRCONN 채널에 지정됨)과 CipherSuites(ConnectionFactory 오브젝트에 지정됨)는 다른 이름 지정 체계를 사용하여 동일한 TLS 암호화 알고리즘을 나타냅니다. 인식된 CipherSpec 이름이 SSLCIPHERSUITE 특성에 지정된 경우, JMSAdmin은 경고를 발행하고 CipherSpec을 동등한 CipherSuite에 맵핑합니다. IBM MQ 및 JMSAdmin에서 인식하는 CipherSpecs 목록은 [IBM MQ classes for JMS의 TLS CipherSpecs 및 CipherSuites](#)의 내용을 참조하십시오.

IBM Java JSSE FIPS 제공자(IBMJSSEFIPS)에서 지원하는 CipherSuite를 사용하기 위해 연결이 필요한 경우에는 연결 팩토리의 SSLFIPSREQUIRED 특성을 YES로 설정하십시오. 이 특성의 기본값은 NO로, 연결이 지원되는 CipherSuite를 사용할 수 있음을 의미합니다. SSLCIPHERSUITE를 설정하지 않으면 특성이 무시됩니다.

SSLPEERNAME은 채널 정의에 설정할 수 있는 SSLPEER 매개변수의 형식과 일치합니다. 쉼표 또는 세미콜론으로 구분된 속성 이름-값 쌍의 목록입니다. 예를 들면, 다음과 같습니다.

```
SSLPEERNAME(CN=QMGR.*, OU=IBM, OU=WEBSPPHERE)
```

이름 및 값 세트가 식별 이름을 구성합니다. 식별 이름 및 IBM MQ에서의 사용에 대한 자세한 정보는 [IBM MQ 보안](#)의 내용을 참조하십시오.

제공된 예에서는 연결 시 서버에 표시된 식별 인증서를 검사합니다. 연결에 성공하려면 인증서에 QMGR로 시작하는 공통 이름이 있어야 합니다. 최소한 두 개의 조직 단위 이름이 있어야 하며 첫 번째는 IBM 이고 두 번째는 WEBSPPHERE입니다. 검사는 대소문자를 구분하지 않습니다.

SSLPEERNAME이 설정되지 않으면 검사가 수행되지 않습니다. SSLCIPHERSUITE를 설정하지 않으면 SSLPEERNAME이 무시됩니다.

SSLCRL 특성은 0개 이상의 CRL(인증서 폐기 목록) 서버를 지정합니다. 이 특성을 사용하려면 Java 2 v1.4에 JVM이 필요합니다. 이것은 쉼표로 구분된 양식 항목 목록입니다.

```
ldap:// hostname:[ port ]
```

선택적으로 뒤에 단일 /가 옵니다. *port* 를 생략하면 기본 LDAP 포트 389가 사용됩니다. 연결 시 서버에서 제공하는 TLS 인증서는 지정된 CRL 서버를 기준으로 검사됩니다. CRL 보안에 대한 자세한 정보는 [IBM MQ 보안](#)의 내용을 참조하십시오.

SSLCRL이 설정되지 않으면 검사가 수행되지 않습니다. SSLCIPHERSUITE를 설정하지 않으면 SSLCRL이 무시됩니다.

SSLRESETCOUNT 특성은 암호화에 사용되는 비밀 키가 재협상되기 전에 연결을 통해 송신 및 수신되는 바이트 합계를 나타냅니다. 송신된 바이트 수는 암호화 전의 바이트 수이며, 수신된 바이트 수는 복호화 후의 바이트 수입니다. 바이트 수에는 IBM MQ classes for JMS에 의해 송신되거나 수신된 제어 정보도 포함됩니다.

예를 들어, 4MB의 데이터가 이동된 이후 재협상된 비밀 키로 TLS 사용 MQI 채널에서 연결을 작성하는 데 사용될 수 있는 ConnectionFactory 오브젝트를 구성하려면 JMSAdmin에 대해 다음 명령을 실행하십시오.

```
ALTER CF(my.cf) SSLRESETCOUNT(4194304)
```

SSLRESETCOUNT 값이 기본값인 0인 경우 비밀 키가 재협상되지 않습니다. SSLCIPHERSUITE를 설정하지 않으면 SSLRESETCOUNT 특성이 무시됩니다.

## IBM MQ Message Service Client (XMS) for .NET 참조

이 참조 절에서는 IBM MQ Message Service Client (XMS) for .NET(XMS .NET) 클래스 인터페이스 및 XMS에서 정의한 오브젝트 특성에 대한 정보를 제공합니다.

### .NET 인터페이스

이 절에서는 .NET 클래스 인터페이스 및 해당 특성과 메소드에 대해 설명합니다.

다음 표에는 IBM.XMS 네임스페이스에 정의된 인터페이스가 요약되어 있습니다.

표 871. .NET 클래스 인터페이스의 요약	
인터페이스	설명
<a href="#">1868 페이지의 『IBytesMessage』</a>	바이트 메시지는 본문이 바이트 스트림을 구성하는 메시지입니다.
<a href="#">1877 페이지의 『IConnection』</a>	Connection 오브젝트는 애플리케이션의 메시징 서버에 대한 활성 연결을 나타냅니다.
<a href="#">1880 페이지의 『IConnectionFactory』</a>	애플리케이션이 연결 팩토리를 사용하여 연결을 작성합니다.
<a href="#">1881 페이지의 『IConnectionMetaData』</a>	ConnectionMetaData 오브젝트는 연결에 대한 정보를 제공합니다.
<a href="#">1882 페이지의 『IDestination』</a>	대상은 애플리케이션이 메시지를 전송하는 위치이거나 애플리케이션이 메시지를 받는 소스이거나 둘 다입니다.
<a href="#">1883 페이지의 『ExceptionListener』</a>	애플리케이션이 예외 리스너를 사용하여 연결 문제점을 비동기적으로 알립니다.
<a href="#">1883 페이지의 『IllegalStateException』</a>	XMS에서는 애플리케이션이 잘못되거나 부적절한 시간에 메소드를 호출하거나 XMS이(가) 요청된 조작에 적절한 상태가 아닌 경우 이 예외를 처리합니다.
<a href="#">1884 페이지의 『InitialContext』</a>	애플리케이션이 InitialContext 오브젝트를 사용하여 관리 오브젝트의 저장소에서 검색되는 오브젝트 정의에서 오브젝트를 작성합니다.
<a href="#">1886 페이지의 『InvalidClientIDException』</a>	애플리케이션이 연결을 위해 클라이언트 ID를 설정하려 했지만 클라이언트 ID가 유효하지 않거나 이미 사용 중이면 XMS에서 이 예외가 처리됩니다.
<a href="#">1886 페이지의 『InvalidDestinationException』</a>	XMS에서는 애플리케이션이 유효하지 않은 대상을 지정한 경우에 이 예외 처리를 합니다.
<a href="#">1886 페이지의 『InvalidSelectorException』</a>	애플리케이션에서 구문이 유효하지 않은 메시지 선택자 표현식을 제공하면 XMS에서 이 예외를 처리합니다.
<a href="#">1886 페이지의 『IMapMessage』</a>	맵 메시지는 본문이 이름-값 쌍 세트로 구성된 메시지이며 각 값에는 연관된 데이터 유형이 있습니다.
<a href="#">1895 페이지의 『IMessage』</a>	Message 오브젝트는 애플리케이션이 송신하거나 수신하는 메시지를 나타냅니다. IMessage는 IMapMessage와 같은 메시지 클래스의 슈퍼클래스입니다.

표 871. .NET 클래스 인터페이스의 요약 (계속)

인터페이스	설명
1901 페이지의 『 <a href="#">IMessageConsumer</a> 』	애플리케이션은 메시지 이용자를 사용하여 대상에 전송된 메시지를 받습니다.
1903 페이지의 『 <a href="#">MessageEOFException</a> 』	애플리케이션이 바이트 메시지의 본문을 읽는 중에 XMS이(가) 바이트 메시지 스트림의 끝에 도달하면 XMS에서 이 예외를 처리합니다.
1903 페이지의 『 <a href="#">MessageFormatException</a> 』	XMS에 올바르지 않은 형식의 메시지가 표시되면 XMS에서 이 예외가 처리됩니다.
1903 페이지의 『 <a href="#">IMessageListener(위임)</a> 』	애플리케이션은 메시지 리스너를 사용하여 메시지를 비동기적으로 수신합니다.
1904 페이지의 『 <a href="#">MessageNotReadableException</a> 』	XMS에서는 애플리케이션이 쓰기 전용인 메시지 본문을 읽으려고 할 경우 이 예외 처리를 합니다.
1904 페이지의 『 <a href="#">MessageNotWritableException</a> 』	XMS에서는 애플리케이션이 쓰기 전용인 메시지 본문에 쓰려고 할 경우 이 예외 처리를 합니다.
1904 페이지의 『 <a href="#">IMessageProducer</a> 』	애플리케이션은 메시지 생성자를 사용하여 메시지를 대상에 전송합니다.
1909 페이지의 『 <a href="#">IObjectMessage</a> 』	오브젝트 메시지는 본문이 직렬화된 Java 또는 .NET 오브젝트로 구성되는 메시지입니다.
1910 페이지의 『 <a href="#">IPropertyContext</a> 』	IPropertyContext는 특성을 가져오고 설정하는 메소드를 포함하는 추상 수퍼클래스입니다. 이러한 메소드는 다른 클래스에서 상속합니다.
1919 페이지의 『 <a href="#">IQueueBrowser</a> 』	애플리케이션이 큐 브라우저를 사용하여 메시지를 제거하지 않고 큐에서 메시지를 찾습니다.
1921 페이지의 『 <a href="#">Requestor</a> 』	애플리케이션이 요청자를 사용하여 요청 메시지를 보내고 잠시 대기한 후 응답을 수신합니다.
1922 페이지의 『 <a href="#">ResourceAllocationException</a> 』	XMS에서는 XMS이(가) 메소드에 필요한 자원을 할당할 수 없는 경우에 이 예외 처리를 합니다.
1922 페이지의 『 <a href="#">SecurityException</a> 』	XMS는 애플리케이션을 인증하기 위해 제공된 사용자 ID 및 비밀번호가 거부되는 경우 이 예외를 처리합니다. XMS는 권한 검사에 실패하여 메소드가 완료되지 않은 경우에도 이 예외를 발생시킵니다.
1923 페이지의 『 <a href="#">ISession</a> 』	세션은 메시지 송신 및 수신을 위한 단일 스레드 컨텍스트입니다.
1933 페이지의 『 <a href="#">IStreamMessage</a> 』	스트림 메시지는 각 값에 연관된 데이터 유형이 있는 값 스트림으로 본문이 구성된 메시지입니다.
1941 페이지의 『 <a href="#">ITextMessage</a> 』	텍스트 메시지는 본문이 문자열로 구성된 메시지입니다.
1942 페이지의 『 <a href="#">TransactionInProgressException</a> 』	XMS에서는 트랜잭션이 진행 중이므로 유효하지 않은 조작을 애플리케이션에서 요청할 경우 이 예외 처리를 합니다.
1942 페이지의 『 <a href="#">TransactionRolledBackException</a> 』	XMS에서는 애플리케이션이 현재 트랜잭션을 확약하기 위해 <code>Session.commit()</code> 를 호출하지만 트랜잭션이 롤백되는 경우에 이 예외를 처리합니다.

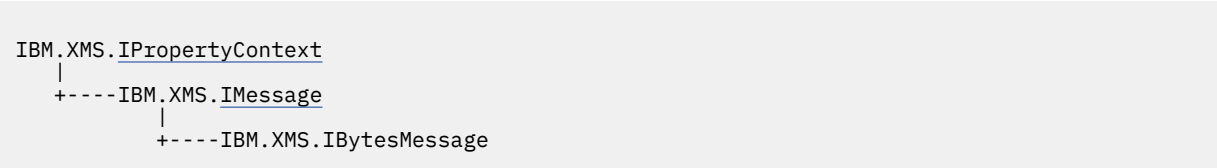
표 871. .NET 클래스 인터페이스의 요약 (계속)	
인터페이스	설명
XMSC	.NET의 경우 XMS 특성 이름 및 값은 이 클래스에서 공용 상수로 정의됩니다. 자세한 정보는 1945 페이지의 『XMS 오브젝트의 특성』의 내용을 참조하십시오.
<a href="#">1943 페이지의 『XMSEException』</a>	XMS이(가) .NET 메소드에 대한 호출을 처리하는 중에 오류를 감지하면 XMS에서 예외 처리합니다. 예외는 오류 정보를 캡슐화하는 오브젝트입니다.  다른 유형의 XMS 예외가 있으며, XMSEException 오브젝트는 하나의 예외 유형일 뿐입니다. 그러나 XMSEException 클래스는 다른 XMS 예외 클래스의 슈퍼클래스입니다. XMS는 다른 유형의 예외가 적합하지 않은 경우에 XMSEException 오브젝트를 처리합니다.
<a href="#">1943 페이지의 『XMSFactoryFactory』</a>	애플리케이션에서 관리 오브젝트를 사용하지 않을 경우 이 클래스를 사용하여 연결 팩토리, 큐, 토픽을 작성하십시오.

각 메소드의 정의는 XMS가 메소드에 대한 호출을 처리하는 동안 오류를 발견할 경우 리턴되는 예외 코드를 나열합니다. 각 예외 코드는 이름 지정된 상수로 표시됩니다. 이 상수는 해당 예외를 가지고 있습니다.

## IBytesMessage

바이트 메시지는 본문이 바이트 스트림을 구성하는 메시지입니다.

상속 계층 구조:



## .NET 특성

*BodyLength* - 본문 길이 가져오기

인터페이스:

```

Int64 BodyLength
{
    get;
}

```

메시지 본문이 읽기 전용인 경우 메시지 본문의 길이를 바이트 단위로 가져옵니다.

메시지 읽기 커서의 현재 위치에 상관없이 리턴값은 전체 본문의 길이입니다.

예외:

- XMSEException
- MessageNotReadableException

## 메소드



*ReadBoolean* - 부울 값 읽기

인터페이스:

```
Boolean ReadBoolean();
```

바이트 메시지 스트림에서 부울 값을 읽습니다.

매개변수:

없음

리턴값:

읽은 부울 값입니다.

예외:

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadSignedByte* - 바이트 읽기

인터페이스:

```
Int16 ReadSignedByte();
```

바이트 메시지 스트림의 다음 바이트를 부호 있는 8비트 정수로 읽습니다.

매개변수:

없음

리턴값:

읽은 바이트입니다.

예외:

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadBytes* - 바이트 읽기

인터페이스:

```
Int32 ReadBytes(Byte[] array);  
Int32 ReadBytes(Byte[] array, Int32 length);
```

커서의 현재 위치에서 시작하여 바이트 메시지 스트림에서 바이트 배열을 읽습니다..

매개변수:

**array (output)**

읽은 바이트 배열을 포함하는 버퍼입니다. 호출하기 전에 스트림에서 읽을 나머지 바이트 수가 버퍼 길이보다 크거나 같을 경우 버퍼가 가득 채워집니다. 그렇지 않을 경우 버퍼가 나머지 모든 바이트로 일부만 채워집니다.

입력 시 널 포인터를 지정하면 메소드가 바이트를 읽지 않고 건너뛵니다. 호출하기 전에 스트림에서 읽을 나머지 바이트 수가 버퍼 길이보다 크거나 같을 경우 건너뛴 바이트 수는 버퍼 길이와 같습니다. 그렇지 않을 경우 나머지 모든 바이트를 건너뛵니다. 커서는 바이트 메시지 스트림의 다음 위치에 남아 있습니다.

**length (input)**

바이트 단위의 버퍼 길이입니다.

**리턴값:**

버퍼로 읽어 들인 바이트 수입니다. 버퍼의 일부만 채워진 경우 값이 버퍼 길이보다 작으며, 이는 더 이상 읽을 바이트가 남아 있지 않음을 나타냅니다. 호출 전에 스트림에서 읽을 바이트가 남아 있지 않은 경우 값은 `XMSC_END_OF_STREAM`입니다.

입력 시 널 포인터를 지정하면 이 메소드가 값을 리턴하지 않습니다.

**예외:**

- `XMSEException`
- `MessageNotReadableException`

*ReadChar* - 문자 읽기**인터페이스:**

```
Char ReadChar();
```

바이트 메시지 스트림의 다음 2바이트를 문자로 읽습니다.

**매개변수:**

없음

**리턴값:**

읽은 문자입니다.

**예외:**

- `XMSEException`
- `MessageNotReadableException`
- `MessageEOFException`

*ReadDouble* - 배 정밀도 부동 소수점 수 읽기**인터페이스:**

```
Double ReadDouble();
```

바이트 메시지 스트림의 다음 8바이트를 배정밀도 부동 소수점 수로 읽습니다.

**매개변수:**

없음

**리턴값:**

읽은 배 정밀도 부동 소수점 수입니다.

**예외:**

- `XMSEException`
- `MessageNotReadableException`
- `MessageEOFException`

*ReadFloat* - 부동 소수점 수 읽기**인터페이스:**

```
Single ReadFloat();
```

바이트 메시지 스트림의 다음 4바이트를 부동 소수점 수로 읽습니다.

**매개변수:**

없음

**리턴값:**

읽은 부동 소수점 수입니다.

**예외:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadInt* - 정수 읽기

**인터페이스:**

```
Int32 ReadInt();
```

바이트 메시지 스트림의 다음 4바이트를 부호 있는 32비트 정수로 읽습니다.

**매개변수:**

없음

**리턴값:**

읽은 정수입니다.

**예외:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadLong* - Long 정수 읽기

**인터페이스:**

```
Int64 ReadLong();
```

바이트 메시지 스트림의 다음 8바이트를 부호 있는 64비트 정수로 읽습니다.

**매개변수:**

없음

**리턴값:**

읽은 Long 정수입니다.

**예외:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadShort* - Short 정수 읽기

**인터페이스:**

```
Int16 ReadShort();
```

바이트 메시지 스트림의 다음 2바이트를 부호 있는 16비트 정수로 읽습니다.

**매개변수:**

없음

**리턴값:**

읽은 Short 정수입니다.

**예외:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadByte* - 부호 없는 바이트 읽기

**인터페이스:**

```
Byte ReadByte();
```

바이트 메시지 스트림의 다음 바이트를 부호 있는 8비트 정수로 읽습니다.

**매개변수:**

없음

**리턴값:**

읽은 바이트입니다.

**예외:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadUnsignedShort* - 부호 없는 *Short* 정수 읽기

**인터페이스:**

```
Int32 ReadUnsignedShort();
```

바이트 메시지 스트림의 다음 2바이트를 부호 있는 16비트 정수로 읽습니다.

**매개변수:**

없음

**리턴값:**

읽은 부호 없는 *Short* 정수입니다.

**예외:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadUTF* - UTF 문자열 읽기

**인터페이스:**

```
String ReadUTF();
```

바이트 메시지 스트림에서 UTF-8로 인코딩된 문자열을 읽습니다.

**참고:** *ReadUTF()*를 호출하기 전에 버퍼의 커서가 바이트 메시지 스트림의 시작 지점에 있는지 확인하십시오.

**매개변수:**

없음

**리턴값:**

읽은 문자열을 캡슐화하는 *String* 오브젝트입니다.

**예외:**

- XMSEException

- MessageNotReadableException
- MessageEOFException

*Reset* - 재설정

인터페이스:

```
void Reset();
```

메시지 본문을 읽기 전용 모드로 설정하고 바이트 메시지 스트림의 시작 부분에서 커서 위치를 변경합니다.

매개변수:

없음

리턴값:

Void

예외:

- XMSEException
- MessageNotReadableException

*WriteBoolean* - 부울 값 쓰기

인터페이스:

```
void WriteBoolean(Boolean value);
```

부울 값을 바이트 메시지 스트림에 씁니다.

매개변수:

**value (input)**

기록할 부울 값입니다.

리턴값:

Void

예외:

- XMSEException
- MessageNotWritableException

*WriteByte* - 바이트 쓰기

인터페이스:

```
void WriteByte(Byte value);
void WriteSignedByte(Int16 value);
```

바이트를 바이트 메시지 스트림에 씁니다.

매개변수:

**value (input)**

기록할 바이트입니다.

리턴값:

Void

예외:

- XMSEException
- MessageNotWritableException

## WriteBytes - 바이트 쓰기

### 인터페이스:

```
void WriteBytes(Byte[] value);
```

바이트 배열을 바이트 메시지 스트림에 씁니다.

### 매개변수:

#### **value (input)**

기록할 바이트 배열입니다.

### 리턴값:

Void

### 예외:

- XMSEException
- MessageNotWritableException

## WriteBytes - 부분 바이트 배열 쓰기

### 인터페이스:

```
void WriteBytes(Byte[] value, int offset, int length);
```

지정된 길이로 정의된 대로 바이트의 부분 배열을 바이트 메시지 스트림에 씁니다.

### 매개변수:

#### **value (input)**

기록할 바이트 배열입니다.

#### **offset (input)**

기록할 바이트 배열의 시작 지점입니다.

#### **length (input)**

기록할 바이트 수입니다.

### 리턴값:

Void

### 예외:

- XMSEException
- MessageNotWritableException

## WriteChar - 문자 쓰기

### 인터페이스:

```
void WriteChar(Char value);
```

바이트 메시지 스트림에 문자를 2바이트로 쓰며, 상위 바이트가 먼저입니다.

### 매개변수:

#### **value (input)**

기록할 문자입니다.

### 리턴값:

Void

### 예외:

- XMSEException

- MessageNotWritableException

*WriteDouble* - 배 정밀도 부동 소수점 수 쓰기

인터페이스:

```
void WriteDouble(Double value);
```

배정밀도 부동 소수점 수를 long 정수로 변환하고 long 정수를 바이트 메시지 스트림에 8바이트로 쓰며, 상위 바이트 먼저 씁니다..

매개변수:

**value (input)**

기록할 배 정밀도 부동 소수점 수입니다.

리턴값:

Void

예외:

- XMSEException
- MessageNotWritableException

*WriteFloat* - 부동 소수점 수 쓰기

인터페이스:

```
void WriteFloat(Single value);
```

부동 소수점 수를 정수로 변환하고 정수를 4바이트로 바이트 메시지 스트림에 쓰며, 상위 바이트 먼저 씁니다.

매개변수:

**value (input)**

기록할 부동 소수점 수입니다.

리턴값:

Void

예외:

- XMSEException
- MessageNotWritableException

*WriteInt* - 정수 쓰기

인터페이스:

```
void WriteInt(Int32 value);
```

바이트 메시지 스트림에 정수를 4바이트로 쓰고, 상위 바이트를 먼저 씁니다.

매개변수:

**value (input)**

기록할 정수입니다.

리턴값:

Void

예외:

- XMSEException
- MessageNotWritableException

## WriteLong - Long 정수 쓰기

### 인터페이스:

```
void WriteLong(Int64 value);
```

바이트 메시지 스트림에 long 정수를 8바이트로 쓰며, 상위 바이트를 먼저 씁니다.

### 매개변수:

#### **value (input)**

기록할 Long 정수입니다.

### 리턴값:

Void

### 예외:

- XMSEException
- MessageNotWritableException

## WriteObject - 오브젝트 쓰기

### 인터페이스:

```
void WriteObject(Object value);
```

지정된 오브젝트를 바이트 메시지 스트림에 씁니다.

### 매개변수:

#### **value (input)**

기록할 오브젝트이며, 원시 유형에 대한 참조여야 합니다.

### 리턴값:

Void

### 예외:

- XMSEException
- MessageNotWritableException

## WriteShort - Short 정수 쓰기

### 인터페이스:

```
void WriteShort(Int16 value);
```

바이트 메시지 스트림에 short 정수를 2바이트로 쓰며, 상위 바이트를 먼저 씁니다.

### 매개변수:

#### **value (input)**

기록할 Short 정수입니다.

### 리턴값:

Void

### 예외:

- XMSEException
- MessageNotWritableException



WriteUTF - UTF 문자열 쓰기

인터페이스:

```
void WriteUTF(String value);
```

바이트 메시지 스트림에 UTF-8로 인코딩된 문자열을 씁니다.

매개변수:

**value (input)**

기록할 문자열을 캡슐화하는 String 오브젝트입니다.

리턴값:

Void

예외:

- XMSEException
- MessageNotWritableException

### 상속된 특성 및 메소드

다음 특성은 IMessage 인터페이스에서 상속됩니다.

[JMSCorrelationID](#), [JMSDeliveryMode](#), [JMSDestination](#), [JMSExpiration](#), [JMSMessageID](#), [JMSPriority](#), [JMSRe교부](#), [JMSReplyTo](#), [JMSTim환원](#), [JMSType](#), 특성

다음 메소드는 IMessage 인터페이스에서 상속됩니다.

[clearBody](#), [clearProperties](#), [PropertyExists](#)

다음 메소드는 IPropertyContext 인터페이스에서 상속됩니다.

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

## IConnection

Connection 오브젝트는 애플리케이션의 메시징 서버에 대한 활성 연결을 나타냅니다.

상속 계층 구조:

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IConnection
```

Connection 오브젝트의 XMS 정의 특성 목록은 [1946 페이지의 『Connection 특성』](#)의 내용을 참조하십시오.

### .NET 특성

ClientID - 클라이언트 ID 가져오기 및 설정

인터페이스:

```
String ClientID
{
    get;
    set;
}
```

연결에 대한 클라이언트 ID를 가져오고 설정합니다.

클라이언트 ID는 관리자가 `ConnectionFactory`에 미리 정의하거나 `ClientID`를 설정하여 지정할 수 있습니다.

클라이언트 ID는 발행/구독 도메인에서 지속 가능한 구독을 지원하는 데만 사용되며 포인트-투-포인트 도메인에서는 무시됩니다.

애플리케이션이 연결에 대한 클라이언트 ID를 설정할 경우 애플리케이션은 연결이 작성된 후 연결에서 다른 작업을 수행하기 전에 바로 클라이언트 ID를 설정해야 합니다. 이 시점 이후에 애플리케이션에서 클라이언트 ID를 설정하려고 하면 호출에서 `IllegalStateException` 예외가 발생합니다.

이 특성은 실시간 브로커 연결에는 유효하지 않습니다.

#### 예외:

- `XMSEException`
- `IllegalStateException`
- `InvalidClientIDException`

*ExceptionListener* - 예외 리스너 가져오기 및 설정

#### 인터페이스:

```
ExceptionListener ExceptionListener
{
    get;
    set;
}
```

연결에 등록된 예외 리스너를 가져오고 연결에 예외 리스너를 등록합니다.

연결에 예외 리스너가 등록되어 있지 않을 경우 이 메소드가 널을 리턴합니다. 예외 리스너가 이미 연결에 등록되어 있을 경우 예외 리스너 대신 널을 지정하여 등록을 취소할 수 있습니다.

예외 리스너 사용에 대한 자세한 정보는 [.NET에서 메시지 및 예외 리스너 사용](#)을 참조하십시오.

#### 예외:

- `XMSEException`

*Metadata* - 메타데이터 가져오기

#### 인터페이스:

```
IConnectionMetaData MetaData
{
    get;
}
```

연결에 대한 메타데이터를 가져옵니다.

#### 예외:

- `XMSEException`

## 메소드

*Close* - 연결 닫기

#### 인터페이스:

```
void Close();
```

연결을 닫습니다.

애플리케이션이 이미 닫힌 연결을 닫으려는 경우 호출이 무시됩니다.

매개변수:

없음

리턴값:

Void

예외:

- XMSEException

CreateSession - 세션 작성

인터페이스:

```
ISession CreateSession(Boolean transacted,  
                        AcknowledgeMode acknowledgeMode);
```

세션을 작성합니다.

매개변수:

**transacted (input)**

True 값을 설정하면 세션이 트랜잭션되며, False 값을 설정하면 세션이 트랜잭션되지 않습니다.

브로커에 대한 실시간 연결의 경우 값은 False여야 합니다.

**acknowledgeMode (input)**

애플리케이션이 수신한 메시지를 수신확인하는 방법을 표시합니다. AcknowledgeMode 열거자에 있는 다음 값 중 하나를 사용해야 합니다.

AcknowledgeMode.AutoAcknowledge

AcknowledgeMode.ClientAcknowledge

AcknowledgeMode.DupsOkAcknowledge

브로커에 대한 실시간 연결의 경우 값은 AcknowledgeMode.AutoAcknowledge 또는 AcknowledgeMode.DupsOkAcknowledge여야 합니다.

트랜잭션된 세션인 경우 이 매개변수는 무시됩니다. 수신확인 모드에 대한 자세한 정보는 [메시지 수신확인을 참조하십시오](#).

리턴값:

Session 오브젝트입니다.

예외:

- XMSEException

Start - 연결 시작

인터페이스:

```
void Start();
```

연결에 대한 수신 메시지 전달을 시작하거나 다시 시작합니다. 연결이 이미 시작된 경우에는 호출이 무시됩니다.

매개변수:

없음

리턴값:

Void

예외:

- XMSEException

Stop - 연결 중지

인터페이스:

```
void Stop();
```

연결에 대한 수신 메시지 전달을 중지합니다. 연결이 이미 중지된 경우에는 호출이 무시됩니다.

매개변수:

없음

리턴값:

Void

예외:

- XMSEException

## 상속된 특성 및 메소드

다음 메소드는 `IPropertyContext` 인터페이스에서 상속됩니다.

`GetBooleanProperty`, `GetByteProperty`, `GetBytesProperty`, `GetCharProperty`, `GetDoubleProperty`, `GetFloatProperty`, `GetIntProperty`, `GetLongProperty`, `GetObjectProperty`, `GetShortProperty`, `GetStringProperty`, `SetBooleanProperty`, `SetByteProperty`, `SetBytesProperty`, `SetCharProperty`, `SetDoubleProperty`, `SetFloatProperty`, `SetIntProperty`, `SetLongProperty`, `SetObjectProperty`, `SetShortProperty`, `SetStringProperty`

## IConnectionFactory

애플리케이션이 연결 팩토리를 사용하여 연결을 작성합니다.

상속 계층 구조:

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IConnectionFactory
```

ConnectionFactory 오브젝트의 XMS 정의 특성 목록은 [1946 페이지의 『ConnectionFactory의 특성』](#)의 내용을 참조하십시오.

## 메소드

*CreateConnection* - 연결 팩토리 작성(기본 사용자 ID 사용)

인터페이스:

```
IConnection CreateConnection();
```

기본 특성을 사용하여 연결 팩토리를 작성합니다.

IBM MQ에 연결하는 중이고 XMSC\_USERID가 설정되지 않은 경우 큐 관리자는 기본적으로 로그인된 사용자의 사용자 ID를 사용합니다. 개별 사용자의 추가 연결 레벨 인증이 필요한 경우 IBM MQ에서 구성되는 클라이언트 인증 엑시트를 작성할 수 있습니다.

매개변수:

없음

예외:

- XMSEException

CreateConnection - 연결 작성(지정된 사용자 ID 사용)

인터페이스:

```
IConnection CreateConnection(String userId, String password);
```

지정된 사용자 ID를 사용하여 연결을 작성합니다.

IBM MQ에 연결하는 중이고 XMSC\_USERID가 설정되지 않은 경우 큐 관리자는 기본적으로 로그인된 사용자의 사용자 ID를 사용합니다. 개별 사용자의 추가 연결 레벨 인증이 필요한 경우 IBM MQ에서 구성되는 클라이언트 인증 엑시트를 작성할 수 있습니다.

연결은 중지됨 모드로 작성됩니다. 애플리케이션이 **Connection.start()**를 호출하기 전에는 메시지가 전달되지 않습니다.

매개변수:

**userID (input)**

애플리케이션 인증에 사용될 사용자 ID를 캡슐화하는 String 오브젝트입니다. 널을 제공하면 인증 없이 연결이 작성됩니다.

**password (input)**

애플리케이션 인증에 사용될 비밀번호를 캡슐화하는 String 오브젝트입니다. 널을 제공하면 인증 없이 연결이 작성됩니다.

리턴값:

Connection 오브젝트입니다.

예외:

- XMSException
- XMS\_X\_SECURITY\_EXCEPTION

### 상속된 특성 및 메소드

다음 메소드는 [IPropertyContext](#) 인터페이스에서 상속됩니다.

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

### IConnectionMetaData

ConnectionMetaData 오브젝트는 연결에 대한 정보를 제공합니다.

상속 계층 구조:

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IConnectionMetaData
```

ConnectionMetaData 오브젝트의 XMS 정의 특성 목록은 [1950 페이지의 『ConnectionMetaData 특성』](#)의 내용을 참조하십시오.

### .NET 특성

JMSXPropertyNames - JMS 정의 메시지 특성 가져오기

인터페이스:

```
System.Collections.IEnumerator JMSXPropertyNames
{
```

```
    get;
}
```

연결에서 지원하는 JMS 정의 메시지 특성의 이름 목록을 리턴합니다.

브로커에 대한 실시간 연결에서는 JMS 정의 메시지 특성이 지원되지 않습니다.

**예외:**

- XMSEException

## 상속된 특성 및 메소드

다음 메소드는 `IPropertyContext` 인터페이스에서 상속됩니다.

`GetBooleanProperty`, `GetByteProperty`, `GetBytesProperty`, `GetCharProperty`, `GetDoubleProperty`, `GetFloatProperty`, `GetIntProperty`, `GetLongProperty`, `GetObjectProperty`, `GetShortProperty`, `GetStringProperty`, `SetBooleanProperty`, `SetByteProperty`, `SetBytesProperty`, `SetCharProperty`, `SetDoubleProperty`, `SetFloatProperty`, `SetIntProperty`, `SetLongProperty`, `SetObjectProperty`, `SetShortProperty`, `SetStringProperty`

## IDestination

대상은 애플리케이션이 메시지를 전송하는 위치이거나 애플리케이션이 메시지를 받는 소스이거나 둘 다입니다.

**상속 계층 구조:**

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IDestination
```

Destination 오브젝트의 XMS 정의 특성 목록은 [1950 페이지의 『Destination 특성』](#)의 내용을 참조하십시오.

## .NET 특성

*Name* - 목적지 이름 가져오기

**인터페이스:**

```
String Name
{
    get;
}
```

대상의 이름을 가져옵니다. 이 이름은 큐 이름이나 토픽 이름을 캡슐화하는 문자열입니다.

**예외:**

- XMSEException

*TypeId* - 목적지 유형 가져오기

**인터페이스:**

```
DestinationType TypeId
{
    get;
}
```

대상의 유형을 가져옵니다. 목적지 유형의 값은 다음 중 하나입니다.

`DestinationType.Queue`  
`DestinationType.Topic`

예외:

- XMSEException

## 상속된 특성 및 메소드

다음 메소드는 `IPropertyContext` 인터페이스에서 상속됩니다.

`GetBooleanProperty`, `GetByteProperty`, `GetBytesProperty`, `GetCharProperty`, `GetDoubleProperty`, `GetFloatProperty`, `GetIntProperty`, `GetLongProperty`, `GetObjectProperty`, `GetShortProperty`, `GetStringProperty`, `SetBooleanProperty`, `SetByteProperty`, `SetBytesProperty`, `SetCharProperty`, `SetDoubleProperty`, `SetFloatProperty`, `SetIntProperty`, `SetLongProperty`, `SetObjectProperty`, `SetShortProperty`, `SetStringProperty`

## ExceptionHandler

애플리케이션이 예외 리스너를 사용하여 연결 문제점을 비동기적으로 알립니다.

상속 계층 구조:

없음

애플리케이션이 메시지를 비동기적으로 이용할 때만 연결을 사용하고 그 외의 용도로는 사용하지 않는 경우 애플리케이션에서 연결과 관련된 문제점을 파악할 수 있는 유일한 방법은 예외 리스너를 사용하는 것입니다. 다른 상황에서 예외 리스너를 사용하면 XMS에 대한 다음 동기 호출까지 기다리는 대신 연결 문제점을 즉시 알 수 있습니다.

## 위임

`ExceptionHandler` - 예외 리스너

인터페이스:

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

애플리케이션에 연결 문제점을 알립니다.

이 위임을 구현하는 메소드는 연결에 등록할 수 있습니다.

예외 리스너 사용에 대한 자세한 정보는 [.NET에서 메시지 및 예외 리스너 사용](#)을 참조하십시오.

매개변수:

**exception (input)**

XMS에서 작성하는 예외 포인터입니다.

리턴값:

Void

## InvalidOperationException

XMS에서는 애플리케이션이 잘못되거나 부적절한 시간에 메소드를 호출하거나 XMS이(가) 요청된 조작에 적절한 상태가 아닌 경우 이 예외를 처리합니다.

상속 계층 구조:

```
IBM.XMS.XMSEException
|
+----IBM.XMS.Exception
|
+----IBM.XMS.InvalidOperationException
```

## 상속된 특성 및 메소드

다음 메소드는 `XMSEException` 인터페이스에서 상속됩니다.

GetError코드, GetLinked예외

## InitialContext

애플리케이션이 InitialContext 오브젝트를 사용하여 관리 오브젝트의 저장소에서 검색되는 오브젝트 정의에서 오브젝트를 작성합니다.

상속 계층 구조:  
없음

## .NET 특성

*Environment* - 환경 가져오기

인터페이스:

```
Hashtable Environment
{
    get;
}
```

환경을 가져옵니다.

예외:

- 예외는 사용 중인 디렉토리 서비스에 따라 다릅니다.

## 구성자

*InitialContext* - 초기 컨텍스트 작성

인터페이스:

```
InitialContext(Hashtable env);
```

InitialContext 오브젝트를 작성합니다.

매개변수:

- 관리 오브젝트 저장소에 대한 연결을 설정하는 데 필요한 정보를 Hashtable 환경의 구성자에 제공합니다.

예외:

- XMSEException

## 메소드

*AddToEnvironment* - 환경에 새 특성 추가

인터페이스:

```
Object AddToEnvironment(String propName, Object propVal);
```

환경에 새 특성을 추가합니다.

매개변수:

**propName (input)**

추가할 특성의 이름을 캡슐화하는 String 오브젝트입니다.

**propVal (input)**

추가할 특성의 값입니다.



**리턴값:**

이전 특성 값입니다.

**예외:**

- 예외는 사용 중인 디렉토리 서비스에 따라 다릅니다.

*Close* - 해당 컨텍스트 닫기

**인터페이스:**

```
void Close()
```

이 컨텍스트를 닫습니다.

**매개변수:**

없음

**리턴값:**

없음

**예외:**

- 예외는 사용 중인 디렉토리 서비스에 따라 다릅니다.

*Lookup* - 초기 컨텍스트에서 오브젝트 검색

**인터페이스:**

```
Object Lookup(String name);
```

관리 오브젝트의 저장소에서 검색되는 오브젝트 정의에서 오브젝트를 작성합니다.

**매개변수:****name (input)**

검색할 관리 오브젝트의 이름을 캡슐화하는 String 오브젝트입니다. 이 이름은 간단한 이름 또는 복잡한 이름일 수 있습니다. 자세한 정보는 [관리 오브젝트 검색](#)을 참조하십시오.

**리턴값:**

검색할 오브젝트의 유형에 따라 *ConnectionFactory* 또는 *IDestination*입니다. 함수가 디렉토리에 액세스 할 수 있지만 필요한 오브젝트를 찾을 수 없는 경우 *Null*이 리턴됩니다.

**예외:**

- 예외는 사용 중인 디렉토리 서비스에 따라 다릅니다.

*RemoveFromEnvironment* - 환경에서 특성 제거

**인터페이스:**

```
Object RemoveFromEnvironment(String propName);
```

환경에서 특성을 제거합니다.

**매개변수:****propName (input)**

제거할 특성의 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**

제거된 오브젝트입니다.

**예외:**

- 예외는 사용 중인 디렉토리 서비스에 따라 다릅니다.

## InvalidClientIDException

애플리케이션이 연결을 위해 클라이언트 ID를 설정하려 했지만 클라이언트 ID가 유효하지 않거나 이미 사용 중이면 XMS에서 이 예외가 처리됩니다.

상속 계층 구조:

```
IBM.XMS.XMSException
|
+----IBM.XMS.XMSException
|
+----IBM.XMS.InvalidClientIDException
```

### 상속된 특성 및 메소드

다음 메소드는 [XMSException](#) 인터페이스에서 상속됩니다.

[GetError코드](#), [GetLinked예외](#)

## InvalidDestinationException

XMS에서는 애플리케이션이 유효하지 않은 대상을 지정한 경우에 이 예외 처리를 합니다.

상속 계층 구조:

```
IBM.XMS.XMSException
|
+----IBM.XMS.XMSException
|
+----IBM.XMS.InvalidDestinationException
```

### 상속된 특성 및 메소드

다음 메소드는 [XMSException](#) 인터페이스에서 상속됩니다.

[GetError코드](#), [GetLinked예외](#)

## InvalidSelectorException

애플리케이션에서 구문이 유효하지 않은 메시지 선택자 표현식을 제공하면 XMS에서 이 예외를 처리합니다.

상속 계층 구조:

```
IBM.XMS.XMSException
|
+----IBM.XMS.XMSException
|
+----IBM.XMS.InvalidSelectorException
```

### 상속된 특성 및 메소드

다음 메소드는 [XMSException](#) 인터페이스에서 상속됩니다.

[GetError코드](#), [GetLinked예외](#)

## IMapMessage

맵 메시지는 본문이 이름-값 쌍 세트로 구성된 메시지이며 각 값에는 연관된 데이터 유형이 있습니다.

상속 계층 구조:

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IMessage
```

```
|  
+----IBM.XMS.IMapMessage
```

애플리케이션이 이름-값 쌍의 값을 가져올 때 값은 XMS에 의해 다른 데이터 유형으로 변환될 수 있습니다. 이러한 내재적 변환 양식에 대한 자세한 정보는 [XMS 메시지의 본문](#)에서 맵 메시지에 대한 정보를 참조하십시오.

## .NET 특성

*MapNames* - 맵 이름 가져오기

인터페이스:

```
System.Collections.IEnumerator MapNames  
{  
    get;  
}
```

맵 메시지 본문에서 이름 목록을 가져옵니다.

예외:

- XMSEException

## 메소드

*GetBoolean* - 부울 값 가져오기

인터페이스:

```
Boolean GetBoolean(String name);
```

맵 메시지 본문에서 이름으로 식별된 부울 값을 가져옵니다.

매개변수:

**name (input)**

부울 값을 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

리턴값:

맵 메시지 본문에서 검색된 부울 값입니다.

예외:

- XMSEException

*GetByte* - 바이트 가져오기

인터페이스:

```
Byte    GetByte(String name);  
Int16   GetSignedByte(String name);
```

맵 메시지 본문에서 이름으로 식별되는 바이트를 가져옵니다.

매개변수:

**name (input)**

바이트를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

리턴값:

맵 메시지 본문에서 검색된 바이트입니다. 바이트에 대한 데이터 변환은 수행되지 않습니다.

예외:

- XMSEException

*GetBytes* - 바이트 가져오기

인터페이스:

```
Byte[] GetBytes(String name);
```

맵 메시지 본문에서 이름으로 식별된 바이트 배열을 가져옵니다.

매개변수:

**name (input)**

바이트 배열을 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

리턴값:

배열의 바이트 수입니다.

예외:

- XMSEException

*GetChar* - 문자 가져오기

인터페이스:

```
Char GetChar(String name);
```

맵 메시지 본문에서 이름으로 식별된 문자를 가져옵니다.

매개변수:

**name (input)**

문자를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

리턴값:

맵 메시지 본문에서 검색된 문자입니다.

예외:

- XMSEException

*GetDouble* - 배 정밀도 부동 소수점 수 가져오기

인터페이스:

```
Double GetDouble(String name);
```

맵 메시지 본문에서 이름으로 식별되는 배정밀도 부동 소수점 수를 가져옵니다.

매개변수:

**name (input)**

배 정밀도 부동 소수점 수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

리턴값:

맵 메시지 본문에서 검색된 배 정밀도 부동 소수점 수입니다.

예외:

- XMSEException

*GetFloat* - 부동 소수점 수 가져오기

인터페이스:

```
Single GetFloat(String name);
```

맵 메시지 본문에서 이름으로 식별된 부동 소수점 수를 가져옵니다.

매개변수:

**name (input)**

부동 소수점 수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

리턴값:

맵 메시지 본문에서 검색된 부동 소수점 수입니다.

예외:

- XMSEException

*GetInt* - 정수 가져오기

인터페이스:

```
Int32  GetInt(String name);
```

맵 메시지 본문에서 이름으로 식별된 정수를 가져옵니다.

매개변수:

**name (input)**

정수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

리턴값:

맵 메시지 본문에서 검색된 정수입니다.

예외:

- XMSEException

*GetLong* - Long 정수 가져오기

인터페이스:

```
Int64  GetLong(String name);
```

맵 메시지 본문에서 이름으로 식별된 Long 정수를 가져옵니다.

매개변수:

**name (input)**

long 정수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

리턴값:

맵 메시지 본문에서 검색된 long 정수입니다.

예외:

- XMSEException

*GetObject* - 오브젝트 가져오기

인터페이스:

```
Object  GetObject(String name);
```

맵 메시지 본문에서 이름-값 쌍 값에 대한 참조를 가져옵니다. 이름-값 쌍은 이름으로 식별됩니다.

매개변수:

**name (input)**

이름-값 쌍의 이름을 캡슐화하는 String 오브젝트입니다.

리턴값:

다음 오브젝트 유형 중 하나인 값입니다.

Boolean

Byte  
Byte[]  
Char  
Double  
Single  
Int32  
Int64  
Int16  
String

**예외:**

XMSEException

*GetShort* - Short 정수 가져오기

**인터페이스:**

```
Int16 GetShort(String name);
```

맵 메시지 본문에서 이름으로 식별된 Short 정수를 가져옵니다.

**매개변수:**

**name (input)**

short 정수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**

맵 메시지 본문에서 검색된 short 정수입니다.

**예외:**

- XMSEException

*GetString* - 문자열 가져오기

**인터페이스:**

```
String GetString(String name);
```

맵 메시지 본문에서 이름으로 식별된 문자열을 가져옵니다.

**매개변수:**

**name (input)**

맵 메시지 본문에서 문자열을 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**

맵 메시지 본문에서 검색된 문자열을 캡슐화하는 String 오브젝트입니다. 데이터 변환이 필수인 경우 이 값은 변환 후 문자열입니다.

**예외:**

- XMSEException

*ItemExists* - 이름-값 쌍이 있는지 검사

**인터페이스:**

```
Boolean ItemExists(String name);
```

맵 메시지 본문에 지정된 이름의 이름-값 쌍이 있는지 검사합니다.

매개변수:

**name (input)**

이름-값 쌍의 이름을 캡슐화하는 String 오브젝트입니다.

리턴값:

- True - 맵 메시지 본문에 지정된 이름의 이름-값 쌍이 있는 경우
- False - 맵 메시지 본문에 지정된 이름의 이름-값 쌍이 없는 경우

예외:

- XMSEException

*SetBoolean* - 부울 값 설정

인터페이스:

```
void SetBoolean(String name, Boolean value);
```

맵 메시지 본문에 부울 값을 설정합니다.

매개변수:

**name (input)**

맵 메시지 본문에서 부울 값을 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**

설정할 부울 값입니다.

리턴값:

Void

예외:

- XMSEException

*SetByte* - 바이트 설정

인터페이스:

```
void SetByte(String name, Byte value);  
void SetSignedByte(String name, Int16 value);
```

맵 메시지 본문에서 바이트를 설정합니다.

매개변수:

**name (input)**

맵 메시지 본문에서 바이트를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**

설정할 바이트입니다.

리턴값:

Void

예외:

- XMSEException

*SetBytes* - 바이트 설정

인터페이스:

```
void SetBytes(String name, Byte[] value);
```

맵 메시지 본문에서 바이트 배열을 설정합니다.

매개변수:

**name (input)**

맵 메시지 본문에서 바이트 배열을 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**

설정할 바이트 배열입니다.

리턴값:

Void

예외:

- XMSEException

*SetChar* - 문자 설정

인터페이스:

```
void SetChar(String name, Char value);
```

맵 메시지 본문에서 2바이트 문자를 설정합니다.

매개변수:

**name (input)**

맵 메시지 본문에서 문자를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**

설정할 문자입니다.

리턴값:

Void

예외:

- XMSEException

*SetDouble* - 배 정밀도 부동 소수점 수 설정

인터페이스:

```
void SetDouble(String name, Double value);
```

맵 메시지 본문에 배정밀도 부동 소수점 수를 설정합니다.

매개변수:

**name (input)**

맵 메시지 본문에서 배 정밀도 부동 소수점 수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**

설정할 배 정밀도 부동 소수점 수입니다.

리턴값:

Void

예외:

- XMSEException

*SetFloat* - 부동 소수점 수 설정

인터페이스:

```
void SetFloat(String name, Single value);
```

맵 메시지 본문에서 부동 소수점 수를 설정합니다.



매개변수:

**name (input)**

맵 메시지 본문에서 부동 소수점 수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**

설정할 부동 소수점 수입니다.

리턴값:

Void

예외:

- XMSEException

*SetInt* - 정수 설정

인터페이스:

```
void SetInt(String name, Int32 value);
```

맵 메시지 본문에서 정수를 설정합니다.

매개변수:

**name (input)**

맵 메시지 본문에서 정수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**

설정할 정수입니다.

리턴값:

Void

예외:

- XMSEException

*SetLong* - Long 정수 설정

인터페이스:

```
void SetLong(String name, Int64 value);
```

맵 메시지 본문에서 long 정수를 설정합니다.

매개변수:

**name (input)**

맵 메시지 본문에서 long 정수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**

설정할 long 정수입니다.

리턴값:

Void

예외:

- XMSEException

*SetObject* - 오브젝트 설정

인터페이스:

```
void SetObject(String name, Object value);
```

맵 메시지 본문에서 값을 XMS 원시 유형으로 설정합니다.

매개변수:

**name (input)**

맵 메시지 본문에서 값을 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**

설정할 값을 포함하는 바이트 배열입니다.

리턴값:

Void

예외:

- XMSEException

*SetShort* - Short 정수 설정

인터페이스:

```
void SetShort(String name, Int16 value);
```

맵 메시지 본문에서 short 정수를 설정합니다.

매개변수:

**name (input)**

맵 메시지 본문에서 short 정수를 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**

설정할 short 정수입니다.

리턴값:

Void

예외:

- XMSEException

*SetString* - 문자열 설정

인터페이스:

```
void SetString(String name, String value);
```

맵 메시지 본문에서 문자열을 설정합니다.

매개변수:

**name (input)**

맵 메시지 본문에서 문자열을 식별하는 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**

설정할 문자열을 캡슐화하는 String 오브젝트입니다.

리턴값:

Void

예외:

- XMSEException

### 상속된 특성 및 메소드

다음 특성은 *IMessage* 인터페이스에서 상속됩니다.

[JMSCorrelationID](#), [JMSDeliveryMode](#), [JMSDestination](#), [JMSExpiration](#), [JMSMessageID](#), [JMSPriority](#), [JMSRe교부](#), [JMSReplyTo](#), [JMSTim환원](#), [JMSType](#), 특성

다음 메소드는 *IMessage* 인터페이스에서 상속됩니다.

[clearBody](#), [clearProperties](#), [PropertyExists](#)

다음 메소드는 [IPropertyContext](#) 인터페이스에서 상속됩니다.

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

## IMessage

Message 오브젝트는 애플리케이션이 송신하거나 수신하는 메시지를 나타냅니다. IMessage는 IMapMessage와 같은 메시지 클래스의 수퍼클래스입니다.

상속 계층 구조:

```
IBM.XMS.IPropertyContext
|
+---- IBM.XMS.IMessage
```

Message 오브젝트에 있는 JMS 메시지 헤더 필드의 목록은 [XMS 메시지의 헤더 필드](#)를 참조하십시오. Message 오브젝트의 JMS 정의 특성 목록은 [메시지의 JMS 정의 특성](#)을 참조하십시오. Message 오브젝트의 IBM 정의 특성 목록은 [메시지의 IBM 정의 특성](#)을 참조하십시오. Message 오브젝트에 대한 JMS\_IBM\_MQMD\* 특성 목록은 [1953 페이지의 『JMS IBM MQMD\\* 특성』](#)의 내용을 참조하십시오.

메시지는 가비지 콜렉터에 의해 삭제됩니다. 메시지가 삭제되면 해당 메시지가 사용하고 있던 자원이 해제됩니다.

## .NET 특성

*GetJMSCorrelationID* - *JMSCorrelationID* 가져오기 및 설정

인터페이스:

```
String JMSCorrelationID
{
    get;
    set;
}
```

메시지의 상관 ID를 가져와서 String 오브젝트로 설정합니다.

예외:

- XMSException

*JMSDeliveryMode* - *JMSDeliveryMode* 가져오기 및 설정

인터페이스:

```
DeliveryMode JMSDeliveryMode
{
    get;
    set;
}
```

메시지 전달 모드를 가져오고 설정합니다.

메시지의 전달 모드는 다음 값 중 하나입니다.

`DeliveryMode.Persistent`  
`DeliveryMode.NonPersistent`

전송되지 않은 새로 작성된 메시지의 경우 전달 모드는 `DeliveryMode.Persistent`지만, 브로커에 대한 실시간 연결의 경우 전달 모드는 `DeliveryMode.NonPersistent`입니다. 수신된 메시지의 경우 이 메소드는 수신하는 애플리케이션이 `JMSDeliveryMode`를 설정하여 전달 모드를 변경하지 않는 한 메시지를 전송할 때 `IMessageProducer.send()` 호출에 의해 설정된 전달 모드를 리턴합니다.

**예외:**

- `XMSEException`

*JMSDestination* - *JMSDestination* 가져오기 및 설정

**인터페이스:**

```
IDestination JMSDestination
{
    get;
    set;
}
```

메시지 대상을 가져오고 설정합니다.

목적지는 메시지를 전송할 때 `IMessageProducer.send()` 호출에 의해 설정됩니다. `JMSDestination`의 값은 무시됩니다. 그러나 `JMSDestination`을 사용하여 수신된 메시지의 목적지를 변경할 수 있습니다.

전송되지 않은 새로 작성된 메시지의 경우 이 메소드는 송신하는 애플리케이션이 `JMSDestination`을 설정하여 목적지를 설정하지 않는 한 널 `Destination` 오브젝트를 리턴합니다. 수신된 메시지의 경우 이 메소드는 수신하는 애플리케이션이 `JMSDestination`을 설정하여 목적지를 변경하지 않는 한 메시지를 전송할 때 `IMessageProducer.send()` 호출에 의해 설정된 목적지에 대한 `Destination` 오브젝트를 리턴합니다.

**예외:**

- `XMSEException`

*JMSEExpiration* - *JMSEExpiration* 가져오기 및 설정

**인터페이스:**

```
Int64 JMSEExpiration
{
    get;
    set;
}
```

메시지의 만기 시간을 가져오고 설정합니다.

만기 시간은 메시지를 전송할 때 `IMessageProducer.send()` 호출에 의해 설정됩니다. 만기 시간 값은 전송 애플리케이션이 지정한 TTL(Time to Live)을 메시지 전송 시간에 더한 값으로, 1970년 1월 1일 00:00:00 GMT 이후부터 밀리초 단위로 표시됩니다.

전송되지 않은 새로 작성된 메시지의 경우 송신하는 애플리케이션이 `JMSEExpiration`을 설정하여 다른 만기 시간을 설정하지 않는 한 만기 시간은 0입니다. 수신된 메시지의 경우 이 메소드는 수신하는 애플리케이션이 `JMSEExpiration`을 설정하여 만기 시간을 변경하지 않는 한 메시지를 전송할 때 `IMessageProducer.send()` 호출에 의해 설정된 만기 시간을 리턴합니다.

TTL(Time to Live)이 0일 경우 `IMessageProducer.send()` 호출은 만기 시간을 0으로 설정하여 메시지가 만기되지 않음을 나타냅니다.

XMS는 만기된 메시지를 버리고 이를 애플리케이션에 전달하지 않습니다.

**예외:**

- `XMSEException`

## JMSMessageID - JMSMessageID 가져오기 및 설정

### 인터페이스:

```
String JMSMessageID
{
    get;
    set;
}
```

메시지의 메시지 ID를 가져와서 메시지 ID를 캡슐화하는 String 오브젝트로 설정합니다.

메시지 ID는 메시지를 전송할 때 IMessageProducer.send() 호출에 의해 설정됩니다. 수신된 메시지의 경우 이 메소드는 수신하는 애플리케이션이 JMSMessageID를 설정하여 메시지 ID를 변경하지 않는 한 메시지를 전송할 때 IMessageProducer.send() 호출에 의해 설정된 메시지 ID를 리턴합니다.

메시지에 메시지 ID가 없을 경우 이 메소드는 널을 리턴합니다.

### 예외:

- XMSEException

## JMSPriority - JMSPriority 가져오기 및 설정

### 인터페이스:

```
Int32 JMSPriority
{
    get;
    set;
}
```

메시지 우선순위를 가져오고 설정합니다.

우선순위는 메시지를 전송할 때 IMessageProducer.send() 호출에 의해 설정됩니다. 값은 0(가장 낮은 우선순위) - 9(가장 높은 우선순위) 범위에 있는 정수입니다.

전송되지 않은 새로 작성된 메시지의 경우 송신하는 애플리케이션이 JMSPriority를 설정하여 다른 우선순위를 설정하지 않는 한 우선순위는 4입니다. 수신된 메시지의 경우 이 메소드는 수신하는 애플리케이션이 JMSPriority를 설정하여 우선순위를 변경하지 않는 한 메시지를 전송할 때 IMessageProducer.send() 호출에 의해 설정된 우선순위를 리턴합니다.

### 예외:

- XMSEException

## JMSRedelivered - JMSRedelivered 가져오기 및 설정

### 인터페이스:

```
Boolean JMSRedelivered
{
    get;
    set;
}
```

메시지가 다시 전달되는지 확인하고 메시지가 다시 전달되는지 표시합니다. 이 표시는 메시지를 수신할 때 IMessageConsumer.receive() 호출에 의해 설정됩니다.

이 특성의 값은 다음과 같습니다.

- True - 메시지가 다시 전달되는 경우
- False - 메시지가 다시 전달되지 않는 경우

브로커에 대한 실시간 연결의 경우 이 값은 항상 False입니다.

메시지가 전송되기 전에 `JMSRedelivered`에 의해 설정된 다시 전달 표시는 메시지가 전송될 때 `IMessageProducer.send()` 호출에 의해 무시되고 메시지가 수신될 때 `IMessageConsumer.receive()` 호출에 의해 무시되고 대체됩니다. 그러나 `JMSRedelivered`를 사용하여 수신된 메시지에 대한 표시를 변경할 수 있습니다.

**예외:**

- `XMSEException`

*JMSReplyTo* - *JMSReplyTo* 가져오기 및 설정

**인터페이스:**

```
IDestination JMSReplyTo
{
    get;
    set;
}
```

메시지의 응답이 송신될 대상을 가져오고 설정합니다.

이 특성의 값은 메시지에 대한 응답을 송신할 목적지에 대한 `Destination` 오브젝트입니다. 널 `Destination` 오브젝트는 응답이 예상되지 않음을 의미합니다.

**예외:**

- `XMSEException`

*JMSTimestamp* - *JMSTimestamp* 가져오기 및 설정

**인터페이스:**

```
Int64 JMSTimestamp
{
    get;
    set;
}
```

메시지가 전송된 시간을 가져오고 설정합니다.

시간 소인은 메시지를 전송할 때 `IMessageProducer.send()` 호출에 의해 설정되며, 1970년 1월 1일 00:00:00 GMT 이후부터 밀리초 단위로 표시됩니다.

전송되지 않은 새로 작성된 메시지의 경우 송신하는 애플리케이션이 `JMSTimestamp`를 설정하여 다른 시간 소인을 설정하지 않는 한 시간 소인은 0입니다. 수신된 메시지의 경우 이 메소드는 수신하는 애플리케이션이 `JMSTimestamp`를 설정하여 시간 소인을 변경하지 않는 한 메시지를 전송할 때 `IMessageProducer.send()` 호출에 의해 설정된 시간 소인을 리턴합니다.

**예외:**

- `XMSEException`

**참고:**

1. 시간 소인이 정의되어 있지 않은 경우 이 메소드는 0만 리턴하고 예외가 발생되지는 않습니다.

*JMSType* - *JMSType* 가져오기 및 설정

**인터페이스:**

```
String JMSType
{
    get;
    set;
}
```

메시지 유형을 가져오고 설정합니다.

JMSType의 값은 메시지 유형을 캡슐화하는 문자열입니다. 데이터 변환이 필수인 경우 이 값은 변환 후 유형입니다.

**예외:**

- XMSEException

*PropertyNames* - 특성 가져오기

**인터페이스:**

```
System.Collections.IEnumerator PropertyNames
{
    get;
}
```

메시지의 이름 특성 목록을 가져옵니다.

**예외:**

- XMSEException

**메소드**

*Acknowledge* - 수신확인

**인터페이스:**

```
void Acknowledge();
```

이 메시지와 세션에서 수신한 이전에 확인되지 않은 모든 메시지를 수신확인합니다.

세션의 수신확인 모드가 AcknowledgeMode.ClientAcknowledge인 경우 애플리케이션은 이 메소드를 호출할 수 있습니다. 세션에 다른 수신확인 모드가 있거나 세션이 트랜잭션된 경우 이 메소드에 대한 호출은 무시됩니다. 수신되었지만 수신확인되지 않은 메시지는 다시 전달될 수 있습니다.

메시지 수신확인에 대한 자세한 정보는 [../develop/xms\\_cmесack.dita#xms\\_cmесack](#)의 내용을 참조하십시오.

**매개변수:**

없음

**리턴값:**

Void

**예외:**

- XMSEException
- IllegalStateException

*ClearBody* - 본문 지우기

**인터페이스:**

```
void ClearBody();
```

메시지 본문을 지웁니다. 헤더 필드와 메시지 특성은 지워지지 않습니다.

애플리케이션이 메시지 본문을 지우면 본문은 새로 작성된 메시지의 빈 본문과 동일한 상태로 남아 있습니다. 새로 작성한 메시지의 빈 본문의 상태는 메시지 본문의 유형에 따라 다릅니다. 자세한 정보는 [XMS 메시지의 본문](#)을 참조하십시오.

애플리케이션은 본문 상태에 관계없이 언제든지 메시지 본문을 지울 수 있습니다. 메시지 본문이 읽기 전용일 경우 애플리케이션이 본문에 기록할 수 있는 유일한 방법은 애플리케이션이 먼저 본문을 지우는 것입니다.

매개변수:

없음

리턴값:

Void

예외:

- XMSEException

*ClearProperties* - 특성 지우기

인터페이스:

```
void ClearProperties();
```

메시지 특성을 지웁니다. 헤더 필드와 메시지 본문은 지워지지 않습니다.

애플리케이션이 메시지 특성을 지울 경우 특성은 읽기 및 쓰기가 가능해집니다.

애플리케이션은 특성의 상태에 관계없이 언제든지 메시지 특성을 지울 수 있습니다. 메시지 특성이 읽기 전용일 경우 쓰기 가능한 특성이 될 수 있는 유일한 방법은 애플리케이션이 먼저 특성을 지우는 것입니다.

매개변수:

없음

리턴값:

Void

예외:

- XMSEException

*PropertyExists* - 특성 존재 여부 검사

인터페이스:

```
Boolean PropertyExists(String propertyName);
```

지정된 이름의 특성이 메시지에 있는지 검사합니다.

매개변수:

**propertyName (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

리턴값:

- True - 지정된 이름의 특성이 메시지에 있는 경우
- False - 지정된 이름의 특성이 메시지에 없는 경우

예외:

- XMSEException

## 상속된 특성 및 메소드

다음 메소드는 [IPropertyContext](#) 인터페이스에서 상속됩니다.

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)



## IMessageConsumer

애플리케이션은 메시지 이용자를 사용하여 대상에 전송된 메시지를 받습니다.

상속 계층 구조:

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IMessageConsumer
```

MessageConsumer 오브젝트의 XMS 정의 특성 목록은 [1957 페이지의 『MessageConsumer 특성』](#)의 내용을 참조하십시오.

## .NET 특성

*MessageListener* - 메시지 리스너 가져오기 및 설정

인터페이스:

```
MessageListener MessageListener
{
    get;
    set;
}
```

메시지 이용자에 등록된 메시지 리스너를 가져오고 메시지 이용자에 메시지 리스너를 등록합니다.

메시지 이용자에 메시지 리스너가 등록되지 않은 경우 *MessageListener*는 널입니다. 메시지 리스너가 이미 메시지 이용자에 등록된 경우 대신 널을 지정하여 등록을 취소할 수 있습니다.

메시지 리스너 사용에 대한 자세한 정보는 [.NET에서 메시지 및 예외 리스너 사용](#)을 참조하십시오.

예외:

- *XMSEException*

*MessageSelector* - 메시지 선택자 가져오기

인터페이스:

```
String MessageSelector
{
    get;
}
```

메시지 이용자에 대한 메시지 선택자를 가져옵니다. 리턴값은 메시지 선택자 표현식을 캡슐화하는 *String* 오브젝트입니다. 데이터 변환이 필수인 경우 이 값은 변환 후 메시지 선택자 표현식입니다. 메시지 이용자에 메시지 선택자가 없는 경우 *MessageSelector*의 값은 널 *String* 오브젝트입니다.

예외:

- *XMSEException*

## 메소드

*Close* - 메시지 이용자 닫기

인터페이스:

```
void Close();
```

메시지 이용자를 닫습니다.

애플리케이션이 이미 닫힌 메시지 이용자를 닫으려는 경우 호출이 무시됩니다.

**매개변수:**

없음

**리턴값:**

Void

**예외:**

- XMSEException

*Receive* - 수신

**인터페이스:**

```
IMessage Receive();
```

메시지 이용자에 대한 다음 메시지를 수신합니다. 호출이 무기한으로 메시지를 기다리거나 메시지 이용자가 닫힐 때까지 기다립니다.

**매개변수:**

없음

**리턴값:**

Message 오브젝트에 대한 포인터입니다. 호출이 메시지를 기다리는 동안 메시지 이용자가 닫힌 경우 이 메소드는 널 Message 오브젝트에 대한 포인터를 리턴합니다.

**예외:**

- XMSEException

*Receive* - 수신(대기 간격 있음)

**인터페이스:**

```
IMessage Receive(Int64 delay);
```

메시지 이용자에 대한 다음 메시지를 수신합니다. 호출이 지정된 기간 동안만 메시지를 기다리거나 메시지 이용자가 닫힐 때까지 기다립니다.

**매개변수:**

**delay (input)**

호출이 메시지를 기다리는 시간(밀리초)입니다. 대기 간격을 0으로 지정하면 호출이 메시지를 무기한으로 기다립니다.

**리턴값:**

Message 오브젝트에 대한 포인터입니다. 대기 간격 중 메시지가 도착하지 않거나 호출이 메시지를 기다리는 동안 메시지 이용자가 닫힌 경우 이 메소드는 널 Message 오브젝트에 대한 포인터를 리턴하지만 예외는 발생하지 않습니다.

**예외:**

- XMSEException

*ReceiveNoWait* - 수신(대기 시간 없음)

**인터페이스:**

```
IMessage ReceiveNoWait();
```

메시지를 즉시 사용할 수 있으면 메시지 이용자의 다음 메시지를 받습니다.

**매개변수:**

없음

## 리턴값:

Message 오브젝트에 대한 포인터입니다. 즉시 사용할 수 있는 메시지가 없는 경우 이 메소드는 널 Message 오브젝트에 대한 포인터를 리턴합니다.

## 예외:

- XMSEException

## 상속된 특성 및 메소드

다음 메소드는 [IPropertyContext](#) 인터페이스에서 상속됩니다.

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

## MessageEOFException

애플리케이션이 바이트 메시지의 본문을 읽는 중에 XMS이(가) 바이트 메시지 스트림의 끝에 도달하면 XMS에서 이 예외를 처리합니다.

### 상속 계층 구조:

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
|
+----IBM.XMS.MessageEOFException
```

## 상속된 특성 및 메소드

다음 메소드는 [XMSEException](#) 인터페이스에서 상속됩니다.

[GetError코드](#), [GetLinked예외](#)

## MessageFormatException

XMS에 올바르지 않은 형식의 메시지가 표시되면 XMS에서 이 예외가 처리됩니다.

### 상속 계층 구조:

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
|
+----IBM.XMS.MessageFormatException
```

## 상속된 특성 및 메소드

다음 메소드는 [XMSEException](#) 인터페이스에서 상속됩니다.

[GetError코드](#), [GetLinked예외](#)

## IMessageListener(위임)

애플리케이션은 메시지 리스너를 사용하여 메시지를 비동기적으로 수신합니다.

### 상속 계층 구조:

없음

## 위임

*MessageListener* - 메시지 리스너

인터페이스:

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

메시지를 메시지 처리자에게 비동기적으로 전달합니다.

이 위임을 구현하는 메소드는 연결에 등록할 수 있습니다.

메시지 리스너 사용에 대한 자세한 정보는 [.NET에서 메시지 및 예외 리스너 사용을 참조하십시오](#).

매개변수:

**mesg (input)**

Message 오브젝트입니다.

리턴값:

Void

## MessageNotReadableException

XMS에서는 애플리케이션이 쓰기 전용인 메시지 본문을 읽으려고 할 경우 이 예외 처리를 합니다.

상속 계층 구조:

```
IBM.XMS.XMSException
|
+----IBM.XMS.XMSException
|
+----IBM.XMS.MessageNotReadableException
```

### 상속된 특성 및 메소드

다음 메소드는 [XMSException](#) 인터페이스에서 상속됩니다.

[GetError코드](#), [GetLinked예외](#)

## MessageNotWritableException

XMS에서는 애플리케이션이 쓰기 전용인 메시지 본문에 쓰려고 할 경우 이 예외 처리를 합니다.

상속 계층 구조:

```
IBM.XMS.XMSException
|
+----IBM.XMS.XMSException
|
+----IBM.XMS.MessageNotWritableException
```

### 상속된 특성 및 메소드

다음 메소드는 [XMSException](#) 인터페이스에서 상속됩니다.

[GetError코드](#), [GetLinked예외](#)

## IMessageProducer

애플리케이션은 메시지 생성자를 사용하여 메시지를 대상에 전송합니다.

상속 계층 구조:

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IMessageProducer
```

MessageProducer 오브젝트의 XMS 정의 특성 목록은 [1957 페이지의 『MessageProducer 특성』](#)의 내용을 참조하십시오.

## **.NET 특성**

*DeliveryMode* - 기본 전달 모드 가져오기 및 설정

**인터페이스:**

```
DeliveryMode DeliveryMode
{
    get;
    set;
}
```

메시지 생성자가 전송한 메시지의 기본 전달 모드를 가져오고 설정합니다.

기본 전달 모드는 다음 값 중 하나입니다.

`DeliveryMode.Persistent`  
`DeliveryMode.NonPersistent`

브로커에 대한 실시간 연결의 경우 값은 `DeliveryMode.NonPersistent`여야 합니다.

기본값은 `DeliveryMode.Persistent`지만, 브로커에 대한 실시간 연결의 기본값은 `DeliveryMode.NonPersistent`입니다.

**예외:**

- `XMSException`

*Destination* - 목적지 가져오기

**인터페이스:**

```
IDestination Destination
{
    get;
}
```

메시지 생성자의 대상을 가져옵니다.

**매개변수:**

없음

**리턴값:**

`Destination` 오브젝트입니다. 메시지 작성자에 목적지가 없으면 이 메소드는 널 `Destination` 오브젝트를 리턴합니다.

**예외:**

- `XMSException`

*DisableMsgID* - 메시지 ID 사용 안함 플래그 가져오기 및 설정

**인터페이스:**

```
Boolean DisableMessageID
{
    get;
    set;
}
```

수신 애플리케이션이 메시지 생성자가 보낸 메시지에 메시지 식별자를 포함해야 하는지 확인하고, 수신 애플리케이션이 메시지 생성자가 보낸 메시지에 메시지 식별자를 포함해야 하는지 표시합니다.

큐 관리자에 대한 연결이나 브로커에 대한 실시간 연결에서는 이 플래그가 무시되고, 서비스 통합 버스에 대한 연결에서는 이 플래그가 적용됩니다.

DisabledMsgID의 값은 다음과 같습니다.

- True - 수신 애플리케이션이 메시지 작성자가 보낸 메시지에 메시지 ID를 포함시킬 필요가 없는 경우
- False - 수신 애플리케이션이 메시지 작성자가 보낸 메시지에 메시지 ID를 포함시켜야 하는 경우

**예외:**

- XMSEException

*DisableMsgTS* - 시간소인 사용 안함 플래그 가져오기 및 설정

**인터페이스:**

```
Boolean DisableMessageTimestamp
{
    get;
    set;
}
```

수신 애플리케이션이 메시지 생성자가 보낸 메시지에 시간소인을 포함해야 하는지 확인하고, 수신 애플리케이션이 메시지 생성자가 보낸 메시지에 시간소인을 포함해야 하는지 표시합니다.

브로커에 대한 실시간 연결에서는 이 플래그가 무시되고, 큐 관리자에 대한 연결이나 서비스 통합 버스에 대한 연결에서는 이 플래그가 적용됩니다.

DisableMsgTS의 값은 다음과 같습니다.

- True - 수신 애플리케이션이 메시지 작성자가 보낸 메시지에 시간소인을 포함시킬 필요가 없는 경우
- False - 수신 애플리케이션이 메시지 작성자가 보낸 메시지에 시간소인을 포함시켜야 하는 경우

**리턴값:**

**예외:**

- XMSEException

*Priority* - 기본 우선순위 가져오기 및 설정

**인터페이스:**

```
Int32 Priority
{
    get;
    set;
}
```

메시지 생성자가 전송한 메시지의 기본 우선순위를 가져오고 설정합니다.

기본 메시지 우선순위 값은 0(가장 낮은 우선순위) - 9(가장 높은 우선순위) 범위에 있는 정수입니다.

브로커에 대한 실시간 연결에서는 메시지의 우선순위가 무시됩니다.

**예외:**

- XMSEException

*TimeToLive* - 기본 TTL(*Time to Live*) 가져오기 및 설정

**인터페이스:**

```
Int64 TimeToLive
{
    get;
    set;
}
```

메시지가 만기될 때까지의 유지되는 기본 시간을 가져오고 설정합니다.

이 시간은 메시지 작성자가 메시지를 보내는 시간부터 측정되며 밀리초 단위의 TTL(Time to Live)입니다. 값 0은 메시지가 만료되지 않음을 의미합니다.

브로커에 대한 실시간 연결의 경우 이 값은 항상 0입니다.

**예외:**

- XMSEException

## 메소드

*Close* - 메시지 작성자 닫기

**인터페이스:**

```
void Close();
```

메시지 생성자를 닫습니다.

애플리케이션이 이미 닫힌 메시지 작성자를 닫으려는 경우 호출이 무시됩니다.

**매개변수:**

없음

**리턴값:**

Void

**예외:**

- XMSEException

*Send* - 송신

**인터페이스:**

```
void Send(IMessage msg) ;
```

메시지 생성자가 작성될 때 지정된 대상으로 메시지를 전송합니다. 메시지 작성자 기본 전달 모드, 우선순위 및 TTL(Time to Live)을 사용하여 메시지를 보냅니다.

**매개변수:**

**msg (input)**

Message 오브젝트입니다.

**리턴값:**

Void

**예외:**

- XMSEException
- MessageFormatException
- InvalidDestinationException

*Send* - 송신(전달 모드, 우선순위 및 TTL(Time to Live) 지정)

**인터페이스:**

```
void Send(IMessage msg,
          DeliveryMode deliveryMode,
          Int32 priority,
          Int64 timeToLive);
```

메시지 생성자가 작성될 때 지정된 대상으로 메시지를 전송합니다. 지정된 전달 모드, 우선순위 및 TTL(Time to Live)을 사용하여 메시지를 보냅니다.

**매개변수:**

**msg (input)**

Message 오브젝트입니다.

**deliveryMode (input)**

메시지의 전달 모드이며, 다음 값 중 하나여야 합니다.

DeliveryMode.Persistent

DeliveryMode.NonPersistent

브로커에 대한 실시간 연결의 경우 값은 DeliveryMode.NonPersistent여야 합니다.

**priority (input)**

메시지의 우선순위입니다. 값은 0(가장 낮은 우선순위) - 9(가장 높은 우선순위) 범위에 있는 정수입니다.

브로커에 대한 실시간 연결에서는 이 값이 무시됩니다.

**timeToLive (input)**

밀리초 단위의 메시지 TTL(Time to Live)입니다. 값 0은 메시지가 만료되지 않음을 의미합니다. 브로커에 대한 실시간 연결의 경우 이 값은 0이어야 합니다.

**리턴값:**

Void

**예외:**

- XMSEException
- MessageFormatException
- InvalidDestinationException
- IllegalStateException

*Send* - 송신(지정된 목적지로 보내기)

**인터페이스:**

```
void Send(IDestination dest, IMessage msg) ;
```

메시지 생성자가 작성될 때 대상이 지정되지 않은 메시지 생성자를 사용하는 경우 지정된 대상으로 메시지를 전송합니다. 메시지 작성자 기본 전달 모드, 우선순위 및 TTL(Time to Live)을 사용하여 메시지를 보냅니다.

일반적으로 메시지 작성자를 생성할 때 목적지를 지정합니다. 그렇지 않을 경우 메시지를 전송할 때마다 목적지를 지정해야 합니다.

**매개변수:**

**dest (input)**

Destination 오브젝트입니다.

**msg (input)**

Message 오브젝트입니다.

**리턴값:**

Void

**예외:**

- XMSEException
- MessageFormatException
- InvalidDestinationException



Send - 송신(지정된 목적지로 보내기, 전달 모드, 우선순위 및 TTL(Time to Live) 지정)

인터페이스:

```
void Send(IDestination dest,
          IMessage msg,
          DeliveryMode deliveryMode,
          Int32 priority,
          Int64 timeToLive) ;
```

메시지 생성자가 작성될 때 대상이 지정되지 않은 메시지 생성자를 사용하는 경우 지정된 대상으로 메시지를 전송합니다. 지정된 전달 모드, 우선순위 및 TTL(Time to Live)을 사용하여 메시지를 보냅니다.

일반적으로 메시지 작성자를 생성할 때 목적지를 지정합니다. 그렇지 않을 경우 메시지를 전송할 때마다 목적지를 지정해야 합니다.

매개변수:

**dest (input)**

Destination 오브젝트입니다.

**msg (input)**

Message 오브젝트입니다.

**deliveryMode (input)**

메시지의 전달 모드이며, 다음 값 중 하나여야 합니다.

DeliveryMode.Persistent  
DeliveryMode.NonPersistent

브로커에 대한 실시간 연결의 경우 값은 DeliveryMode.NonPersistent여야 합니다.

**priority (input)**

메시지의 우선순위입니다. 값은 0(가장 낮은 우선순위) - 9(가장 높은 우선순위) 범위에 있는 정수입니다. 브로커에 대한 실시간 연결에서는 이 값이 무시됩니다.

**timeToLive (input)**

밀리초 단위의 메시지 TTL(Time to Live)입니다. 값 0은 메시지가 만료되지 않음을 의미합니다. 브로커에 대한 실시간 연결의 경우 이 값은 0이어야 합니다.

리턴값:

Void

예외:

- XMSEException
- MessageFormatException
- InvalidDestinationException
- IllegalStateException

## 상속된 특성 및 메소드

다음 메소드는 [IPropertyContext](#) 인터페이스에서 상속됩니다.

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

## IObjectMessage

오브젝트 메시지는 본문이 직렬화된 Java 또는 .NET 오브젝트로 구성되는 메시지입니다.

상속 계층 구조:

```
IBM.XMS.IPropertyContext
|
+---- IBM.XMS.IMessage
      |
      +---- IBM.XMS.IObjectMessage
```

## .NET 특성

*Object* - 오브젝트를 바이트로 가져오기 및 설정

인터페이스:

```
System.Object Object
{
    get;
    set;
}

Byte[] GetObject();
```

오브젝트 메시지의 본문을 구성하는 오브젝트를 가져오고 설정합니다.

예외:

- [XMSException](#)
- [MessageNotReadableException](#)
- [MessageEOFException](#)
- [MessageNotWritableException](#)

## 상속된 특성 및 메소드

다음 특성은 [IMessage](#) 인터페이스에서 상속됩니다.

[JMSCorrelationID](#), [JMSDeliveryMode](#), [JMSDestination](#), [JMSExpiration](#), [JMSMessageID](#), [JMSPriority](#), [JMSSubject](#), [JMSReplyTo](#), [JMSTimeout](#), [JMSType](#), 특성

다음 메소드는 [IMessage](#) 인터페이스에서 상속됩니다.

[clearBody](#), [clearProperties](#), [PropertyExists](#)

다음 메소드는 [IPropertyContext](#) 인터페이스에서 상속됩니다.

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

## IPropertyContext

[IPropertyContext](#)는 특성을 가져오고 설정하는 메소드를 포함하는 추상 슈퍼클래스입니다. 이러한 메소드는 다른 클래스에서 상속합니다.

상속 계층 구조:

없음

메소드

*GetBooleanProperty* - 부울 특성 가져오기

인터페이스:

```
Boolean GetBooleanProperty(String property_name);
```

지정된 이름의 부울 특성 값을 가져옵니다.

매개변수:

**property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

리턴값:

특성의 값입니다.

스레드 컨텍스트:

서브클래스에 의해 결정됨

예외:

- XMSEException

*GetByteProperty* - 바이트 특성 가져오기

인터페이스:

```
Byte    GetByteProperty(String property_name) ;  
Int16   GetSignedByteProperty(String property_name) ;
```

이름으로 식별된 바이트 특성 값을 가져옵니다.

매개변수:

**property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

리턴값:

특성의 값입니다.

스레드 컨텍스트:

서브클래스에 의해 결정됨

예외:

- XMSEException

*GetBytesProperty* - 바이트 배열 특성 가져오기

인터페이스:

```
Byte[]  GetBytesProperty(String property_name) ;
```

이름으로 식별된 바이트 배열 특성 값을 가져옵니다.

매개변수:

**property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

리턴값:

배열의 바이트 수입니다.

스레드 컨텍스트:

서브클래스에 의해 결정됨

예외:

- XMSEException

*GetCharProperty* - 문자 특성 가져오기

인터페이스:

```
Char GetCharProperty(String property_name) ;
```

이름으로 식별된 2바이트 문자 특성 값을 가져옵니다.

매개변수:

**property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

리턴값:

특성의 값입니다.

스레드 컨텍스트:

서브클래스에 의해 결정됨

예외:

- XMSEException

*GetDoubleProperty* - 배 정밀도 부동 소수점 특성 가져오기

인터페이스:

```
Double GetDoubleProperty(String property_name) ;
```

이름으로 식별된 배정밀도 부동 소수점 특성 값을 가져옵니다.

매개변수:

**property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

리턴값:

특성의 값입니다.

스레드 컨텍스트:

서브클래스에 의해 결정됨

예외:

- XMSEException

*GetFloatProperty* - 부동 소수점 특성 가져오기

인터페이스:

```
Single GetFloatProperty(String property_name) ;
```

이름으로 식별된 부동 소수점 특성 값을 가져옵니다.

매개변수:

**property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

리턴값:

특성의 값입니다.

스레드 컨텍스트:

서브클래스에 의해 결정됨

예외:

- XMSEException

## GetIntProperty - GetIntProperty

### 인터페이스:

```
Int32 GetIntProperty(String property_name) ;
```

이름으로 식별된 정수 특성 값을 가져옵니다.

### 매개변수:

#### **property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

### 리턴값:

특성의 값입니다.

### 스레드 컨텍스트:

서브클래스에 의해 결정됨

### 예외:

- XMSEException

## GetLongProperty - Long 정수 특성 가져오기

### 인터페이스:

```
Int64 GetLongProperty(String property_name) ;
```

이름으로 식별된 Long 정수 특성 값을 가져옵니다.

### 매개변수:

#### **property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

### 리턴값:

특성의 값입니다.

### 스레드 컨텍스트:

서브클래스에 의해 결정됨

### 예외:

- XMSEException

## GetObjectProperty - 오브젝트 특성 가져오기

### 인터페이스:

```
Object GetObjectProperty( String property_name) ;
```

이름으로 식별된 특성의 값과 데이터 유형을 가져옵니다.

### 매개변수:

#### **property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

### 리턴값:

특성의 값이며, 다음 오브젝트 유형 중 하나입니다.

Boolean  
Byte  
Byte[]  
Char  
Double

Single  
Int32  
Int64  
Int16  
String

**스레드 컨텍스트:**

서브클래스에 의해 결정됨

**예외:**

- XMSEException

*GetShortProperty* - Short 정수 특성 가져오기

**인터페이스:**

```
Int16 GetShortProperty(String property_name) ;
```

이름으로 식별된 Short 정수 특성 값을 가져옵니다.

**매개변수:**

**property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**

특성의 값입니다.

**스레드 컨텍스트:**

서브클래스에 의해 결정됨

**예외:**

- XMSEException

*GetStringProperty* - *GetStringProperty*

**인터페이스:**

```
String GetStringValue(String property_name) ;
```

이름으로 식별된 문자열 특성의 값을 가져옵니다.

**매개변수:**

**property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

**리턴값:**

특성 값 문자열을 캡슐화하는 String 오브젝트입니다. 데이터 변환이 필수인 경우 이 값은 변환 후 문자열입니다.

**스레드 컨텍스트:**

서브클래스에 의해 결정됨

**예외:**

- XMSEException

*SetBooleanProperty* - 부울 특성 설정

**인터페이스:**

```
void SetBooleanProperty( String property_name, Boolean value) ;
```

이름으로 식별된 부울 특성의 값을 설정합니다.

매개변수:

**property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**

특성의 값입니다.

리턴값:

Void

스레드 컨텍스트:

서브클래스에 의해 결정됨

예외:

- XMSEException
- MessageNotWritableException

*SetByteProperty* - 바이트 특성 설정

인터페이스:

```
void SetByteProperty( String property_name, Byte value) ;  
void SetSignedByteProperty( String property_name, Int16 value) ;
```

이름으로 식별된 바이트 특성 값을 설정합니다.

매개변수:

**property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**

특성의 값입니다.

리턴값:

Void

스레드 컨텍스트:

서브클래스에 의해 결정됨

예외:

- XMSEException
- MessageNotWritableException

*SetBytesProperty* - 바이트 배열 특성 설정

인터페이스:

```
void SetBytesProperty( String property_name, Byte[] value ) ;
```

이름으로 식별된 바이트 배열 특성의 값을 설정합니다.

매개변수:

**property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**

바이트 배열을 나타내는 특성의 값입니다.

리턴값:

Void

스레드 컨텍스트:

서브클래스에 의해 결정됨

#### 예외:

- XMSEException
- MessageNotWritableException

*SetCharProperty* - 문자 특성 설정

#### 인터페이스:

```
void SetCharProperty( String property_name, Char value) ;
```

이름으로 식별된 2바이트 문자 특성 값을 설정합니다.

#### 매개변수:

##### **property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

##### **value (input)**

특성의 값입니다.

#### 리턴값:

Void

#### 스레드 컨텍스트:

서브클래스에 의해 결정됨

#### 예외:

- XMSEException
- MessageNotWritableException

*SetDoubleProperty* - 배 정밀도 부동 소수점 특성 설정

#### 인터페이스:

```
void SetDoubleProperty( String property_name, Double value) ;
```

이름으로 식별된 배정밀도 부동 소수점 특성 값을 설정합니다.

#### 매개변수:

##### **property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

##### **value (input)**

특성의 값입니다.

#### 리턴값:

Void

#### 스레드 컨텍스트:

서브클래스에 의해 결정됨

#### 예외:

- XMSEException
- MessageNotWritableException

*SetFloatProperty* - 부동 소수점 특성 설정

#### 인터페이스:

```
void SetFloatProperty( String property_name, Single value) ;
```

이름으로 식별된 부동 소수점 특성 값을 설정합니다.



매개변수:

**property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**

특성의 값입니다.

리턴값:

Void

스레드 컨텍스트:

서브클래스에 의해 결정됨

예외:

- XMSEException
- MessageNotWritableException

*SetIntProperty* - 정수 특성 설정

인터페이스:

```
void SetIntProperty( String property_name, Int32 value) ;
```

이름으로 식별된 정수 특성 값을 설정합니다.

매개변수:

**property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**

특성의 값입니다.

리턴값:

Void

스레드 컨텍스트:

서브클래스에 의해 결정됨

예외:

- XMSEException
- MessageNotWritableException

*SetLongProperty* - Long 정수 특성 설정

인터페이스:

```
void SetLongProperty( String property_name, Int64 value) ;
```

이름으로 식별된 long 정수 특성 값을 설정합니다.

매개변수:

**property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**

특성의 값입니다.

리턴값:

Void

스레드 컨텍스트:

서브클래스에 의해 결정됨

#### 예외:

- XMSEException
- MessageNotWritableException

*SetObjectProperty* - 오브젝트 특성 설정

#### 인터페이스:

```
void SetObjectProperty( String property_name, Object value) ;
```

이름으로 식별된 특성의 값과 데이터 유형을 설정합니다.

#### 매개변수:

##### **property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

##### **objectType (input)**

특성의 값이며, 다음 오브젝트 유형 중 하나여야 합니다.

Boolean  
Byte  
Byte[]  
Char  
Double  
Single  
Int32  
Int64  
Int16  
String

##### **value (input)**

바이트 배열을 나타내는 특성의 값입니다.

##### **length (input)**

배열의 바이트 수입니다.

#### 리턴값:

Void

#### 스레드 컨텍스트:

서브클래스에 의해 결정됨

#### 예외:

- XMSEException
- MessageNotWritableException

*SetShortProperty* - Short 정수 특성 설정

#### 인터페이스:

```
void SetShortProperty( String property_name, Int16 value) ;
```

이름으로 식별된 Short 정수 특성 값을 설정합니다.

#### 매개변수:

##### **property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

##### **value (input)**

특성의 값입니다.

**리턴값:**

Void

**스레드 컨텍스트:**

서브클래스에 의해 결정됨

**예외:**

- XMSEException
- MessageNotWritableException

*SetStringProperty* - 문자열 특성 설정**인터페이스:**

```
void SetStringProperty( String property_name, String value);
```

이름으로 식별된 문자열 특성의 값을 설정합니다.

**매개변수:****property\_name (input)**

특성의 이름을 캡슐화하는 String 오브젝트입니다.

**value (input)**

특성 값 문자열을 캡슐화하는 String 오브젝트입니다.

**리턴값:**

Void

**스레드 컨텍스트:**

서브클래스에 의해 결정됨

**예외:**

- XMSEException
- MessageNotWritableException

## IQueueBrowser

애플리케이션이 큐 브라우저를 사용하여 메시지를 제거하지 않고 큐에서 메시지를 찾습니다.

**상속 계층 구조:**

```
IBM.XMS.IPropertyContext
System.Collections.IEnumerable
|
+---- IBM.XMS.IQueueBrowser
```

## .NET 특성

*MessageSelector* - 메시지 선택자 가져오기**인터페이스:**

```
String MessageSelector
{
    get;
}
```

큐 브라우저에 대한 메시지 선택자를 가져옵니다.

메시지 선택자는 메시지 선택자 표현식을 캡슐화하는 String 오브젝트입니다. 데이터 변환이 필수인 경우 이 값은 변환 후 메시지 선택자 표현식입니다. 큐 브라우저에 메시지 선택자가 없으면 이 메소드는 널 String 오브젝트를 리턴합니다.

## 예외:

- XMSEException

*Queue* - 큐 가져오기

## 인터페이스:

```
IDestination Queue
{
    get;
}
```

큐 브라우저와 연관된 큐를 큐를 나타내는 대상 오브젝트로 가져옵니다.

## 예외:

- XMSEException

## 메소드

*Close* - 큐 브라우저 닫기

## 인터페이스:

```
void Close();
```

큐 브라우저를 닫습니다.

애플리케이션이 이미 닫힌 큐 브라우저를 닫으려는 경우 호출이 무시됩니다.

## 매개변수:

없음

## 리턴값:

Void

## 예외:

- XMSEException

*GetEnumerator* - 메시지 가져오기

## 인터페이스:

```
IEnumerator GetEnumerator();
```

큐에 있는 메시지 목록을 가져옵니다.

이 메소드는 **Message** 오브젝트의 목록을 캡슐화하는 열거자를 리턴합니다. **Message** 오브젝트의 순서는 큐에서 메시지를 검색하는 순서와 동일합니다. 애플리케이션은 열거자를 사용하여 각 메시지를 교대로 찾아볼 수 있습니다.

열거자는 메시지가 큐에 놓여지고 큐에서 제거될 때 동적으로 업데이트됩니다. 애플리케이션이 **IEnumerator.MoveNext()**를 호출하여 큐의 다음 메시지를 찾아볼 때마다 메시지는 큐의 현재 콘텐츠를 반영합니다.

애플리케이션이 이 메소드를 큐 브라우저에 두 번 이상 호출하는 경우 각 호출은 새 열거자를 리턴합니다. 그러므로 애플리케이션은 둘 이상의 열거자를 사용하여 큐의 메시지를 찾아보고 큐 내에 여러 위치를 유지보수할 수 있습니다.

## 매개변수:

없음

## 리턴값:

Iterator 오브젝트입니다.

예외:

- XMSEException

## 상속된 특성 및 메소드

다음 메소드는 `IPropertyContext` 인터페이스에서 상속됩니다.

`GetBooleanProperty`, `GetByteProperty`, `GetBytesProperty`, `GetCharProperty`, `GetDoubleProperty`, `GetFloatProperty`, `GetIntProperty`, `GetLongProperty`, `GetObjectProperty`, `GetShortProperty`, `GetStringProperty`, `SetBooleanProperty`, `SetByteProperty`, `SetBytesProperty`, `SetCharProperty`, `SetDoubleProperty`, `SetFloatProperty`, `SetIntProperty`, `SetLongProperty`, `SetObjectProperty`, `SetShortProperty`, `SetStringProperty`

## Requestor

애플리케이션이 요청자를 사용하여 요청 메시지를 보내고 잠시 대기한 후 응답을 수신합니다.

상속 계층 구조:

없음

## 구성자

*Requestor* - 요청자 작성

인터페이스:

```
Requestor(ISession sess, IDestination dest);
```

요청자를 작성합니다.

매개변수:

**sess (input)**

Session 오브젝트입니다. 세션은 변환할 수 없으며 다음 수신확인 모드 중 하나를 가지고 있어야 합니다.

`AcknowledgeMode.AutoAcknowledge`

`AcknowledgeMode.DupsOkAcknowledge`

**dest (input)**

애플리케이션이 요청 메시지를 보낼 수 있는 목적지를 나타내는 Destination 오브젝트입니다.

스레드 컨텍스트:

요청자와 연관된 세션입니다.

예외:

- XMSEException

## 메소드

*Close* - 요청자 닫기

인터페이스:

```
void Close();
```

요청자를 닫습니다.

애플리케이션이 이미 닫힌 요청자를 닫으려는 경우 호출이 무시됩니다.

**참고:** 애플리케이션이 요청자를 닫아도 연관된 세션은 닫히지 않습니다. 이런 점에서 XMS는 JMS와 다르게 작동합니다.

매개변수:

없음

리턴값:

Void

스레드 컨텍스트:

임의

예외:

- XMSEException

*Request* - 응답 요청

인터페이스:

```
IMessage Request(IMessage requestMessage);
```

요청 메시지를 전송한 다음, 기다렸다가 요청 메시지를 수신하는 애플리케이션에서 응답을 수신합니다. 응답을 수신하거나 세션이 끝날 때까지(둘 중에서 빠른 쪽이 적용됨) 이 메소드에 대한 호출은 차단됩니다.

매개변수:

**requestMessage (input)**

요청 메시지를 캡슐화하는 Message 오브젝트입니다.

리턴값:

응답 메시지를 캡슐화하는 Message 오브젝트에 대한 포인터입니다.

스레드 컨텍스트:

요청자와 연관된 세션입니다.

예외:

- XMSEException

## ResourceAllocationException

XMS에서는 XMS이(가) 메소드에 필요한 자원을 할당할 수 없는 경우에 이 예외 처리를 합니다.

상속 계층 구조:

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
|
+----IBM.XMS.ResourceAllocationException
```

## 상속된 특성 및 메소드

다음 메소드는 XMSEException 인터페이스에서 상속됩니다.

GetError코드, GetLinked예외

## SecurityException

XMS 는 애플리케이션을 인증하기 위해 제공된 사용자 ID 및 비밀번호가 거부되는 경우 이 예외를 처리합니다. XMS는 권한 검사에 실패하여 메소드가 완료되지 않은 경우에도 이 예외를 발생시킵니다.

상속 계층 구조:

```
IBM.XMS.XMSEException
|
+----IBM.XMS.XMSEException
|
+----IBM.XMS.SecurityException
```

## 상속된 특성 및 메소드

다음 메소드는 [XMSException](#) 인터페이스에서 상속됩니다.

[GetError코드](#), [GetLinked예외](#)

## ISession

세션은 메시지 송신 및 수신을 위한 단일 스레드 컨텍스트입니다.

상속 계층 구조:

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.ISession
```

Session 오브젝트의 XMS 정의 특성 목록은 [1957 페이지의 『Session 특성』](#)의 내용을 참조하십시오.

## .NET 특성

*AcknowledgeMode* - 수신확인 모드 가져오기

인터페이스:

```
AcknowledgeMode AcknowledgeMode
{
    get;
}
```

세션의 수신확인 모드를 가져옵니다.

수신확인 모드는 세션이 작성될 때 지정됩니다.

세션이 트랜잭션되지 않으면 수신확인 모드는 다음 값 중 하나입니다.

```
AcknowledgeMode.AutoAcknowledge
AcknowledgeMode.ClientAcknowledge
AcknowledgeMode.DupsOkAcknowledge
```

수신확인 모드에 대한 자세한 정보는 [메시지 수신확인](#)을 참조하십시오.

트랜잭션된 세션에는 수신확인 모드가 없습니다. 세션이 트랜잭션되면 메소드가 `AcknowledgeMode.SessionTransacted`를 대신 리턴합니다.

예외:

- `XMSException`

*Transacted* - 트랜잭션되었는지 여부를 판별

인터페이스:

```
Boolean Transacted
{
    get;
}
```

세션이 트랜잭션되었는지 판별합니다.

트랜잭션 상태는 다음과 같습니다.

- `True` - 세션이 트랜잭션된 경우
- `False` - 세션이 트랜잭션되지 않은 경우

브로커에 대한 실시간 연결의 경우 이 메소드는 항상 `False`를 리턴합니다.

#### 예외:

- XMSEException

### 메소드

*Close* - 세션 닫기

#### 인터페이스:

```
void Close();
```

세션을 닫습니다. 세션이 트랜잭션되면 진행 중인 트랜잭션이 롤백됩니다.

애플리케이션이 이미 닫힌 세션을 닫으려는 경우 호출이 무시됩니다.

#### 매개변수:

없음

#### 리턴값:

Void

#### 스레드 컨텍스트:

임의

#### 예외:

- XMSEException

*Commit* - 커밋

#### 인터페이스:

```
void Commit();
```

현재 트랜잭션에서 처리된 모든 메시지를 요약합니다.

세션은 트랜잭션된 세션이어야 합니다.

#### 매개변수:

없음

#### 리턴값:

Void

#### 예외:

- XMSEException
- IllegalStateException
- TransactionRolledBackException

*CreateBrowser* - 큐 브라우저 작성

#### 인터페이스:

```
IQueueBrowser CreateBrowser(IDestination queue) ;
```

지정된 큐에 맞는 큐 브라우저를 작성합니다.

#### 매개변수:

##### **queue (input)**

큐를 나타내는 Destination 오브젝트입니다.

#### 리턴값:

QueueBrowser 오브젝트입니다.



**예외:**

- XMSEException
- InvalidDestinationException

*CreateBrowser* - 큐 브라우저 작성(메시지 선택자 사용)

**인터페이스:**

```
IQueueBrowser CreateBrowser(IDestination queue, String selector) ;
```

메시지 선택자를 사용하여 지정된 큐에 대한 큐 브라우저를 작성합니다.

**매개변수:**

**queue (input)**

큐를 나타내는 Destination 오브젝트입니다.

**selector (input)**

메시지 선택자 표현식을 캡슐화하는 String 오브젝트입니다. 메시지 선택자 표현식과 일치하는 특성이 있는 메시지만 큐 브라우저에 전달됩니다.

널 String 오브젝트는 큐 브라우저에 대한 메시지 선택자가 없음을 의미합니다.

**리턴값:**

QueueBrowser 오브젝트입니다.

**예외:**

- XMSEException
- InvalidDestinationException
- InvalidSelectorException

*CreateBytesMessage* - 바이트 메시지 작성

**인터페이스:**

```
IBytesMessage CreateBytesMessage();
```

바이트 메시지를 작성합니다.

**매개변수:**

없음

**리턴값:**

BytesMessage 오브젝트입니다.

**예외:**

- XMSEException
- IllegalStateException(세션이 닫힘)

*CreateConsumer* - 이용자 작성

**인터페이스:**

```
IMessageConsumer CreateConsumer(IDestination dest) ;
```

지정된 대상의 메시지 이용자를 작성합니다.

**매개변수:**

**dest (input)**

Destination 오브젝트입니다.

**리턴값:**

MessageConsumer 오브젝트입니다.

**예외:**

- XMSEException
- InvalidDestinationException

*CreateConsumer* - 이용자 작성(메시지 선택자 사용)

**인터페이스:**

```

IMessageConsumer CreateConsumer(IDestination dest,
                                String selector) ;

```

메시지 선택자를 사용하여 지정된 대상의 메시지 이용자를 작성합니다.

**매개변수:****dest (input)**

Destination 오브젝트입니다.

**selector (input)**

메시지 선택자 표현식을 캡슐화하는 String 오브젝트입니다. 메시지 선택자 표현식과 일치하는 특성이 있는 메시지만 메시지 이용자에 전달됩니다.

널 String 오브젝트는 메시지 이용자에 대한 메시지 선택자가 없음을 의미합니다.

**리턴값:**

MessageConsumer 오브젝트입니다.

**예외:**

- XMSEException
- InvalidDestinationException
- InvalidSelectorException

*CreateConsumer* - 이용자 작성(메시지 선택자 및 로컬 메시지 플래그 사용)

**인터페이스:**

```

IMessageConsumer CreateConsumer(IDestination dest,
                                String selector,
                                Boolean noLocal) ;

```

메시지 선택자를 사용하고 대상이 토픽인 경우 메시지 이용자가 자체 연결을 통해 공개된 메시지를 수신할지도 지정하여 지정된 대상의 메시지 이용자를 작성합니다.

**매개변수:****dest (input)**

Destination 오브젝트입니다.

**selector (input)**

메시지 선택자 표현식을 캡슐화하는 String 오브젝트입니다. 메시지 선택자 표현식과 일치하는 특성이 있는 메시지만 메시지 이용자에 전달됩니다.

널 String 오브젝트는 메시지 이용자에 대한 메시지 선택자가 없음을 의미합니다.

**noLocal (input)**

값을 True로 설정하면 메시지 이용자가 자체 연결을 통해 공개된 메시지를 수신하지 않습니다. 값을 False로 설정하면 메시지 이용자가 자체 연결을 통해 공개된 메시지를 수신합니다. 기본값은 False입니다.

**리턴값:**

MessageConsumer 오브젝트입니다.

#### 예외:

- XMSEException
- InvalidDestinationException
- InvalidSelectorException

*CreateDurableSubscriber* - 지속 가능한 구독자 작성

#### 인터페이스:

```
IMessageConsumer CreateDurableSubscriber(IDestination dest,  
                                         String subscription) ;
```

지정된 토픽의 지속 가능 구독자를 작성합니다.

이 메소드는 실시간 브로커 연결에는 유효하지 않습니다.

지속 가능한 구독자에 대한 자세한 정보는 [지속 가능한 구독자를 참조하십시오](#).

#### 매개변수:

##### **dest (input)**

토픽을 나타내는 Destination 오브젝트입니다. 토픽은 임시 토픽이 아니어야 합니다.

##### **subscription (input)**

지속 가능한 구독을 식별하는 이름을 캡슐화하는 String 오브젝트입니다. 이 이름은 연결에 대한 클라이언트 ID 내에서 고유해야 합니다.

#### 리턴값:

지속 가능한 구독자를 나타내는 MessageConsumer 오브젝트입니다.

#### 예외:

- XMSEException
- InvalidDestinationException

*CreateDurableSubscriber* - 지속 가능한 구독자 작성(메시지 선택자 및 로컬 메시지 플러그 사용)

#### 인터페이스:

```
IMessageConsumer CreateDurableSubscriber(IDestination dest,  
                                         String subscription,  
                                         String selector,  
                                         Boolean noLocal) ;
```

메시지 선택자를 사용하고 지속 가능 구독자가 자체 연결을 통해 게시된 메시지를 수신하는지 지정하여 지정된 토픽에 대한 지속 가능 구독자를 작성하십시오.

이 메소드는 실시간 브로커 연결에는 유효하지 않습니다.

지속 가능한 구독자에 대한 자세한 정보는 [지속 가능한 구독자를 참조하십시오](#).

#### 매개변수:

##### **dest (input)**

토픽을 나타내는 Destination 오브젝트입니다. 토픽은 임시 토픽이 아니어야 합니다.

##### **subscription (input)**

지속 가능한 구독을 식별하는 이름을 캡슐화하는 String 오브젝트입니다. 이 이름은 연결에 대한 클라이언트 ID 내에서 고유해야 합니다.

##### **selector (input)**

메시지 선택자 표현식을 캡슐화하는 String 오브젝트입니다. 메시지 선택자 표현식과 일치하는 특성이 있는 메시지만 지속 가능한 구독자에 전달됩니다.

널 String 오브젝트는 지속 가능한 구독자에 대한 메시지 선택자가 없음을 의미합니다.

**noLocal (input)**

값을 True로 설정하면 지속 가능한 구독자가 자체 연결을 통해 공개된 메시지를 수신하지 않습니다. 값을 False로 설정하면 지속 가능한 구독자가 자체 연결을 통해 공개된 메시지를 수신합니다. 기본값은 False입니다.

**리턴값:**

지속 가능한 구독자를 나타내는 MessageConsumer 오브젝트입니다.

**예외:**

- XMSEException
- InvalidDestinationException
- InvalidSelectorException

*CreateMapMessage* - 맵 메시지 작성

**인터페이스:**

```
IMapMessage CreateMapMessage();
```

맵 메시지를 작성합니다.

**매개변수:**

없음

**리턴값:**

MapMessage 오브젝트입니다.

**예외:**

- XMSEException
- IllegalStateException(세션이 닫힘)

*CreateMessage* - 메시지 작성

**인터페이스:**

```
IMessage CreateMessage();
```

본문이 없는 메시지를 작성합니다.

**매개변수:**

없음

**리턴값:**

Message 오브젝트입니다.

**예외:**

- XMSEException
- IllegalStateException(세션이 닫힘)

*CreateObject* 메시지-오브젝트 메시지 작성

**인터페이스:**

```
IObjectMessage CreateObjectMessage();
```

오브젝트 메시지를 작성합니다.

**매개변수:**

없음

**리턴값:**

ObjectMessage 오브젝트입니다.

**예외:**

- XMSEException
- IllegalStateException(세션이 닫힘)

*CreateProducer* - 작성자 작성

**인터페이스:**

```
IMessageProducer CreateProducer(IDestination dest) ;
```

지정된 대상으로 메시지를 전송하는 메시지 생성자를 작성합니다.

**매개변수:**

**dest (input)**

Destination 오브젝트입니다.

널 Destination 오브젝트를 지정하면 메시지 작성자가 목적지 없이 작성됩니다. 이 경우 애플리케이션은 메시지 작성자를 사용하여 메시지를 전송할 때마다 목적지를 지정해야 합니다.

**리턴값:**

MessageProducer 오브젝트입니다.

**예외:**

- XMSEException
- InvalidDestinationException

*CreateQueue* - 큐 작성

**인터페이스:**

```
IDestination CreateQueue(String queue) ;
```

메시지 서버의 큐를 나타내는 Destination 오브젝트를 작성합니다.

이 메소드는 메시지 서버에 큐를 작성하지 않습니다. 따라서 애플리케이션이 이 메소드를 호출하려면 먼저 큐를 작성해야 합니다.

**매개변수:**

**queue (input)**

큐 이름을 캡슐화하거나 큐를 식별하는 URI(Uniform Resource Identifier)를 캡슐화하는 String 오브젝트입니다.

**리턴값:**

큐를 나타내는 Destination 오브젝트입니다.

**예외:**

- XMSEException

*CreateStreamMessage* - 스트림 메시지 작성

**인터페이스:**

```
IStreamMessage CreateStreamMessage();
```

스트림 메시지를 작성합니다.

**매개변수:**

없음

**리턴값:**

StreamMessage 오브젝트입니다.

**예외:**

- XMSEException
- XMS\_ILLEGAL\_STATE\_EXCEPTION

*CreateTemporaryQueue* - 임시 큐 작성

**인터페이스:**

```
IDestination CreateTemporaryQueue() ;
```

임시 큐를 작성합니다.

임시 큐의 범위는 연결입니다. 연결에 의해 작성된 세션만 임시 큐를 사용할 수 있습니다.

임시 큐는 명시적으로 삭제되거나 연결이 끊길 때까지(둘 중에서 빠른 쪽이 적용됨) 그대로 유지됩니다.

임시 큐에 대한 자세한 정보는 [임시 목적지](#)를 참조하십시오.

**매개변수:**

없음

**리턴값:**

임시 큐를 나타내는 Destination 오브젝트입니다.

**예외:**

- XMSEException

*CreateTemporaryTopic* - 임시 토픽 작성

**인터페이스:**

```
IDestination CreateTemporaryTopic() ;
```

임시 토픽을 작성합니다.

임시 토픽의 범위는 연결입니다. 연결에 의해 작성된 세션만 임시 토픽을 사용할 수 있습니다.

임시 토픽은 명시적으로 삭제되거나 연결이 끊길 때까지(둘 중에서 빠른 쪽이 적용됨) 그대로 유지됩니다.

임시 토픽에 대한 자세한 정보는 [임시 목적지](#)를 참조하십시오.

**매개변수:**

없음

**리턴값:**

임시 토픽을 나타내는 Destination 오브젝트입니다.

**예외:**

- XMSEException

*CreateTextMessage* - 텍스트 메시지 작성

**인터페이스:**

```
ITextMessage CreateTextMessage();
```

본문이 비어 있는 텍스트 메시지를 작성합니다.

**매개변수:**

없음

**리턴값:**

TextMessage 오브젝트입니다.

**예외:**

- XMSEException

*CreateTextMessage* - 텍스트 메시지 작성(초기화됨)

**인터페이스:**

```
ITextMessage CreateTextMessage(String initialValue);
```

지정된 텍스트로 본문이 초기화되는 텍스트 메시지를 작성합니다.

**매개변수:**

**initialValue (input)**

텍스트 메시지의 본문을 초기화하는 텍스트를 캡슐화하는 String 오브젝트입니다.

없음

**리턴값:**

TextMessage 오브젝트입니다.

**예외:**

- XMSEException

*CreateTopic* - 토픽 작성

**인터페이스:**

```
IDestination CreateTopic(String topic) ;
```

토픽을 나타내는 Destination 오브젝트를 작성합니다.

**매개변수:**

**topic (input)**

토픽 이름을 캡슐화하거나 토픽을 식별하는 URI(Uniform Resource Identifier)를 캡슐화하는 String 오브젝트입니다.

**리턴값:**

토픽을 나타내는 Destination 오브젝트입니다.

**예외:**

- XMSEException

*Recover* - 복구

**인터페이스:**

```
void Recover();
```

세션을 복구합니다. 메시지 전달이 중지되었다가 수신확인되지 않은 가장 오래된 메시지를 사용하여 재시작됩니다.

세션은 트랜잭션된 세션이 아니어야 합니다.

세션 복구에 대한 자세한 정보는 [메시지 수신확인](#)을 참조하십시오.

**매개변수:**

없음

**리턴값:**

Void

**예외:**

- XMSEException

- `IllegalStateException`

*Rollback* - 롤백

인터페이스:

```
void Rollback();
```

현재 트랜잭션에서 처리된 모든 메시지를 롤백합니다.

세션은 트랜잭션된 세션이어야 합니다.

매개변수:

없음

리턴값:

`Void`

예외:

- `XMSEException`
- `IllegalStateException`

*Unsubscribe* - 구독 해제

인터페이스:

```
void Unsubscribe(String subscription);
```

지속 가능한 구독을 삭제합니다. 메시징 서버는 유지보수 중인 지속 가능 구독의 레코드를 삭제하고 지속 가능 구독자에게 추가 메시지를 전송하지 않습니다.

다음과 같은 경우 애플리케이션은 지속 가능한 구독을 삭제할 수 없습니다.

- 지속 가능한 구독에 대해 활성 메시지 이용자가 있는 경우
- 이용된 메시지가 보류 중인 트랜잭션의 일부인 경우
- 이용된 메시지가 수신확인되지 않은 경우

이 메소드는 실시간 브로커 연결에는 유효하지 않습니다.

매개변수:

**subscription (input)**

지속 가능한 구독을 식별하는 이름을 캡슐화하는 `String` 오브젝트입니다.

리턴값:

`Void`

예외:

- `XMSEException`
- `InvalidDestinationException`
- `IllegalStateException`

## 상속된 특성 및 메소드

다음 메소드는 `IPropertyContext` 인터페이스에서 상속됩니다.

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)



## IStreamMessage

스트림 메시지는 각 값에 연관된 데이터 유형이 있는 값 스트림으로 본문이 구성된 메시지입니다. 본문의 콘텐츠는 순차적으로 쓰여지고 읽힙니다.

상속 계층 구조:

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IMessage
|
+----IBM.XMS.IStreamMessage
```

애플리케이션이 메시지 스트림에서 값을 읽을 때 값은 XMS에 의해 다른 데이터 유형으로 변환될 수 있습니다. 이러한 내재적 변환 양식에 대한 자세한 정보는 [XMS 메시지의 본문](#)을 참조하십시오.

## 메소드

*ReadBoolean* - 부울 값 읽기

인터페이스:

```
Boolean ReadBoolean();
```

메시지 스트림에서 부울 값을 읽습니다.

매개변수:

없음

리턴값:

읽은 부울 값입니다.

예외:

- XMSException
- MessageNotReadableException
- MessageEOFException

*ReadByte* - 바이트 읽기

인터페이스:

```
Int16 ReadSignedByte();
Byte ReadByte();
```

메시지 스트림에서 부호가 있는 8비트 정수를 읽습니다.

매개변수:

없음

리턴값:

읽은 바이트입니다.

예외:

- XMSException
- MessageNotReadableException
- MessageEOFException

## ReadBytes - 바이트 읽기

### 인터페이스:

```
Int32 ReadBytes(Byte[] array);
```

메시지 스트림에서 바이트 배열을 읽습니다.

### 매개변수:

#### array (input)

바이트 배열을 포함하는 읽은 버퍼를 나타내며 바이트 단위의 버퍼 길이입니다.

배열의 바이트 수가 버퍼 길이보다 작거나 같을 경우 전체 배열이 버퍼로 읽힙니다. 배열의 바이트 수가 버퍼 길이보다 클 경우 버퍼가 배열의 일부로 채워지고 내부 커서가 다음으로 읽을 바이트의 위치를 표시합니다. readBytes()에 대한 이후 호출은 배열에서 현재 커서 위치에서 시작하는 바이트를 읽습니다.

입력 시 널 포인터를 지정하면 해당 호출이 바이트 배열을 읽지 않고 건너 뜁니다.

### 리턴값:

버퍼로 읽어 들인 바이트 수입니다. 버퍼의 일부만 채워진 경우 값이 버퍼 길이보다 작으며, 이는 배열에 더 이상 읽을 바이트가 남아 있지 않음을 나타냅니다. 호출하기 전에 배열에서 읽을 나머지 바이트가 없을 경우 값은 XMSC\_END\_OF\_BYTEARRAY입니다.

입력 시 널 포인터를 지정하면 이 메소드가 값을 리턴하지 않습니다.

### 예외:

- XMSEException
- MessageNotReadableException
- MessageEOFException

## ReadChar - 문자 읽기

### 인터페이스:

```
Char ReadChar();
```

메시지 스트림에서 2바이트 문자를 읽습니다.

### 매개변수:

없음

### 리턴값:

읽은 문자입니다.

### 예외:

- XMSEException
- MessageNotReadableException
- MessageEOFException

## ReadDouble - 배 정밀도 부동 소수점 수 읽기

### 인터페이스:

```
Double ReadDouble();
```

메시지 스트림에서 8바이트 배정밀도 부동 소수점 수를 읽습니다.

### 매개변수:

없음

### 리턴값:

읽은 배 정밀도 부동 소수점 수입니다.

**예외:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadFloat* - 부동 소수점 수 읽기

**인터페이스:**

```
Single ReadFloat();
```

메시지 스트림에서 4바이트 부동 소수점 수를 읽습니다.

**매개변수:**

없음

**리턴값:**

읽은 부동 소수점 수입니다.

**예외:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadInt* - 정수 읽기

**인터페이스:**

```
Int32 ReadInt();
```

메시지 스트림에서 부호 있는 32비트 정수를 읽습니다.

**매개변수:**

없음

**리턴값:**

읽은 정수입니다.

**예외:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadLong* - Long 정수 읽기

**인터페이스:**

```
Int64 ReadLong();
```

메시지 스트림에서 부호 있는 64비트 정수를 읽습니다.

**매개변수:**

없음

**리턴값:**

읽은 Long 정수입니다.

**예외:**

- XMSEException
- MessageNotReadableException

- MessageEOFException

*ReadObject* - 오브젝트 읽기

**인터페이스:**

```
Object ReadObject();
```

메시지 스트림에서 값을 읽고 해당 데이터 유형을 리턴합니다.

**매개변수:**

없음

**리턴값:**

다음 오브젝트 유형 중 하나인 값입니다.

Boolean  
Byte  
Byte[]  
Char  
Double  
Single  
Int32  
Int64  
Int16  
String

**예외:**

XMSEException

*ReadShort* - Short 정수 읽기

**인터페이스:**

```
Int16 ReadShort();
```

메시지 스트림에서 부호가 있는 16비트 정수를 읽습니다.

**매개변수:**

없음

**리턴값:**

읽은 Short 정수입니다.

**예외:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*ReadString* - 문자열 읽기

**인터페이스:**

```
String ReadString();
```

메시지 스트림에서 문자열을 읽습니다. 필요한 경우 XMS에서 문자열의 문자를 로컬 코드 페이지로 변환합니다.

**매개변수:**

없음

**리턴값:**

읽은 문자열을 캡슐화하는 String 오브젝트입니다. 데이터 변환이 필수인 경우 이 매개변수는 변환 후 문자열입니다.

**예외:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*Reset* - 재설정

**인터페이스:**

```
void Reset();
```

메시지 본문을 읽기 전용 모드로 설정하고 메시지 스트림의 시작 부분에서 커서 위치를 변경합니다.

**매개변수:**

없음

**리턴값:**

Void

**예외:**

- XMSEException
- MessageNotReadableException
- MessageEOFException

*WriteBoolean* - 부울 값 쓰기

**인터페이스:**

```
void WriteBoolean(Boolean value);
```

부울 값을 메시지 스트림에 씁니다.

**매개변수:****value (input)**

기록할 부울 값입니다.

**리턴값:**

Void

**예외:**

- XMSEException
- MessageNotWritableException

*WriteByte* - 바이트 쓰기

**인터페이스:**

```
void WriteByte(Byte value);
void WriteSignedByte(Int16 value);
```

바이트를 메시지 스트림에 씁니다.

**매개변수:****value (input)**

기록할 바이트입니다.

**리턴값:**

Void

**예외:**

- XMSEException
- MessageNotWritableException

*WriteBytes* - 바이트 쓰기**인터페이스:**

```
void WriteBytes(Byte[] value);
```

바이트 배열을 메시지 스트림에 씁니다.

**매개변수:****value (input)**

기록할 바이트 배열입니다.

**length (input)**

배열의 바이트 수입니다.

**리턴값:**

Void

**예외:**

- XMSEException
- MessageNotWritableException

*WriteChar* - 문자 쓰기**인터페이스:**

```
void WriteChar(Char value);
```

메시지 스트림에 문자를 2바이트로 쓰며, 상위 바이트를 먼저 씁니다.

**매개변수:****value (input)**

기록할 문자입니다.

**리턴값:**

Void

**예외:**

- XMSEException
- MessageNotWritableException

*WriteDouble* - 배 정밀도 부동 소수점 수 쓰기**인터페이스:**

```
void WriteDouble(Double value);
```

배정밀도 부동 소수점 수를 long 정수로 변환하고 long 정수를 메시지 스트림에 8바이트로 쓰며, 상위 바이트 먼저 씁니다..

**매개변수:****value (input)**

기록할 배 정밀도 부동 소수점 수입니다.

**리턴값:**

Void

**예외:**

- XMSEException
- MessageNotWritableException

*WriteFloat* - 부동 소수점 수 쓰기**인터페이스:**

```
void WriteFloat(Single value);
```

부동 소수점 수를 정수로 변환하고 정수를 4바이트로 메시지 스트림에 쓰며, 상위 바이트 먼저 씁니다.

**매개변수:****value (input)**

기록할 부동 소수점 수입니다.

**리턴값:**

Void

**예외:**

- XMSEException
- MessageNotWritableException

*WriteInt* - 정수 쓰기**인터페이스:**

```
void WriteInt(Int32 value);
```

메시지 스트림에 정수를 4바이트로 쓰고, 상위 바이트를 먼저 씁니다.

**매개변수:****value (input)**

기록할 정수입니다.

**리턴값:**

Void

**예외:**

- XMSEException
- MessageNotWritableException

*WriteLong* - Long 정수 쓰기**인터페이스:**

```
void WriteLong(Int64 value);
```

메시지 스트림에 long 정수를 8바이트로 쓰고, 상위 바이트를 먼저 씁니다.

**매개변수:****value (input)**

기록할 Long 정수입니다.

**리턴값:**

Void

**예외:**

- XMSEException
- MessageNotWritableException

*WriteObject* - 오브젝트 쓰기

**인터페이스:**

```
void WriteObject(Object value);
```

지정된 데이터 유형의 값을 메시지 스트림에 씁니다.

**매개변수:****objectType (input)**

다음 오브젝트 유형 중 하나인 값입니다.

Boolean  
Byte  
Byte[]  
Char  
Double  
Single  
Int32  
Int64  
Int16  
String

**value (input)**

기록할 값을 포함하는 바이트 배열입니다.

**length (input)**

배열의 바이트 수입니다.

**리턴값:**

Void

**예외:**

- XMSEException

*WriteShort* - Short 정수 쓰기

**인터페이스:**

```
void WriteShort(Int16 value);
```

메시지 스트림에 short 정수를 2바이트로 쓰고, 상위 바이트를 먼저 씁니다.

**매개변수:****value (input)**

기록할 Short 정수입니다.

**리턴값:**

Void

**예외:**

- XMSEException
- MessageNotWritableException



*WriteString* - 문자열 쓰기

인터페이스:

```
void WriteString(String value);
```

문자열을 메시지 스트림에 씁니다.

매개변수:

**value (input)**

기록할 문자열을 캡슐화하는 String 오브젝트입니다.

리턴값:

Void

예외:

- XMSException
- MessageNotWritableException

## 상속된 특성 및 메소드

다음 특성은 *IMessage* 인터페이스에서 상속됩니다.

[JMSCorrelationID](#), [JMSDeliveryMode](#), [JMSDestination](#), [JMSExpiration](#), [JMSMessageID](#), [JMSPriority](#), [JMSRe교부](#), [JMSReplyTo](#), [JMSTim환원](#), [JMSType](#), 특성

다음 메소드는 *IMessage* 인터페이스에서 상속됩니다.

[clearBody](#), [clearProperties](#), [PropertyExists](#)

다음 메소드는 *IPropertyContext* 인터페이스에서 상속됩니다.

[GetBooleanProperty](#), [GetByteProperty](#), [GetBytesProperty](#), [GetCharProperty](#), [GetDoubleProperty](#), [GetFloatProperty](#), [GetIntProperty](#), [GetLongProperty](#), [GetObjectProperty](#), [GetShortProperty](#), [GetStringProperty](#), [SetBooleanProperty](#), [SetByteProperty](#), [SetBytesProperty](#), [SetCharProperty](#), [SetDoubleProperty](#), [SetFloatProperty](#), [SetIntProperty](#), [SetLongProperty](#), [SetObjectProperty](#), [SetShortProperty](#), [SetStringProperty](#)

## ITextMessage

텍스트 메시지는 본문이 문자열로 구성된 메시지입니다.

상속 계층 구조:

```
IBM.XMS.IPropertyContext
|
+----IBM.XMS.IMessage
|
+----IBM.XMS.ITextMessage
```

## .NET 특성

*Text* - 텍스트 가져오기 및 설정

인터페이스:

```
String Text
{
    get;
    set;
}
```

텍스트 메시지의 본문을 구성하는 문자열을 가져오고 설정합니다.

필요한 경우 XMS에서 문자열의 문자를 로컬 코드 페이지로 변환합니다.

예외:

- XMSException
- MessageNotReadableException
- MessageNotWritableException
- MessageEOFException

## 상속된 특성 및 메소드

다음 특성은 `IMessage` 인터페이스에서 상속됩니다.

`JMSCorrelationID`, `JMSDeliveryMode`, `JMSDestination`, `JMSExpiration`, `JMSMessageID`, `JMSPriority`, `JMSRe교부`, `JMSReplyTo`, `JMSTim환원`, `JMSType`, 특성

다음 메소드는 `IMessage` 인터페이스에서 상속됩니다.

`clearBody`, `clearProperties`, `PropertyExists`

다음 메소드는 `IPropertyContext` 인터페이스에서 상속됩니다.

`GetBooleanProperty`, `GetByteProperty`, `GetBytesProperty`, `GetCharProperty`, `GetDoubleProperty`, `GetFloatProperty`, `GetIntProperty`, `GetLongProperty`, `GetObjectProperty`, `GetShortProperty`, `GetStringProperty`, `SetBooleanProperty`, `SetByteProperty`, `SetBytesProperty`, `SetCharProperty`, `SetDoubleProperty`, `SetFloatProperty`, `SetIntProperty`, `SetLongProperty`, `SetObjectProperty`, `SetShortProperty`, `SetStringProperty`

## TransactionInProgressException

XMS에서는 트랜잭션이 진행 중이므로 유효하지 않은 조작을 애플리케이션에서 요청할 경우 이 예외 처리를 합니다.

상속 계층 구조:

```
IBM.XMS.XMSException
|
+----IBM.XMS.XMSException
      |
      +----IBM.XMS.TransactionInProgressException
```

## 상속된 특성 및 메소드

다음 메소드는 `XMSException` 인터페이스에서 상속됩니다.

`GetError코드`, `GetLinked예외`

## TransactionRolledBackException

XMS에서는 애플리케이션이 현재 트랜잭션을 확약하기 위해 `Session.commit()`를 호출하지만 트랜잭션이 롤백 되는 경우에 이 예외를 처리합니다.

상속 계층 구조:

```
IBM.XMS.XMSException
|
+----IBM.XMS.XMSException
      |
      +----IBM.XMS.TransactionRolledBackException
```

## 상속된 특성 및 메소드

다음 메소드는 `XMSException` 인터페이스에서 상속됩니다.

`GetError코드`, `GetLinked예외`

## XMSException

XMS이(가) .NET 메소드에 대한 호출을 처리하는 중에 오류를 감지하면 XMS에서 예외 처리합니다. 예외는 오류 정보를 캡슐화하는 오브젝트입니다.

상속 계층 구조:

```
System.Exception
|
+----IBM.XMS.XMSException
```

다른 유형의 XMS 예외가 있으며, XMSException 오브젝트는 하나의 예외 유형일 뿐입니다. 그러나 XMSException 클래스는 다른 XMS 예외 클래스의 슈퍼클래스입니다. XMS는 다른 유형의 예외가 적합하지 않은 경우에 XMSException 오브젝트를 처리합니다.

## .NET 특성

*ErrorCode* - 오류 코드 가져오기

인터페이스:

```
public String ErrorCode
{
    get {return errorCode_;}
}
```

오류 코드를 가져옵니다.

예외:

- XMSException

*LinkedException* - 링크된 예외 가져오기

인터페이스:

```
public Exception LinkedException
{
    get { return linkedException_;}
    set { linkedException_ = value;}
}
```

예외 체인에서 다음 예외를 가져옵니다.

체인에 더 이상 예외가 없으면 메소드는 널을 리턴합니다.

예외:

- XMSException

## XMSFactoryFactory

애플리케이션에서 관리 오브젝트를 사용하지 않을 경우 이 클래스를 사용하여 연결 팩토리, 큐, 토픽을 작성하십시오.

상속 계층 구조:

없음

## .NET 특성

*Metadata* - 메타데이터 검색

인터페이스:

```
IConnectionMetaData MetaData
```

XMSFactoryFactory 오브젝트의 연결 유형에 적합한 메타데이터를 가져옵니다.

예외:

없음

메소드

*CreateConnectionFactory* - 연결 팩토리 작성

인터페이스:

```
IConnectionFactory CreateConnectionFactory();
```

선언된 유형의 ConnectionFactory 오브젝트를 작성합니다.

매개변수:

없음

리턴값:

ConnectionFactory 오브젝트입니다.

예외:

- XMSEException

*CreateQueue* - 큐 작성

인터페이스:

```
IDestination CreateQueue(String name);
```

메시지 서버의 큐를 나타내는 Destination 오브젝트를 작성합니다.

이 메소드는 메시지 서버에 큐를 작성하지 않습니다. 따라서 애플리케이션이 이 메소드를 호출하려면 먼저 큐를 작성해야 합니다.

매개변수:

**name (input)**

큐 이름을 캡슐화하거나 큐를 식별하는 URI(Uniform Resource Identifier)를 캡슐화하는 String 오브젝트입니다.

리턴값:

큐를 나타내는 Destination 오브젝트입니다.

예외:

- XMSEException

*CreateTopic* - 토픽 작성

인터페이스:

```
IDestination CreateTopic(String name);
```

토픽을 나타내는 Destination 오브젝트를 작성합니다.

매개변수:

**name (input)**

토픽 이름을 캡슐화하거나 토픽을 식별하는 URI(Uniform Resource Identifier)를 캡슐화하는 String 오브젝트입니다.

리턴값:

토픽을 나타내는 Destination 오브젝트입니다.

예외:

- XMSEException

*GetInstance* - XMSFactoryFactory의 인스턴스 가져오기

인터페이스:

```
static XMSFactoryFactory GetInstance(int connectionType);
```

XMSFactoryFactory의 인스턴스를 작성합니다. XMS 애플리케이션은 XMSFactoryFactory 오브젝트를 사용하여 필요한 프로토콜 유형에 적합한 ConnectionFactory 오브젝트에 대한 참조를 가져옵니다. 그러면, 이 ConnectionFactory 오브젝트는 해당 프로토콜 유형에 대한 연결만 생성할 수 있습니다.

매개변수:

**connectionType (input)**

ConnectionFactory 오브젝트가 연결을 생성하는 연결 유형입니다.

- XMSC.CT\_WPM
- XMSC.CT\_RTT
- XMSC.CT\_WMQ

리턴값:

선언된 연결 유형에만 사용되는 XMSFactoryFactory 오브젝트입니다.

예외:

- NotSupportedException

## XMS 오브젝트의 특성

이 절에서는 XMS가 정의하는 오브젝트 특성에 대해 설명합니다.

이 절에는 다음 오브젝트 유형에 대한 정보가 포함되어 있습니다.

- [1946 페이지의 『Connection 특성』](#)
- [1946 페이지의 『ConnectionFactory의 특성』](#)
- [1950 페이지의 『ConnectionMetaData 특성』](#)
- [1950 페이지의 『Destination 특성』](#)
- [1952 페이지의 『InitialContext 특성』](#)
- [1952 페이지의 『Message 특성』](#)
- [1957 페이지의 『MessageConsumer 특성』](#)
- [1957 페이지의 『MessageProducer 특성』](#)
- [1957 페이지의 『Session 특성』](#)

각 오브젝트 유형에 대한 설명은 지정된 유형의 오브젝트 특성을 나열하고 각 특성에 대한 간단한 설명을 제공합니다.

또한 이 절에서는 각 특성의 정의를 제공합니다([1957 페이지의 『특성 정의』](#) 참조).

애플리케이션이 이 절에서 설명하는 오브젝트의 특성을 정의하는 경우 오류가 발생하지 않지만 예기치 않은 결과가 발생할 수 있습니다.

**참고:** 이 섹션의 특성 이름과 값은 C 및 C++에 사용되는 양식인 XMSC.NAME 양식으로 표시됩니다. 그러나 .NET에서 특성 이름의 양식은 사용하는 방법에 따라 XMSC.NAME 또는 XMSC\_NAME일 수 있습니다.

- 특성을 지정하는 경우 특성 이름은 다음 예에 표시된 대로 XMSC.NAME 양식이어야 합니다.

```
cf.SetStringProperty(XMSC.WMQ_CHANNEL, "DOTNET.SVRCONN");
```

- 문자열을 지정하는 경우 특성 이름은 다음 예에 표시된 대로 XMSC\_NAME 양식이어야 합니다.

```
cf.SetStringProperty("XMSC_WMQ_CHANNEL", "DOTNET.SVRCONN");
```

.NET에서는 특성 이름 및 값이 XMSC 클래스의 상수로 제공됩니다. 이러한 상수는 문자열을 식별하며 XMS.NET 애플리케이션에서 사용될 수 있습니다. 사전정의된 상수를 사용하는 경우 특성 이름과 값은 XMSC.NAME 양식으로 사용됩니다. 예를 들어 XMSC\_USERID가 아닌 XMSC.USERID를 사용합니다.

또한 데이터 유형은 C/C++에 사용되는 양식입니다. .NET의 데이터 유형에서 .NET에 해당하는 값을 찾을 수 있습니다.

## Connection 특성

Connection 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

표 872. Connection 특성	
특성 이름	설명
1989 페이지의 『XMSC_WMQ_RESOLVED_QUEUE_MANAGER』	이 특성을 사용하여 연결된 큐 관리자의 이름을 얻을 수 있습니다.
1989 페이지의 『XMSC_WMQ_RESOLVED_QUEUE_MANAGER_ID』	이 특성은 연결 후에 큐 관리자의 ID로 채워집니다.
XMSC_WPM_CONNECTION_PROTOCOL	메시징 엔진에 연결하는 데 사용되는 통신 프로토콜입니다. 이 특성은 읽기 전용입니다.
XMSC_WPM_HOST_NAME	애플리케이션이 연결되는 메시징 엔진이 포함된 시스템의 호스트 이름 또는 IP 주소입니다. 이 특성은 읽기 전용입니다.
XMSC_WPM_ME_NAME	애플리케이션이 연결되는 메시징 엔진의 이름입니다. 이 특성은 읽기 전용입니다.
XMSC_WPM_PORT	애플리케이션이 연결되는 메시징 엔진이 대기하는 포트의 번호입니다. 이 특성은 읽기 전용입니다.

연결 오브젝트에도 연결하기 위해 사용된 연결 팩토리의 특성에서 파생된 읽기 전용 특성이 있습니다. 이러한 특성은 연결될 시점에 설정된 연결 팩토리 특성에서 파생된 것 뿐만 아니라 설정되지 않은 특성의 기본값에서 파생된 것도 있습니다. 특성에는 애플리케이션이 연결된 메시징 서버의 유형과 관련된 특성만 포함됩니다. 특성의 이름은 연결 팩토리 특성의 이름과 동일합니다.

## ConnectionFactory의 특성

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

표 873. ConnectionFactory의 특성	
특성 이름	설명
1966 페이지의 『XMSC_ASYNC_EXCEPTIONS』	이 특성은 연결이 끊기거나 XMS API 호출에 비동기적으로 예외가 발생하는 경우에만 XMS에서 ExceptionListener에 알리는지 여부를 판별합니다. 이 특성은 등록된 ExceptionListener가 있고 이 ConnectionFactory에서 작성된 모든 연결에 적용됩니다.

표 873. ConnectionFactory의 특성 (계속)	
특성 이름	설명
1974 페이지의 『XMSC_WMQ_BALANCING_APPLICATION_TYPE』	밸런싱 옵션 유형
1974 페이지의 『XMSC_WMQ_BALANCING_OPTIONS』	issuing 애플리케이션에서 설정한 밸런싱 옵션
1975 페이지의 『XMSC_WMQ_BALANCING_TIMEOUT』	리밸런싱으로 인해 애플리케이션 활동이 중단될 수 있는 제한 시간 초과입니다.
XMSC_CLIENT_ID	연결에 사용하는 클라이언트 ID입니다.
XMSC_CONNECTION_TYPE	애플리케이션이 연결되는 메시징 서버의 유형입니다.
XMSC_PASSWORD	애플리케이션이 메시징 서버에 연결할 때 해당 애플리케이션을 인증하는 데 사용할 수 있는 비밀번호입니다.
1971 페이지의 『XMSC_RTT_BROKER_PING_INTERVAL』	이 시간 간격(밀리초) 후 XMS.NET에서 활동을 감지하기 위해 실시간 메시징 서버에 대한 연결을 검사합니다.
XMSC_RTT_CONNECTION_PROTOCOL	브로커에 대한 실시간 연결에 사용되는 통신 프로토콜입니다.
XMSC_RTT_HOST_NAME	브로커가 실행하는 시스템의 호스트 이름 또는 IP 주소입니다.
XMSC_RTT_LOCAL_ADDRESS	브로커에 대한 실시간 연결에 사용되는 로컬 네트워크 인터페이스의 호스트 이름 또는 IP 주소입니다.
XMSC_RTT_MULTICAST	연결 팩토리 또는 목적지의 멀티캐스트 설정입니다.
XMSC_RTT_PORT	브로커가 수신 요청을 청취하는 포트 번호입니다.
XMSC_USERID	애플리케이션이 메시징 서버에 연결할 때 해당 애플리케이션을 인증하는 데 사용할 수 있는 사용자 ID입니다.
XMSC_WMQ_BROKER_CONTROLQ	브로커가 사용하는 제어 큐의 이름입니다.
XMSC_WMQ_BROKER_PUBQ	브로커가 모니터하는 큐의 이름입니다. 애플리케이션은 발행하는 메시지를 이 큐로 전송합니다.
XMSC_WMQ_BROKER_QMGR	브로커가 연결되는 큐 관리자의 이름입니다.
XMSC_WMQ_BROKER_SUBQ	지속 불가능한 메시지 이용자의 구독자 큐 이름입니다.
XMSC_WMQ_BROKER_VERSION	애플리케이션이 연결이나 목적지에 사용하는 브로커의 유형입니다.
1977 페이지의 『XMSC_WMQ_CCDTURL』	클라이언트 채널 정의 테이블을 포함하는 파일의 이름과 위치를 식별하고 이 파일의 액세스 방법을 지정하는 URL(Uniform Resource Locator)입니다.
XMSC_WMQ_CHANNEL	연결에 사용되는 채널 이름입니다.
1978 페이지의 『XMSC_WMQ_CLIENT_RECONNECT_OPTIONS』	이 특성은 이 팩토리에서 작성된 새 연결에 대한 클라이언트 다시 연결 옵션을 지정합니다.
1978 페이지의 『XMSC_WMQ_CLIENT_RECONNECT_TIMEOUT』	이 특성은 클라이언트 연결이 다시 연결하려고 시도하는 시간(초)을 지정합니다.
XMSC_WMQ_CONNECTION_MODE	애플리케이션이 큐 관리자에 연결할 때 사용되는 모드입니다.
1979 페이지의 『XMSC_WMQ_CONNECTION_NAME_LIST』	이 특성은 연결이 끊어진 후 클라이언트가 다시 연결하려고 시도하는 호스트를 지정합니다.

표 873. ConnectionFactory의 특성 (계속)	
특성 이름	설명
<a href="#">XMSC_WMQ_FAIL_IF_QUIESCE</a>	애플리케이션이 연결된 큐 관리자가 정지 상태인 경우 특정 메소드에 대한 호출의 실패 여부를 지정합니다.
<a href="#">XMSC_WMQ_HOST_NAME</a>	큐 관리자가 실행하는 시스템의 호스트 이름 또는 IP 주소입니다.
<a href="#">XMSC_WMQ_LOCAL_ADDRESS</a>	큐 관리자에 대한 연결의 경우 이 특성은 사용할 로컬 네트워크 인터페이스를 지정하거나, 사용할 로컬 포트나 로컬 포트의 범위 또는 둘 다를 지정합니다.
<a href="#">XMSC_WMQ_MESSAGE_SELECTION</a>	메시지 선택이 수행되는지 여부를 결정합니다.XMS 클라이언트 또는 브로커에 의해.
<a href="#">XMSC_WMQ_MSG_BATCH_SIZE</a>	비동기 메시지 전달을 사용할 때 하나의 일괄처리로 큐에서 검색되는 최대 메시지 수입니다.
<a href="#">XMSC_WMQ_POLLING_INTERVAL</a>	세션에 있는 각 메시지 리스너의 큐에 적합한 메시지가 없는 경우, 이 값은 각 메시지 리스너가 해당 큐에서 메시지 가져오기를 다시 시도하기 전에 경과하는 최대 간격(밀리세컨드)입니다.
1987 페이지의 <a href="#">『XMSC_WMQ_PROVIDER_VERSION』</a>	애플리케이션이 연결하려고 하는 큐 관리자의 버전, 릴리스, 수정 레벨 및 수정팩입니다.
<a href="#">XMSC_WMQ_PORT</a>	큐 관리자 리스너가 수신 요청을 청취하는 포트 번호입니다.
<a href="#">XMSC_WMQ_PUB_ACK_INTERVAL</a>	게시자가 게시하기 전에 게시한 메시지 수XMS 클라이언트는 브로커에게 승인을 요청합니다.
1983 페이지의 <a href="#">『XMSC_WMQ_PUT_ASYNC_ALLOWED』</a>	이 특성은 메시지 생성자가 비동기 Put을 사용하여 메시지를 이 목적지로 송신할 수 있는지 여부를 판별합니다.
<a href="#">XMSC_WMQ_QMGR_CCSID</a>	MQI(Message Queue Interface)에 정의된 문자 데이터 필드가 다음 사이에서 교환되는 코드화된 문자 세트 또는 코드 페이지의 식별자(CCSID)입니다.XMS 클라이언트와IBM MQ 고객.
<a href="#">XMSC_WMQ_QUEUE_MANAGER</a>	연결된 큐 관리자의 이름.
<a href="#">XMSC_WMQ_RECEIVE_EXIT</a>	실행할 채널 수신 종료를 식별합니다.
<a href="#">XMSC_WMQ_RECEIVE_EXIT_INIT</a>	호출 시 채널 수신 종료에 전달되는 사용자 데이터입니다.
<a href="#">XMSC_WMQ_SECURITY_EXIT</a>	채널 보안 엑시트를 식별합니다.
<a href="#">XMSC_WMQ_SECURITY_EXIT_INIT</a>	호출 시 채널 보안 엑시트로 전달되는 사용자 데이터입니다.
1991 페이지의 <a href="#">『XMSC_WMQ_SEND_CHECK_COUNT』</a>	변환되지 않은 하나의 XMS 세션에서 비동기 put 오류를 검사하는 사이에 허용할 수 있는 송신 호출 수입니다.
<a href="#">XMSC_WMQ_SEND_EXIT</a>	채널 전송 종료를 식별합니다.
<a href="#">XMSC_WMQ_SEND_EXIT_INIT</a>	호출 시 채널 송신 엑시트로 전달되는 사용자 데이터입니다.
1991 페이지의 <a href="#">『XMSC_WMQ_SHARE_CONV_ALLOWED』</a>	클라이언트 연결이 다른 최상위 레벨과 소켓을 공유할 수 있는지 여부XMS 채널 정의가 일치하는 경우 동일한 프로세스에서 동일한 큐 관리자로 연결됩니다. 이 특성은 애플리케이션 개발 또는 유지보수 또는 경영 목적을 위해 필요한 경우 별도의 소켓에서 연결의 완전한 격리를 허용하기 위해 제공됩니다.
<a href="#">XMSC_WMQ_SSL_CERT_STORES</a>	큐 관리자에 대한 SSL(Secure Socket Layer) 연결에 사용되는 인증서 폐기 목록(CRL)이 있는 서버의 위치입니다.
<a href="#">XMSC_WMQ_SSL_CIPHER_SPEC</a>	큐 관리자에 대한 보안 연결에 사용할 CipherSpec의 이름.



표 873. ConnectionFactory의 특성 (계속)	
특성 이름	설명
<u>XMSC_WMQ_SSL_CIPHER_SUITE</u>	큐 관리자에 대한 TLS 연결에 사용될 CipherSuite의 이름입니다. 보안 연결 협상에 사용되는 프로토콜은 지정된 CipherSuite에 따라 달라집니다.
<u>XMSC_WMQ_SSL_CRYPTO_HW</u>	클라이언트 시스템에 연결된 암호화 하드웨어의 구성 세부사항입니다.
<u>XMSC_WMQ_SSL_FIPS_REQUIRED</u>	이 특성의 값은 애플리케이션이 비FIPS 준수 암호 스위트를 사용할 수 있는지 여부를 판별합니다. 이 특성이 true로 설정되는 경우 FIPS 알고리즘만 클라이언트-서버 연결에 사용됩니다.
<u>XMSC_WMQ_SSL_KEY_REPOSITORY</u>	키 및 인증서가 저장되는 키 데이터베이스 파일의 위치입니다.
<u>XMSC_WMQ_SSL_KEY_RESETCOUNT</u>	KeyResetCount는 비밀 키가 재협상되기 전에 SSL 통신에서 송신되고 수신된 암호화되지 않은 총 바이트 수를 나타냅니다.
<u>XMSC_WMQ_SSL_PEER_NAME</u>	큐 관리자에 대한 SSL(Secure Socket Layer) 연결에 사용되는 피어 이름입니다.
<u>XMSC_WMQ_SYNCPOINT_ALL_GETS</u>	동기점 제어에 있는 큐에서 모든 메시지를 검색해야 하는지 여부를 결정합니다.
1997 페이지의 『XMSC_WMQ_TARGET_CLIENT』	
<u>XMSC_WMQ_TEMP_Q_PREFIX</u>	이름을 형성하는 데 사용되는 접두사IBM MQ 애플리케이션이 생성될 때 생성되는 동적 큐XMS 임시 대기열.
<u>XMSC_WMQ_TEMP_TOPIC_PREFIX</u>	임시 주제를 만들 때,XMS " 형식의 토픽 문자열을 생성합니다.TEMP/TEMPTOPICPREFIX/unique_id" 또는 이 속성에 기본값이 포함된 경우 이 문자열은 "TEMP/unique_id"가 생성됩니다. 비어 있지 않은 값을 지정하면 이 연결 아래에 작성된 임시 토픽에 대한 구독자의 관리 큐를 작성하기 위한 특정 모델 큐를 정의할 수 있습니다.
<u>XMSC_WMQ_TEMPORARY_MODEL</u>	이름은IBM MQ 애플리케이션이 생성될 때 동적 큐가 생성되는 모델 큐XMS 임시 대기열.
<u>XMSC_WPM_BUS_NAME</u>	연결 팩토리의 경우, 애플리케이션이 연결하는 서비스 통합 버스의 이름이거나, 또는 목적지의 경우에는 목적지가 존재하는 서비스 통합 버스의 이름입니다.
<u>XMSC_WPM_CONNECTION_PROXIMITY</u>	연결에 대한 연결 근접성 설정입니다.
<u>XMSC_WPM_DUR_SUB_HOME</u>	목적지나 연결에 대한 모든 지속 가능한 구독을 관리하는 메시징 엔진의 이름입니다.
<u>XMSC_WPM_LOCAL_ADDRESS</u>	서비스 통합 버스에 대한 연결의 경우 이 특성은 사용할 로컬 네트워크 인터페이스를 지정하거나, 사용할 로컬 포트나 로컬 포트의 범위 또는 둘 다를 지정합니다.
<u>XMSC_WPM_NON_PERSISTENT_MAP</u>	연결을 사용하여 송신되는 비지속적 메시지의 신뢰도 레벨입니다.
<u>XMSC_WPM_PERSISTENT_MAP</u>	연결을 사용하여 송신되는 지속 메시지의 신뢰도 레벨입니다.
<u>XMSC_WPM_PROVIDER_ENDPOINTS</u>	부트스트랩 서버에 사용되는 하나 이상의 엔드포인트 주소의 순서입니다.
<u>XMSC_WPM_TARGET_GROUP</u>	메시징 엔진의 대상 그룹 이름입니다.
<u>XMSC_WPM_TARGET_SIGNIFICANCE</u>	메시징 엔진의 대상 그룹에 대한 중요성입니다.

표 873. <i>ConnectionFactory</i> 의 특성 (계속)	
특성 이름	설명
<u>XMSC_WPM_TARGET_TRANSPORT_CHAIN</u>	애플리케이션이 메시징 엔진에 연결하는 데 사용해야 하는 인바운드 전송 체인의 이름입니다.
<u>XMSC_WPM_TARGET_TYPE</u>	메시징 엔진의 대상 그룹 유형입니다.
<u>XMSC_WPM_TEMP_Q_PREFIX</u>	애플리케이션이 서비스 통합 버스를 생성할 때 생성되는 임시 대기열의 이름을 구성하는 데 사용되는 접두사입니다.XMS 임시 대기열.
<u>XMSC_WPM_TEMP_TOPIC_PREFIX</u>	애플리케이션이 작성한 임시 토픽의 이름을 구성하는 데 사용되는 접두부입니다.

## ConnectionMetaData 특성

ConnectionMetaData 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

표 874. <i>ConnectionMetaData</i> 특성	
특성 이름	설명
<u>XMSC_JMS_MAJOR_VERSION</u>	주요 버전 번호JMS 사양XMS 기반으로합니다. 이 특성은 읽기 전용입니다.
<u>XMSC_JMS_MINOR_VERSION</u>	부 버전 번호JMS 사양XMS 기반으로합니다. 이 특성은 읽기 전용입니다.
<u>XMSC_JMS_VERSION</u>	버전 식별자JMS 사양XMS 기반으로합니다. 이 특성은 읽기 전용입니다.
<u>XMSC_MAJOR_VERSION</u>	버전 번호XMS 고객. 이 특성은 읽기 전용입니다.
<u>XMSC_MINOR_VERSION</u>	릴리스 번호XMS 고객. 이 특성은 읽기 전용입니다.
<u>XMSC_PROVIDER_NAME</u>	공급자는XMS 고객. 이 특성은 읽기 전용입니다.
<u>XMSC_VERSION</u>	cli의 버전 식별자XMS 엔트. 이 속성은 읽기 전용입니다.

## Destination 특성

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

표 875. <i>Destination</i> 특성	
특성 이름	설명
<u>XMSC_DELIVERY_MODE</u>	목적지에 전송되는 메시지의 전달 모드입니다.
<u>XMSC_PRIORITY</u>	목적지에 전송되는 메시지의 우선순위입니다.
<u>XMSC_RTT_MULTICAST</u>	연결 팩토리 또는 목적지의 멀티캐스트 설정입니다.
<u>XMSC_TIME_TO_LIVE</u>	목적지에 송신되는 메시지의 TTL(Time to Live).
<u>XMSC_WMQ_BROKER_VERSION</u>	애플리케이션이 연결이나 목적지에 사용하는 브로커의 유형입니다.
<u>XMSC_WMQ_CCSID</u>	메시지 본문의 문자 데이터 문자열이 있는 코드화된 문자 세트 또는 코드 페이지의 ID(CCSID)입니다.XMS 클라이언트는 메시지를 대상으로 전달합니다.

표 875. Destination 특성 (계속)	
특성 이름	설명
<u>XMSC_WMQ_DUR_SUBQ</u>	목적지에서 메시지를 수신하는 지속 가능한 구독자의 구독자 큐 이름입니다.  <b>참고:</b> 이 특성은 IBM Message Service Client for .NET 버전 2.0에 사용할 수 있지만, 연결 팩토리의 XMSC_WMQ_PROVIDER_VERSION 특성이 7 미만의 버전 번호로 설정되는 경우를 제외하고 IBM WebSphere MQ 7.0 큐 관리자에 연결된 애플리케이션에 대해서는 영향을 미치지 않습니다.
<u>XMSC_WMQ_ENCODING</u>	메시지 본문의 숫자 데이터가 표시되는 방식 XMS 클라이언트는 메시지를 대상으로 전달합니다.
<u>XMSC_WMQ_FAIL_IF_QUIESCE</u>	애플리케이션이 연결된 큐 관리자가 정지 상태인 경우 특정 메소드에 대한 호출의 실패 여부를 지정합니다.
1981 페이지의 『 <u>XMSC_WMQ_MESSAGE_BODY</u> 』	이 속성은 XMS 애플리케이션이 처리하는 MQRFH2 ~의 IBM MQ 메시지 페이로드의 일부(즉, 메시지 본문의 일부)로 메시지를 보냅니다.
1981 페이지의 『 <u>XMSC_WMQ_MQMD_MESSAGE_CONTEXT</u> 』	메시지 컨텍스트의 수준을 결정합니다. XMS 애플리케이션. 이 특성이 적용되려면 애플리케이션이 적합한 컨텍스트 권한으로 실행 중이어야 합니다.
1982 페이지의 『 <u>XMSC_WMQ_MQMD_READ_ENABLED</u> 』	이 속성은 XMS 애플리케이션은 MQMD 필드의 값을 추출할 수 있는지 여부를 결정합니다.
1982 페이지의 『 <u>XMSC_WMQ_MQMD_WRITE_ENABLED</u> 』	이 속성은 XMS 애플리케이션은 MQMD 필드의 값을 설정할지 여부를 결정할 수 있습니다.
1983 페이지의 『 <u>XMSC_WMQ_READ_AHEAD_ALLOWED</u> 』	이 특성은 메시지를 수신하기 전에 메시지 이용자와 큐 브라우저가 이 목적지에서 내부 버퍼로 트랜잭트되지 않은 비지속적 메시지를 가져오기 위해 미리 읽기를 사용할 수 있는지 여부를 판별합니다.
1984 페이지의 『 <u>XMSC_WMQ_READ_AHEAD_CLOSE_POLICY</u> 』	비동기 메시지 리스너로 전달되는 메시지의 경우, 메시지 이용자를 닫을 때 내부 미리 읽기 버퍼에서 메시지에 일어나는 사항을 이 특성에서 판별합니다.
1988 페이지의 『 <u>XMSC_WMQ_RECEIVE_CCSID</u> 』	큐 관리자 메시지 변환용 대상 CCSID를 설정하는 목적지 특성입니다. XMSC_WMQ_RECEIVE_CONVERSION을 WMQ_RECEIVE_CONVERSION_QMGR로 설정하지 않은 경우, 값이 무시됩니다.
1988 페이지의 『 <u>XMSC_WMQ_RECEIVE_CONVERSION</u> 』	큐 관리자가 데이터 변환을 수행할지 판별하는 목적지 특성입니다.
<u>XMSC_WMQ_TARGET_CLIENT</u>	목적지에 송신되는 메시지에 MQRFH2 헤더를 포함할지 여부를 지정합니다.
<u>XMSC_WMQ_TEMP_TOPIC_PREFIX</u>	임시 주제를 만들 때, XMS " 형식의 토픽 문자열을 생성합니다. TEMP/TEMPTOPICPREFIX/unique_id " 또는 이 속성에 기본값이 포함된 경우 이 문자열은 "TEMP/unique_id"가 생성됩니다. 비어 있지 않은 값을 지정하면 이 연결 아래에 작성된 임시 토픽에 대한 구독자의 관리 큐를 작성하기 위한 특정 모델 큐를 정의할 수 있습니다.

표 875. Destination 특성 (계속)	
특성 이름	설명
<u>XMSC_WPM_BUS_NAME</u>	연결 팩토리의 경우, 애플리케이션이 연결하는 서비스 통합 버스의 이름이거나, 또는 목적지의 경우에는 목적지가 존재하는 서비스 통합 버스의 이름입니다.
<u>XMSC_WPM_TOPIC_SPACE</u>	토픽을 포함하는 토픽 영역의 이름입니다.

## InitialContext 특성

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

표 876. InitialContext 특성	
특성 이름	설명
<u>XMSC_IC_PROVIDER_URL</u>	이를 사용하여 JNDI 네이밍 디렉토리를 찾으므로 COS 네이밍 서비스가 웹 서비스와 동일한 서버에 있을 필요가 없습니다.
<u>XMSC_IC_SECURITY_AUTHENTICATION</u>	를 기반으로Java 컨텍스트 인터페이스 SECURITY_AUTHENTICATION. 이 특성은 COS 네이밍 컨텍스트에만 적용할 수 있습니다.
<u>XMSC_IC_SECURITY_CREDENTIALS</u>	를 기반으로Java 컨텍스트 인터페이스 SECURITY_CREDENTIALS. 이 특성은 COS 네이밍 컨텍스트에만 적용할 수 있습니다.
<u>XMSC_IC_SECURITY_PRINCIPAL</u>	를 기반으로Java 컨텍스트 인터페이스 SECURITY_PRINCIPAL. 이 특성은 COS 네이밍 컨텍스트에만 적용할 수 있습니다.
<u>XMSC_IC_SECURITY_PROTOCOL</u>	를 기반으로Java 컨텍스트 인터페이스 SECURITY_PROTOCOL 이 특성은 COS 네이밍 컨텍스트에만 적용할 수 있습니다.
<u>XMSC_IC_URL</u>	LDAP 및 FileSystem 컨텍스트의 경우 관리 오브젝트가 들어 있는 저장소의 주소입니다. COS 네이밍 컨텍스트의 경우 디렉토리에서 오브젝트를 찾는 웹 서비스의 주소입니다.

## Message 특성

Message 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

표 877. Message 특성	
특성 이름	설명
<u>JMS_IBM_CHARACTER_SET</u>	메시지 본문의 문자 데이터 문자열이 있는 코드화된 문자 세트 또는 코드 페이지의 ID(CCSID)입니다.XMS 클라이언트는 메시지를 의도한 대상으로 전달합니다. XMS에서 이 특성은 숫자 값을 보유하며 CCSID에 매핑됩니다. 단, 이 특성은 JMS 특성을 기반으로 하므로 문자열 유형 값을 가지며 이 숫자 CCSID를 나타내는 Java 문자 세트에 매핑됩니다.
<u>JMS_IBM_ENCODING</u>	메시지 본문의 숫자 데이터가 표시되는 방식XMS 클라이언트는 메시지를 의도한 대상으로 전달합니다.
<u>JMS_IBM_EXCEPTIONMESSAGE</u>	메시지가 예외 목적지로 송신된 이유를 설명하는 텍스트입니다. 이 특성은 읽기 전용입니다.
<u>JMS_IBM_EXCEPTIONPROBLEMDESTINATION</u>	메시지가 예외 목적지에 송신되기 전에 있었던 목적지 이름입니다.

표 877. Message 특성 (계속)	
특성 이름	설명
<u>JMS_IBM_EXCEPTIONREASON</u>	메시지가 예외 목적지에 송신된 이유를 나타내는 이유 코드입니다.
<u>JMS_IBM_EXCEPTIONTIMESTAMP</u>	메시지가 예외 목적지로 전송된 시간입니다.
<u>JMS_IBM_FEEDBACK</u>	보고서 메시지의 속성을 나타내는 코드입니다.
<u>JMS_IBM_FORMAT</u>	메시지에 있는 애플리케이션 데이터의 네이처입니다.
<u>JMS_IBM_LAST_MSG_IN_GROUP</u>	메시지가 메시지 그룹에서 마지막 메시지인지 나타냅니다.
<u>JMS_IBM_MSGTYPE</u>	메시지의 유형입니다.
<u>JMS_IBM_PUTAPPLTYPE</u>	메시지를 송신한 애플리케이션의 유형입니다.
<u>JMS_IBM_PUTDATE</u>	메시지가 송신된 날짜입니다.
<u>JMS_IBM_PUTTIME</u>	메시지가 송신된 시간입니다.
<u>JMS_IBM_REPORT_COA</u>	보고서 메시지에 포함해야 하는 원래 메시지의 애플리케이션 데이터 양을 지정하는 '도착 확인' 보고서 메시지를 요청합니다.
<u>JMS_IBM_REPORT_COD</u>	보고서 메시지에 포함해야 하는 원래 메시지의 애플리케이션 데이터 양을 지정하는 '전달 확인' 보고서 메시지를 요청합니다.
<u>JMS_IBM_REPORT_DISCARD_MSG</u>	메시지를 의도한 목적지에 전달할 수 없을 경우 메시지를 버리는 요청입니다.
<u>JMS_IBM_REPORT_EXCEPTION</u>	보고서 메시지에 포함해야 하는 원래 메시지의 애플리케이션 데이터 양을 지정하는 예외 보고서 메시지를 요청합니다.
<u>JMS_IBM_REPORT_EXPIRATION</u>	보고서 메시지에 포함해야 하는 원래 메시지의 애플리케이션 데이터 양을 지정하는 만기 보고서 메시지를 요청합니다.
<u>JMS_IBM_REPORT_NAN</u>	부정적인 조치 보고서 메시지를 요청합니다.
<u>JMS_IBM_REPORT_PAN</u>	긍정적인 조치 알림 보고서 메시지를 요청합니다.
<u>JMS_IBM_REPORT_PASS_CORREL_ID</u>	보고서나 응답 메시지의 상관 ID가 원래 메시지의 상관 ID와 동일하도록 요청합니다.
<u>JMS_IBM_REPORT_PASS_MSG_ID</u>	보고서나 응답 메시지의 메시지 ID가 원래 메시지의 메시지 ID와 동일하도록 요청합니다.
<u>JMS_IBM_RETAIN</u>	이 특성을 설정하면 메시지를 보유한 발행물로 처리하도록 큐 관리자에 표시합니다.
<u>JMS_IBM_SYSTEM_MESSAGEID</u>	서비스 통합 버스 내에서 메시지를 고유하는 식별하는 ID입니다. 이 특성은 읽기 전용입니다.
<u>JMSX_APPID</u>	메시지를 송신한 애플리케이션의 이름입니다.
<u>JMSX_DELIVERY_COUNT</u>	메시지 전달 시도 횟수입니다.
<u>JMSX_GROUPID</u>	메시지가 속한 메시지 그룹의 ID입니다.
<u>JMSX_GROUPSEQ</u>	메시지 그룹에 있는 메시지의 순서 번호입니다.
<u>JMSX_USERID</u>	메시지를 송신한 애플리케이션과 연관된 사용자 ID입니다.

### JMS\_IBM\_MQMD\* 특성

IBM Message Service Client for .NET을 사용하면 클라이언트 애플리케이션이 API를 사용하여 MQMD 필드를 읽고 쓸 수 있습니다. 또한 MQ 메시지 데이터에 액세스할 수도 있습니다. 기본적으로 MQMD에 대한 액세스는 사

용 불가능한 상태이므로 Destination 특성 XMSC\_WMQ\_MQMD\_WRITE\_ENABLED 및 XMSC\_WMQ\_MQMD\_READ\_ENABLED를 사용하여 애플리케이션이 명시적으로 사용하도록 설정해야 합니다. 이 두 특성은 서로 독립적입니다.

StrucId 및 Version을 제외한 모든 MQMD 필드는 추가 Message 오브젝트 특성으로 표시되며 JMS\_IBM\_MQMD가 접두부로 사용됩니다.

JMS\_IBM\_MQMD\* 특성은 이전 표에 설명된 JMS\_IBM\*과 같은 다른 특성보다 더 우선합니다.

## 메시지 송신

StrucId 및 Version을 제외한 모든 MQMD 필드가 표시됩니다. 이러한 특성은 MQMD 필드만 참조합니다. 여기서 특성은 MQMD 및 MQRFH2 헤더 모두에서 발생하며, MQRFH2의 버전은 설정되거나 추출되지 않습니다.

JMS\_IBM\_MQMD\_BackoutCount를 제외한 이러한 특성을 설정할 수 있습니다.

JMS\_IBM\_MQMD\_BackoutCount에 대해 설정된 값은 무시됩니다.

특성이 최대 길이를 보유하며 너무 긴 값을 제공하는 경우에는 값이 잘립니다.

특정 특성의 경우, 목적지 오브젝트에서 XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT 특성을 설정해야 할 수도 있습니다. 이 특성이 적용되려면 애플리케이션이 적합한 컨텍스트 권한으로 실행 중이어야 합니다.

XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT를 적합한 값으로 설정하지 않으면 특성 값이 무시됩니다.

XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT를 적합한 값으로 설정하지만 큐 관리자에 대해 충분한 컨텍스트 권한이 없는 경우에는 예외가 발생합니다. XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT의 특정 값을 요구하는 특성은 다음과 같습니다.

다음 특성은 XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT가 XMSC\_WMQ\_MDCTX\_SET\_IDENTITY\_CONTEXT 또는 XMSC\_WMQ\_MDCTX\_SET\_ALL\_CONTEXT로 설정되도록 요구합니다.

- JMS\_IBM\_MQMD\_UserIdentifier
- JMS\_IBM\_MQMD\_AccountingToken
- JMS\_IBM\_MQMD\_ApplIdentityData

다음 특성은 XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT가 XMSC\_WMQ\_MDCTX\_SET\_ALL\_CONTEXT로 설정되도록 요구합니다.

- JMS\_IBM\_MQMD\_PutApplType
- JMS\_IBM\_MQMD\_PutApplName
- JMS\_IBM\_MQMD\_PutDate
- JMS\_IBM\_MQMD\_PutTime
- JMS\_IBM\_MQMD\_ApplOriginData

## 메시지 수신

생성 애플리케이션이 설정한 실제 특성과는 무관하게 XMSC\_WMQ\_MQMD\_READ\_ENABLED 특성이 true로 설정된 경우 이 모든 특성은 수신된 메시지에서 사용 가능합니다. JMS 스펙에 따라 우선 모든 특성이 선택 취소되지 않는 한 애플리케이션은 수신된 메시지의 특성을 수정할 수 없습니다. 수신된 메시지는 특성의 수정 없이 전달될 수 없습니다.

**참고:** XMSC\_WMQ\_MQMD\_READ\_ENABLED 특성이 true로 설정된 목적지에서 애플리케이션이 메시지를 수신하고 XMSC\_WMQ\_MQMD\_WRITE\_ENABLED가 true로 설정된 목적지에 이를 전달하는 경우, 이 결과에 따라 수신된 메시지의 모든 MQMD 필드 값은 전달된 메시지에 복사됩니다. 특성 표

표 878. MQMD 필드를 표시하는 Message 오브젝트		
특성	설명	유형
JMS_IBM_MQMD_REPORT	보고 메시지의 옵션	System.Int32
JMS_IBM_MQMD_MSGTYPE	메시지 유형	System.Int32
JMS_IBM_MQMD_EXPIRY	메시지 수명	System.Int32

표 878. MQMD 필드를 표시하는 Message 오브젝트 (계속)		
특성	설명	유형
JMS_IBM_MQMD_FEEDBACK	피드백 또는 이유 코드	System.Int32
JMS_IBM_MQMD_ENCODING	메시지 데이터의 숫자 인코딩	System.Int32
JMS_IBM_MQMD_CODEDCHARSETID	메시지 데이터의 문자 세트 ID	System.Int32
JMS_IBM_MQMD_FORMAT	메시지 데이터의 형식 이름	System.String
JMS_IBM_MQMD_PRIORITY	메시지 우선순위	System.Int32
참고: 0-9 범위에 없는 JMS_IBM_MQMD_PRIORITY로 값이 지정되는 경우, 이 값은 JMS 스펙을 위반합니다.		
JMS_IBM_MQMD_PERSISTENCE	메시지 지속성	System.Int32
JMS_IBM_MQMD_MSGID	메시지 ID	바이트 배열
참고: JMS 스펙은 메시지 ID가 JMS 제공자에 의해 설정되어야 하며 고유하거나 널이어야 함을 지정합니다. 값을 JMS_IBM_MQMD_MSGID로 지정하는 경우, 이 값은 JMSMessageID에 복사됩니다. 따라서 이는 JMS 제공자에 의해 설정되지 않으며 고유하지 않을 수 있습니다. 이 값은 JMS 스펙을 위반합니다.		참고: 메시지에서 바이트 배열 특성의 사용은 JMS 스펙을 위반합니다.
JMS_IBM_MQMD_CORRELID	상관 ID	바이트 배열
참고: 'ID:' 문자열로 시작되는 JMS_IBM_MQMD_CORRELID로 값을 지정하는 경우, 이 값은 JMS 스펙을 위반합니다.		참고: 메시지에서 바이트 배열 특성의 사용은 JMS 스펙을 위반합니다.
JMS_IBM_MQMD_BACKOUTCOUNT	백아웃 카운터	System.Int32
JMS_IBM_MQMD_REPLYTOQ	응답 큐의 이름	System.String
JMS_IBM_MQMD_REPLYTOQMGR	응답 큐 관리자의 이름	System.String
JMS_IBM_MQMD_USERIDENTIFIER	사용자 ID	System.String
JMS_IBM_MQMD_ACCOUNTINGTOKEN	계정 토큰	바이트 배열
		참고: 메시지에서 바이트 배열 특성의 사용은 JMS 스펙을 위반합니다.
JMS_IBM_MQMD_APPLIDENTITYDATA	ID 관련 애플리케이션 데이터	System.String
JMS_IBM_MQMD_PUTAPPLTYPE	메시지를 넣는 애플리케이션의 유형	System.Int32
JMS_IBM_MQMD_PUTAPPLNAME	메시지를 넣는 애플리케이션 이름	System.String
JMS_IBM_MQMD_PUTDATE	메시지를 넣은 날짜	System.String
JMS_IBM_MQMD_PUTTIME	메시지를 넣은 시간	System.String
JMS_IBM_MQMD_APPLORIGINDATA	원본 관련 애플리케이션 데이터	System.String



표 878. MQMD 필드를 표시하는 Message 오브젝트 (계속)		
특성	설명	유형
JMS_IBM_MQMD_GROUPID	그룹 ID	바이트 배열 <b>참고:</b> 메시지에서 바이트 배열 특성의 사용은 JMS 스펙을 위반합니다.
JMS_IBM_MQMD_MSGSEQNUMBER	그룹 내 로컬 메시지의 순서 번호	System.Int32
JMS_IBM_MQMD_OFFSET	논리 메시지의 시작에서 실제 메시지의 데이터의 오프셋	System.Int32
JMS_IBM_MQMD_MSGFLAGS	메시지 플래그	System.Int32
JMS_IBM_MQMD_ORIGINALLENGTH	원래 메시지의 길이	System.Int32

추가 세부사항은 [MQMD](#)를 참조하십시오.

## 예

이 예제의 결과로 MQMD.UserIdentifer가 "JoeBloggs"로 설정된 큐 또는 토픽에 메시지를 넣습니다.

```
// Create a ConnectionFactory, connection, session, producer, message
// ...

// Create a destination
// ...

// Enable MQMD write
dest.setBooleanProperty(XMSC_WMQ_MQMD_WRITE_ENABLED,
    XMSC_WMQ_MQMD_WRITE_ENABLED_YES);

// Optionally, set a message context if applicable for this MD field
dest.setIntProperty(XMSC_WMQ_MQMD_MESSAGE_CONTEXT,
    XMSC_WMQ_MDCTX_SET_IDENTITY_CONTEXT);

// On the message, set property to provide custom UserId
msg.setStringProperty(JMS_IBM_MQMD_USERIDENTIFIER, "JoeBloggs");

// Send the message
// ...
```

JMS\_IBM\_MQMD\_USERIDENTIFIER를 설정하기 전에 XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT를 설정해야 합니다. XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT 사용에 대한 자세한 정보는 Message 오브젝트 특성을 참조하십시오.

이와 유사하게, 메시지를 수신하기 전에 XMSC\_WMQ\_MQMD\_READ\_ENABLED를 true로 설정한 후에 메시지의 Get 메소드(예: getStringProperty)를 사용하여 MQMD 필드의 콘텐츠를 추출할 수 있습니다. 수신된 특성은 읽기 전용입니다.

이 예에서는 value 필드가 큐나 토픽에서 가져온 메시지의 MQMD.ApplIdentityData 필드의 값을 보유하게 됩니다.

```
// Create a ConnectionFactory, connection, session, consumer
// ...

// Create a destination
// ...

// Enable MQMD read
dest.setBooleanProperty(XMSC_WMQ_MQMD_READ_ENABLED, XMSC_WMQ_MQMD_READ_ENABLED_YES);

// Receive a message
// ...
```



```
// Get required MQMD field value using a property
System.String value = rcvMsg.getStringProperty(JMS_IBM_MQMD_APPLIDENTITYDATA);
```

## MessageConsumer 특성

MessageConsumer 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

표 879. MessageConsumer 특성	
특성 이름	설명
XMSC_IS_SUBSCRIPTION_MULTICAST	다음을 사용하여 메시지가 메시지 소비자에게 전달되는지 여부를 나타냅니다. WebSphere MQ Multicast Transport . 이 특성은 읽기 전용입니다.
XMSC_IS_SUBSCRIPTION_RELIABLE_MULTICAST	다음을 사용하여 메시지가 메시지 소비자에게 전달되는지 여부를 나타냅니다. WebSphere MQ Multicast Transport 믿을 수 있는 서비스 품질로 이 특성은 읽기 전용입니다.

자세한 정보는 [IMessageConsumer](#)의 .NET 특성을 참조하십시오.

## MessageProducer 특성

MessageProducer 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

자세한 내용은 [IMessageProducer](#)의 .NET 특성을 참조하십시오.

## Session 특성

Session 오브젝트의 특성에 대한 개요이며 상세한 참조 정보에 대한 링크가 있습니다.

자세한 내용은 [ISession](#)의 .NET 특성을 참조하십시오.

## 특성 정의

이 절에서는 각 오브젝트 특성의 정의를 제공합니다.

각 특성 정의에는 다음 정보가 포함되어 있습니다.

- 특성의 데이터 유형
- 특성이 있는 오브젝트의 유형
- Destination의 특성의 경우 URI(Uniform Resource Identifier)에서 사용할 수 있는 이름
- 특성의 자세한 설명
- 특성의 올바른 값
- 특성의 기본값

이름이 다음 접두부 중 하나로 시작되는 특성은 지정된 연결 유형에 대해서만 관련됩니다.

### XMSC\_RTT

실시간 브로커 연결에 대해서만 특성이 관련됩니다. 특성 이름은 헤더 파일 `xmsc_rtt.h`에서 이름 지정된 상수로 정의됩니다.

### XMSC\_WMQ

애플리케이션이 IBM MQ 큐 관리자에 연결되는 경우에만 특성이 관련됩니다. 특성 이름은 헤더 파일 `xmsc_wmq.h`에서 이름 지정된 상수로 정의됩니다.

### XMSC\_WPM

애플리케이션이 WebSphere 서비스 통합 버스에 연결되는 경우에만 특성이 관련됩니다. 특성 이름은 헤더 파일 `xmsc_wpm.h`에서 이름 지정된 상수로 정의됩니다.

정의에서 별도의 언급이 없는 한 나머지 특성은 모든 유형의 연결과 관련됩니다. 특성 이름은 헤더 파일 `xmsc.h`에서 이름 지정된 상수로 정의됩니다. 접두부 `MSX`로 시작하는 이름의 특성은 메시지의 JMS 정의 특성이며, 접두

부 JMS\_IBM으로 시작하는 이름의 특성은 메시지의 IBM 정의 특성입니다. 메시지의 특성에 대한 자세한 정보는 [XMS 메시지의 특성을 참조하십시오](#).

정의에서 달리 언급되지 않는 한, 각 특성은 포인트-투-포인트 및 발행/구독 도메인 둘 다와 관련됩니다.

특성이 읽기 전용으로 지정되지 않은 한 애플리케이션이 특성의 값을 가져와서 설정할 수 있습니다.

## JMS\_IBM\_CHARACTER\_SET

데이터 유형:

System.Int32

특성:

메시지

메시지 본문의 문자 데이터 문자열이 있는 코드화된 문자 세트 또는 코드 페이지의 ID(CCSID)입니다. XMS 클라이언트는 메시지를 의도한 대상으로 전달합니다. XMS에서 이 특성은 숫자 값을 보유하며 CCSID에 매핑됩니다. 단, 이 특성은 JMS 특성을 기반으로 하므로 문자열 유형 값을 가지며 이 숫자 CCSID를 나타내는 Java 문자 세트에 매핑됩니다. 이 특성은 [XMSC WMQ CCSID](#) 특성에서 목적지에 대해 지정되는 CCSID를 대체합니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되는 경우에는 관련되지 않습니다.

## JMS\_IBM\_ENCODING

데이터 유형:

System.Int32


특성:

메시지

메시지 본문의 숫자 데이터가 표시되는 방식 XMS 클라이언트는 메시지를 의도한 대상으로 전달합니다. 이 특성은 [XMSC WMQ\\_ENCODING](#) 특성에 의해 목적지에 대해 지정된 인코딩을 대체합니다. 이 특성은 2진 정수, 팩형 10진수 정수 및 부동 소수점 숫자의 표시를 지정합니다.

특성의 올바른 값은 메시지 디스크립터의 **Encoding** 필드에 지정할 수 있는 값과 동일합니다.

애플리케이션은 다음과 같은 이름 지정된 상수를 사용하여 특성을 설정할 수도 있습니다.

이름 지정된 상수	의미
MQENC_INTEGER_NORMAL	표준 정수 인코딩
MQENC_INTEGER_REVERSED	역방향 정수 인코딩
MQENC_DECIMAL_NORMAL	표준 팩형 10진수 인코딩
MQENC_DECIMAL_REVERSED	역방향 팩형 10진수 인코딩
MQENC_FLOAT_IEEE_NORMAL	표준 IEEE 부동 소수점 인코딩
MQENC_FLOAT_IEEE_REVERSED	역방향 IEEE 부동 소수점 인코딩
 MQENC_FLOAT_S390	z/OS 아키텍처 부동 소수점 인코딩
MQENC_NATIVE	기본 시스템 인코딩

특성 값을 구성하기 위해 애플리케이션은 다음과 같이 세 개의 상수를 추가할 수 있습니다.

- 이름이 MQENC\_INTEGER로 시작하는 상수. 2진 정수의 표시를 지정합니다.
- 이름이 MQENC\_DECIMAL로 시작하는 상수. 팩형 10진 정수의 표시를 지정합니다.
- 이름이 MQENC\_FLOAT로 시작하는 상수. 부동 소수점 수를 지정합니다.

또는 애플리케이션이 특성을 MQENC\_NATIVE로 설정할 수 있습니다. 이 값은 환경에 따라 다릅니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되는 경우에는 관련되지 않습니다.

## **JMS\_IBM\_EXCEPTIONMESSAGE**

데이터 유형:

문자열

특성:

메시지

메시지가 예외 목적지로 송신된 이유를 설명하는 텍스트입니다. 이 특성은 읽기 전용입니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되고 예외 목적지에서 메시지를 받는 경우에만 관련됩니다.

## **JMS\_IBM\_EXCEPTIONPROBLEMDESTINATION**

데이터 유형:

문자열

특성:

메시지

메시지가 예외 목적지에 송신되기 전에 있었던 목적지 이름입니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되고 예외 목적지에서 메시지를 받는 경우에만 관련됩니다.

## **JMS\_IBM\_EXCEPTIONREASON**

데이터 유형:

System.Int32

특성:

메시지

메시지가 예외 목적지에 송신된 이유를 나타내는 이유 코드입니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되고 예외 목적지에서 메시지를 받는 경우에만 관련됩니다.

## **JMS\_IBM\_EXCEPTIONTIMESTAMP**

데이터 유형:

System.Int64

특성:

메시지

메시지가 예외 목적지로 전송된 시간입니다.

이 시간은 1970년 1월 1일 00:00:00 GMT 이후부터 밀리초 단위로 표시됩니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되고 예외 목적지에서 메시지를 받는 경우에만 관련됩니다.

## **JMS\_IBM\_FEEDBACK**

데이터 유형:

System.Int32

특성:

메시지

보고서 메시지의 속성을 나타내는 코드입니다.

특성의 유효값은 메시지 디스크립터의 **Feedback** 필드에서 지정할 수 있는 피드백 코드와 이유 코드입니다.

기본적으로 특성이 설정되어 있지 않습니다.

## **JMS\_IBM\_FORMAT**

데이터 유형:

문자열

**특성:**

메시지

메시지에 있는 애플리케이션 데이터의 네이처입니다.

특성의 올바른 값은 메시지 디스크립터의 **Format** 필드에 지정할 수 있는 값과 동일합니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되는 경우에는 관련되지 않습니다.

**JMS\_IBM\_LAST\_MSG\_IN\_GROUP****데이터 유형:**

System.Boolean

**특성:**

메시지

메시지가 메시지 그룹에서 마지막 메시지인지 나타냅니다.

메시지가 메시지 그룹의 마지막 메시지인 경우 이 특성을 true로 설정하십시오. 그렇지 않으면 이 특성을 false로 설정하거나 이 특성을 설정하지 마십시오. 기본적으로 특성이 설정되어 있지 않습니다.

true 값은 메시지 디스크립터의 **MsgFlags** 필드에 지정할 수 있는 상태 플래그 MQMF\_LAST\_MSG\_IN\_GROUP에 해당합니다.

이 특성은 발행/구독 도메인에서는 무시되며, 애플리케이션이 서비스 통합 버스에 연결되는 경우에는 관련되지 않습니다.

**JMS\_IBM\_MSGTYPE****데이터 유형:**

System.Int32

**특성:**

메시지

메시지의 유형입니다.

특성의 올바른 값은 다음과 같습니다.

올바른 값	의미
MQMT_DATAGRAM	응답을 요청하지 않는 메시지입니다.
MQMT_REQUEST	응답이 필요한 메시지입니다.
MQMT_REPLY	응답 메시지입니다.
MQMT_REPORT	보고 메시지입니다.

이러한 값은 메시지 디스크립터의 **MsgType** 필드에 지정할 수 있는 메시지 유형에 해당합니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되는 경우에는 관련되지 않습니다.

**JMS\_IBM\_PUTAPPLTYPE****데이터 유형:**

System.Int32

**특성:**

메시지

메시지를 송신한 애플리케이션의 유형입니다.

특성의 유효값은 메시지 디스크립터의 **PutApp1Type** 필드에서 지정할 수 있는 애플리케이션 유형입니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되는 경우에는 관련되지 않습니다.

### **JMS\_IBM\_PUTDATE**

데이터 유형:

문자열

특성:

메시지

메시지가 송신된 날짜입니다.

특성의 올바른 값은 메시지 디스크립터의 **PutDate** 필드에 지정할 수 있는 값과 동일합니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되는 경우에는 관련되지 않습니다.

### **JMS\_IBM\_PUTTIME**

데이터 유형:

문자열

특성:

메시지

메시지가 송신된 시간입니다.

특성의 올바른 값은 메시지 디스크립터의 **PutTime** 필드에 지정할 수 있는 값과 동일합니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되는 경우에는 관련되지 않습니다.

### **JMS\_IBM\_REPORT\_COA**

데이터 유형:

System.Int32

특성:

메시지

보고서 메시지에 포함해야 하는 원래 메시지의 애플리케이션 데이터 양을 지정하는 '도착 확인' 보고서 메시지를 요청합니다.

특성의 올바른 값은 다음과 같습니다.

올바른 값	의미
MQRO_COA	원래 메시지의 애플리케이션 데이터가 보고 메시지에 포함되지 않은 '도착 시 확인' 보고 메시지를 요청합니다.
MQRO_COA_WITH_DATA	원래 메시지의 애플리케이션 데이터의 처음 100바이트가 보고 메시지에 포함된 '도착 시 확인' 보고 메시지를 요청합니다.
MQRO_COA_WITH_FULL_DATA	원래 메시지의 모든 애플리케이션 데이터가 보고 메시지에 포함된 '도착 시 확인' 보고 메시지를 요청합니다.

이러한 값은 메시지 디스크립터의 **Report** 필드에서 지정할 수 있는 보고서 옵션에 해당합니다. 이러한 옵션에 대한 자세한 정보는 [Report \(MQLONG\)](#)를 참조하십시오.

기본적으로 특성이 설정되어 있지 않습니다.

### **JMS\_IBM\_REPORT\_COD**

데이터 유형:

System.Int32

**특성:**

메시지

보고서 메시지에 포함해야 하는 원래 메시지의 애플리케이션 데이터 양을 지정하는 '전달 확인' 보고서 메시지를 요청합니다.

특성의 올바른 값은 다음과 같습니다.

**올바른 값**

MQRO\_COD

MQRO\_COD\_WITH\_DATA

MQRO\_COD\_WITH\_FULL\_DATA

**의미**

원래 메시지의 애플리케이션 데이터가 보고 메시지에 포함되지 않은 '전달 시 확인' 보고 메시지를 요청합니다.

원래 메시지의 애플리케이션 데이터의 처음 100바이트가 보고 메시지에 포함된 '전달 시 확인' 보고 메시지를 요청합니다.

원래 메시지의 모든 애플리케이션 데이터가 보고 메시지에 포함된 '전달 시 확인' 보고 메시지를 요청합니다.

이 값은 메시지 디스크립터의 **Report** 필드에 지정할 수 있는 보고서 옵션에 해당합니다.

기본적으로 특성이 설정되어 있지 않습니다.

**JMS\_IBM\_REPORT\_DISCARD\_MSG****데이터 유형:**

System.Int32

**특성:**

메시지

메시지를 의도한 목적지에 전달할 수 없을 경우 메시지를 버리는 요청입니다.

메시지를 원하는 목적지에 전달할 수 없는 경우 메시지를 버리도록 요청하려면 특성을 MQRO\_DISCARD\_MSG로 설정하십시오. 메시지를 데드-레터 큐에 넣거나 예외 목적지로 전송해야 하는 경우 이 특성을 설정하지 마십시오. 기본적으로 특성이 설정되어 있지 않습니다.

MQRO\_DISCARD\_MSG값은 메시지 디스크립터의 **Report** 필드에 지정할 수 있는 보고서 옵션에 해당합니다.

**JMS\_IBM\_REPORT\_EXCEPTION****데이터 유형:**

System.Int32

**특성:**

메시지

보고서 메시지에 포함해야 하는 원래 메시지의 애플리케이션 데이터 양을 지정하는 예외 보고서 메시지를 요청합니다.

특성의 올바른 값은 다음과 같습니다.

**올바른 값**

MQRO\_EXCEPTION

MQRO\_EXCEPTION\_WITH\_DATA

MQRO\_EXCEPTION\_WITH\_FULL\_DATA

**의미**

원래 메시지의 애플리케이션 데이터가 보고 메시지에 포함되지 않은 예외 보고 메시지를 요청합니다.

원래 메시지의 애플리케이션 데이터의 처음 100바이트가 보고 메시지에 포함된 예외 보고 메시지를 요청합니다.

원래 메시지의 모든 애플리케이션 데이터가 보고 메시지에 포함된 예외 보고 메시지를 요청합니다.

이 값은 메시지 디스크립터의 **Report** 필드에 지정할 수 있는 보고서 옵션에 해당합니다.

기본적으로 특성이 설정되어 있지 않습니다.

## **JMS\_IBM\_REPORT\_EXPIRATION**

데이터 유형:

System.Int32

특성:

메시지

보고서 메시지에 포함해야 하는 원래 메시지의 애플리케이션 데이터 양을 지정하는 만기 보고서 메시지를 요청합니다.

특성의 올바른 값은 다음과 같습니다.

**올바른 값**

MQRO\_EXPIRATION

MQRO\_EXPIRATION\_WITH\_DATA

MQRO\_EXPIRATION\_WITH\_FULL\_DATA

**의미**

원래 메시지의 애플리케이션 데이터가 보고 메시지에 포함되지 않은 만기 보고 메시지를 요청합니다.

원래 메시지의 애플리케이션 데이터의 처음 100바이트가 보고 메시지에 포함된 만기 보고 메시지를 요청합니다.

원래 메시지의 모든 애플리케이션 데이터가 보고 메시지에 포함된 만기 보고 메시지를 요청합니다.

이 값은 메시지 디스크립터의 **Report** 필드에 지정할 수 있는 보고서 옵션에 해당합니다.

기본적으로 특성이 설정되어 있지 않습니다.

## **JMS\_IBM\_REPORT\_NAN**

데이터 유형:

System.Int32

특성:

메시지

부정적인 조치 보고서 메시지를 요청합니다.

부정적 조치 알림 보고 메시지를 요청하려면 이 특성을 MQRO\_NAN으로 설정하십시오. 부정적 조치 알림 보고 메시지가 필요하지 않으면 이 특성을 설정하지 마십시오. 기본적으로 특성이 설정되어 있지 않습니다.

MQRO\_NAN값은 메시지 디스크립터의 **Report** 필드에 지정할 수 있는 보고서 옵션에 해당합니다.

## **JMS\_IBM\_REPORT\_PAN**

데이터 유형:

System.Int32

특성:

메시지

긍정적인 조치 알림 보고서 메시지를 요청합니다.

긍정적 조치 알림 보고 메시지를 요청하려면 이 특성을 MQRO\_PAN으로 설정하십시오. 긍정적 조치 알림 보고 메시지가 필요하지 않으면 이 특성을 설정하지 마십시오. 기본적으로 특성이 설정되어 있지 않습니다.

MQRO\_PAN값은 메시지 디스크립터의 **Report** 필드에 지정할 수 있는 보고서 옵션에 해당합니다.

## **JMS\_IBM\_REPORT\_PASS\_CORREL\_ID**

데이터 유형:

System.Int32

특성:

메시지

보고서나 응답 메시지의 상관 ID가 원래 메시지의 상관 ID와 동일하도록 요청합니다.

특성의 올바른 값은 다음과 같습니다.

올바른 값	의미
MQRO_PASS_CORREL_ID	모든 보고서나 응답 메시지의 상관 ID가 원래 메시지의 상관 ID와 동일하도록 요청합니다.
MQRO_COPY_MSG_ID_TO_CORREL_ID	모든 보고서나 응답 메시지의 상관 ID가 원래 메시지의 메시지 ID와 동일하도록 요청합니다.

이러한 값은 메시지 디스크립터의 **Report** 필드에 지정할 수 있는 보고서 옵션에 해당합니다.

이 특성의 기본값은 MQRO\_COPY\_MSG\_ID\_TO\_CORREL\_ID입니다.

### **JMS\_IBM\_REPORT\_PASS\_MSG\_ID**

데이터 유형:  
System.Int32

특성:  
메시지

보고서나 응답 메시지의 메시지 ID가 원래 메시지의 메시지 ID와 동일하도록 요청합니다.

특성의 올바른 값은 다음과 같습니다.

올바른 값	의미
MQRO_PASS_MSG_ID	모든 보고서나 응답 메시지의 메시지 ID가 원래 메시지의 메시지 ID와 동일하도록 요청합니다.
MQRO_NEW_MSG_ID	각 보고서나 응답 메시지에 대해 새 메시지 ID가 생성되도록 요청합니다.

이러한 값은 메시지 디스크립터의 **Report** 필드에서 지정할 수 있는 보고서 옵션에 해당합니다.

이 특성의 기본값은 MQRO\_NEW\_MSG\_ID입니다.

### **JMS\_IBM\_RETAIN**

데이터 유형:  
System.Int32

특성:  
메시지

이 특성을 설정하면 메시지를 보유한 발행물로 처리하도록 큐 관리자에 표시합니다. 구독자가 토픽에서 메시지를 수신하면 이전 릴리스에서 수신된 메시지 외에 구독한 후 즉시 추가 메시지를 수신할 수 있습니다. 이러한 메시지는 구독하는 토픽에 대해 선택적으로 보유한 발행물입니다. 구독과 일치하는 각 토픽에 보유한 발행물이 있는 경우 이 발행물을 구독 중인 메시지 이용자에 전달할 수 있게 됩니다.

이 특성에 유효한 유일한 값은 RETAIN\_PUBLICATION입니다. 기본적으로 이 특성은 설정되어 있지 않습니다.

**참고:** 이 특성은 발행/구독 도메인에만 관련됩니다.

### **JMS\_IBM\_SYSTEM\_MESSAGEID**

데이터 유형:  
문자열

특성:  
메시지

서비스 통합 버스 내에서 메시지를 고유하는 식별하는 ID입니다. 이 특성은 읽기 전용입니다.

이 특성은 애플리케이션이 서비스 통합 버스에 연결되는 경우에만 관련됩니다.



## **JMSX\_APPID**

**데이터 유형:**

문자열

**특성:**

메시지

메시지를 송신한 애플리케이션의 이름입니다.

이 특성은 JMS 이름이 JMSXAppID인 JMS 정의 특성입니다. 특성에 대한 자세한 정보는 *Java Message Service Specification*, 버전 1.1을 참조하십시오.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 실시간 브로커 연결에는 유효하지 않습니다.

## **JMSX\_DELIVERY\_COUNT**

**데이터 유형:**

System.Int32

**특성:**

메시지

메시지 전달 시도 횟수입니다.

이 특성은 JMS 이름이 JMSXDeliveryCount인 JMS 정의 특성입니다. 특성에 대한 자세한 정보는 *Java Message Service* 스펙, 버전 1.1을 참조하십시오.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 실시간 브로커 연결에는 유효하지 않습니다.

## **JMSX\_GROUPID**

**데이터 유형:**

문자열

**특성:**

메시지

메시지가 속한 메시지 그룹의 ID입니다.

이 특성은 JMS 이름이 JMSXGroupID인 JMS 정의 특성입니다. 특성에 대한 자세한 정보는 *Java Message Service* 스펙, 버전 1.1을 참조하십시오.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 실시간 브로커 연결에는 유효하지 않습니다.

## **JMSX\_GROUPSEQ**

**데이터 유형:**

System.Int32

**특성:**

메시지

메시지 그룹에 있는 메시지의 순서 번호입니다.

이 특성은 JMS 이름이 JMSXGroupSeq인 JMS 정의 특성입니다. 특성에 대한 자세한 정보는 *Java Message Service* 스펙, 버전 1.1을 참조하십시오.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 실시간 브로커 연결에는 유효하지 않습니다.

## **JMSX\_USERID**

**데이터 유형:**

문자열

**특성:**

메시지

메시지를 송신한 애플리케이션과 연관된 사용자 ID입니다.

이 특성은 JMS 이름이 JMSXUserID인 JMS 정의 특성입니다. 특성에 대한 자세한 정보는 *Java Message Service 스펙, 버전 1.1*을 참조하십시오.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 실시간 브로커 연결에는 유효하지 않습니다.

## **XMSC\_ASYNC\_EXCEPTIONS**

**데이터 유형:**

System.Int32

**특성:**

ConnectionFactory

**적용 가능한 오브젝트:**

JMS 관리 도구 긴 이름: ASYNCEXCEPTION

JMS 관리 도구 짧은 이름: AEX

이 특성은 연결이 끊기거나 XMS API 호출에 비동기적으로 예외가 발생하는 경우에만 XMS에서 ExceptionListener에 알리는지 여부를 판별합니다. 이 특성은 등록된 ExceptionListener가 있고 이 ConnectionFactory에서 작성된 모든 연결에 적용됩니다.

이 특성의 올바른 값은 다음과 같습니다.

### **XMSC\_ASYNC\_EXCEPTIONS\_ALL**

동기 API 호출의 범위 밖에서 비동기적으로 발견된 예외와, 연결이 끊긴 모든 예외가 ExceptionListener로 전송됩니다.

### **XMSC\_ASYNC\_EXCEPTIONS\_CONNECTIONBROKEN**

끊긴 연결을 나타내는 예외만 ExceptionListener로 전송됩니다. 비동기 처리 중에 발생한 다른 예외는 ExceptionListener에 보고되지 않으므로 애플리케이션에 이 예외 정보가 제공되지 않습니다.

기본적으로 이 특성은 XMSC\_ASYNC\_EXCEPTIONS\_ALL로 설정됩니다.

## **XMSC\_CLIENT\_ID**

**데이터 유형:**

문자열

**특성:**

ConnectionFactory

**적용 가능한 오브젝트:**

JMS 관리 도구 긴 이름: CLIENTID

JMS 관리 도구 짧은 이름: CID

연결에 사용하는 클라이언트 ID입니다.

클라이언트 ID는 발행/구독 도메인에서 지속 가능한 구독을 지원하는 데만 사용되며 포인트-투-포인트 도메인에서는 무시됩니다. 클라이언트 ID 설정에 대한 자세한 정보는 [ConnectionFactoryies](#) 및 [Connection](#) 오브젝트를 참조하십시오.

이 특성은 실시간 브로커 연결에는 관련되지 않습니다.

## ***XMSC\_CONNECTION\_TYPE***

### **데이터 유형:**

System.Int32

### **특성:**

ConnectionFactory

애플리케이션이 연결되는 메시징 서버의 유형입니다.

특성의 올바른 값은 다음과 같습니다.

<b>올바른 값</b>	<b>의미</b>
XMSC_CT_RTT	브로커에 대한 실시간 연결입니다.
XMSC_CT_WMQ	IBM MQ 큐 관리자에 대한 연결입니다.
XMSC_CT_WPM	WebSphere Application Server service integration bus에 대한 연결입니다.

기본적으로 특성이 설정되어 있지 않습니다.

## ***XMSC\_DELIVERY\_MODE***

### **데이터 유형:**

System.Int32

### **특성:**

목적지

### **URI에서 사용되는 이름:**

persistence(IBM MQ 목적지의 경우)

deliveryMode(WebSphere 기본 메시징 제공자 목적지의 경우)

### **적용 가능한 오브젝트:**

JMS 관리 도구 긴 이름: PERSISTENCE

JMS 관리 도구 짧은 이름: PER

목적지에 전송되는 메시지의 전달 모드입니다.

특성의 올바른 값은 다음과 같습니다.

<b>올바른 값</b>	<b>의미</b>
XMSC_DELIVERY_NOT_PERSISTENT	목적지에 전송된 메시지가 비지속적입니다. 메시지 작성자의 기본 전달 모드 또는 전송 호출에서 지정된 모든 전달 모드가 무시됩니다. 목적지가 IBM MQ 큐인 경우 큐 속성인 <i>DefPersistence</i> 값도 무시됩니다.
XMSC_DELIVERY_PERSISTENT	목적지에 전송된 메시지가 지속적입니다. 메시지 작성자의 기본 전달 모드 또는 전송 호출에서 지정된 모든 전달 모드가 무시됩니다. 목적지가 IBM MQ 큐인 경우 큐 속성인 <i>DefPersistence</i> 값도 무시됩니다.
XMSC_DELIVERY_AS_APP	목적지에 전송된 메시지에 전송 호출에서 지정된 전달 모드가 적용됩니다. 전송 호출이 전달 모드를 지정하지 않으면 메시지 작성자의 기본 전달 모드가 대신 사용됩니다. 목적지가 IBM MQ 큐인 경우 큐 속성인 <i>DefPersistence</i> 값이 무시됩니다.

## 올바른 값

XMSC\_DELIVERY\_AS\_DEST

## 의미

목적지가 IBM MQ 큐인 경우 큐에 놓인 메시지에 큐 속성인 *DefPersistence* 값에서 지정된 전달 모드가 적용됩니다. 메시지 작성자의 기본 전달 모드 또는 전송 호출에서 지정된 모든 전달 모드가 무시됩니다.

목적지가 IBM MQ 큐가 아닌 경우에는 XMSC\_DELIVERY\_AS\_APP의 내용이 동일하게 적용됩니다.

기본값은 XMSC\_DELIVERY\_AS\_APP입니다.

## **XMSC\_IC\_PROVIDER\_URL**

데이터 유형:

문자열

특성:

InitialContext

이를 사용하여 JNDI 네이밍 디렉토리를 찾으므로 COS 네이밍 서비스가 웹 서비스와 동일한 서버에 있을 필요가 없습니다.

## **XMSC\_IC\_SECURITY\_AUTHENTICATION**

데이터 유형:

문자열

특성:

InitialContext

를 기반으로 Java 컨텍스트 인터페이스 SECURITY\_AUTHENTICATION. 이 특성은 COS 네이밍 컨텍스트에만 적용할 수 있습니다.

## **XMSC\_IC\_SECURITY\_CREDENTIALS**

데이터 유형:

문자열

특성:

InitialContext

를 기반으로 Java 컨텍스트 인터페이스 SECURITY\_CREDENTIALS. 이 특성은 COS 네이밍 컨텍스트에만 적용할 수 있습니다.

## **XMSC\_IC\_SECURITY\_PRINCIPAL**

데이터 유형:

문자열

특성:

InitialContext

를 기반으로 Java 컨텍스트 인터페이스 SECURITY\_PRINCIPAL. 이 특성은 COS 네이밍 컨텍스트에만 적용할 수 있습니다.

## **XMSC\_IC\_SECURITY\_PROTOCOL**

데이터 유형:

문자열

특성:

InitialContext

를 기반으로Java 컨텍스트 인터페이스 SECURITY\_PROTOCOL 이 특성은 COS 네이밍 컨텍스트에만 적용할 수 있습니다.

### ***XMSC\_IC\_URL***

데이터 유형:

문자열

특성:

InitialContext

LDAP 및 FileSystem 컨텍스트의 경우 관리 오브젝트가 포함된 저장소의 주소입니다.

LDAP 및 FileSystem 컨텍스트의 경우 관리 오브젝트가 들어 있는 저장소의 주소입니다.

### ***XMSC\_IS\_SUBSCRIPTION\_MULTICAST***

데이터 유형:

System.Boolean

특성:

MessageConsumer

다음을 사용하여 메시지가 메시지 소비자에게 전달되는지 여부를 나타냅니다.WebSphere MQ Multicast Transport . 이 특성은 읽기 전용입니다.

메시지가 WebSphere MQ Multicast Transport를 사용하여 메시지 이용자에게 전달되는 경우 이 특성의 값은 true입니다. 그렇지 않은 경우 값은 false입니다.

이 특성은 브로커에 대한 실시간 연결에만 관련됩니다.

### ***XMSC\_IS\_SUBSCRIPTION\_RELIABLE\_MULTICAST***

데이터 유형:

System.Boolean

특성:

MessageConsumer

다음을 사용하여 메시지가 메시지 소비자에게 전달되는지 여부를 나타냅니다.WebSphere MQ Multicast Transport 믿을 수 있는 서비스 품질로 이 특성은 읽기 전용입니다.

메시지가 신뢰할 수 있는 품질로 WebSphere MQ Multicast Transport를 사용하여 메시지 이용자에게 전달되는 경우 이 특성의 값은 true입니다. 그렇지 않은 경우 값은 false입니다.

이 특성은 브로커에 대한 실시간 연결에만 관련됩니다.

### ***XMSC\_JMS\_MAJOR\_VERSION***

데이터 유형:

System.Int32

특성:

ConnectionMetaData

주요 버전 번호JMS 사양XMS 기반으로합니다. 이 특성은 읽기 전용입니다.

### ***XMSC\_JMS\_MINOR\_VERSION***

데이터 유형:

System.Int32

특성:

ConnectionMetaData

부 버전 번호JMS 사양XMS 기반으로합니다. 이 특성은 읽기 전용입니다.

## ***XMSC\_JMS\_VERSION***

데이터 유형:

문자열

특성:

ConnectionMetaData

버전 식별자 JMS 사양 XMS 기반으로 합니다. 이 특성은 읽기 전용입니다.

## ***XMSC\_MAJOR\_VERSION***

데이터 유형:

System.Int32

특성:

ConnectionMetaData

버전 번호 XMS 고객. 이 특성은 읽기 전용입니다.

## ***XMSC\_MINOR\_VERSION***

데이터 유형:

System.Int32

특성:

ConnectionMetaData

릴리스 번호 XMS 고객. 이 특성은 읽기 전용입니다.

## ***XMSC\_PASSWORD***

데이터 유형:


바이트 배열


특성:

ConnectionFactory

애플리케이션이 메시징 서버에 연결할 때 해당 애플리케이션을 인증하는 데 사용할 수 있는 비밀번호입니다. 비밀번호는 [XMSC\\_USERID](#) 특성과 함께 사용됩니다.

기본적으로 특성이 설정되어 있지 않습니다.

 멀티플랫폼에서 IBM MQ 에 연결 중이고 연결 팩토리의 XMSC\_USERID 특성을 설정한 경우 로그인한 사용자의 **userid** 와 일치해야 합니다. 이러한 특성을 설정하지 않으면 큐 관리자는 기본적으로 로그인된 사용자의 **userid** 를 사용합니다. 개별 사용자의 추가 연결 레벨 인증이 필요한 경우 IBM MQ에서 구성되는 클라이언트 인증 엑시트를 작성할 수 있습니다. 클라이언트 인증 엑시트 작성에 대한 자세한 정보는 [클라이언트 애플리케이션에 대한 인증 계획을 참조하십시오](#).

 IBM MQ for z/OS에 연결할 때 사용자를 인증하려면 보안 엑시트를 사용해야 합니다.

## ***XMSC\_PRIORITY***

데이터 유형:

System.Int32

특성:

목적지

**URI**에서 사용되는 이름:

priority

목적지에 전송되는 메시지의 우선순위입니다.

특성의 올바른 값은 다음과 같습니다.

## 올바른 값

가장 낮은 우선순위는 0이며 가장 높은 우선순위는 9인 정수 범위

## 의미

목적지로 전송되는 메시지에 지정된 우선순위가 있습니다. 메시지 작성자의 기본 우선순위 또는 Send 호출에서 지정된 우선순위는 무시됩니다. 목적지가 IBM MQ 큐인 경우 큐 속성 **DefPriority**의 값도 무시됩니다.

### XMSC\_PRIORITY\_AS\_APP

목적지로 전송된 메시지에 Send 호출에서 지정된 우선순위가 있습니다. Send 호출이 우선순위를 지정하지 않으면 메시지 작성자의 기본 우선순위가 대신 사용됩니다. 목적지가 IBM MQ 큐인 경우 큐 속성 **DefPriority**의 값이 무시됩니다.

### XMSC\_PRIORITY\_AS\_DEST

목적지가 IBM MQ 큐인 경우 큐에 놓인 메시지에 큐 속성 **DefPriority**의 값에서 지정된 우선순위가 적용됩니다. 메시지 작성자의 기본 우선순위 또는 Send 호출에서 지정된 우선순위는 무시됩니다.

목적지가 IBM MQ 큐가 아닌 경우에는 XMSC\_PRIORITY\_AS\_APP의 내용이 동일하게 적용됩니다.

기본값은 XMSC\_PRIORITY\_AS\_APP입니다.

WebSphere MQ Real-Time Transport 및 WebSphere MQ Multicast Transport는 메시지의 우선순위에 따라 아무런 조치도 수행하지 않습니다.

## **XMSC\_PROVIDER\_NAME**

### 데이터 유형:

문자열

### 특성:

ConnectionMetaData

공급자는 XMS 고객. 이 특성은 읽기 전용입니다.

## **XMSC\_RTT\_BROKER\_PING\_INTERVAL**

### 데이터 유형:

System.Int32

### 특성:

ConnectionFactory

이 시간 간격(밀리초) 후 XMS .NET에서 활동을 감지하기 위해 실시간 메시징 서버에 대한 연결을 검사합니다. 활동이 감지되지 않으면 클라이언트가 ping을 시작합니다. ping에 대한 응답이 감지되지 않으면 연결이 닫힙니다.

이 특성의 기본값은 30000입니다.

## **XMSC\_RTT\_CONNECTION\_PROTOCOL**

### 데이터 유형:

System.Int32

### 특성:

ConnectionFactory

브로커에 대한 실시간 연결에 사용되는 통신 프로토콜입니다.

특성의 값은 XMSC\_RTT\_CP\_TCP여야 합니다. 이 값은 TCP/IP를 통해 브로커에 실시간으로 연결됨을 의미합니다. 기본값은 XMSC\_RTT\_CP\_TCP입니다.

## **XMSC\_RTT\_HOST\_NAME**

### 데이터 유형:

문자열

**특성:**

ConnectionFactory

브로커가 실행하는 시스템의 호스트 이름 또는 IP 주소입니다.

이 특성은 `XMSC_RTT_PORT` 특성과 함께 사용하여 브로커를 식별합니다.

기본적으로 특성이 설정되어 있지 않습니다.

**XMSC\_RTT\_LOCAL\_ADDRESS****데이터 유형:**

문자열

**특성:**

ConnectionFactory

브로커에 대한 실시간 연결에 사용되는 로컬 네트워크 인터페이스의 호스트 이름 또는 IP 주소입니다.

이 특성은 애플리케이션이 실행 중인 시스템에 두 개 이상의 네트워크 인터페이스가 있는 경우에만 유용하며 사용자가 실시간 연결에 사용되는 인터페이스를 지정할 수 있어야 합니다. 시스템에 네트워크 인터페이스가 하나 뿐이면 해당 인터페이스만 사용할 수 있습니다. 시스템에 네트워크 인터페이스가 둘 이상이며 특성이 설정되어 있지 않으면 인터페이스가 임의로 선택됩니다.

기본적으로 특성이 설정되어 있지 않습니다.

**XMSC\_RTT\_MULTICAST****데이터 유형:**

System.Int32

**특성:**

ConnectionFactory 및 Destination

**URI에서 사용되는 이름:**

multicast

연결 팩토리 또는 목적지의 멀티캐스트 설정입니다. 토픽인 대상에만 이 특성이 있을 수 있습니다.

애플리케이션은 이 특성을 사용하여 브로커에 대한 실시간 연결과 관련하여 멀티캐스트를 사용할 수 있도록 하며 멀티캐스트가 사용 가능한 경우 멀티캐스트를 사용하여 메시지를 브로커에서 메시지 이용자로 전달하는 방법을 정확하게 지정합니다. 이 특성은 메시지 작성자가 브로커에 메시지를 전송하는 방법에는 영향을 미치지 않습니다.

특성의 올바른 값은 다음과 같습니다.

**올바른 값**

XMSC\_RTT\_MULTICAST\_DISABLED

XMSC\_RTT\_MULTICAST\_ASCF

XMSC\_RTT\_MULTICAST\_ENABLED

**의미**

메시지가 WebSphere MQ Multicast Transport를 사용하여 메시지 이용자에 전달되지 않습니다. 이 값은 ConnectionFactory 오브젝트의 기본값입니다.

메시지가 메시지 이용자와 연관된 연결 팩토리의 멀티캐스트 설정에 따라 메시지 이용자로 전달됩니다. 연결 팩토리의 멀티캐스트 설정은 연결될 때 표시됩니다. 이 값은 Destination 오브젝트에만 유효하며 Destination 오브젝트의 기본값입니다.

브로커에서 토픽에 멀티캐스트가 구성된 경우, 메시지가 WebSphere MQ Multicast Transport를 사용하여 메시지 이용자로 전달됩니다. 토픽에 신뢰할 수 있는 멀티캐스트가 구성되면 신뢰할 수 있는 서비스 품질(QoS)이 사용됩니다.



## 올바른 값

XMSC\_RTT\_MULTICAST\_RELIABLE

XMSC\_RTT\_MULTICAST\_NOT\_RELIABLE

## 의미

브로커에서 토픽에 신뢰할 수 있는 멀티캐스트가 구성된 경우, 메시지가 신뢰할 수 있는 서비스 품질(QoS)의 WebSphere MQ Multicast Transport를 사용하여 메시지 이용자로 전달됩니다. 토픽에 신뢰할 수 있는 멀티캐스트가 구성되지 않으면 토픽에 대한 메시지 이용자를 작성할 수 없습니다.

브로커에서 토픽에 멀티캐스트가 구성된 경우, 메시지가 WebSphere MQ Multicast Transport를 사용하여 메시지 이용자로 전달됩니다. 토픽에 신뢰할 수 있는 멀티캐스트가 구성되더라도 신뢰할 수 있는 서비스 품질(QoS)이 사용되지 않습니다.

## **XMSC\_RTT\_PORT**

### 데이터 유형:

System.Int32

### 특성:

ConnectionFactory

브로커가 수신 요청을 청취하는 포트 번호입니다. 브로커에서 이 포트에서 청취할 Real-timeInput 또는 Real-timeOptimizedFlow 메시지 처리 노드를 구성해야 합니다.

이 특성은 XMSC\_RTT\_HOST\_NAME 특성과 함께 사용하여 브로커를 식별합니다.

이 특성의 기본값은 XMSC\_RTT\_DEFAULT\_PORT 또는 1506입니다.

## **XMSC\_TIME\_TO\_LIVE**

### 데이터 유형:

System.Int32

### 특성:

목적지

### URI에서 사용되는 이름:

expiry(IBM MQ 목적지의 경우)

timeToLive(WebSphere 기본 메시징 제공자 목적지의 경우)

목적지에 송신되는 메시지의 TTL(Time to Live).

특성의 올바른 값은 다음과 같습니다.

## 올바른 값

0

양수

XMSC\_TIME\_TO\_LIVE\_AS\_APP

## 의미

목적지에 전송되는 메시지가 만기되지 않습니다.

목적지로 전송되는 메시지에 밀리초 단위로 지정된 TTL(Time to Live)이 있습니다. 메시지 작성자의 기본 TTL 또는 Send 호출에서 지정된 모든 TTL이 무시됩니다.

목적지에 전송되는 메시지에 Send 호출에서 지정된 TTL이 있습니다. Send 호출이 TTL을 지정하지 않으면 메시지 작성자의 기본 TTL이 대신 사용됩니다.

기본값은 XMSC\_TIME\_TO\_LIVE\_AS\_APP입니다.

## **XMSC\_USERID**

### 데이터 유형:

문자열

### 특성:

ConnectionFactory

애플리케이션이 메시징 서버에 연결할 때 해당 애플리케이션을 인증하는 데 사용할 수 있는 사용자 ID입니다. 사용자 ID는 `XMSC_PASSWORD` 특성과 함께 사용됩니다.

기본적으로 특성이 설정되어 있지 않습니다.

**Multi** IBM MQ for Multiplatforms에 연결하는 중이고 연결 팩토리의 `XMSC_USERID` 특성을 설정한 경우 해당 특성이 로그인된 사용자의 **userid**와 일치해야 합니다. 이러한 특성을 설정하지 않으면 큐 관리자는 기본적으로 로그인된 사용자의 **userid**를 사용합니다. 개별 사용자의 추가 연결 레벨 인증이 필요한 경우 IBM MQ에서 구성되는 클라이언트 인증 엑시트를 작성할 수 있습니다. 클라이언트 인증 엑시트 작성에 대한 자세한 정보는 [클라이언트 애플리케이션에 대한 인증 계획을 참조하십시오](#).

**z/OS** IBM MQ for z/OS에 연결할 때 사용자를 인증하려면 보안 엑시트를 사용해야 합니다.

## **XMSC\_VERSION**

### 데이터 유형:

문자열

### 특성:

ConnectionMetaData

cli의 버전 식별자XMS 엔트. 이 속성은 읽기 전용입니다.

## **XMSC\_WMQ\_BALANCING\_APPLICATION\_TYPE**

### 데이터 유형:

System.Int32

### 특성:

ConnectionFactory

특성의 올바른 값은 다음과 같습니다.

#### 올바른 값

`XMSC_WMQ_BALANCING_APPLICATION_TYPE_SIMPLE`

`XMSC_WMQ_BALANCING_APPLICATION_TYPE_REQUEST_REPLY`

#### 의미

단순 밸런싱. 균일한 클러스터에서 애플리케이션 리밸런싱에 미치는 영향에 설명된 규칙 외에 특정 규칙이 적용되지 않습니다. 이는 기본값입니다.

요청-응답 밸런싱. 각 MQPUT 호출 후 응답 메시지에 대해 일치하는 MQGET 호출이 예상됩니다. 이 메시지가 수신되거나 요청 메시지 EXPIRY가 초과될 때까지 밸런싱이 지연됩니다.

또한 이 특성은 `client.ini` 파일에서 설정할 수 있습니다. 환경 설정의 순서는 다음과 같습니다.

1. 애플리케이션에 설정된 특성
2. `mqclient.ini` 파일에서 이름이 지정된 [애플리케이션 스탠자](#)와 일치합니다.
3. `mqclient.ini` 파일의 [애플리케이션 기본값 스탠자](#).

## **XMSC\_WMQ\_BALANCING\_OPTIONS**

### 데이터 유형:

System.Int32

### 특성:

ConnectionFactory

특성의 올바른 값은 다음과 같습니다.

### 올바른 값

XMSC\_WMQ\_BALANCING\_OPTIONS\_NONE

XMSC\_WMQ\_BALANCING\_OPTIONS\_IGNORE\_TRANSACTION

### 해당 값

옵션이 설정되지 않았습니다. (기본값)

이 옵션을 설정하면 트랜잭션 중간에 있는 경우에도 애플리케이션을 다시 밸런싱할 수 있습니다.

또한 이 특성은 `client.ini` 파일에서 설정할 수 있습니다. 환경 설정의 순서는 다음과 같습니다.

1. 애플리케이션에 설정된 특성
2. `mqclient.ini` 파일에서 이름이 지정된 애플리케이션 스탠자와 일치합니다.
3. `mqclient.ini` 파일의 애플리케이션 기본값 스탠자.

## **XMSC\_WMQ\_BALANCING\_TIMEOUT**

### 데이터 유형:

System.Int32

### 특성:

ConnectionFactory

특성의 올바른 값은 다음과 같습니다.

### 올바른 값

XMSC\_WMQ\_BALANCING\_TIMEOUT\_IMMEDIATE

XMSC\_WMQ\_BALANCING\_TIMEOUT\_AS\_DEFAULT

XMSC\_WMQ\_BALANCING\_TIMEOUT\_NEVER

### 의미

즉시 제한시간이 초과됨

설정된 기본 제한시간 값입니다. (기본값)

제한시간이 초과되지 않음

**참고:** 정의된 값 또는 0-999999999초의 값에서만 하나의 값을 제공해야 합니다.

또한 이 특성은 `client.ini` 파일에서 설정할 수 있습니다. 환경 설정의 순서는 다음과 같습니다.

1. 애플리케이션에 설정된 특성
2. `mqclient.ini` 파일에서 이름이 지정된 애플리케이션 스탠자와 일치합니다.
3. `mqclient.ini` 파일의 애플리케이션 기본값 스탠자.

## **XMSC\_WMQ\_BROKER\_CONTROLQ**

### 데이터 유형:

문자열

### 특성:

ConnectionFactory

브로커가 사용하는 제어 큐의 이름입니다.

이 특성의 기본값은 SYSTEM.BROKER.CONTROL.QUEUE입니다.

이 특성은 발행/구독 도메인에만 관련됩니다.

## **XMSC\_WMQ\_BROKER\_PUBQ**

### 데이터 유형:

문자열

### 특성:

ConnectionFactory

브로커가 모니터링하는 큐의 이름입니다. 애플리케이션은 발행하는 메시지를 이 큐로 전송합니다.

이 특성의 기본값은 SYSTEM.BROKER.DEFAULT.STREAM입니다.

이 특성은 발행/구독 도메인에만 관련됩니다.

### ***XMSC\_WMQ\_BROKER\_QMGR***

**데이터 유형:**

문자열

**특성:**

ConnectionFactory

브로커가 연결되는 큐 관리자의 이름입니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 발행/구독 도메인에만 관련됩니다.

### ***XMSC\_WMQ\_BROKER\_SUBQ***

**데이터 유형:**

문자열

**특성:**

ConnectionFactory

지속 불가능한 메시지 이용자의 구독자 큐 이름입니다.

구독자 큐 이름은 다음 문자로 시작되어야 합니다.

SYSTEM.JMS.ND.

지속 불가능한 모든 이용자가 구독자 큐를 공유하도록 하려면 공유 큐의 완전한 이름을 지정하십시오. 지정된 이름의 큐가 존재해야 애플리케이션이 지속 불가능한 메시지를 생성할 수 있습니다.

각 비지속 메시지 이용자가 고유한 독점 구독자 큐에서 메시지를 검색하게 하려면 별표(\*)로 끝나는 큐 이름을 지정하십시오. 그런 다음 애플리케이션이 지속 불가능한 메시지를 생성할 때 XMS 클라이언트는 메시지 이용자가 독점적으로 사용할 동적 큐를 작성합니다. XMS 클라이언트는 이 특성 값을 사용하여 동적 큐를 작성하는데 사용되는 오브젝트 디스크립터에 있는 **DynamicQName** 필드의 콘텐츠를 설정합니다.

특성의 기본값은 SYSTEM.JMS.ND.SUBSCRIBER.QUEUE- XMS 가 기본적으로 공유 큐 접근 방식을 사용함을 의미합니다.

이 특성은 발행/구독 도메인에만 관련됩니다.

### ***XMSC\_WMQ\_BROKER\_VERSION***

**데이터 유형:**

System.Int32

**특성:**

ConnectionFactory 및 Destination

**URI에서 사용되는 이름:**

brokerVersion

애플리케이션이 연결이나 목적지에 사용하는 브로커의 유형입니다. 토픽인 대상에만 이 특성이 있을 수 있습니다.

특성의 올바른 값은 다음과 같습니다.

#### **올바른 값**

XMSC\_WMQ\_BROKER\_V1

#### **의미**

애플리케이션이 IBM MQ 발행/구독 브로커를 사용합니다.

IBM MQ 발행/구독에서 WebSphere Message Broker로 마이그레이션하지만 애플리케이션을 변경하지 않은 경우 애플리케이션은 이 값을 사용할 수도 있습니다.

XMSC\_WMQ\_BROKER\_V2

애플리케이션이 IBM Integration Bus의 브로커를 사용합니다.

## 올바른 값

XMSC\_WMQ\_BROKER\_UNSPECIFIED

## 의미

브로커를 마이그레이션한 후 RFH2 헤더가 더 이상 사용되지 않도록 이 특성을 설정하십시오. 마이그레이션하고 나면 이 특성은 더 이상 적절하지 않습니다.

connectionfactory의 기본값은 XMSC\_WMQ\_BROKER\_UNSPECIFIED이지만 기본적으로 이 특성은 목적지에 대해서는 설정되지 않습니다. 목적지에 대한 특성을 설정하면 연결 팩토리 특성에서 지정한 값을 모두 대체합니다.

### ***XMSC\_WMQ\_CCDTURL***

#### 데이터 유형:

System.String

#### 특성:

ConnectionFactory

#### 적용 가능한 오브젝트:

JMS 관리 도구 긴 이름: CCDTURL

JMS 관리 도구 짧은 이름: CCDT

클라이언트 채널 정의 테이블을 포함하는 파일의 이름과 위치를 식별하고 이 파일의 액세스 방법을 지정하는 URL(Uniform Resource Locator)입니다.

기본적으로 이 특성은 설정되어 있지 않습니다.

### ***XMSC\_WMQ\_CCSID***

#### 데이터 유형:

System.Int32

#### 특성:

목적지

#### URI에서 사용되는 이름:

CCSID

메시지 본문의 문자 데이터 문자열이 있는 코드화된 문자 세트 또는 코드 페이지의 ID(CCSID)입니다. XMS 클라이언트는 메시지를 대상으로 전달합니다. 개별 메시지에 대해 설정된 경우 JMS IBM CHARACTER SET 특성이 이 특성에서 목적지에 대해 지정한 CCSID를 대체합니다.

이 특성의 기본값은 1208입니다.

이 특성은 목적지에서 수신한 메시지가 아니라 목적지로 전송된 메시지에만 관련됩니다.

### ***XMSC\_WMQ\_CHANNEL***

#### 데이터 유형:

문자열

#### 특성:

ConnectionFactory

#### 적용 가능한 오브젝트:

JMS 관리 도구 긴 이름: CHANNEL

JMS 관리 도구 짧은 이름: CHAN

연결에 사용되는 채널 이름입니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 클라이언트 모드의 큐 관리자에 연결되는 경우에만 관련됩니다.

## ***XMSC\_WMQ\_CLIENT\_RECONNECT\_OPTIONS***

데이터 유형:

문자열

특성:

ConnectionFactory

적용 가능한 오브젝트:

JMS 관리 도구 긴 이름: CLIENTRECONNECTOPTIONS

JMS 관리 도구 짧은 이름: CROPT

이 특성은 이 팩토리에서 작성된 새 연결에 대한 클라이언트 다시 연결 옵션을 지정합니다. 이는 XMSC에 있으며 다음 중 하나입니다.

- WMQ\_CLIENT\_RECONNECT\_AS\_DEF(기본값). mqclient.ini 파일에 지정된 값을 사용하십시오. 채널 스탠자 내의 **DefRecon** 특성을 사용하여 값을 설정하십시오. 다음 중 하나로 설정할 수 있습니다.
  1. 예 WMQ\_CLIENT\_RECONNECT 옵션으로 사용
  2. 아니오. 기본값. 다시 연결 옵션을 지정하지 않음
  3. QMGR. WMQ\_CLIENT\_RECONNECT\_Q\_MGR 옵션으로 사용
  4. DISABLED. WMQ\_CLIENT\_RECONNECT\_DISABLED 옵션으로 사용
- WMQ\_CLIENT\_RECONNECT. 연결 이름 목록에 지정된 큐 관리자에 다시 연결합니다.
- WMQ\_CLIENT\_RECONNECT\_Q\_MGR. 원래 연결된 동일한 큐 관리자에 다시 연결합니다. 연결하려고 하는 큐 관리자(연결 이름 목록에서 지정됨)에 원래 연결되는 큐 관리자와 다른 QMID가 있는 경우 MQRC\_RECONNECT\_QMID\_MISMATCH가 리턴됩니다.
- WMQ\_CLIENT\_RECONNECT\_DISABLED. 다시 연결을 사용할 수 없습니다.

## ***XMSC\_WMQ\_CLIENT\_RECONNECT\_TIMEOUT***

데이터 유형:

문자열

특성:

ConnectionFactory

적용 가능한 오브젝트:

JMS 관리 도구 긴 이름: CLIENTRECONNECTTIMEOUT

JMS 관리 도구 짧은 이름: CRT

XMSC\_WMQ\_CLIENT\_RECONNECT\_TIMEOUT 특성은 관리 XMS .NET 클라이언트에만 유효합니다.

이 특성은 클라이언트 연결이 다시 연결하려고 시도하는 시간(초)을 지정합니다.

이 시간 동안 다시 연결하려고 시도한 후에는 클라이언트가 MQRC\_RECONNECT\_FAILED와 함께 실패합니다. 이 특성의 기본 설정은 XMSC.WMQ\_CLIENT\_RECONNECT\_TIMEOUT\_DEFAULT입니다.

이 특성의 기본값은 1800입니다.

## ***XMSC\_WMQ\_CONNECTION\_MODE***

데이터 유형:

System.Int32

특성:

ConnectionFactory

애플리케이션이 큐 관리자에 연결할 때 사용되는 모드입니다.

특성의 올바른 값은 다음과 같습니다.

올바른 값	의미
XMSC_WMQ_CM_BINDINGS	최적의 성능을 위한 바인딩 모드의 큐 관리자에 대한 연결입니다. 이 값은 C/C++에 대한 기본값입니다.
XMSC_WMQ_CM_CLIENT	완전한 관리 스택을 위한 클라이언트 모드의 큐 관리자에 대한 연결입니다. 이 값은 .NET에 대한 기본값입니다.
XMSC_WMQ_CM_CLIENT_UNMANAGED(.NET 만 해당)	비관리 클라이언트 스택을 강제 실행하는 큐 관리자에 대한 연결입니다.

### **XMSC\_WMQ\_CONNECTION\_NAME\_LIST**

데이터 유형:

문자열

특성:

ConnectionFactory

적용 가능한 오브젝트:

JMS 관리 도구 긴 이름: CONNECTIONNAMELIST

JMS 관리 도구 짧은 이름: CNLIST

이 특성은 연결이 끊어진 후 클라이언트가 다시 연결하려고 시도하는 호스트를 지정합니다.

연결 이름 목록은 쉼표로 구분된 호스트/IP 포트 쌍의 목록입니다. 이 특성의 기본 설정은 WMQ\_CONNECTION\_NAME\_LIST\_DEFAULT입니다.

예를 들어 127.0.0.1(1414), host2.example.com(1400)입니다.

이 특성의 기본 설정은 localhost(1414)입니다.

### **XMSC\_WMQ\_DUR\_SUBQ**

데이터 유형:

문자열

특성:

목적지

목적지에서 메시지를 수신하는 지속 가능한 구독자의 구독자 큐 이름입니다. 토픽인 대상에만 이 특성이 있을 수 있습니다.

구독자 큐 이름은 다음 문자로 시작되어야 합니다.

SYSTEM.JMS.D.

지속 가능한 모든 구독자가 구독자 큐를 공유하도록 하려면 공유 큐의 완전한 이름을 지정하십시오. 지정된 이름의 큐가 존재해야 애플리케이션이 지속 가능한 구독자를 작성할 수 있습니다.

지속 가능한 각 구독자가 고유한 독점 구독자 큐에서 메시지를 검색하게 하려면 별표(\*)로 끝나는 큐 이름을 지정하십시오. 그런 다음 애플리케이션이 지속 가능한 구독자를 작성할 때 XMS 클라이언트는 지속 가능한 구독자가 독점 사용할 동적 큐를 작성합니다. XMS 클라이언트는 이 특성 값을 사용하여 동적 큐를 작성하는 데 사용되는 오브젝트 디스크립터에 있는 **DynamicQName** 필드의 콘텐츠를 설정합니다.

특성의 기본값은 SYSTEM.JMS.D.SUBSCRIBER.QUEUE- XMS 가 기본적으로 공유 큐 접근 방식을 사용함을 의미합니다.

이 특성은 발행/구독 도메인에만 관련됩니다.

### **XMSC\_WMQ\_ENCODING**

데이터 유형:

System.Int32


특성:

목적지

메시지 본문의 숫자 데이터가 표시되는 방식 XMS 클라이언트는 메시지를 대상으로 전달합니다. 개별 메시지에 대해 설정된 경우 `JMS IBM_ENCODING` 특성이 이 특성에서 목적지에 대해 지정한 인코딩을 대체합니다. 이 특성은 2진 정수, 팩형 10진수 정수 및 부동 소수점 숫자의 표시를 지정합니다.

특성의 올바른 값은 메시지 디스크립터의 **Encoding** 필드에 지정할 수 있는 값과 동일합니다.

애플리케이션은 다음과 같은 이름 지정된 상수를 사용하여 특성을 설정할 수도 있습니다.

이름 지정된 상수	의미
<code>MQENC_INTEGER_NORMAL</code>	표준 정수 인코딩
<code>MQENC_INTEGER_REVERSED</code>	역방향 정수 인코딩
<code>MQENC_DECIMAL_NORMAL</code>	표준 팩형 10진수 인코딩
<code>MQENC_DECIMAL_REVERSED</code>	역방향 팩형 10진수 인코딩
<code>MQENC_FLOAT_IEEE_NORMAL</code>	표준 IEEE 부동 소수점 인코딩
<code>MQENC_FLOAT_IEEE_REVERSED</code>	역방향 IEEE 부동 소수점 인코딩
 <code>MQENC_FLOAT_S390</code>	z/OS 아키텍처 부동 소수점 인코딩
<code>MQENC_NATIVE</code>	기본 시스템 인코딩

특성 값을 구성하기 위해 애플리케이션은 다음과 같이 세 개의 상수를 추가할 수 있습니다.

- 이름이 `MQENC_INTEGER`로 시작하는 상수. 2진 정수의 표시를 지정합니다.
- 이름이 `MQENC_DECIMAL`로 시작하는 상수. 팩형 10진 정수의 표시를 지정합니다.
- 이름이 `MQENC_FLOAT`로 시작하는 상수. 부동 소수점 수를 지정합니다.

또는 애플리케이션이 특성을 `MQENC_NATIVE`로 설정할 수 있습니다. 이 값은 환경에 따라 다릅니다.

이 특성의 기본값은 `MQENC_NATIVE`입니다.

이 특성은 목적지에서 수신한 메시지가 아니라 목적지로 전송된 메시지에만 관련됩니다.

### ***XMSC\_WMQ\_FAIL\_IF\_QUIESCE***

데이터 유형:

`System.Int32`

특성:

`ConnectionFactory` 및 `Destination`

URI에서 사용되는 이름:

`failIfQuiesce`

적용 가능한 오브젝트:

JMS 관리 도구 긴 이름: `FAILIFQUIESCE`

JMS 관리 도구 짧은 이름: `FIQ`

애플리케이션이 연결된 큐 관리자가 정지 상태인 경우 특정 메소드에 대한 호출의 실패 여부를 지정합니다.

특성의 올바른 값은 다음과 같습니다.

올바른 값	의미
<code>XMSC_WMQ_FIQ_YES</code>	큐 관리자가 정지(quiescing) 상태에 있는 경우 특정 메소드에 대한 호출이 실패합니다. 애플리케이션은 큐 관리자가 정지 상태임을 감지하면 즉시 태스크를 완료하고 연결을 닫아 큐 관리자를 중지할 수 있습니다.
<code>XMSC_WMQ_FIQ_NO</code>	큐 관리자가 정지 상태이기 때문에 메소드에 대한 호출이 실패하지 않습니다. 이 값을 지정하면 애플리케이션이 큐 관리자가 정지 상태임을 감지할 수 없습니다. 애플리케이션은 큐 관리자에 대한 조작을 계속 수행합니다. 따라서 큐 관리자가 중지하는 것을 방지할 수 있습니다.



연결 팩토리의 기본값은 XMSC\_WMQ\_FIQ\_YES이지만 기본적으로 이 특성은 목적지에 대해서는 설정되지 않습니다. 목적지에 대한 특성을 설정하면 연결 팩토리 특성에서 지정한 값을 모두 대체합니다.

### **XMSC\_WMQ\_MESSAGE\_BODY**

**데이터 유형:**  
System.Int32

**특성:**  
목적지

이 속성은 XMS 애플리케이션이 처리하는 MQRFH2 ~의 IBM MQ 메시지 페이로드의 일부(즉, 메시지 본문의 일부)로 메시지를 보냅니다.

**참고:** 목적지로 메시지를 전송할 때, XMSC\_WMQ\_MESSAGE\_BODY 특성은 기존 XMS 목적지 특성 XMSC\_WMQ\_TARGET\_CLIENT를 대체합니다.

이 특성의 올바른 값은 다음과 같습니다.

### **XMSC\_WMQ\_MESSAGE\_BODY\_JMS**

**수신:** 인바운드 XMS 메시지 유형 및 본문은 수신된 IBM MQ 메시지에 있는 MQRFH2(있는 경우) 또는 MQMD(MQRFH2가 없는 경우)의 콘텐츠에 의해 결정됩니다.

**송신:** 아웃바운드 XMS 메시지 본문에는 XMS 메시지 특성 및 헤더 필드를 기반으로 하여 접두부가 붙거나 자동 생성된 MQRFH2 헤더가 포함됩니다.

### **XMSC\_WMQ\_MESSAGE\_BODY\_MQ**

**수신:** 인바운드 XMS 메시지 유형은 수신된 IBM MQ 메시지의 콘텐츠 또는 수신된 MQMD의 형식 필드에 상관 없이 항상 ByteMessage입니다. XMS 메시지 본문은 기본 메시징 제공자 API 호출에 의해 리턴된 변경되지 않은 메시지 데이터입니다. 메시지 본문에 있는 데이터의 문자 세트와 인코딩은 MQMD의 CodedCharSetId 및 Encoding 필드에 의해 판별됩니다. 메시지 본문에 있는 데이터의 형식은 MQMD의 Format 필드에 의해 판별됩니다.

**송신:** 아웃바운드 XMS 메시지 본문에는 애플리케이션 페이로드(payload)가 그대로 포함됩니다. 자동 생성된 IBM MQ 헤더는 본문에 추가되지 않습니다.

### **XMSC\_WMQ\_MESSAGE\_BODY\_UNSPECIFIED**

**수신:** XMS 클라이언트는 이 특성에 적합한 값을 판별합니다. 수신 경로에서 이 값은 WMQ\_MESSAGE\_BODY\_JMS 특성 값입니다.

**송신:** XMS 클라이언트는 이 특성에 적합한 값을 판별합니다. 송신 경로에서 이 값은 XMSC\_WMQ\_TARGET\_CLIENT 특성 값입니다.

기본적으로 이 특성은 XMSC\_WMQ\_MESSAGE\_BODY\_UNSPECIFIED로 설정됩니다.

### **XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT**

**데이터 유형:**  
System.Int32

**특성:**  
목적지

메시지 컨텍스트의 수준을 결정합니다. XMS 애플리케이션. 이 특성이 적용되려면 애플리케이션이 적합한 컨텍스트 권한으로 실행 중이어야 합니다.

이 특성의 올바른 값은 다음과 같습니다.

### **XMSC\_WMQ\_MDCTX\_DEFAULT**

아웃바운드 메시지의 경우, MQOPEN API 호출과 MQPMO 구조는 명시적인 메시지 컨텍스트 옵션을 지정하지 않습니다.

### **XMSC\_WMQ\_MDCTX\_SET\_IDENTITY\_CONTEXT**

MQOPEN API 호출이 메시지 컨텍스트 옵션 MQOO\_SET\_IDENTITY\_CONTEXT를 지정하고 MQPMO 구조가 MQPMO\_SET\_IDENTITY\_CONTEXT를 지정합니다.

### **XMSC\_WMQ\_MDCTX\_SET\_ALL\_CONTEXT**

MQOPEN API 호출이 메시지 컨텍스트 옵션 MQOO\_SET\_ALL\_CONTEXT를 지정하고 MQPMO 구조가 MQPMO\_SET\_ALL\_CONTEXT를 지정합니다.

기본적으로 이 특성은 XMSC\_WMQ\_MDCTX\_DEFAULT로 설정됩니다.

**참고:** 이 특성은 애플리케이션이 WebSphere Application Server service integration bus에 연결된 경우에는 관련되지 않습니다.

메시지를 보낼 때 다음 특성이 적용되도록 하려면 XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT 특성을 XMSC\_WMQ\_MDCTX\_SET\_IDENTITY\_CONTEXT 특성 값 또는 XMSC\_WMQ\_MDCTX\_SET\_ALL\_CONTEXT 특성 값으로 설정해야 합니다.

- JMS\_IBM\_MQMD\_USERIDENTIFIER
- JMS\_IBM\_MQMD\_ACCOUNTINGTOKEN
- JMS\_IBM\_MQMD\_APPLIDENTITYDATA

메시지를 보낼 때 다음 특성이 적용되도록 하려면 XMSC\_WMQ\_MQMD\_MESSAGE\_CONTEXT 특성을 XMSC\_WMQ\_MDCTX\_SET\_ALL\_CONTEXT 특성 값으로 설정해야 합니다.

- JMS\_IBM\_MQMD\_PUTAPPLTYPE
- JMS\_IBM\_MQMD\_PUTAPPLNAME
- JMS\_IBM\_MQMD\_PUTDATE
- JMS\_IBM\_MQMD\_PUTTIME
- JMS\_IBM\_MQMD\_APPLORIGINDATA

### **XMSC\_WMQ\_MQMD\_READ\_ENABLED**

데이터 유형:

System.Int32

특성:

목적지

이 속성은 XMS 애플리케이션은 MQMD 필드의 값을 추출할 수 있는지 여부를 결정합니다.

이 특성의 올바른 값은 다음과 같습니다.

### **XMSC\_WMQ\_READ\_ENABLED\_NO**

메시지를 송신할 때 송신된 메시지의 JMS\_IBM\_MQMD\* 특성이 MQMD에서 업데이트된 필드 값을 반영하도록 업데이트되지 않습니다.

메시지를 수신할 때 송신자가 일부 또는 모든 특성을 설정했다더라도 JMS\_IBM\_MQMD\* 특성을 수신 메시지에서 사용할 수 없습니다.

### **XMSC\_WMQ\_READ\_ENABLED\_YES**

메시지를 송신할 때 송신자가 명시적으로 설정하지 않은 해당 특성을 포함하여 MQMD에서 업데이트된 필드 값을 반영하기 위해 송신된 메시지의 모든 JMS\_IBM\_MQMD\* 특성이 업데이트됩니다.

메시지를 수신할 때 송신자가 명시적으로 설정하지 않은 해당 특성을 포함하여 모든 JMS\_IBM\_MQMD\* 특성을 수신 메시지에서 사용할 수 있습니다.

기본적으로 이 특성은 XMSC\_WMQ\_READ\_ENABLED\_NO로 설정됩니다.

### **XMSC\_WMQ\_MQMD\_WRITE\_ENABLED**

데이터 유형:

System.Int32

특성:

목적지

이 속성은 XMS 애플리케이션은 MQMD 필드의 값을 설정할지 여부를 결정할 수 있습니다.

이 특성의 올바른 값은 다음과 같습니다.

**XMSC\_WMQ\_WRITE\_ENABLED\_NO**

모든 JMS\_IBM\_MQMD\* 특성이 무시되며, 해당 값이 기본 MQMD 구조로 복사되지 않습니다.

**XMSC\_WMQ\_WRITE\_ENABLED\_YES**

JMS\_IBM\_MQMD\* 특성이 처리됩니다. 해당 값이 기본 MQMD 구조로 복사됩니다.

기본적으로 이 특성은 XMSC\_WMQ\_WRITE\_ENABLED\_NO로 설정됩니다.

**XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED**

데이터 유형:

System.Int32

특성:

목적지

이 특성은 메시지 생성자가 비동기 Put을 사용하여 메시지를 이 목적지로 송신할 수 있는지 여부를 판별합니다.

이 특성의 올바른 값은 다음과 같습니다.

**XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_AS\_DEST**

큐 정의 또는 토픽 정의를 참조하여 비동기 Put이 허용되는지 여부를 판별합니다.

**XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_AS\_Q\_DEF**

큐 정의를 참조하여 비동기 Put이 허용되는지 여부를 판별합니다.

**XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_AS\_TOPIC\_DEF**

토픽 정의를 참조하여 비동기 Put이 허용되는지 여부를 판별합니다.

**XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_DISABLED**

비동기 Put이 허용되지 않습니다.

**XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_ENABLED**

비동기 Put이 허용됩니다.

기본적으로 이 특성은 XMSC\_WMQ\_PUT\_ASYNC\_ALLOWED\_AS\_DEST로 설정됩니다.

**참고:** 이 특성은 애플리케이션이 WebSphere Application Server service integration bus에 연결된 경우에는 관련되지 않습니다.

**XMSC\_WMQ\_READ\_AHEAD\_ALLOWED**

데이터 유형:

System.Int32

특성:

목적지

이 특성은 메시지를 수신하기 전에 메시지 이용자와 큐 브라우저가 이 목적지에서 내부 버퍼로 트랜잭트되지 않은 비지속적 메시지를 가져오기 위해 미리 읽기를 사용할 수 있는지 여부를 판별합니다.

이 특성의 올바른 값은 다음과 같습니다.

**XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_AS\_Q\_DEF**

큐 정의를 참조하여 미리 읽기가 허용되는지 여부를 판별합니다.

**XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_AS\_TOPIC\_DEF**

토픽 정의를 참조하여 미리 읽기가 허용되는지 여부를 판별합니다.

**XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_AS\_DEST**

큐 정의 또는 토픽 정의를 참조하여 미리 읽기가 허용되는지 여부를 판별합니다.

### **XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_DISABLED**

메시지를 이용하거나 찾아보는 동안 미리 읽기가 허용되지 않습니다.

### **XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_ENABLED**

미리 읽기가 허용됩니다.

기본적으로 이 특성은 XMSC\_WMQ\_READ\_AHEAD\_ALLOWED\_AS\_DEST로 설정됩니다.

### **XMSC\_WMQ\_READ\_AHEAD\_CLOSE\_POLICY**

데이터 유형:

System.Int32

특성:

목적지

비동기 메시지 리스너로 전달되는 메시지의 경우, 메시지 이용자를 닫을 때 내부 미리 읽기 버퍼에서 메시지에 일어나는 사항을 이 특성에서 판별합니다.

이 특성은 목적지에서 메시지를 이용할 때 큐 닫기 옵션을 지정하는 데 사용되지만 목적지로 메시지를 보내는 경우에는 적용되지 않습니다.

찾아보는 중에 큐에서 메시지를 계속 사용할 수 있으므로 큐 브라우저의 경우 이 특성이 무시됩니다.

이 특성의 올바른 값은 다음과 같습니다.

### **XMSC\_WMQ\_READ\_AHEAD\_CLOSE\_POLICY\_DELIVER\_CURRENT**

리턴하기 전에 현재 메시지 리스너 호출만 완료하고, 내부 미리 읽기 버퍼에 메시지를 남길 수 있습니다. 이 메시지는 이후 제거됩니다.

### **XMSC\_WMQ\_READ\_AHEAD\_CLOSE\_POLICY\_DELIVER\_ALL**

리턴하기 전에 내부 미리 읽기 버퍼의 모든 메시지가 애플리케이션 메시지 리스너로 전달됩니다.

기본적으로 이 특성은 XMSC\_WMQ\_READ\_AHEAD\_CLOSE\_POLICY\_DELIVER\_CURRENT로 설정됩니다.

참고:

#### **비정상 애플리케이션 종료**

XMS 애플리케이션이 비정상적으로 종료되면 미리 읽기 버퍼의 모든 메시지가 유실됩니다.

#### **트랜잭션의 영향**

애플리케이션에서 트랜잭션을 사용하는 경우 미리 읽기를 사용할 수 없습니다. 따라서 애플리케이션이 트랜잭션된 세션을 사용할 때 동작에서 차이가 없습니다.

#### **세션 수신확인 모드의 영향**

수신확인 모드가 XMSC\_AUTO\_ACKNOWLEDGE 또는 XMSC\_DUPS\_OK\_ACKNOWLEDGE인 경우 트랜잭션되지 않은 세션에서 미리 읽기를 사용할 수 있습니다. 세션 수신확인 모드가 XMSC\_CLIENT\_ACKNOWLEDGE인 경우 트랜잭션된 세션 또는 트랜잭션되지 않은 세션인지에 상관없이 미리 읽기를 사용할 수 없습니다.

#### **큐 브라우저 및 큐 브라우저 선택자의 영향**

XMS 애플리케이션에서 사용되는 큐 브라우저 및 큐 브라우저 선택자는 미리 읽기를 통한 성능상 이점을 얻을 수 있습니다. 이후 조작을 위해 큐에서 메시지를 여전히 사용할 수 있으므로 큐 브라우저를 닫아도 성능이 저하되지 않습니다. 미리 읽기의 성능상 이점을 제외하고 큐 브라우저와 큐 브라우저 선택자에 대한 다른 영향은 없습니다.

### **XMSC\_WMQ\_HOST\_NAME**

데이터 유형:

문자열

특성:

ConnectionFactory

적용 가능한 오브젝트:

JMS 관리 도구 긴 이름: HOSTNAME

JMS 관리 도구 짧은 이름: HOST

큐 관리자가 실행하는 시스템의 호스트 이름 또는 IP 주소입니다.

이 특성은 애플리케이션이 클라이언트 모드의 큐 관리자에 연결되는 경우에만 사용됩니다. 이 특성은 `XMSC_WMQ_PORT` 특성과 함께 사용하여 큐 관리자를 식별합니다.

이 특성의 기본값은 `localhost`입니다.

## **XMSC\_WMQ\_LOCAL\_ADDRESS**

데이터 유형:

문자열

특성:

ConnectionFactory

적용 가능한 오브젝트:

JMS 관리 도구 긴 이름: LOCALADDRESS

JMS 관리 도구 짧은 이름: LA

큐 관리자에 대한 연결의 경우 이 특성은 사용할 로컬 네트워크 인터페이스를 지정하거나, 사용할 로컬 포트나 로컬 포트의 범위 또는 둘 다를 지정합니다.

이 특성의 값은 다음 형식의 문자열입니다.

`[host_name][(low_port)[,high_port]]`

변수의 의미는 다음과 같습니다.

### **host\_name**

연결에 사용하는 로컬 네트워크 인터페이스의 호스트 이름 또는 IP 주소입니다.

이 정보는 애플리케이션이 실행 중인 시스템에 두 개 이상의 네트워크 인터페이스가 있는 경우에만 필요하며 사용자가 연결에 사용되는 인터페이스를 지정할 수 있어야 합니다. 시스템에 네트워크 인터페이스가 하나뿐이면 해당 인터페이스만 사용할 수 있습니다. 시스템에 네트워크 인터페이스가 둘 이상이고 사용해야 할 인터페이스를 지정하지 않으면 인터페이스가 임의로 선택됩니다.

### **low\_port**

연결에 사용되는 로컬 포트의 번호입니다.

`high_port`도 함께 지정하면 `low_port`가 포트 번호 범위 중에서 가장 낮은 번호로 해석됩니다.

### **high\_port**

포트 번호 범위에서 가장 높은 포트 번호입니다. 지정된 범위의 포트 중 하나를 연결에 사용해야 합니다.

문자열의 최대 길이는 48자입니다.

다음은 특성의 올바른 값에 대한 예입니다.

```
JUPITER
9.20.4.98
JUPITER(1000)
9.20.4.98(1000,2000)
(1000)
(1000,2000)
```

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 클라이언트 모드의 큐 관리자에 연결되는 경우에만 관련됩니다.

## **XMSC\_WMQ\_MESSAGE\_SELECTION**

데이터 유형:

System.Int32

특성:

ConnectionFactory

메시지 선택이 수행되는지 여부를 결정합니다.XMS 클라이언트 또는 브로커에 의해.

특성의 올바른 값은 다음과 같습니다.

올바른 값	의미
XMSC_WMQ_MSEL_CLIENT	XMS 클라이언트에서 메시지를 선택합니다.
XMSC_WMQ_MSEL_BROKER	브로커를 통해 메시지가 선택됩니다.

기본값은 XMSC\_WMQ\_MSEL\_CLIENT입니다.

이 특성은 발행/구독 도메인에만 관련됩니다. XMSC\_WMQ\_BROKER\_VERSION 특성이 XMSC\_WMQ\_BROKER\_V1로 설정된 경우 브로커의 메시지 선택이 지원되지 않습니다.

## **XMSC\_WMQ\_MSG\_BATCH\_SIZE**

데이터 유형:

System.Int32

특성:

ConnectionFactory

비동기 메시지 전달을 사용할 때 하나의 일괄처리로 큐에서 검색되는 최대 메시지 수입니다.

애플리케이션이 비동기 메시지 전달을 사용하는 경우 특정 조건에서는 XMS 클라이언트가 각 메시지를 개별적으로 애플리케이션에 전달하기 전에 큐에서 메시지 배치를 검색합니다. 이 특성은 배치에 있을 수 있는 최대 메시지 수를 지정합니다.

특성 값은 양의 정수이며 기본값은 10입니다. 해결해야 하는 특정한 성능 문제점이 있는 경우에만 특성을 다른 값으로 설정하십시오.

애플리케이션이 네트워크를 통해 큐 관리자에 연결된 경우 이 특성의 값을 높이면 네트워크 오버헤드 및 응답 시간을 감소시킬 수 있습니다. 그러나 클라이언트 시스템에 메시지를 저장하는 데 필요한 메모리 양이 증가합니다. 반대로 이 특성의 값을 낮추면 네트워크 오버헤드 및 응답 시간이 증가하지만 메시지를 저장하는 데 필요한 메모리의 양을 감소시킬 수 있습니다.

## **XMSC\_WMQ\_POLLING\_INTERVAL**

데이터 유형:

System.Int32

특성:

ConnectionFactory

세션에 있는 각 메시지 리스너의 큐에 적합한 메시지가 없는 경우, 이 값은 각 메시지 리스너가 해당 큐에서 메시지를 가져오기를 다시 시도하기 전에 경과하는 최대 간격(밀리세컨드)입니다.

세션의 메시지 리스너에 사용 가능한 적합한 메시지가 없는 경우가 자주 발생하면 이 특성 값을 높여 보십시오.

특성 값은 양의 정수입니다. 기본값은 5000입니다.

## **XMSC\_WMQ\_PORT**

데이터 유형:

System.Int32

특성:

ConnectionFactory

적용 가능한 오브젝트:

JMS 관리 도구 긴 이름: PORT

JMS 관리 도구 짧은 이름: PORT

큐 관리자 리스너가 수신 요청을 청취하는 포트 번호입니다.

이 특성은 애플리케이션이 클라이언트 모드의 큐 관리자에 연결되는 경우에만 사용됩니다. 이 특성은 XMSC\_WMQ\_HOST\_NAME 특성과 함께 사용하여 큐 관리자를 식별합니다.

이 특성의 기본값은 XMSC\_WMQ\_DEFAULT\_CLIENT\_PORT 또는 1414입니다.

## ***XMSC\_WMQ\_PROVIDER\_VERSION***

데이터 유형:

문자열

특성:

ConnectionFactory

이 특성의 올바른 애플리케이션이 연결하려고 하는 큐 관리자의 버전, 릴리스, 수정 레벨 및 수정팩입니다. 값은 다음과 같습니다.

- 지정되지 않음

또는 다음 형식 중 하나의 문자열

- V.R.M.F
- V.R.M
- V.R
- V

여기서 V, R, M 및 F는 0 이상의 정수 값입니다.

기본적으로 이 특성은 "unspecified"로 설정됩니다.

IBM MQ 클라이언트의 버전은 XMS 클라이언트 애플리케이션이 IBM MQ 특정 기능을 사용할 수 있는지 여부에 중요한 역할을 합니다.

**참고:** 시스템 특성 `XMSC_WMQ_OVERRIDEPROVIDERVERSION`은 `XMSC_WMQ_PROVIDER_VERSION` 특성을 대체합니다. 이 특성은 연결 팩토리 설정을 변경할 수 없는 경우에 사용할 수 있습니다.

## ***XMSC\_WMQ\_PUB\_ACK\_INTERVAL***

데이터 유형:

System.Int32

특성:

ConnectionFactory

게시자가 게시하기 전에 게시한 메시지 수 XMS 클라이언트는 브로커에게 승인을 요청합니다.

이 특성의 값을 낮추면 클라이언트가 수신확인을 더 빈번하게 요청하므로 발행자의 성능이 저하됩니다. 값을 올리는 경우 브로커가 실패하면 클라이언트가 예외를 처리하는 데 더 많은 시간이 걸립니다.

특성 값은 양의 정수입니다. 기본 값은 25입니다.

## ***XMSC\_WMQ\_QMGR\_CCSID***

데이터 유형:

System.Int32

특성:

ConnectionFactory

MQI(Message Queue Interface)에 정의된 문자 데이터 필드가 다음 사이에서 교환되는 코드화된 문자 세트 또는 코드 페이지의 식별자(CCSID)입니다. XMS 클라이언트와 IBM MQ 고객. 이 특성은 메시지 본문의 문자 데이터 문자열에 적용되지 않습니다.

XMS 애플리케이션이 클라이언트 모드의 큐 관리자에 연결될 때 XMS 클라이언트가 IBM MQ 클라이언트에 연결됩니다. 두 클라이언트 사이에서 교환되는 정보에는 MQI에서 정의된 문자 데이터의 필드가 포함됩니다. 정상적인 상황에서는 IBM MQ 클라이언트는 이러한 필드가 클라이언트가 실행 중인 시스템의 코드 페이지에 있다고 가정합니다. XMS 클라이언트가 다른 코드 페이지에 있는 이러한 필드를 제공하거나 수신할 것으로 예상되는 경우에는 IBM MQ 클라이언트에 알리도록 이 특성을 설정해야 합니다.

IBM MQ 클라이언트가 문자 데이터의 해당 필드를 큐 관리자에 전달하는 경우 필요하면 큐 관리자가 사용하는 코드 페이지로 필드 내의 데이터를 변환해야 합니다. 이와 유사하게 IBM MQ 클라이언트가 큐 관리자로부터 해당 필드를 수신하는 경우 필요하면 XMS 클라이언트가 데이터를 수신할 것으로 예상하는 코드 페이지로 필드 내의 데이터를 변환해야 합니다. IBM MQ 클라이언트는 이 특성을 사용하여 이러한 데이터 변환을 수행합니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성을 설정하는 것은 기본 IBM MQ 클라이언트 애플리케이션을 지원하는 IBM MQ 클라이언트에 대해 MQCCSID 환경 변수를 설정하는 것과 동일합니다. 이 환경 변수에 대한 자세한 정보는 [MQCCSID](#)를 참조하십시오.

### ***XMSC\_WMQ\_QUEUE\_MANAGER***

**데이터 유형:**

문자열

**특성:**

ConnectionFactory

**적용 가능한 오브젝트:**

JMS 관리 도구 긴 이름: QMANAGER

JMS 관리 도구 짧은 이름: QMGR

연결된 큐 관리자의 이름.

기본적으로 특성이 설정되어 있지 않습니다.

### ***XMSC\_WMQ\_RECEIVE\_CCSID***

큐 관리자 메시지 변환용 대상 CCSID를 설정하는 목적지 특성입니다. XMSC\_WMQ\_RECEIVE\_CONVERSION을 WMQ\_RECEIVE\_CONVERSION\_QMGR로 설정하지 않은 경우, 값이 무시됩니다.

**데이터 유형:**

정수

**값:**

임의의 양의 정수입니다.

기본값은 1208입니다.

메시지에 GMO\_CONVERT 값을 지정하는 것은 선택사항입니다. GMO\_CONVERT 값을 지정한 경우 지정된 값에 따라 변환이 수행됩니다.

### ***XMSC\_WMQ\_RECEIVE\_CONVERSION***

큐 관리자가 데이터 변환을 수행할지 판별하는 목적지 특성입니다.

**데이터 유형:**

정수

**값:**

XMSC\_WMQ\_RECEIVE\_CONVERSION\_CLIENT\_MSG(기본값): XMS 클라이언트에서만 데이터 변환을 수행합니다. 변환은 항상 코드 페이지 1208을 사용하여 수행됩니다.

XMSC\_WMQ\_RECEIVE\_CONVERSION\_QMGR: XMS 클라이언트로 메시지를 보내기 전에 큐 관리자에서 데이터 변환을 수행합니다.

### ***XMSC\_WMQ\_RECEIVE\_EXIT***

**데이터 유형:**

문자열

**특성:**

ConnectionFactory

실행할 채널 수신 종료를 식별합니다.

이 특성의 값은 채널 수신 엑시트를 식별하는 문자열입니다. 형식은 다음과 같습니다.

**libraryName**(entryPointName)

여기서,

- **libraryName**은 관리 엑시트 .dll의 전체 경로입니다.



- `entryPointName`은 네임스페이스에 의해 규정된 클래스 이름입니다.

예: `C:\MyReceiveExit.dll(MyReceiveExitNameSpace.MyReceiveExitClassName)`

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 관리 클라이언트 모드의 큐 관리자에 연결되는 경우에만 관련됩니다. 또한 관리 엑시트만 지원됩니다.

### ***XMSC\_WMQ\_RECEIVE\_EXIT\_INIT***

데이터 유형:

문자열

특성:

ConnectionFactory

호출 시 채널 수신 종료에 전달되는 사용자 데이터입니다.

이 특성의 값은 문자열입니다. 기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 관리 클라이언트 모드의 큐 관리자에 연결되고 [1988 페이지의 『XMSC\\_WMQ\\_RECEIVE\\_EXIT』](#) 특성이 설정된 경우에만 관련됩니다.

### ***XMSC\_WMQ\_RESOLVED\_QUEUE\_MANAGER***

데이터 유형:

문자열

특성:

ConnectionFactory

이 특성을 사용하여 연결된 큐 관리자의 이름을 얻을 수 있습니다.

CCDT(클라이언트 채널 정의 표)에서 사용하는 경우 이 이름은 연결 팩토리에 지정된 큐 관리자 이름과 다를 수 있습니다.

### ***XMSC\_WMQ\_RESOLVED\_QUEUE\_MANAGER\_ID***

데이터 유형:

문자열

특성:

ConnectionFactory

이 특성은 연결 후에 큐 관리자의 ID로 채워집니다.

### ***XMSC\_WMQ\_SECURITY\_EXIT***

데이터 유형:

문자열

특성:

ConnectionFactory

채널 보안 엑시트를 식별합니다.

이 특성의 값은 채널 보안 엑시트를 식별하는 문자열입니다. 형식은 다음과 같습니다.

**libraryName**(entryPointName)

여기서,

- **libraryName**은 관리 엑시트 .dll의 전체 경로입니다.
- `entryPointName`은 네임스페이스에 의해 규정된 클래스 이름입니다.

예: `C:\MySecurityExit.dll(MySecurityExitNameSpace.MySecurityExitClassName)`

문자열의 최대 길이는 128자입니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 관리 클라이언트 모드의 큐 관리자에 연결되는 경우에만 관련됩니다. 또한 관리 엑시트만 지원됩니다.

### ***XMSC\_WMQ\_SECURITY\_EXIT\_INIT***

데이터 유형:

문자열

특성:

ConnectionFactory

호출 시 채널 보안 엑시트로 전달되는 사용자 데이터입니다.

사용자 데이터 문자열의 최대 길이는 32자입니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 관리 클라이언트 모드의 큐 관리자에 연결되고 [1989 페이지의 『XMSC WMQ SECURITY EXIT』](#) 특성이 설정된 경우에만 관련됩니다.

### ***XMSC\_WMQ\_SEND\_EXIT***

데이터 유형:

문자열

특성:

ConnectionFactory

채널 전송 종료를 식별합니다.

이 특성의 값은 문자열입니다. 채널 송신 엑시트의 형식은 다음과 같습니다.

**libraryName**(entryPointName)

여기서,

- **libraryName**은 관리 엑시트 .dll의 전체 경로입니다.
- **entryPointName**은 네임스페이스에 의해 규정된 클래스 이름입니다.

예: C:\MySendExit.dll(MySendExitNameSpace.MySendExitClassName)

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 애플리케이션이 관리 클라이언트 모드의 큐 관리자에 연결되는 경우에만 관련됩니다. 또한 관리 엑시트만 지원됩니다.

### ***XMSC\_WMQ\_SEND\_EXIT\_INIT***

데이터 유형:

문자열

특성:

ConnectionFactory

호출 시 채널 송신 엑시트로 전달되는 사용자 데이터입니다.

이 특성 값은 쉼표로 구분된 하나 이상의 사용자 데이터 항목의 문자열입니다. 기본적으로 특성이 설정되어 있지 않습니다.

채널 송신 엑시트의 순서에 전달되는 사용자 데이터를 지정하는 규칙은 채널 수신 엑시트의 순서에 전달되는 사용자 데이터를 지정하는 규칙과 동일합니다. 따라서 해당 규칙에 대해서는 [1989 페이지의 『XMSC WMQ RECEIVE\\_EXIT\\_INIT』](#)의 내용을 참조하십시오.

이 특성은 애플리케이션이 관리 클라이언트 모드의 큐 관리자에 연결되고 [1990 페이지의 『XMSC WMQ\\_SEND\\_EXIT』](#) 특성이 설정된 경우에만 관련됩니다.

## **XMSC\_WMQ\_SEND\_CHECK\_COUNT**

데이터 유형:

System.Int32

특성:

ConnectionFactory

변환되지 않은 하나의 XMS 세션에서 비동기 put 오류를 검사하는 사이에 허용할 수 있는 송신 호출 수입니다.

기본적으로 이 특성은 0으로 설정됩니다.

## **XMSC\_WMQ\_SHARE\_CONV\_ALLOWED**

데이터 유형:

System.Int32

특성:

ConnectionFactory

적용 가능한 오브젝트:

JMS 관리 도구 긴 이름: SHARECONVALLOWED

JMS 관리 도구 짧은 이름: SCALD

클라이언트 연결이 다른 최상위 레벨과 소켓을 공유할 수 있는지 여부 XMS 채널 정의가 일치하는 경우 동일한 프로세스에서 동일한 큐 관리자로 연결됩니다. 이 특성은 애플리케이션 개발 또는 유지보수 또는 경영 목적을 위해 필요한 경우 별도의 소켓에서 연결의 완전한 격리를 허용하기 위해 제공됩니다. 이 특성을 설정하면 XMS에 기본 소켓을 공유하도록 표시합니다. 이는 단일 소켓을 공유하는 연결 수를 표시하지는 않습니다. 소켓을 공유하는 연결의 수는 IBM MQ 클라이언트와 IBM MQ 서버 간에 협상되는 SHARECNV 값에 의해 판별됩니다.

애플리케이션은 다음과 같은 이름 지정된 상수로 이 특성을 설정할 수 있습니다.

- XMSC\_WMQ\_SHARE\_CONV\_ALLOWED\_FALSE - 연결이 소켓을 공유하지 않습니다.
- XMSC\_WMQ\_SHARE\_CONV\_ALLOWED\_TRUE - 연결이 소켓을 공유합니다.

기본적으로 이 특성은 XMSC\_WMQ\_SHARE\_CONV\_ALLOWED\_ENABLED로 설정됩니다.

이 특성은 애플리케이션이 클라이언트 모드의 큐 관리자에 연결되는 경우에만 관련됩니다.

## **XMSC\_WMQ\_SSL\_CERT\_STORES**

데이터 유형:

문자열

특성:

ConnectionFactory

큐 관리자에 대한 SSL(Secure Socket Layer) 연결에 사용되는 인증서 폐기 목록(CRL)이 있는 서버의 위치입니다.

이 특성 값은 쉼표로 구분된 하나 이상의 URL 목록입니다. 각 URL의 형식은 다음과 같습니다.

```
[user[/password]@]ldap://[serveraddress][:portnum][, ...]
```

이 형식은 호환 가능하지만 기본 MQJMS 형식에서 확장됩니다.

serveraddress를 빈 상태로 둘 수 있습니다. 이 경우, XMS는 값이 문자열 "localhost"라고 가정합니다.

목록의 예는 다음과 같습니다.

```
myuser/mypassword@ldap://server1.mycom.com:389
ldap://server1.mycom.com
ldap://
ldap://:389
```

.NET 전용: IBM MQ (WMQ\_CM\_CLIENT) 에 대한 관리 연결 및 IBM MQ (WMQ\_CM\_CLIENT\_UNMANAGED) 에 대한 비관리 연결은 둘 다 TLS/SSL 연결을 지원합니다.

기본적으로 특성이 설정되어 있지 않습니다.

**관련 개념**

[비관리 .NET 클라이언트에 대한 SSL 및 TLS 지원](#)

[관리 .NET 클라이언트에 대한 SSL 및 TLS 지원](#)

***XMSC\_WMQ\_SSL\_CIPHER\_SPEC***

**데이터 유형:**

문자열

**특성:**

ConnectionFactory

큐 관리자에 대한 보안 연결에 사용할 CipherSpec의 이름.

IBM MQ TLS 지원과 함께 사용할 수 있는 암호 스펙이 다음 표에 나열됩니다. 개인 인증서를 요청할 때 공용 및 개인 키 쌍의 키 크기를 지정합니다. SSL 데이터 교환 동안에 사용되는 키 크기는 인증서에 저장된 크기입니다(표에 명시된 것처럼 CipherSpec으로 판별되는 경우는 제외). 기본적으로 이 특성은 설정되어 있지 않습니다.

CipherSpec 이름	사용되는 프로토콜	해시 알고리즘	암호화 알고리즘	암호화 비트	FIPS <sup>1</sup>	스위트 B 128 비트	스위트 B 192비트
TLS_RSA_WITH_AES_128_CBC_SHA	TLS 1.0	SHA-1	AES	128	예	아니오	아니오
TLS_RSA_WITH_AES_256_CBC_SHA <sup>2</sup>	TLS 1.0	SHA-1	AES	256	예	아니오	아니오
TLS_RSA_WITH_DES_CBC_SHA	TLS 1.0	SHA-1	DES	56	아니오	아니오	아니오
TLS_RSA_WITH_3DES_EDE_CBC_SHA <sup>4</sup>	TLS 1.0	SHA-1	3DES	168	예	아니오	아니오
TLS_RSA_WITH_AES_128_GCM_SHA256	TLS 1.2	SHA-256	AES	128	예	아니오	아니오
TLS_RSA_WITH_AES_256_GCM_SHA384	TLS 1.2	SHA-384	AES	256	예	아니오	아니오
TLS_RSA_WITH_AES_128_CBC_SHA256	TLS 1.2	SHA-256	AES	128	예	아니오	아니오
TLS_RSA_WITH_AES_256_CBC_SHA256	TLS 1.2	SHA-256	AES	256	예	아니오	아니오
ECDHE_ECDSA_RC4_128_SHA256	TLS 1.2	SHA-256	RC4	128	아니오	아니오	아니오
ECDHE_ECDSA_3DES_EDE_CBC_SHA256	TLS 1.2	SHA-256	3DES	168	예	아니오	아니오
ECDHE_RSA_RC4_128_SHA256	TLS 1.2	SHA-256	RC4	128	아니오	아니오	아니오
ECDHE_RSA_3DES_EDE_CBC_SHA256	TLS 1.2	SHA-256	3DES	168	예	아니오	아니오
ECDHE_ECDSA_AES_128_CBC_SHA256	TLS 1.2	SHA-256	AES	128	예	아니오	아니오
ECDHE_ECDSA_AES_256_CBC_SHA384	TLS 1.2	SHA-384	AES	256	예	아니오	아니오

CipherSpec 이름	사용되는 프로토콜	해시 알 고리즘	암호화 알고리즘	암호화 비트	FIPS <sup>1</sup>	스위트 B 128 비트	스위트 B 192비트
ECDHE_RSA_AES_128_CBC_SHA256	TLS 1.2	SHA-256	AES	128	예	아니오	아니오
ECDHE_RSA_AES_256_CBC_SHA384	TLS 1.2	SHA-384	AES	256	예	아니오	아니오
ECDHE_ECDSA_AES_128_GCM_SHA256	TLS 1.2	SHA-256	AES	128	예	예	아니오
ECDHE_ECDSA_AES_256_GCM_SHA384	TLS 1.2	SHA-384	AES	256	예	아니오	예
ECDHE_RSA_AES_128_GCM_SHA256	TLS 1.2	SHA-256	AES	128	예	아니오	아니오
ECDHE_RSA_AES_256_GCM_SHA384	TLS 1.2	SHA-384	AES	256	예	아니오	아니오
TLS_RSA_WITH_NULL_SHA256	TLS 1.2	SHA-256	없음	0	아니오	아니오	아니오
ECDHE_RSA_NULL_SHA256	TLS 1.2	SHA-256	없음	0	아니오	아니오	아니오
ECDHE_ECDSA_NULL_SHA256	TLS 1.2	SHA-256	없음	0	아니오	아니오	아니오
TLS_RSA_WITH_NULL_NULL	TLS 1.2	없음	없음	0	아니오	아니오	아니오
TLS_RSA_WITH_RC4_128_SHA256	TLS 1.2	SHA-256	RC4	128	아니오	아니오	아니오

**참고:**

1. CipherSpec이 FIPS(Federal Information Processing Standard) 140-2를 준수하는지 여부를 지정합니다. FIPS에 대한 설명과 FIPS 140-2 준수 조작을 위해 IBM MQ를 구성하는 방법에 대해서는 [FIPS\(Federal Information Processing Standard\)](#)를 참조하십시오.
2. 이 CipherSpec은 적절한 제한되지 않은 정책 파일이 IBM MQ Explorer에서 사용되는 JRE에 적용되지 않으면 IBM MQ Explorer에서 큐 관리자로의 연결을 보안하는 데 사용할 수 없습니다.
3. 이 CipherSpec은 2007년 5월 19일 이전에 FIPS 140-2 인증되었습니다.
4. FIPS 140-2 준수 조작을 위해 IBM MQ를 구성한 경우 AMQ9288 오류로 인해 연결이 종료되기 전까지는 이 CipherSpec을 사용하여 최대 32GB의 데이터를 전송할 수 있습니다. 이 오류를 방지하려면 3중 DES(더 이상 사용되지 않음)를 사용하지 않거나, FIPS 140-2 구성에서 이 CipherSpec을 사용할 때 비밀 키 재설정을 사용으로 설정하십시오.

**관련 개념**

[메시지의 데이터 무결성](#)

**관련 태스크**

[보안 설정](#)

[CipherSpec 지정](#)

***XMSC\_WMQ\_SSL\_CIPHER\_SUITE***

**데이터 유형:**

문자열

**특성:**

ConnectionFactory

큐 관리자에 대한 TLS 연결에 사용될 CipherSuite의 이름입니다. 보안 연결 협상에 사용되는 프로토콜은 지정된 CipherSuite에 따라 달라집니다.

이 특성의 표준 값은 다음과 같습니다.

- SSL\_RSA\_WITH\_DES\_CBC\_SHA
- SSL\_RSA\_EXPORT1024\_WITH\_DES\_CBC\_SHA
- SSL\_RSA\_EXPORT1024\_WITH\_RC4\_56\_SHA
- SSL\_RSA\_EXPORT\_WITH\_RC4\_40\_MD5
- SSL\_RSA\_WITH\_RC4\_128\_MD5
- SSL\_RSA\_WITH\_RC4\_128\_SHA
- SSL\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA
- SSL\_RSA\_WITH\_AES\_128\_CBC\_SHA
- SSL\_RSA\_WITH\_AES\_256\_CBC\_SHA
- SSL\_RSA\_WITH\_DES\_CBC\_SHA
- SSL\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA

이 값은 `XMSC_WMQ_SSL_CIPHER_SPEC` 대신 제공될 수 있습니다.

`XMSC_WMQ_SSL_CIPHER_SPEC`에 비어 있지 않은 값이 지정되는 경우 이 값이 `XMSC_WMQ_SSL_CIPHER_SUITE`에 대한 설정을 대체합니다. `XMSC_WMQ_SSL_CIPHER_SPEC`에 값이 없는 경우 `XMSC_WMQ_SSL_CIPHER_SUITE`값이 IBM Global Security Kit (GSKit)에 제공될 암호 스위트로 사용됩니다. 이 경우 값은 IBM MQ 큐 관리자에 대한 XMS 연결의 CipherSuite 및 CipherSpec 이름 매핑에 설명된 대로 동등한 CipherSpec 값에 매핑됩니다.

`XMSC_WMQ_SSL_CIPHER_SPEC` 및 `XMSC_WMQ_SSL_CIPHER_SUITE`가 모두 비어 있는 경우 `pChDef->SSLCipherSpec` 필드가 공백으로 채워집니다.

.NET 전용: IBM MQ (WMQ\_CM\_CLIENT)에 대한 관리 연결 및 IBM MQ (WMQ\_CM\_CLIENT\_UNMANAGED)에 대한 비관리 연결은 둘 다 TLS/SSL 연결을 지원합니다.

기본적으로 특성이 설정되어 있지 않습니다.

### 관련 개념

[비관리 .NET 클라이언트에 대한 SSL 및 TLS 지원](#)

[관리 .NET 클라이언트에 대한 SSL 및 TLS 지원](#)

## ***XMSC\_WMQ\_SSL\_CRYPTO\_HW***

데이터 유형:

문자열

특성:

ConnectionFactory

클라이언트 시스템에 연결된 암호화 하드웨어의 구성 세부사항입니다.

이 특성의 표준 값은 다음과 같습니다.

- GSK\_ACCELERATOR\_RAINBOW\_CS\_OFF
- GSK\_ACCELERATOR\_RAINBOW\_CS\_ON
- GSK\_ACCELERATOR\_NCIPHER\_NF\_OFF
- GSK\_ACCELERATOR\_NCIPHER\_NF\_ON

PKCS11 암호화 하드웨어에 대한 특수한 형식이 있습니다(DriverPath, TokenLabel 및 TokenPassword가 사용자 지정 문자열임).

```
GSK_PKCS11=PKCS#11 DriverPath; PKCS#11 TokenLabel;PKCS#11 TokenPassword
```

XMS가 문자열 콘텐츠를 해석하거나 변경하지 않습니다. 최대 256개의 1바이트 문자로 제공된 값을 MQSCO.CryptoHardware 필드에 복사합니다.

.NET 전용: IBM MQ (WMQ\_CM\_CLIENT) 에 대한 관리 연결 및 IBM MQ (WMQ\_CM\_CLIENT\_UNMANAGED) 에 대한 비관리 연결은 둘 다 TLS/SSL 연결을 지원합니다.

기본적으로 특성이 설정되어 있지 않습니다.

#### 관련 개념

[비관리 .NET 클라이언트에 대한 SSL 및 TLS 지원](#)

[관리 .NET 클라이언트에 대한 SSL 및 TLS 지원](#)

### ***XMSC\_WMQ\_SSL\_FIPS\_REQUIRED***

데이터 유형:

부울

특성:

ConnectionFactory

이 특성의 값은 애플리케이션이 비FIPS 준수 암호 스위트를 사용할 수 있는지 여부를 판별합니다. 이 특성이 true 로 설정되는 경우 FIPS 알고리즘만 클라이언트-서버 연결에 사용됩니다.

이 특성 값은 다음과 같으며 MQSCO.FipsRequired의 두 표준 값으로 변환됩니다.

표 880. MQSCO.FlipsRequired 특성 값 표		
값	설명	해당하는 MQSCO.FipsRequired 값
false	모든 CipherSpec을 사용할 수 있습니다.	MQSSL_FIPS_NO(기본값)
true	이 클라이언트 연결에 적용되는 CipherSpec에서는 FIPS 인증 암호화 알고리즘만 사용할 수 있습니다.	MQSSL_FIPS_YES

XMS는 MQCONNX 호출 전에 해당 값을 MQSCO.FipsRequired에 복사합니다.

.NET 전용: IBM MQ (WMQ\_CM\_CLIENT) 에 대한 관리 연결 및 IBM MQ (WMQ\_CM\_CLIENT\_UNMANAGED) 에 대한 비관리 연결은 둘 다 TLS/SSL 연결을 지원합니다.

#### 관련 개념

[비관리 .NET 클라이언트에 대한 SSL 및 TLS 지원](#)

[관리 .NET 클라이언트에 대한 SSL 및 TLS 지원](#)

### ***XMSC\_WMQ\_SSL\_KEY\_REPOSITORY***

데이터 유형:

문자열

특성:

ConnectionFactory

키 및 인증서가 저장되는 키 데이터베이스 파일의 위치입니다.

XMS는 최대 256개의 1바이트 문자로 된 문자열을 MQSCO.KeyRepository 필드에 복사합니다. IBM MQ는 전체 경로를 포함하여 파일 이름으로 이 문자열을 해석합니다.

.NET 전용: IBM MQ (WMQ\_CM\_CLIENT) 에 대한 관리 연결 및 IBM MQ (WMQ\_CM\_CLIENT\_UNMANAGED) 에 대한 비관리 연결은 둘 다 TLS/SSL 연결을 지원합니다.

기본적으로 특성이 설정되어 있지 않습니다.

#### 관련 개념

[비관리 .NET 클라이언트에 대한 SSL 및 TLS 지원](#)

[관리 .NET 클라이언트에 대한 SSL 및 TLS 지원](#)

## ***XMSC\_WMQ\_SSL\_KEY\_RESETCOUNT***

### **데이터 유형:**

System.Int32

### **특성:**

ConnectionFactory

KeyResetCount는 비밀 키가 재협상되기 전에 SSL 통신에서 송신되고 수신된 암호화되지 않은 총 바이트 수를 나타냅니다. 바이트 수에는 MCA에서 보낸 제어 정보가 포함됩니다.

XMS는 MQCONNX 호출 전 사용자가 이 특성에 대해 제공한 값을 MQSCO.KeyResetCount로 복사합니다.

MQSCO.KeyRestCount 매개변수는 IBM MQ 버전 6에서만 사용할 수 있습니다. IBM MQ 버전 5.3인 경우 이 특성이 설정되면 XMS에서 큐 관리자에 대한 연결을 시도하지 않으며 대신 적절하게 예외 처리합니다.

.NET 전용: IBM MQ (WMQ\_CM\_CLIENT) 에 대한 관리 연결 및 IBM MQ (WMQ\_CM\_CLIENT\_UNMANAGED) 에 대한 비관리 연결은 둘 다 TLS/SSL 연결을 지원합니다.

이 특성의 기본값은 0입니다. 즉, 시크릿 키가 재조정되지 않습니다.

### **관련 개념**

[비관리 .NET 클라이언트에 대한 SSL 및 TLS 지원](#)

[관리 .NET 클라이언트에 대한 SSL 및 TLS 지원](#)

## ***XMSC\_WMQ\_SSL\_PEER\_NAME***

### **데이터 유형:**

문자열

### **특성:**

ConnectionFactory

큐 관리자에 대한 SSL(Secure Socket Layer) 연결에 사용되는 피어 이름입니다.

이 특성은 표준 값 목록이 없습니다. 대신 SSLPEER에 대한 규칙에 따라 이 문자열을 빌드해야 합니다.

다음은 피어 이름의 예입니다.

```
"CN=John Smith, O=IBM ,OU=Test , C=GB"
```

XMS는 문자열을 올바른 1바이트 코드 페이지로 복사하고 MQCONNX를 호출하기 전에 올바른 값을 MQCD.SSLPeerNamePtr 및 MQCD.SSLPeerNameLength에 배치합니다.

이 특성은 애플리케이션이 클라이언트 모드의 큐 관리자에 연결되는 경우에만 관련됩니다.

.NET 전용: IBM MQ (WMQ\_CM\_CLIENT) 에 대한 관리 연결 및 IBM MQ (WMQ\_CM\_CLIENT\_UNMANAGED) 에 대한 비관리 연결은 둘 다 TLS/SSL 연결을 지원합니다.

기본적으로 특성이 설정되어 있지 않습니다.

### **관련 개념**

[비관리 .NET 클라이언트에 대한 SSL 및 TLS 지원](#)

[관리 .NET 클라이언트에 대한 SSL 및 TLS 지원](#)

### **관련 참조**

[SSLPEERNAME](#)

## ***XMSC\_WMQ\_SYNCPOINT\_ALL\_GETS***

### **데이터 유형:**

System.Boolean

### **특성:**

ConnectionFactory

동기점 제어에 있는 큐에서 모든 메시지를 검색해야 하는지 여부를 결정합니다.



특성의 올바른 값은 다음과 같습니다.

올바른 값	의미
false	상황이 적절한 경우 XMS 클라이언트는 동기점 제어 외부에 있는 큐에서 메시지를 검색할 수 있습니다.
true	XMS 클라이언트가 동기점 제어 내의 큐에서 모든 메시지를 검색해야 합니다.

기본값은 false입니다.

## ***XMSC\_WMQ\_TARGET\_CLIENT***

데이터 유형:

System.Int32

특성:

목적지

URI에서 사용되는 이름:

targetClient

목적지에 송신되는 메시지에 MQRFH2 헤더를 포함할지 여부를 지정합니다.

애플리케이션이 MQRFH2 헤더를 포함하는 메시지를 전송하는 경우 수신하는 애플리케이션이 헤더를 처리할 수 있어야 합니다.

특성의 올바른 값은 다음과 같습니다.

올바른 값	의미
XMSC_WMQ_TARGET_DEST_JMS	목적지에 전송되는 메시지가 MQRFH2 헤더를 포함합니다. 애플리케이션이 다른 XMS 애플리케이션, IBM MQ classes for JMS 애플리케이션 또는 MQRFH2 헤더를 처리하도록 설계된 고유 IBM MQ 애플리케이션으로 메시지를 전송하는 경우 이 값을 지정하십시오.
XMSC_WMQ_TARGET_DEST_MQ	목적지에 전송되는 메시지가 MQRFH2 헤더를 포함하지 않습니다. 애플리케이션이 MQRFH2 헤더를 처리하도록 설계되지 않은 고유 IBM MQ 애플리케이션으로 메시지를 전송하는 경우 이 값을 지정하십시오.

기본값은 XMSC\_WMQ\_TARGET\_DEST\_JMS입니다.

## ***XMSC\_WMQ\_TEMP\_Q\_PREFIX***

데이터 유형:

문자열

특성:

ConnectionFactory

이름을 형성하는 데 사용되는 접두사 IBM MQ 애플리케이션이 생성될 때 생성되는 동적 큐 XMS 임시 대기열.

접두부를 형성하는 규칙은 오브젝트 디스크립터에서 **DynamicQName** 필드의 콘텐츠를 형성하는 규칙과 동일하지만 마지막 비공백 문자는 별표 (\*) 여야 합니다. 기본적으로 이 특성은 설정되지 않습니다. 설정되지 않은 경우 다음 값이 사용됩니다.

- ▶ **Multi** AMQ.\* Multiplatforms
- ▶ **z/OS** z/OS에서 CSQ.\*

이 특성은 포인트-투-포인트 도메인에만 관련됩니다.

## ***XMSC\_WMQ\_TEMP\_TOPIC\_PREFIX***

데이터 유형:

문자열

**특성:**

ConnectionFactory, Destination

임시 주제를 만들 때, XMS " 형식의 토픽 문자열을 생성합니다. TEMP/TEMPTOPICPREFIX/unique\_id " 또는 이 속성에 기본값이 포함된 경우 이 문자열은 "TEMP/unique\_id "가 생성됩니다. 비어 있지 않은 값을 지정하면 이 연결 아래에 작성된 임시 토픽에 대한 구독자의 관리 큐를 작성하기 위한 특정 모델 큐를 정의할 수 있습니다.

이 특성의 올바른 값은 IBM MQ 토픽 문자열에 유효한 문자만으로 구성된 널이 아닌 문자열입니다.

기본적으로 이 특성은 ""(빈 문자열)로 설정됩니다.

**참고:** 이 특성은 발행/구독 도메인에만 관련됩니다.

***XMSC\_WMQ\_TEMPORARY\_MODEL*****데이터 유형:**

문자열

**특성:**

ConnectionFactory

이름은 IBM MQ 애플리케이션이 생성될 때 동적 큐가 생성되는 모델 큐 XMS 임시 대기열.

이 특성의 기본값은 SYSTEM.DEFAULT.MODEL.QUEUE입니다.

이 특성은 포인트-투-포인트 도메인에만 관련됩니다.

***XMSC\_WMQ\_WILDCARD\_FORMAT*****데이터 유형:**

System.Int32

**특성:**

ConnectionFactory, Destination

이 특성은 사용되는 와일드카드 구문 버전을 판별합니다.

IBM MQ '\*' 및 '?' 와 함께 발행/구독을 사용하는 경우 와일드카드로 처리합니다. 반면 '#' 및 '+'는 IBM Integration Bus에서 발행/구독을 사용할 때 와일드카드로 처리됩니다. 이 특성은 XMSC\_WMQ\_BROKER\_VERSION 특성을 대체합니다.

이 특성의 올바른 값은 다음과 같습니다.

***XMSC\_WMQ\_WILDCARD\_TOPIC\_ONLY***

Recognizes the topic level wildcards only i.e. '#' 및 '+' 는 와일드카드로 처리됩니다. 이 값은 XMSC\_WMQ\_BROKER\_V2와 동일합니다.

***XMSC\_WMQ\_WILDCARD\_CHAR\_ONLY***

Recognizes the character wildcards only i.e. '\*' 및 '?' 와일드카드로 처리합니다. 이 값은 XMSC\_WMQ\_BROKER\_V1과 동일합니다.

기본적으로 이 특성은 XMSC\_WMQ\_WILDCARD\_TOPIC\_ONLY로 설정됩니다.

***XMSC\_WPM\_BUS\_NAME*****데이터 유형:**

문자열

**특성:**

ConnectionFactory 및 Destination

**URI에서 사용되는 이름:**

busName

연결 팩토리의 경우, 애플리케이션이 연결하는 서비스 통합 버스의 이름이거나, 또는 목적지의 경우에는 목적지가 존재하는 서비스 통합 버스의 이름입니다.

토픽인 목적지의 경우 이 특성은 연관된 토픽 공간이 존재하는 서비스 통합 버스의 이름입니다. 이 토픽 공간은 `XMSC_WPM_TOPIC_SPACE` 특성에 의해 지정됩니다.

특성에 목적지가 설정되지 않으면 큐 또는 연관된 토픽 공간이 애플리케이션이 연결하는 서비스 통합 버스에 존재한다고 가정합니다.

기본적으로 특성이 설정되어 있지 않습니다.

## ***XMSC\_WPM\_CONNECTION\_PROTOCOL***

**데이터 유형:**  
System.Int32

**특성:**  
연결

메시징 엔진에 연결하는 데 사용되는 통신 프로토콜입니다. 이 특성은 읽기 전용입니다.

이 특성의 가능한 값은 다음과 같습니다.

값	의미
XMSC_WPM_CP_HTTP	연결이 TCP/IP에서 HTTP를 사용합니다.
XMSC_WPM_CP_TCP	연결이 TCP/IP를 사용합니다.

## ***XMSC\_WPM\_CONNECTION\_PROXIMITY***

**데이터 유형:**  
System.Int32

**특성:**  
ConnectionFactory

연결에 대한 연결 근접성 설정입니다. 이 특성은 애플리케이션이 연결하는 메시징 엔진이 부트스트랩 서버에 얼마나 가까이 있어야 하는지 결정합니다.

특성의 올바른 값은 다음과 같습니다.

올바른 값	연결 근접성 설정
XMSC_WPM_CONNECTION_PROXIMITY_BUS	버스
XMSC_WPM_CONNECTION_PROXIMITY_CLUSTER	클러스터
XMSC_WPM_CONNECTION_PROXIMITY_HOST	호스트
XMSC_WPM_CONNECTION_PROXIMITY_SERVER	서버

기본값은 `XMSC_WPM_CONNECTION_PROXIMITY_BUS`입니다.

## ***XMSC\_WPM\_DUR\_SUB\_HOME***

**데이터 유형:**  
문자열

**특성:**  
ConnectionFactory

**URI에서 사용되는 이름:**  
durableSubscriptionHome

목적지나 연결에 대한 모든 지속 가능한 구독을 관리하는 메시징 엔진의 이름입니다. 지속 가능한 구독자에게 전달될 메시지는 동일한 메시징 엔진의 공개 위치에 저장됩니다.

연결에 대한 지속 가능 구독 홈을 지정해야 애플리케이션에서 연결을 사용하는 지속 가능 구독자를 작성할 수 있습니다. 목적지에 지정한 값은 연결에 지정한 값을 대체합니다.

기본적으로 특성이 설정되어 있지 않습니다.

이 특성은 발행/구독 도메인에만 관련됩니다.

### ***XMSC\_WPM\_HOST\_NAME***

데이터 유형:  
문자열

특성:  
연결

애플리케이션이 연결되는 메시징 엔진이 포함된 시스템의 호스트 이름 또는 IP 주소입니다. 이 특성은 읽기 전용입니다.

### ***XMSC\_WPM\_LOCAL\_ADDRESS***

데이터 유형:  
문자열

특성:  
ConnectionFactory

서비스 통합 버스에 대한 연결의 경우 이 특성은 사용할 로컬 네트워크 인터페이스를 지정하거나, 사용할 로컬 포트나 로컬 포트의 범위 또는 둘 다를 지정합니다.

이 특성의 값은 다음 형식의 문자열입니다.

`[host_name][(low_port)[,high_port]]`

변수의 의미는 다음과 같습니다.

#### ***host\_name***

연결에 사용하는 로컬 네트워크 인터페이스의 호스트 이름 또는 IP 주소입니다.

이 정보는 애플리케이션이 실행 중인 시스템에 두 개 이상의 네트워크 인터페이스가 있는 경우에만 필요하며 사용자가 연결에 사용되는 인터페이스를 지정할 수 있어야 합니다. 시스템에 네트워크 인터페이스가 하나뿐이면 해당 인터페이스만 사용할 수 있습니다. 시스템에 네트워크 인터페이스가 둘 이상이고 사용해야 할 인터페이스를 지정하지 않으면 인터페이스가 임의로 선택됩니다.

#### ***low\_port***

연결에 사용되는 로컬 포트의 번호입니다.

*high\_port*도 함께 지정하면 *low\_port*가 포트 번호 범위 중에서 가장 낮은 번호로 해석됩니다.

#### ***high\_port***

포트 번호 범위에서 가장 높은 포트 번호입니다. 지정된 범위의 포트 중 하나를 연결에 사용해야 합니다.

다음은 특성의 올바른 값에 대한 예입니다.

JUPITER  
9.20.4.98  
JUPITER(1000)  
9.20.4.98(1000,2000)  
(1000)  
(1000,2000)

기본적으로 특성이 설정되어 있지 않습니다.

### ***XMSC\_WPM\_ME\_NAME***

데이터 유형:  
문자열

특성:  
연결

애플리케이션이 연결되는 메시징 엔진의 이름입니다. 이 특성은 읽기 전용입니다.

## ***XMSC\_WPM\_NON\_PERSISTENT\_MAP***

**데이터 유형:**

System.Int32

**특성:**

ConnectionFactory

연결을 사용하여 송신되는 비지속적 메시지의 신뢰도 레벨입니다.

특성의 올바른 값은 다음과 같습니다.

### **올바른 값**

XMSC\_WPM\_MAPPING\_AS\_DESTINATION

XMSC\_WPM\_MAPPING\_BEST\_EFFORT\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_EXPRESS\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_RELIABLE\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_RELIABLE\_PERSISTENT

XMSC\_WPM\_MAPPING\_ASSURED\_PERSISTENT

기본값은 XMSC\_WPM\_MAPPING\_EXPRESS\_NON\_PERSISTENT입니다.

### **신뢰도 레벨**

서비스 통합 버스의 큐 또는 토픽 공간에 지정된 기본 신뢰도 레벨로 판별됨

최상의 비지속적 상태

명확한 비지속적 상태

신뢰 가능한 비지속적 상태

신뢰 가능한 지속적 상태

확실한 지속적 상태

## ***XMSC\_WPM\_PERSISTENT\_MAP***

**데이터 유형:**

System.Int32

**특성:**

ConnectionFactory

연결을 사용하여 송신되는 지속 메시지의 신뢰도 레벨입니다.

특성의 올바른 값은 다음과 같습니다.

### **올바른 값**

XMSC\_WPM\_MAPPING\_AS\_DESTINATION

XMSC\_WPM\_MAPPING\_BEST\_EFFORT\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_EXPRESS\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_RELIABLE\_NON\_PERSISTENT

XMSC\_WPM\_MAPPING\_RELIABLE\_PERSISTENT

XMSC\_WPM\_MAPPING\_ASSURED\_PERSISTENT

### **신뢰도 레벨**

서비스 통합 버스의 큐 또는 토픽 공간에 지정된 기본 신뢰도 레벨로 판별됨

최상의 비지속적 상태

명확한 비지속적 상태

신뢰 가능한 비지속적 상태

신뢰 가능한 지속적 상태

확실한 지속적 상태

기본값은 XMSC\_WPM\_MAPPING\_RELIABLE\_PERSISTENT입니다.

### **XMSC\_WPM\_PORT**

데이터 유형:  
System.Int32

특성:  
연결

애플리케이션이 연결되는 메시징 엔진이 대기하는 포트의 번호입니다. 이 특성은 읽기 전용입니다.

### **XMSC\_WPM\_PROVIDER\_ENDPOINTS**

데이터 유형:  
문자열

특성:  
ConnectionFactory

부트스트랩 서버에 사용되는 하나 이상의 엔드포인트 주소의 순서입니다. 엔드포인트 주소는 쉼표로 구분됩니다.

부트스트랩 서버는 애플리케이션이 연결되는 메시징 엔진을 선택하는 애플리케이션 서버입니다. 부트스트랩 서버의 엔드포인트 주소 형식은 다음과 같습니다.

*host\_name:port\_number:chain\_name*

엔드포인트 주소 구성요소의 의미는 다음과 같습니다.

#### **host\_name**

부트스트랩 서버가 상주하는 시스템의 호스트 이름 또는 IP 주소입니다. 호스트 이름 또는 IP 주소가 지정되지 않은 경우 기본값은 localhost입니다.

#### **port\_number**

부트스트랩 서버가 수신 요청을 청취하는 포트 번호입니다. 포트 번호가 지정되지 않은 경우 기본값은 7276입니다.

#### **chain\_name**

부트스트랩 서버에서 사용하는 부트스트랩 전송 체인의 이름입니다. 올바른 값은 다음과 같습니다.

올바른 값	부트스트랩 전송 체인의 이름
XMSC_WPM_BOOTSTRAP_HTTP	BootstrapTunneledMessaging
XMSC_WPM_BOOTSTRAP_HTTPS	BootstrapTunneledSecureMessaging
XMSC_WPM_BOOTSTRAP_SSL	BootstrapSecureMessaging
XMSC_WPM_BOOTSTRAP_TCP	BootstrapBasicMessaging

이름이 지정되지 않은 경우 기본값은 XMSC\_WPM\_BOOTSTRAP\_TCP입니다.

엔드포인트 주소가 지정되지 않은 경우 기본값은 localhost:7276:BootstrapBasicMessaging입니다.

### **XMSC\_WPM\_SSL\_CIPHER\_SUITE**

데이터 유형:  
문자열

특성:  
ConnectionFactory

이름은 CipherSuite TLS 연결에 사용됩니다. WebSphere Application Server service integration bus 메시징 엔진. 보안 연결 협상에 사용되는 프로토콜은 지정된 CipherSuite에 따라 달라집니다.

표 881. WebSphere Application Server service integration bus 메시징 엔진에 연결하는 데 필요한 CipherSuite 옵션	
암호화 스위트	사용되는 프로토콜
TLS_RSA_WITH_DES_CBC_SHA	TLSv1
TLS_RSA_WITH_3DES_EDE_CBC_SHA	TLSv1
TLS_RSA_WITH_AES_128_CBC_SHA	TLSv1
TLS_RSA_WITH_AES_256_CBC_SHA	TLSv1

**참고:**

1. **Windows** TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA 및 TLS\_RSA\_WITH\_AES\_256\_CBC\_SHA CipherSuites는 Windows에서만 지원됩니다 (이는 GSKit에 의해 좌우됩니다.)
2. **Deprecated** TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA는 더 이상 사용되지 않습니다. 그러나 AMQ9288 오류로 인해 연결이 종료되기 전까지는 이 CipherSpec을 사용하여 최대 32GB의 데이터를 전송할 수 있습니다. 이 오류를 방지하려면 3중 DES를 사용하지 않거나 이 CipherSpec을 사용할 때 비밀 키 재설정을 사용 가능하게 하십시오.

이 특성에는 기본값이 없습니다. SSL 또는 TLS를 사용하려면 이 특성에 대한 값을 지정해야 하고, 그렇지 않을 경우 애플리케이션이 서버에 연결할 수 없습니다.

***XMSC\_WPM\_SSL\_FIPS\_REQUIRED***

**참고:** AIX, Linux, and Windows에서 IBM MQ 는 IBM Crypto for C (ICC) 암호화 모듈을 통해 FIPS 140-2준수를 제공합니다. 이 모듈의 인증서가 히스토리 상태로 이동되었습니다. 고객은 [IBM Crypto for C \(ICC\) 인증서](#) 를 보고 NIST에서 제공하는 조언을 알고 있어야 합니다. 대체 FIPS 140-2모듈이 현재 진행 중이며 해당 상태는 [프로세스 목록의 NIST CMVP 모듈](#)에서 검색하여 볼 수 있습니다.

IBM MQ Operator 3.2.0 및 큐 관리자 컨테이너 이미지 9.4.0.0 이상은 UBI 9를 기반으로 합니다. FIPS 140-2 준수가 현재 보류 중이며 해당 상태는 [프로세스 목록의 NIST CMVP 모듈](#)에서 "Red Hat Enterprise Linux 9-OpenSSL FIPS 제공자" 를 검색하여 볼 수 있습니다.

**데이터 유형:**

부울

**특성:**

ConnectionFactory

이 특성의 값은 애플리케이션이 비FIPS 준수 암호 스위트를 사용할 수 있는지 또는 사용할 수 없는지 여부를 판별합니다. 이 특성이 true로 설정되는 경우 클라이언트-서버 연결에 FIPS 알고리즘만 사용됩니다. 이 특성의 값을 TRUE로 설정하면 애플리케이션이 비FIPS 준수 암호 스위트를 사용하지 않도록 방지합니다.

기본적으로 이 특성은 FALSE(즉, FIPS 모드 사용 안함)로 설정되어 있습니다.

***XMSC\_WPM\_SSL\_KEY\_REPOSITORY***

**데이터 유형:**

문자열

**특성:**

ConnectionFactory

보안 연결에 사용되는 공개 또는 개인 키가 포함되어 있는 키 링 파일의 파일 경로입니다.

키 링 파일 특성을 XMSC\_WPM\_SSL\_MS\_CERTIFICATE\_STORE의 특수 값으로 설정하면 Microsoft Windows 키 데이터베이스를 사용하도록 지정하는 것입니다. Microsoft Windows 키 데이터베이스([제어판 > 인터넷 옵션 > 콘텐츠 > 인증서](#)에 위치)를 사용하면 별도의 키 파일 데이터베이스에 대한 필요가 없어집니다. Windows x64 및 기타 플랫폼에서는 이 상수를 사용할 수 없습니다.

기본적으로 특성이 설정되어 있지 않습니다.

## ***XMSC\_WPM\_SSL\_KEYRING\_LABEL***

데이터 유형:

문자열

특성:

ConnectionFactory

서버 인증 시 사용되는 인증서입니다.값이 지정되지 않은 경우 기본 인증서가 사용됩니다.

기본적으로 특성이 설정되어 있지 않습니다.

## ***XMSC\_WPM\_SSL\_KEYRING\_PW***

데이터 유형:

문자열

특성:

ConnectionFactory

키 링 파일의 비밀번호입니다.

이 특성을 XMSC\_WPM\_SSL\_KEYRING\_STASH\_FILE 대신 사용하여 키 링 파일의 비밀번호를 구성할 수 있습니다.

기본적으로 특성이 설정되어 있지 않습니다.

## ***XMSC\_WPM\_SSL\_KEYRING\_STASH\_FILE***

데이터 유형:

문자열

특성:

ConnectionFactory

키 저장소 파일의 비밀번호가 있는 2진 파일의 이름입니다.

이 특성을 XMSC\_WPM\_SSL\_KEYRING\_PW 대신 사용하여 키 링 파일의 비밀번호를 구성할 수 있습니다.

기본적으로 특성이 설정되어 있지 않습니다.

## ***XMSC\_WPM\_TARGET\_GROUP***

데이터 유형:

문자열

특성:

ConnectionFactory

메시징 엔진의 대상 그룹 이름입니다. 대상 그룹의 네이치는 XMSC\_WPM\_TARGET\_TYPE 특성으로 판별됩니다.

서비스 통합 버스에서 메시징 엔진의 하위 그룹으로 메시징 엔진 검색을 제한하려면 이 특성을 설정하십시오. 애플리케이션이 서비스 통합 버스에서 메시징 엔진에 연결할 수 있도록 하려면 이 특성을 설정하지 마십시오.

기본적으로 특성이 설정되어 있지 않습니다.

## ***XMSC\_WPM\_TARGET\_SIGNIFICANCE***

데이터 유형:

System.Int32

특성:

ConnectionFactory

메시징 엔진의 대상 그룹에 대한 중요성입니다.

특성의 올바른 값은 다음과 같습니다.



### 올바른 값

XMSC\_WPM\_TARGET\_SIGNIFICANCE\_PREFERRED

XMSC\_WPM\_TARGET\_SIGNIFICANCE\_REQUIRED

### 의미

사용 가능한 경우 대상 그룹에서 메시징 엔진이 선택됩니다. 그렇지 않으면 동일한 서비스 통합 버스에서 대상 그룹 외부의 메시징 엔진이 선택됩니다.

선택된 메시징 엔진은 대상 그룹에 있어야 합니다. 대상 그룹의 메시징 엔진이 사용 가능하지 않으면 연결 프로세스가 실패합니다.

이 특성의 기본값은 XMSC\_WPM\_TARGET\_SIGNIFICANCE\_PREFERRED입니다.

## **XMSC\_WPM\_TARGET\_TRANSPORT\_CHAIN**

### 데이터 유형:

문자열

### 특성:

ConnectionFactory

애플리케이션이 메시징 엔진에 연결하는 데 사용해야 하는 인바운드 전송 체인의 이름입니다.

이 특성의 값은 메시징 엔진을 호스트하는 애플리케이션 서버에서 사용 가능한 인바운드 전송 체인의 이름일 수 있습니다. 다음 이름 지정된 상수가 사전정의된 인바운드 전송 체인 중 하나에 제공됩니다.

### 이름 지정된 상수

XMSC\_WPM\_TARGET\_TRANSPORT\_CHAIN\_BASIC

### 전송 체인의 이름

InboundBasicMessaging

이 특성의 기본값은 XMSC\_WPM\_TARGET\_TRANSPORT\_CHAIN\_BASIC입니다.

## **XMSC\_WPM\_TARGET\_TYPE**

### 데이터 유형:

System.Int32

### 특성:

ConnectionFactory

메시징 엔진의 대상 그룹 유형입니다. 이 특성은 XMSC\_WPM\_TARGET\_GROUP 특성으로 식별되는 대상 그룹의 네이처를 판별합니다.

특성의 올바른 값은 다음과 같습니다.

### 올바른 값

XMSC\_WPM\_TARGET\_TYPE\_BUSMEMBER

XMSC\_WPM\_TARGET\_TYPE\_CUSTOM

XMSC\_WPM\_TARGET\_TYPE\_ME

### 의미

대상 그룹의 이름은 버스 멤버의 이름입니다. 대상 그룹은 버스 멤버의 모든 메시징 엔진입니다.

대상 그룹의 이름은 메시징 엔진의 사용자 정의 그룹 이름입니다. 대상 그룹은 사용자 정의 그룹에 등록된 모든 메시징 엔진입니다.

대상 그룹의 이름은 메시징 엔진의 이름입니다. 대상 그룹은 지정된 메시징 엔진입니다.

기본적으로 특성이 설정되어 있지 않습니다.

## **XMSC\_WPM\_TEMP\_Q\_PREFIX**

### 데이터 유형:

문자열

### 특성:

ConnectionFactory

애플리케이션이 서비스 통합 버스를 생성할 때 생성되는 임시 대기열의 이름을 구성하는 데 사용되는 접두사입니다. XMS 임시 대기열 접두부는 최대 12자를 포함할 수 있습니다.

임시 큐의 이름은 "\_Q" 문자로 시작하고 그 뒤에 접두부가 옵니다. 이름의 나머지 부분은 시스템 생성 문자로 구성됩니다.

기본적으로 특성이 설정되어 있지 않습니다. 이는 임시 큐의 이름에 접두부가 없음을 의미합니다.

이 특성은 포인트-투-포인트 도메인에만 관련됩니다.

### **XMSC\_WPM\_TEMP\_TOPIC\_PREFIX**

**데이터 유형:**

문자열

**특성:**

ConnectionFactory

애플리케이션이 작성한 임시 토픽의 이름을 구성하는 데 사용되는 접두부입니다. 접두부는 최대 12자를 포함할 수 있습니다.

임시 토픽의 이름은 "\_T" 문자로 시작하고 그 뒤에 접두부가 옵니다. 이름의 나머지 부분은 시스템 생성 문자로 구성됩니다.

기본적으로 특성이 설정되어 있지 않습니다. 이는 임시 토픽의 이름에 접두부가 없음을 의미합니다.

이 특성은 발행/구독 도메인에만 관련됩니다.

### **XMSC\_WPM\_TOPIC\_SPACE**

**데이터 유형:**

문자열

**특성:**

목적지

**URI에서 사용되는 이름:**

topicSpace

토픽을 포함하는 토픽 영역의 이름입니다. 토픽인 대상에만 이 특성이 있을 수 있습니다.

기본적으로 특성이 설정되어 있지 않습니다. 이는 기본 토픽 공간이 가정됨을 의미합니다.

이 특성은 발행/구독 도메인에만 관련됩니다.

## **Managed File Transfer 애플리케이션 참조 개발**

Managed File Transfer용 애플리케이션을 개발하는 데 도움이 되는 참조 정보입니다.

### **fteCreateTransfer를 사용한 프로그램 시작 예**

**fteCreateTransfer** 명령을 사용하여 전송 전후에 실행할 프로그램을 지정할 수 있습니다.

**fteCreateTransfer**의 사용 외에 전송 전이나 후에 프로그램을 호출하는 다른 방법이 있습니다. 자세한 정보는 **MFT**와 함께 실행할 프로그램 지정을 참조하십시오.

이러한 모든 예에서는 다음 구문을 사용하여 프로그램을 지정합니다.

```
[type:]commandspec[, [retrycount][, [retrywait][, successrc]]]
```

이 구문에 대한 자세한 정보는 **fteCreateTransfer: 새 파일 전송 시작**을 참조하십시오.

## 실행 가능 프로그램 실행

다음 예에서는 mycommand라는 실행 가능 프로그램을 지정하고 두 개의 인수 a 및 b를 프로그램에 전달합니다.

```
mycommand(a,b)
```

전송 시작 전에 소스 에이전트 AGENT1에서 이 프로그램을 실행하려면 다음 명령을 사용하십시오.

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -presrc mycommand(a,b)
destinationSpecification sourceSpecification
```

## 실행 가능 프로그램 실행 및 재시도

다음 예에서는 인수를 사용하지 않는 simple이라는 실행 가능 프로그램을 지정합니다. retrycount에는 값 1이 지정되고 retrywait에는 값 5가 지정됩니다. 이러한 값은 완료 리턴 코드를 리턴하지 않으면 5분 대기한 후 프로그램을 다시 한 번 시도함을 의미합니다. successsrc에는 값이 지정되지 않으므로 기본값 0만 완료 리턴 코드입니다.

```
executable:simple,1,5
```

전송 완료 후에 소스 에이전트 AGENT1에서 이 프로그램을 실행하려면 다음 명령을 사용하십시오.

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -postsrc executable:simple,1,5
destinationSpecification sourceSpecification
```

## Ant 스크립트 실행 및 완료 리턴 코드 지정


다음 예제는 myscript 라는 Ant 스크립트를 지정하고 두 개의 특성을 스크립트에 전달합니다. 스크립트는 **fteAnt** 명령을 사용하여 실행됩니다. successsrc의 값은 >2&<7&!5|0|14로 지정되며, 리턴 코드 0, 3, 4, 6 및 14는 성공을 나타내도록 지정합니다.

```
antscript:myscript(prop1=fred,prop2=bob),,,>2&<7&!5|0|14
```

전송 시작 전에 목적지 에이전트 AGENT2에서 이 프로그램을 실행하려면 다음 명령을 사용하십시오.

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -predst
"antscript:myscript(prop1=fred,prop2=bob),,,>2&<7&!5|0|14"destinationSpecification sourceSpecification
```

## Ant 스크립트 실행 및 호출할 대상 지정

다음 예제는 script2 라는 Ant 스크립트와 호출할 두 개의 대상 target1 및 target2를 지정합니다. prop1 특성도 전달되며 값은 recmfm(F,B)입니다. 괄호, 쉼표 (,) 및 백슬래시 (\) 는 MFT 명령에서 특수 문자이며 백슬래시 (\) 문자로 이스케이프해야 합니다.  Windows의 파일 경로는 이중 백슬래시 (\\) 를 구분 기호로 사용하거나 단일 슬래시 (/) 를 사용하여 지정할 수 있습니다. 다음 예제에서 쉼표 (,) 및 괄호는 백슬래시 문자 (\) 를 사용하여 이스케이프됩니다.

```
antscript:script2(target1,target2,prop1=recmfm\F\B\),,,>2&<7&!5|0|14
```

전송 완료 후에 목적지 에이전트 AGENT2에서 이 프로그램을 실행하려면 다음 명령을 사용하십시오.

```
fteCreateTransfer -sa AGENT1 -da AGENT2
-postdst "antscript:script2(target1,target2,prop1=recmfm\F\B\),,,>2&<7&!5|0|14"
destinationSpecification sourceSpecification
```

## Ant 스크립트에서 메타데이터 사용

전송을 위한 다음 호출 중 하나로 Ant 태스크를 지정할 수 있습니다.

- 사전 소스

- 사후 소스
- 사전 목적지
- 사후 목적지

Ant 태스크가 실행될 때, 전송의 사용자 메타 데이터가 환경 변수를 사용하여 사용 가능하게 됩니다. 예를 들어 다음 코드를 사용하여 이 데이터에 액세스할 수 있습니다.

```
<property environment="environment" />
<echo>${environment.mymetadata}</echo>
```

여기서 mymetadata는 전송에 삽입되는 일부 메타 데이터의 이름입니다.

### JCL 스크립트 실행

다음 예제에서는 ZOSBATCH라는 JCL 스크립트를 지정합니다. retrycount에는 값 3, retrywait에는 값 30, successrc에는 값 0이 지정됩니다. 이러한 값은 리턴 코드 0을 리턴하지 않는 경우 시도 사이에 30초 대기한 후 스크립트를 재시도함을 의미합니다.

```
jcl:ZOSBATCH,3,30,0
```

여기서 ZOSBATCH는 MYSYS.JCL이라는 PDS 멤버이며, agent.properties 파일에는 commandPath=...:/'MYSYS.JCL':... 행이 있습니다.

전송 완료 후에 소스 에이전트 AGENT1에서 이 프로그램을 실행하려면 다음 명령을 사용하십시오.

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -postsrc jcl:ZOSBATCH,3,30,0
destinationSpecification sourceSpecification
```

### 관련 태스크

MFT와 함께 실행할 프로그램 지정

### 관련 참조

[fteCreateTransfer](#): 새 파일 전송 시작

## fteAnt: MFT 에서 Ant 태스크 실행

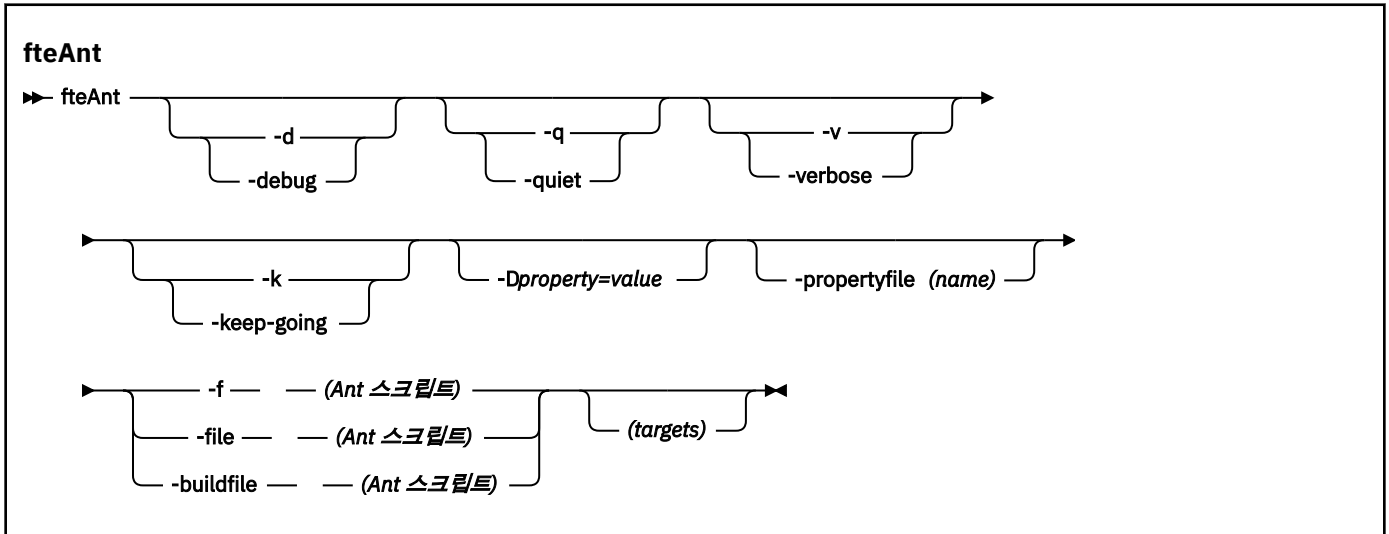
**fteAnt** 명령은 Managed File Transfer Ant 태스크가 사용 가능한 환경에서 Ant 스크립트를 실행합니다. 표준 **ant** 명령과는 달리 **fteAnt**를 실행하려면 스크립트 파일을 정의해야 합니다.

### MFT Ant 태스크 및 중첩 매개변수

Managed File Transfer에서는 파일 전송 기능에 액세스하기 위해 사용할 수 있는 Ant 태스크를 제공합니다. 또한 사용 가능한 중첩 매개변수 세트도 있습니다. 이 매개변수는 몇 가지 제공된 Ant 태스크에서 공통으로 사용되는 중첩 요소 세트에 대해 설명합니다.

**fteAnt** 명령 구문, 매개변수, 사용 예와 리턴 코드는 이 주제의 나머지 부분에서 설명되어 있습니다. MFT에서 제공하는 Ant 태스크 및 중첩 매개변수에 대한 자세한 정보는 하위 주제를 참조하십시오.

## fteAnt 구문



### 매개변수

#### **-debug** 또는 **-d**

선택사항. 디버깅 출력을 생성합니다.

#### **-quiet** 또는 **-q**

선택사항. 간단한 출력을 생성합니다.

#### **-verbose** 또는 **-v**

선택사항. 자세한 출력을 생성합니다.

#### **-keep-going** 또는 **-k**

선택사항. 실패한 대상에 의존하지 않는 모든 대상을 실행합니다.

#### **-D property=value**

선택사항. 지정된 *property*의 *value*를 사용하십시오. **-D**(으)로 설정된 특성이 특성 파일에 설정된 특성보다 우선합니다.

**com.ibm.wmqfte.propertyset** 속성을 사용하여 Ant 태스크에 사용되는 구성 옵션 세트를 지정합니다. 기본값이 아닌 조정 큐 관리자의 이름을 이 특성의 값으로 사용하십시오. 그러면 Ant 태스크는 기본값이 아닌 조정 큐 관리자와 연관된 구성 옵션 세트를 사용합니다. 이 특성을 지정하지 않는 경우에는 기본 조정 큐 관리자를 기반으로 한 기본 구성 옵션 세트가 사용됩니다. Ant 태스크에 **cmdqm** 속성을 지정하는 경우 이 속성은 **fteAnt** 명령에 지정된 구성 옵션 세트보다 우선합니다. 이 작동은 기본 구성 옵션 세트를 사용하는지 아니면 **com.ibm.wmqfte.propertyset** 특성으로 세트를 지정하는지 여부에 관계없이 적용됩니다.

#### **-propertyfile(name)**

선택사항. **-D** 특성이 우선하는 파일에서 모든 특성을 로드합니다.

#### **-f(Ant script)**, **-file(Ant script)** 또는 **-buildfile(Ant script)**

필수입니다. 실행할 Ant 스크립트의 이름을 지정합니다.

#### **targets**

선택사항. Ant 스크립트에서 실행할 하나 이상의 대상 이름입니다. 이 매개변수의 값을 지정하지 않는 경우, 스크립트의 기본 대상이 실행됩니다.

#### **-version**

선택사항. Managed File Transfer 명령 및 Ant 버전을 표시합니다.

#### **-? 또는 -h**

선택사항. 명령 구문을 표시합니다.

## 예

이 예제에서는 Ant 스크립트 `fte_script.xml`의 대상 **copy**이(가) 실행되고 명령이 디버깅 출력을 표준 출력에 씁니다.

```
fteAnt -d -f fte_script.xml copy
```

## 리턴 코드

### 0

명령이 성공적으로 완료되었습니다.

### 1

명령이 성공하지 못하고 종료되었습니다.

Ant 스크립트에서 다른 상태 리턴 코드를 지정할 수도 있습니다(예를 들어, Ant 실패 태스크 사용).

자세한 정보는 [실패](#)의 내용을 참조하십시오.

## 관련 개념

[MFT을\(를\) 사용하여 Ant 스크립트를 사용하여 시작하기](#)

## 관련 태스크

[MFT과\(와\) 함께 Apache Ant 사용](#)

## 관련 참조

[MFT 용 샘플 Ant 태스크](#)

## fte:awaitoutcome Ant 태스크

**fte:filecopy**, **fte:filemove** 또는 **fte:call** 조작이 완료될 때까지 기다립니다.

## 속성

### id

필수입니다. 결과 수신을 위해 대기하고 있는 전송을 식별합니다. 일반적으로 이 특성은 [fte:filecopy](#), [fte:filemove](#) 또는 [fte:call](#) 태스크의 `idProperty` 속성을 사용하여 설정합니다.

### rcproperty

필수입니다. **fte:awaitoutcome** 태스크의 리턴 코드를 저장할 특성의 이름을 지정합니다.

### timeout

선택사항. 조작이 완료될 때까지 대기하는 최대 시간(초)입니다. 최소 제한시간은 1초입니다. 제한시간 값을 지정하지 않으면 **fte:awaitoutcome** 태스크가 조작의 결과가 판별될 때까지 무한정 기다립니다.

## 예

이 예제에서는 파일 사본이 시작되고 해당 ID는 `copy.id` 특성에 저장됩니다. 복사를 진행하는 동안 다른 처리가 발생할 수 있습니다. **fte:awaitoutcome**문은 복사 조작이 완료될 때까지 대기하기 위해 사용됩니다.

**fte:awaitoutcome**문은 `copy.id` 특성에 저장된 ID를 사용하여 대기할 조작을 식별합니다.

**fte:awaitoutcome**은 복사 조작의 결과를 표시하는 리턴 코드를 `copy.result`라는 특성에 저장합니다.

```
<!-- issue a file copy request -->
<fte:filecopy
  src="AGENT1@QM1"
  dst="AGENT2@QM2"
  idproperty="copy.id"
  outcome="defer">

  <fte:filespec
    srcfilespec="/home/fteuser1/file.bin"
    dstdir="/home/fteuser2"/>

</fte:filecopy>

<fte:awaitoutcome id="{copy.id}" rcProperty="copy.rc"/>
```

```
<echo>Copy id=${copy.id} rc=${copy.rc}</echo>
```

## 관련 태스크

MFT과(와) 함께 Apache Ant 사용

## fte:call Ant 태스크

**fte:call** 태스크를 사용하면 스크립트 및 프로그램을 원격으로 호출할 수 있습니다.

이 태스크를 사용하면 **fte:call** 요청을 에이전트에 송신할 수 있습니다. 에이전트는 스크립트나 프로그램을 실행하고 결과를 리턴하여 해당 요청을 처리합니다. 호출할 명령은 에이전트에 액세스할 수 있어야 합니다. `agent.properties` 파일의 `commandPath` 특성 값에 호출할 명령의 위치가 포함되어 있는지 확인하십시오. 명령 중첩 요소에 지정된 경로 정보는 `commandPath` 특성에 지정된 위치에 상대적이어야 합니다. 기본적으로 `commandPath`는 비어 있으므로 에이전트가 명령을 호출할 수 없습니다. 이 특성에 대한 자세한 정보는 [commandPath MFT 특성](#)을 참조하십시오.

`agent.properties` 파일에 대한 자세한 정보는 [MFT agent.properties 파일](#)의 내용을 참조하십시오.

## 속성

### 에이전트

필수입니다. **fte:call** 요청을 제출할 에이전트를 지정합니다. 에이전트 정보를 `agentname@qmgrname` 양식으로 지정하십시오. 여기서 `agentname`은(는) 에이전트의 이름이고 `qmgrname`은(는) 이 에이전트가 직접 연결된 큐 관리자의 이름입니다.

### cmdqm

선택사항. 요청을 제출할 대상 명령 큐 관리자입니다. 이 정보를 `qmgrname@host@port@channel` 양식으로 지정하십시오. 여기서,

- `qmgrname`은(는) 큐 관리자의 이름입니다.
- `host`은(는) 큐 관리자가 실행 중인 시스템의 선택적 호스트 이름입니다.
- `port`은(는) 큐 관리자가 대기 중인 선택적 포트 번호입니다.
- `channel`은(는) 사용할 선택적 SVRCONN 채널입니다.

명령 큐 관리자에 대한 `host`, `port` 또는 `channel` 정보를 생략하면 `command.properties` 파일에 지정된 연결 정보가 사용됩니다.



**주의:** 다음 변수에 대해 값이 지정되지 않는 경우 다음이 사용됩니다.

- `host` 변수: 바인딩 모드가 사용됨
- `port` 변수: 값 1414가 사용됨
- `channel` 변수: SYSTEM.DEF.SVRCONN이 사용됨

자세한 정보는 [MFT command.properties 파일](#)을 참조하십시오.

그러나 중간에 있는 속성은 건너뛸 수 없습니다(예: `qmgrname@host@@channel`). 예를 들어, `qmgrname@host` 또는 `qmgrname@host@port` 또는 `qmgrname@hostport@@channel`을(를) 사용할 수 있습니다.

MFT은(는) @ 구분 기호를 사용하여 주어진 속성을 분할합니다. 발견된 토큰 수에 따라 첫 번째 토큰은 `qmgrname`로, 두 번째 토큰은 `host`로, 세 번째 토큰은 `port`로, 그리고 마지막 토큰은 `channel`로 사용됩니다.

자세한 정보는 [MFT command.properties 파일](#)을 참조하십시오.

**com.ibm.wmqfte.propertySet** 특성을 사용하여 사용할 `command.properties` 파일을 지정할 수 있습니다. 자세한 정보는 [com.ibm.wmqfte.propertySet](#)를 참조하십시오.

`cmdqm` 속성을 사용하지 않는 경우, 이 특성이 설정된 경우에 `com.ibm.wmqfte.ant.commandQueueManager` 특성을 사용하도록 태스크가 기본 설정됩니다. `com.ibm.wmqfte.ant.commandQueueManager` 특성이 설정되지 않은 경우, `command.properties`



파일에 정의된 기본 큐 관리자에 대한 연결이 시도됩니다.  
com.ibm.wmqfte.ant.commandQueueManager 특성의 형식은 cmdqm 속성과 동일합니다. 즉,  
qmgrname@host@port@channel입니다.

### idproperty

defer의 outcome을(를) 지정하지 않은 경우 선택사항입니다. 전송 ID를 지정할 특성의 이름을 지정합니다. 전송 ID는 전송 요청을 제출할 때 생성되며, 전송 ID를 사용하여 전송 진행 상태를 추적하고 전송 문제점을 진단하며 전송을 취소할 수 있습니다.

ignore의 outcome 특성도 지정한 경우 이 특성을 지정할 수 없습니다. 그러나 defer의 outcome 특성도 지정한 경우 idproperty을(를) 지정해야 합니다.

### jobname

선택사항. **fte:call** 요청에 작업 이름을 지정합니다. 해당 작업 이름을 사용하여 전송의 논리 그룹을 작성할 수 있습니다. 의사(pseudo) 고유 작업 이름을 생성하려면 2023 페이지의 『[fte:uuid Ant 태스크](#)』 태스크를 사용하십시오. jobname 속성을 사용하지 않는 경우, 이 특성이 설정된 경우에 com.ibm.wmqfte.ant.jobName 특성 값을 사용하도록 태스크가 기본 설정됩니다. 이 특성을 설정하지 않으면 **fte:call** 요청과 연관되는 작업 이름이 없습니다.

### origuser

선택사항. **fte:call** 요청과 연관시킬 원래 사용자 ID를 지정합니다. origuser 속성을 사용하지 않는 경우, Ant 스크립트를 실행할 때 사용되는 사용자 ID를 사용하도록 태스크가 기본 설정됩니다.

### outcome

선택사항. Ant 스크립트에 제어를 리턴하기 전에 태스크가 **fte:call** 조작이 완료될 때까지 대기하는지 여부를 판별합니다. 다음 옵션 중 하나를 지정합니다.

#### await

제어를 되돌리기 전에 **fte:call** 조작이 완료될 때까지 태스크가 대기합니다. await의 outcome이(가) 지정된 경우 idproperty 속성은 선택사항입니다.

#### defer

태스크는 **fte:call** 요청이 제출되는 즉시 리턴되며 나중에 awaitoutcome 또는 ignoreoutcome 태스크를 사용하여 호출 조작의 결과가 처리된다고 가정합니다. defer의 outcome이(가) 지정되면 idproperty 속성이 필요합니다.

#### ignore

**fte:call** 조작의 결과가 중요하지 않은 경우 ignore의 값을 지정할 수 있습니다. 그러면 명령의 결과를 추적하기 위해 자원을 할당하지 않고 **fte:call** 요청이 제출된 후 바로 태스크가 제어를 되돌립니다. ignore의 outcome이(가) 지정되면 idproperty 속성을 지정할 수 없습니다.

outcome 속성을 지정하지 않는 경우, await 값을 사용하도록 태스크가 기본 설정됩니다.

### rcproperty

선택사항. **fte:call** 요청의 결과 코드를 지정할 특성의 이름을 지정합니다. 결과 코드는 **fte:call** 요청의 전체 결과를 반영합니다.

ignore 또는 defer의 outcome 특성도 지정한 경우 이 특성을 지정할 수 없습니다. 그러나 await의 결과를 지정한 경우에는 rcproperty을(를) 지정해야 합니다.

## 중첩 요소로 지정되는 매개변수

### fte:command

에이전트가 호출할 명령을 지정합니다. 단일 fte:command 요소를 특정 **fte:call** 조작과 연관시킬 수만 있습니다. 호출할 명령은 에이전트의 agent.properties 파일에 있는 commandPath 특성에 지정된 경로에 있어야 합니다.

### fte:metadata

호출 조작과 연관시킬 메타데이터를 지정할 수 있습니다. 이 메타데이터는 호출 조작에서 생성한 로그 메시지에 기록됩니다. 메타데이터의 단일 블록을 특정 전송 요소와 연관시킬 수만 있습니다. 그러나 이 블록은 많은 메타데이터를 포함할 수 있습니다.



## 예

이 예제는 큐 관리자 QM1에서 실행 중인 AGENT1에서 명령을 호출하는 방법을 보여줍니다. 호출할 명령은 `command.sh` 스크립트이며 스크립트는 `xyz`의 단일 인수로 호출됩니다. `command.sh` 명령은 에이전트의 `agent.properties` 파일에 있는 `commandPath` 특성에 지정된 경로에 있습니다.

```
<fte:call cmdqm="QM0@localhost@1414@SYSTEM.DEF.SVRCONN"
  agent="AGENT1@QM1"
  rcproperty="call.rc"
  origuser="bob"
  jobname="{job.id}">

  <fte:command command="command.sh" successrc="1" retrycount="5" retrywait="30">
    <fte:arg value="xyz"/>
  </fte:command>

  <fte:metadata>
    <fte:entry name="org.foo.accountName" value="BDG3R"/>
  </fte:metadata>

</fte:call>
```

## 관련 태스크

[MFT과\(와\) 함께 Apache Ant 사용](#)

## fte:cancel Ant 태스크

Managed File Transfer 관리 전송 또는 관리 호출을 취소합니다. 관리 전송은 **fte:filecopy** 또는 **fte:filemove** 태스크를 사용하여 작성되었을 수 있습니다. 관리 호출은 **fte:call** 태스크를 사용하여 작성되었을 수 있습니다.

## 속성

### agent

필수입니다. **fte:cancel** 요청을 제출할 에이전트를 지정합니다. 값은 `agentname@qmgrname` 양식입니다. 여기서 `agentname`은(는) 에이전트의 이름이고 `qmgrname`은(는) 이 에이전트가 직접 연결된 큐 관리자의 이름입니다.

### cmdqm

선택사항. 요청을 제출할 대상 명령 큐 관리자입니다. 이 정보를 `qmgrname@host@port@channel` 양식으로 지정하십시오. 여기서,

- `qmgrname`은(는) 큐 관리자의 이름입니다.
- `host`은(는) 큐 관리자가 실행 중인 시스템의 선택적 호스트 이름입니다.
- `port`은(는) 큐 관리자가 대기 중인 선택적 포트 번호입니다.
- `channel`은(는) 사용할 선택적 SVRCONN 채널입니다.

명령 큐 관리자에 대한 `host`, `port` 또는 `channel` 정보를 생략하면 `command.properties` 파일에 지정된 연결 정보가 사용됩니다.



**주의:** 다음 변수에 대해 값이 지정되지 않는 경우 다음이 사용됩니다.

- `host` 변수: 바인딩 모드가 사용됨
- `port` 변수: 값 1414가 사용됨
- `channel` 변수: SYSTEM.DEF.SVRCONN이 사용됨

자세한 정보는 [MFT command.properties](#) 파일을 참조하십시오.

그러나 중간에 있는 속성은 건너뛸 수 없습니다(예: `qmgrname@host@@channel`). 예를 들어, `qmgrname@host` 또는 `qmgrname@host@port` 또는 `qmgrname@hostport@@channel`을(를) 사용할 수 있습니다.

MFT은(는) @ 구분 기호를 사용하여 주어진 속성을 분할합니다. 발견된 토큰 수에 따라 첫 번째 토큰은 *qmgrname*로, 두 번째 토큰은 *host*로, 세 번째 토큰은 *port*로, 그리고 마지막 토큰은 *channel*로 사용됩니다.

자세한 정보는 [MFT command.properties](#) 파일을 참조하십시오.

**com.ibm.wmqfte.propertySet** 특성을 사용하여 사용할 `command.properties` 파일을 지정할 수 있습니다. 자세한 정보는 `com.ibm.wmqfte.propertySet`를 참조하십시오.

`cmdqm` 속성을 사용하지 않는 경우, 이 특성이 설정된 경우에 `com.ibm.wmqfte.ant.commandQueueManager` 특성을 사용하도록 태스크가 기본 설정됩니다. `com.ibm.wmqfte.ant.commandQueueManager` 특성이 설정되지 않은 경우, `command.properties` 파일에 정의된 기본 큐 관리자에 대한 연결이 시도됩니다. `com.ibm.wmqfte.ant.commandQueueManager` 특성의 형식은 `cmdqm` 속성과 동일합니다. 즉, `qmgrname@host@port@channel`입니다.

#### id

필수입니다. 취소할 전송의 전송 ID를 지정합니다. 전송 식별자는 `fte:filecopy` 및 `fte:filemove` 태스크 모두를 통해 전송 요청이 제출될 때 생성됩니다.

#### origuser

선택사항. **cancel** 요청과 연관시킬 원래 사용자 ID를 지정합니다. `origuser` 속성을 사용하지 않는 경우, Ant 스크립트를 실행할 때 사용되는 사용자 ID를 사용하도록 태스크가 기본 설정됩니다.

#### 예

예제에서는 **fte:cancel** 요청을 명령 큐 관리자 `qm0`에 송신합니다. **fte:cancel** 요청의 대상은 `transfer.id` 변수에 의해 채워진 전송 ID에 대한 큐 관리자 `qm1`의 `agent1`으로 지정됩니다. 요청은 "bob" 사용자 ID를 사용하여 실행됩니다.

```
<fte:cancel cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
  agent="agent1@qm1"
  id="{transfer.id}"
  origuser="bob"/>
```

#### 관련 태스크

[MFT과\(와\) 함께 Apache Ant 사용](#)

### fte:filecopy Ant 태스크

**fte:filecopy** 태스크는 Managed File Transfer 에이전트 사이에서 파일을 복사합니다. 파일은 소스 에이전트에서 삭제되지 않습니다.

#### 속성

##### cmdqm

선택사항. 요청을 제출할 대상 명령 큐 관리자입니다. 이 정보를 `qmgrname@host@port@channel` 양식으로 지정하십시오. 여기서,

- *qmgrname*은(는) 큐 관리자의 이름입니다.
- *host*은(는) 큐 관리자가 실행 중인 시스템의 선택적 호스트 이름입니다.
- *port*은(는) 큐 관리자가 대기 중인 선택적 포트 번호입니다.
- *channel*은(는) 사용할 선택적 SVRCONN 채널입니다.

명령 큐 관리자에 대한 *host*, *port* 또는 *channel* 정보를 생략하면 `command.properties` 파일에 지정된 연결 정보가 사용됩니다.



**주의:** 다음 변수에 대해 값이 지정되지 않는 경우 다음이 사용됩니다.

- *host* 변수: 바인딩 모드가 사용됨
- *port* 변수: 값 1414가 사용됨

- *channel* 변수: SYSTEM.DEF.SVRCONN이 사용됨

자세한 정보는 [MFT command.properties](#) 파일을 참조하십시오.

그러나 중간에 있는 속성은 건너뛴 수 없습니다(예: `qmgrname@host@@channel`). 예를 들어, `qmgrname@host` 또는 `qmgrname@host@port` 또는 `qmgrname@hostport@@channel`을(를) 사용할 수 있습니다.

MFT은(는) @ 구분 기호를 사용하여 주어진 속성을 분할합니다. 발견된 토큰 수에 따라 첫 번째 토큰은 *qmgrname*로, 두 번째 토큰은 *host*로, 세 번째 토큰은 *port*로, 그리고 마지막 토큰은 *channel*로 사용됩니다.

자세한 정보는 [MFT command.properties](#) 파일을 참조하십시오.

**com.ibm.wmqfte.propertySet** 특성을 사용하여 사용할 `command.properties` 파일을 지정할 수 있습니다. 자세한 정보는 [com.ibm.wmqfte.propertySet](#)를 참조하십시오.

`cmdqm` 속성을 사용하지 않는 경우, 이 특성이 설정된 경우에

`com.ibm.wmqfte.ant.commandQueueManager` 특성을 사용하도록 태스크가 기본 설정됩니다.

`com.ibm.wmqfte.ant.commandQueueManager` 특성이 설정되지 않은 경우, `command.properties` 파일에 정의된 기본 큐 관리자에 대한 연결이 시도됩니다.

`com.ibm.wmqfte.ant.commandQueueManager` 특성의 형식은 `cmdqm` 속성과 동일합니다. 즉, `qmgrname@host@port@channel`입니다.

#### dst

필수입니다. 복사 조작의 목적지 에이전트를 지정합니다. `agentname@qmgrname` 양식으로 이 정보를 지정하십시오. 여기서 `agentname`은(는) 목적지 에이전트의 이름이고 `qmgrname`은(는) 이 에이전트가 직접 연결되는 큐 관리자의 이름입니다.

#### idproperty

`defer`의 `outcome`을(를) 지정하지 않은 경우 선택사항입니다. 전송 ID를 지정할 특성의 이름을 지정합니다. 전송 ID는 전송 요청을 제출할 때 생성되며, 전송 ID를 사용하여 전송 진행 상태를 추적하고 전송 문제점을 진단하며 전송을 취소할 수 있습니다.

`ignore`의 `outcome` 특성도 지정한 경우 이 특성을 지정할 수 없습니다. 그러나 `defer`의 `outcome` 특성도 지정한 경우 `idproperty`을(를) 지정해야 합니다.

#### jobname

선택사항. 복사 요청에 작업 이름을 지정합니다. 해당 작업 이름을 사용하여 전송의 논리 그룹을 작성할 수 있습니다. 의사(pseudo) 고유 작업 이름을 생성하려면 2023 페이지의 『[fte:uuid Ant 태스크](#)』 태스크를 사용하십시오. `jobname` 속성을 사용하지 않는 경우, 이 특성이 설정된 경우에

`com.ibm.wmqfte.ant.jobName` 특성 값을 사용하도록 태스크가 기본 설정됩니다. 이 특성을 설정하지 않은 경우 복사 요청과 연관되는 작업 이름이 없습니다.

#### origuser

선택사항. 복사 요청과 연관시킬 원래 사용자 ID를 지정합니다. `origuser` 속성을 사용하지 않는 경우, Ant 스크립트를 실행할 때 사용되는 사용자 ID를 사용하도록 태스크가 기본 설정됩니다.

#### outcome

선택사항. Ant 스크립트로 제어를 되돌리기 전에 이동 복사 조작이 완료될 때까지 태스크가 대기하는지 여부를 판별합니다. 다음 옵션 중 하나를 지정합니다.

##### await

제어를 되돌리기 전에 복사 조작이 완료될 때까지 태스크가 대기합니다. `await`의 `outcome`이(가) 지정된 경우 `idproperty` 속성은 선택사항입니다.

##### defer

태스크는 복사 요청이 제출된 후 바로 제어를 되돌리고 2010 페이지의 『[fte:awaitoutcome Ant 태스크](#)』 또는 2021 페이지의 『[fte:ignoreoutcome Ant 태스크](#)』 태스크를 사용하여 나중에 복사 조작의 결과가 처리된다고 가정합니다. `defer`의 `outcome`이(가) 지정되면 `idproperty` 속성이 필요합니다.

##### ignore

복사 조작의 결과가 중요하지 않은 경우 `ignore` 값을 지정할 수 있습니다. 그러면 태스크는 전송 결과를 추적하기 위해 자원을 할당하지 않고 복사 요청이 제출된 후 바로 제어를 되돌립니다. `ignore`의 `outcome`이(가) 지정되면 `idproperty` 속성을 지정할 수 없습니다.

outcome 속성을 지정하지 않는 경우, await 값을 사용하도록 태스크가 기본 설정됩니다.

### priority

선택사항. 복사 요청과 연관시킬 우선순위를 지정합니다. 일반적으로, 우선순위가 높은 전송 요청이 우선순위가 낮은 요청보다 우선 처리됩니다. 우선순위 값은 0 - 9 사이의 값이어야 합니다(두 값 포함). 우선순위 값 0이 가장 낮은 우선순위이고 우선순위 값 9가 가장 높은 우선순위입니다. priority 속성을 지정하지 않은 경우 전송 기본값은 우선순위 0입니다.

### rcproperty

선택사항. 복사 요청의 결과 코드를 지정할 특성의 이름을 지정합니다. 결과 코드는 복사 요청의 전체 결과를 반영합니다.

ignore 또는 defer의 outcome 특성도 지정한 경우 이 특성을 지정할 수 없습니다. 그러나 await의 결과를 지정하는 경우 rcproperty을(를) 지정해야 합니다.

### transferRecoveryTimeout

선택사항. 소스 에이전트가 중단된 파일 전송의 복구를 계속 시도하는 기간(초)을 설정합니다. 다음 옵션 중 하나를 지정합니다.

#### -1

전송이 완료될 때까지 에이전트는 중단된 전송의 복구를 계속 시도합니다. 이 옵션을 사용할 경우 특성이 설정되지 않았을 때 에이전트의 기본 동작과 같습니다.

#### 0

에이전트는 복구를 시작하자마자 파일 전송을 중지합니다.

#### >0

에이전트는 지정된 양의 정수 값으로 설정된 기간(초) 동안 중단된 전송의 복구를 계속 시도합니다. 예:

```
<fte:filecopy cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
  src="agent1@qm1" dst="agent2@qm2"
  rcproperty="copy.result" transferRecoveryTimeout="21600">
  <fte:filespec srcfilespec="/home/fteuser1/file.bin" dstfile="/home/fteuser2/
file.bin"/>
</fte:filecopy>
```

은 에이전트가 복구를 시작한 시간부터 6시간 동안 전송 복구를 계속 시도함을 나타냅니다. 이 속성의 최대값은 999999999입니다.

이런 방식으로 전송 복구 제한시간 값을 지정하면 전송을 기반으로 제한시간이 설정됩니다. Managed File Transfer 네트워크의 모든 전송에 대한 글로벌 값을 설정하기 위해 [전송 복구 제한시간](#) 특성에 특성을 추가할 수 있습니다. 자세한 정보는 [복구 중인 전송에 대한 제한시간 옵션](#)을 참조하십시오.

### src

필수입니다. 복사 조작의 소스 에이전트를 지정합니다. 이 정보는 *agentname@qmgrname* 형식으로 지정하십시오. 여기서 *agentname*은 소스 에이전트의 이름이고 *qmgrname*은 이 에이전트가 직접 연결되는 큐 관리자의 이름입니다.

## 중첩 요소로 지정되는 매개변수

### fte:filespec

필수입니다. 복사할 파일을 식별하는 파일 스펙을 하나 이상 지정해야 합니다. 필요하면 둘 이상의 파일 스펙을 지정할 수 있습니다. 자세한 정보는 [2023 페이지의 『fte:filespec Ant 중첩 요소』](#)의 내용을 참조하십시오.

### fte:metadata

복사 조작과 연관시킬 메타데이터를 지정할 수 있습니다. 이 메타데이터는 전송과 함께 전달되며 전송에서 생성한 로그 메시지에 기록됩니다. 메타데이터의 단일 블록을 특정 전송 요소와 연관시킬 수만 있습니다. 그러나 이 블록은 많은 메타데이터를 포함할 수 있습니다. 자세한 정보는 [fte:metadata](#) 토픽을 참조하십시오.

### fte:presrc

전송 시작 전에 소스 에이전트에서 발생하는 프로그램 호출을 지정합니다. 단일 fte:presrc 요소만 지정된 전송과 연관시킬 수 있습니다. 자세한 정보는 [프로그램 호출 주제](#)를 참조하십시오.

### **fte:predst**

전송 시작 전에 목적지 에이전트에서 발생할 프로그램 호출을 지정합니다. 단일 `fte:predst` 요소만 지정된 전송과 연관시킬 수 있습니다. 자세한 정보는 [프로그램 호출](#) 주제를 참조하십시오.

### **fte:postsrc**

전송 완료 후에 소스 에이전트에서 발생하는 프로그램 호출을 지정합니다. 단일 `fte:postsrc` 요소만 지정된 전송과 연관시킬 수 있습니다. 자세한 정보는 [프로그램 호출](#) 주제를 참조하십시오.

### **fte:postdst**

전송 완료 후에 목적지 에이전트에서 발생하는 프로그램 호출을 지정합니다. 단일 `fte:postdst` 요소만 지정된 전송과 연관시킬 수 있습니다. 자세한 정보는 [프로그램 호출](#) 주제를 참조하십시오.

`fte:presrc`, `fte:predst`, `fte:postsrc`, `fte:postdst` 및 엑시트가 성공 상태를 리턴하지 않는 경우 규칙은 다음과 같으며 지정된 순서를 따릅니다.

1. 소스 시작 엑시트를 실행하십시오. 소스 시작 엑시트에 실패하는 경우 전송에 실패하고 추가적인 조치가 실행되지 않습니다.
2. 사전 소스 호출이 제공되는 경우 이를 실행하십시오. 사전 소스 호출에 실패하는 경우 전송에 실패하고 추가적인 조치가 실행되지 않습니다.
3. 목적지 시작 엑시트를 실행하십시오. 목적지 시작 엑시트에 실패하는 경우 전송에 실패하고 추가적인 조치가 실행되지 않습니다.
4. 사전 목적지 호출이 제공되는 경우 이를 실행하십시오. 사전 목적지 호출에 실패하는 경우 전송에 실패하고 추가적인 조치가 실행되지 않습니다.
5. 파일 전송을 수행하십시오.
6. 목적지 종료 엑시트를 실행하십시오. 이러한 엑시트에 해당하는 실패 상태가 없습니다.
7. 전송에 성공할 경우(일부 파일의 전송이 완료되면 전송에 성공한 것으로 간주됨) 사후 목적지 호출(존재하는 경우)을 실행하십시오. 사후 목적지 호출에 실패하는 경우 전송에 실패합니다.
8. 소스 종료 엑시트를 실행하십시오. 이러한 엑시트에 해당하는 실패 상태가 없습니다.
9. 전송에 성공할 경우 사후 소스 호출(존재하는 경우)을 실행하십시오. 사후 소스 호출에 실패하는 경우 전송에 실패합니다.

### **예**

이 예제는 `agent1`과(와) `agent2` 간의 기본 파일 전송을 보여줍니다. 파일 전송을 시작하는 명령은 클라이언트 전송 모드 연결을 사용하여 `qm0`, (이)라는 큐 관리자로 전송됩니다. 파일 전송 조작의 결과는 `copy.result` 특성에 지정됩니다.

```
<fte:filecopy cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
  src="agent1@qm1" dst="agent2@qm2"
  rcproperty="copy.result">

  <fte:filespec srcfilespec="/home/fteuser1/file.bin" dstfile="/home/fteuser2/file.bin"/>
</fte:filecopy>
```

이 예에서는 동일하지만 전송 완료 후 소스 에이전트에서 수행되는 프로그램 시작과 메타데이터가 추가된 파일 전송을 보여 줍니다.

```
<fte:filecopy cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
  src="agent1@qm1" dst="agent2@qm2"
  rcproperty="copy.result">

  <fte:metadata>
    <fte:entry name="org.example.departId" value="ACCOUNTS"/>
    <fte:entry name="org.example.batchGroup" value="A1"/>
  </fte:metadata>

  <fte:filespec srcfilespec="/home/fteuser1/file.bin" dstfile="/home/fteuser2/file.bin"/>

  <fte:postsrc command="/home/fteuser2/scripts/post.sh" successsrc="1" >
    <fte:arg value="/home/fteuser2/file.bin"/>
  </fte:postsrc>
</fte:filecopy>
```

```
</fte:postsrc>  
</fte:filecopy>
```

## 관련 개념

복구 시 파일 전송에 대한 제한시간 초과 옵션

## 관련 태스크

MFT과(와) 함께 Apache Ant 사용

## fte:filemove Ant 태스크

**fte:filemove** 태스크는 Managed File Transfer 에이전트 사이에서 파일을 이동시킵니다. 소스 에이전트에서 목적지 에이전트로 파일이 성공적으로 전송되면 소스 에이전트에서 해당 파일이 삭제됩니다.

## 속성

### cmdqm

선택사항. 요청을 제출할 대상 명령 큐 관리자입니다. 이 정보를 `qmgrname@host@port@channel` 양식으로 지정하십시오. 여기서,

- `qmgrname`은(는) 큐 관리자의 이름입니다.
- `host`은(는) 큐 관리자가 실행 중인 시스템의 선택적 호스트 이름입니다.
- `port`은(는) 큐 관리자가 대기 중인 선택적 포트 번호입니다.
- `channel`은(는) 사용할 선택적 SVRCONN 채널입니다.

명령 큐 관리자에 대한 `host`, `port` 또는 `channel` 정보를 생략하면 `command.properties` 파일에 지정된 연결 정보가 사용됩니다.



**주의:** 다음 변수에 대해 값이 지정되지 않는 경우 다음이 사용됩니다.

- `host` 변수: 바인딩 모드가 사용됨
- `port` 변수: 값 1414가 사용됨
- `channel` 변수: SYSTEM.DEF.SVRCONN이 사용됨

자세한 정보는 [MFT command.properties](#) 파일을 참조하십시오.

그러나 중간에 있는 속성은 건너뛴 수 없습니다(예: `qmgrname@host@@channel`). 예를 들어, `qmgrname@host` 또는 `qmgrname@host@port` 또는 `qmgrname@hostport@@channel`을(를) 사용할 수 있습니다.

MFT은(는) @ 구분 기호를 사용하여 주어진 속성을 분할합니다. 발견된 토큰 수에 따라 첫 번째 토큰은 `qmgrname`로, 두 번째 토큰은 `host`로, 세 번째 토큰은 `port`로, 그리고 마지막 토큰은 `channel`로 사용됩니다.

자세한 정보는 [MFT command.properties](#) 파일을 참조하십시오.

**com.ibm.wmqfte.propertySet** 특성을 사용하여 사용할 `command.properties` 파일을 지정할 수 있습니다. 자세한 정보는 `com.ibm.wmqfte.propertySet`를 참조하십시오.

`cmdqm` 속성을 사용하지 않는 경우, 이 특성이 설정된 경우에 `com.ibm.wmqfte.ant.commandQueueManager` 특성을 사용하도록 태스크가 기본 설정됩니다. `com.ibm.wmqfte.ant.commandQueueManager` 특성이 설정되지 않은 경우, `command.properties` 파일에 정의된 기본 큐 관리자에 대한 연결이 시도됩니다. `com.ibm.wmqfte.ant.commandQueueManager` 특성의 형식은 `cmdqm` 속성과 동일합니다. 즉, `qmgrname@host@port@channel`입니다.

### dst

필수입니다. 복사 조작의 목적지 에이전트를 지정합니다. `agentname@qmgrname` 양식으로 이 정보를 지정하십시오. 여기서 `agentname`은(는) 목적지 에이전트의 이름이고 `qmgrname`은(는) 이 에이전트가 직접 연결되는 큐 관리자의 이름입니다.



## idproperty

defer의 outcome을(를) 지정하지 않은 경우 선택사항입니다. 전송 ID를 지정할 특성의 이름을 지정합니다. 전송 ID는 전송 요청을 제출할 때 생성되며, 전송 ID를 사용하여 전송 진행 상태를 추적하고 전송 문제점을 진단하며 전송을 취소할 수 있습니다.

ignore의 outcome 특성도 지정한 경우 이 특성을 지정할 수 없습니다. 그러나 defer의 outcome 특성도 지정한 경우 idproperty을(를) 지정해야 합니다.

## jobname

선택사항. 이동 요청에 작업 이름을 지정합니다. 해당 작업 이름을 사용하여 전송의 논리 그룹을 작성할 수 있습니다. fte:uuid 태스크를 사용하여 의사(pseudo) 고유 작업 이름을 생성하십시오. jobname 속성을 사용하지 않는 경우, 이 특성이 설정된 경우에 com.ibm.wmqfte.ant.jobName 특성 값을 사용하도록 태스크가 기본 설정됩니다. 이 특성을 설정하지 않은 경우 이동 요청과 연관되는 작업 이름이 없습니다.

## origuser

선택사항. 이동 요청과 연관시킬 원래 사용자 ID를 지정합니다. origuser 속성을 사용하지 않는 경우, Ant 스크립트를 실행할 때 사용되는 사용자 ID를 사용하도록 태스크가 기본 설정됩니다.

## outcome

선택사항. Ant 스크립트로 제어를 되돌리기 전에 이동 조작이 완료될 때까지 태스크가 대기하는지 여부를 판별합니다. 다음 옵션 중 하나를 지정합니다.

### await

제어를 되돌리기 전에 이동 조작이 완료될 때까지 태스크가 대기합니다. await의 outcome이(가) 지정된 경우 idproperty 속성은 선택사항입니다.

### defer

태스크는 이동 요청이 제출된 후 바로 제어를 되돌리고 2010 페이지의 『fte:awaitoutcome Ant 태스크』 또는 2021 페이지의 『fte:ignoreoutcome Ant 태스크』 태스크를 사용하여 나중에 이동 조작의 결과가 처리된다고 가정합니다. defer의 outcome이(가) 지정되면 idproperty 속성이 필요합니다.

### ignore

이동 조작의 결과가 중요하지 않은 경우 ignore 값을 지정할 수 있습니다. 그러면 태스크는 전송 결과를 추적하기 위해 자원을 할당하지 않고 이동 요청이 제출된 후 바로 제어를 되돌립니다. ignore의 outcome이(가) 지정되면 idproperty 속성을 지정할 수 없습니다.

outcome 속성을 지정하지 않는 경우, await 값을 사용하도록 태스크가 기본 설정됩니다.

## priority

선택사항. 이동 요청과 연관시킬 우선순위를 지정합니다. 일반적으로, 우선순위가 높은 전송 요청이 우선순위가 낮은 요청보다 우선 처리됩니다. 우선순위 값은 0 - 9 사이의 값이어야 합니다(두 값 포함). 우선순위 값 0이 가장 낮은 우선순위이고 우선순위 값 9가 가장 높은 우선순위입니다. priority 속성을 지정하지 않은 경우 전송 기본값은 우선순위 0입니다.

## rcproperty

선택사항. 이동 요청의 결과 코드를 지정할 특성의 이름을 지정합니다. 결과 코드는 이동 요청의 전체 결과를 반영합니다.

ignore 또는 defer의 outcome 특성도 지정한 경우 이 특성을 지정할 수 없습니다. 그러나 await의 결과를 지정한 경우에는 rcproperty을(를) 지정해야 합니다.

## transferRecoveryTimeout

선택사항. 소스 에이전트가 중단된 파일 전송의 복구를 계속 시도하는 기간(초)을 설정합니다. 다음 옵션 중 하나를 지정합니다.

### -1

전송이 완료될 때까지 에이전트는 중단된 전송의 복구를 계속 시도합니다. 이 옵션을 사용할 경우 특성이 설정되지 않았을 때 에이전트의 기본 동작과 같습니다.

### 0

에이전트는 복구를 시작하자마자 파일 전송을 중지합니다.

### >0

에이전트는 지정된 양의 정수 값으로 설정된 기간(초) 동안 중단된 전송의 복구를 계속 시도합니다. 예:

```
<fte:filemove cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
src=agent1@qm1 dst="agent2@qm2"
```

```
rcproperty="move.result" transferRecoveryTimeout="21600">
  <fte:filespec srcfilespec="/home/fteuser1/file.bin" dstfile="/home/fteuser2/
file.bin"/>
</fte:filemove
```

은 에이전트가 복구를 시작한 시간부터 6시간 동안 전송 복구를 계속 시도함을 나타냅니다. 이 속성의 최대값은 999999999입니다.

이런 방식으로 전송 복구 제한시간 값을 지정하면 전송을 기반으로 제한시간이 설정됩니다. Managed File Transfer 네트워크의 모든 전송에 대한 글로벌 값을 설정하기 위해 전송 복구 제한시간 특성에 특성을 추가할 수 있습니다. 자세한 정보는 [복구 중인 전송에 대한 제한시간 옵션](#)을 참조하십시오.

#### src

필수입니다. 이동 조작의 소스 에이전트를 지정합니다. 다음 양식으로 이 정보를 지정하십시오. *agentname@qmgrname* 여기서 *agentname*은(는) 소스 에이전트의 이름이고 *qmgrname*은(는) 이 에이전트가 직접 연결되는 큐 관리자의 이름입니다.

### 중첩 요소로 지정되는 매개변수

#### fte:filespec

필수입니다. 이동시킬 파일을 식별하는 파일 스펙을 하나 이상 지정해야 합니다. 필요하다면 둘 이상의 파일 스펙을 지정할 수 있습니다. 자세한 정보는 2023 페이지의 『[fte:filespec Ant 중첩 요소](#)』의 내용을 참조하십시오.

#### fte:metadata

선택사항. 파일 이동 조작과 연관시킬 메타데이터를 지정할 수 있습니다. 이 메타데이터는 전송과 함께 전달되며 전송에서 생성한 로그 메시지에 기록됩니다. 메타데이터의 단일 블록을 특정 전송 요소와 연관시킬 수만 있습니다. 그러나 이 블록은 많은 메타데이터를 포함할 수 있습니다. 자세한 정보는 [fte:metadata](#) 토크를 참조하십시오.

#### fte:presrc

선택사항. 전송 시작 전에 소스 에이전트에서 발생하는 프로그램 호출을 지정합니다. 단일 *fte:presrc* 요소만 지정된 전송과 연관시킬 수 있습니다. 자세한 정보는 [프로그램 호출](#) 주제를 참조하십시오.

#### fte:predst

선택사항. 전송 시작 전에 목적지 에이전트에서 발생할 프로그램 호출을 지정합니다. 단일 *fte:predst* 요소만 지정된 전송과 연관시킬 수 있습니다. 자세한 정보는 [프로그램 호출](#) 주제를 참조하십시오.

#### fte:postsrc

선택사항. 전송 완료 후에 소스 에이전트에서 발생하는 프로그램 호출을 지정합니다. 단일 *fte:postsrc* 요소만 지정된 전송과 연관시킬 수 있습니다. 자세한 정보는 [프로그램 호출](#) 주제를 참조하십시오.

#### fte:postdst

선택사항. 전송 완료 후에 목적지 에이전트에서 발생하는 프로그램 호출을 지정합니다. 단일 *fte:postdst* 요소만 지정된 전송과 연관시킬 수 있습니다. 자세한 정보는 [프로그램 호출](#) 주제를 참조하십시오.

*fte:presrc*, *fte:predst*, *fte:postsrc*, *fte:postdst* 및 엑시트가 성공 상태를 리턴하지 않는 경우 규칙은 다음과 같으며 지정된 순서를 따릅니다.

1. 소스 시작 엑시트를 실행하십시오. 소스 시작 엑시트에 실패하는 경우 전송에 실패하고 추가적인 조치가 실행되지 않습니다.
2. 사전 소스 호출이 제공되는 경우 이를 실행하십시오. 사전 소스 호출에 실패하는 경우 전송에 실패하고 추가적인 조치가 실행되지 않습니다.
3. 목적지 시작 엑시트를 실행하십시오. 목적지 시작 엑시트에 실패하는 경우 전송에 실패하고 추가적인 조치가 실행되지 않습니다.
4. 사전 목적지 호출이 제공되는 경우 이를 실행하십시오. 사전 목적지 호출에 실패하는 경우 전송에 실패하고 추가적인 조치가 실행되지 않습니다.
5. 파일 전송을 수행하십시오.
6. 목적지 종료 엑시트를 실행하십시오. 이러한 엑시트에 해당하는 실패 상태가 없습니다.



7. 전송에 성공할 경우(일부 파일의 전송이 완료되면 전송에 성공한 것으로 간주됨) 사후 목적지 호출(존재하는 경우)을 실행하십시오. 사후 목적지 호출에 실패하는 경우 전송에 실패합니다.
8. 소스 종료 엑시트를 실행하십시오. 이러한 엑시트에 해당하는 실패 상태가 없습니다.
9. 전송에 성공할 경우 사후 소스 호출(존재하는 경우)을 실행하십시오. 사후 소스 호출에 실패하는 경우 전송에 실패합니다.

## 예

이 예제는 agent1과(와) agent2 사이의 기본 파일 이동을 보여줍니다. 파일 이동을 시작하는 명령은 클라이언트 전송 모드 연결을 사용하여 qm0,(이)라는 큐 관리자로 전송됩니다. 파일 전송 조작의 결과는 `move.result` 특성에 지정됩니다.

```
<fte:filemove cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
  src="agent1@qm1" dst="agent2@qm2"
  rcproperty="move.result">

  <fte:filespec srcfilespec="/home/fteuser1/file.bin" dstfile="/home/fteuser2/file.bin"/>
</fte:filemove>
```

## 관련 개념

[복구 시 파일 전송에 대한 제한시간 초과 옵션](#)

## 관련 태스크

[MFT과\(와\) 함께 Apache Ant 사용](#)

## fte:ignoreoutcome Ant 태스크

**fte:filecopy**, **fte:filemove** 또는 **fte:call** 명령의 결과를 무시합니다. **fte:filecopy**, **fte:filemove** 또는 **fte:call** 태스크를 지정하여 `defer` 결과를 얻는 경우, Ant 태스크는 이 결과 추적에 자원을 할당합니다. 더 이상 결과가 중요하지 않은 경우에는 **fte:ignoreoutcome** 태스크를 사용하여 해당 자원을 해제할 수 있습니다.

## 속성

### id

필수입니다. 더 이상 중요하지 않은 결과를 식별합니다. 일반적으로 2014 페이지의 『[fte:filecopy Ant 태스크](#)』, 2018 페이지의 『[fte:filemove Ant 태스크](#)』 또는 2011 페이지의 『[fte:call Ant 태스크](#)』 태스크의 `idproperty` 속성을 사용하여 설정한 특성을 사용하여 이 ID를 지정합니다.

## 예

이 예에서는 `fte:ignoreoutcome` 태스크를 사용하여 이전 2014 페이지의 『[fte:filecopy Ant 태스크](#)』 태스크의 결과 추적에 할당된 자원을 해제하는 방법을 보여 줍니다.

```
<!-- issue a file copy request -->
<fte:filecopy cmdqm="qm1@localhost@1414@SYSTEM.DEF.SVRCONN"
  src="agent1@qm1" dst="agent1@qm1"
  idproperty="copy.id"
  outcome="defer"/>

<!-- do some other things -->

<!-- decide that the result of the copy is not interesting -->
<fte:ignoreoutcome id="{copy.id}"/>
```

## 관련 태스크

[MFT과\(와\) 함께 Apache Ant 사용](#)

## fte:ping Ant 태스크

IBM MQ Managed File Transfer Ant 태스크는 응답을 유도하도록 에이전트를 PING하고 에이전트가 전송을 처리할 수 있는지 판별합니다.

**참고:** IBMWebSphereMQ FTE(파일 전송 에디션)는 더 이상 지원되지 않는 제품입니다. FTE에서 IBM MQ의 Managed File Transfer 컴포넌트로 마이그레이션하려면 [Managed File Transfer 마이그레이션을 참조하십시오.](#)

### 속성

#### agent

필수입니다. **fte:ping** 요청을 제출할 에이전트를 지정합니다. 값은 `agentname@qmgrname` 양식입니다. 여기서 `agentname`은(는) 에이전트의 이름이고 `qmgrname`은(는) 이 에이전트가 직접 연결된 큐 관리자의 이름입니다.

#### cmdqm

선택사항. 요청을 제출할 대상 명령 큐 관리자입니다. 이 정보를 `qmgrname@host@port@channel` 양식으로 지정하십시오. 여기서,

- `qmgrname`은(는) 큐 관리자의 이름입니다.
- `host`은(는) 큐 관리자가 실행 중인 시스템의 선택적 호스트 이름입니다.
- `port`은(는) 큐 관리자가 대기 중인 선택적 포트 번호입니다.
- `channel`은(는) 사용할 선택적 SVRCONN 채널입니다.

명령 큐 관리자에 대한 `host`, `port` 또는 `channel` 정보를 생략하면 `command.properties` 파일에 지정된 연결 정보가 사용됩니다.



**주의:** 다음 변수에 대해 값이 지정되지 않는 경우 다음이 사용됩니다.

- `host` 변수: 바인딩 모드가 사용됨
- `port` 변수: 값 1414가 사용됨
- `channel` 변수: SYSTEM.DEF.SVRCONN이 사용됨

자세한 정보는 [MFT command.properties](#) 파일을 참조하십시오.

그러나 중간에 있는 속성은 건너뛴 수 없습니다(예: `qmgrname@host@@channel`). 예를 들어, `qmgrname@host` 또는 `qmgrname@host@port` 또는 `qmgrname@hostport@@channel`을(를) 사용할 수 있습니다.

MFT은(는) @ 구분 기호를 사용하여 주어진 속성을 분할합니다. 발견된 토큰 수에 따라 첫 번째 토큰은 `qmgrname`로, 두 번째 토큰은 `host`로, 세 번째 토큰은 `port`로, 그리고 마지막 토큰은 `channel`로 사용됩니다.

자세한 정보는 [MFT command.properties](#) 파일을 참조하십시오.

**com.ibm.wmqfte.propertySet** 특성을 사용하여 사용할 `command.properties` 파일을 지정할 수 있습니다. 자세한 정보는 `com.ibm.wmqfte.propertySet`를 참조하십시오.

`cmdqm` 속성을 사용하지 않는 경우, 이 특성이 설정된 경우에

`com.ibm.wmqfte.ant.commandQueueManager` 특성을 사용하도록 태스크가 기본 설정됩니다.

`com.ibm.wmqfte.ant.commandQueueManager` 특성이 설정되지 않은 경우, `command.properties` 파일에 정의된 기본 큐 관리자에 대한 연결이 시도됩니다.

`com.ibm.wmqfte.ant.commandQueueManager` 특성의 형식은 `cmdqm` 속성과 동일합니다. 즉, `qmgrname@host@port@channel`입니다.

#### rcproperty

필수입니다. **ping** 조작의 리턴 코드를 저장할 특성을 지정합니다.

#### timeout

선택사항. 태스크가 에이전트 응답을 기다리는 최대 시간(초)입니다. 최소 제한시간은 0초이지만 명령이 에이전트가 응답할 때까지 무한정 기다리도록 제한시간을 -1로 지정할 수도 있습니다. `timeout`에 값이 지정되지 않은 경우 기본값은 에이전트가 응답할 때까지 최대 5초를 기다리는 것입니다.

## 예

이 예에서는 qm1에서 호스트하는 agent1에 **fte:ping** 요청을 송신합니다. **fte:ping** 요청은 15초 동안 에이전트의 응답을 기다립니다. **fte:ping** 요청의 결과는 ping.rc라는 특성에 저장됩니다.

```
<fte:ping agent="agent1@qm1" rcproperty="ping.rc" timeout="15"/>
```

## 리턴 코드

### 0

명령이 성공적으로 완료되었습니다.

### 2

명령 시간이 종료되었습니다.

## 관련 태스크

[MFT과\(와\) 함께 Apache Ant 사용](#)

## fte:uuid Ant 태스크

의사(pseudo) 무작위 고유 ID를 생성하고 제공된 특성에 지정합니다. 예를 들어, 이 ID를 사용하여 다른 파일 전송 조작의 작업 이름을 생성할 수 있습니다.

## 속성

### 길이

필수입니다. 생성할 UUID의 숫자 길이입니다. 이 길이 값은 **prefix** 매개변수로 지정된 접두부의 길이를 포함하지 않습니다.

### property

필수입니다. 생성된 UUID를 지정할 특성의 이름입니다.

### 접두어

선택사항. 생성된 UUID에 추가할 접두부입니다. 이 접두어는 **length** 매개변수에 지정된 대로 UUID 길이의 일부로 계산되지 않습니다.

## 예



이 예제는 문자 ABC(으)로 시작하는 UUID와 뒤에 16개의 의사 무작위 16진수 문자를 정의합니다. UUID는 uuid.property 특성에 지정됩니다.

```
<fte:uuid length="16" property="uuid.property" prefix="ABC"/>
```

## 관련 태스크

[MFT과\(와\) 함께 Apache Ant 사용](#)

## fte:filespec Ant 중첩 요소

**fte:filespec** 매개변수는 다른 태스크에서 중첩된 요소로 사용됩니다. **fte:filespec**을(를) 사용하면 하나 이상의 소스 파일, 디렉토리  또는 데이터 세트 및 목적지 간의 매핑을 설명할 수 있습니다. 일반적으로 이 요소는 이동시키거나 복사할 파일 세트, 디렉토리  또는 데이터 세트를 표현할 때 사용됩니다.

## 중첩 주체

- [fte:filecopy](#) 태스크
- [fte:filemove](#) 태스크

## 소스 스펙 속성

srcfilespec 또는 srcqueue 중 하나를 지정해야 합니다.

### srcfilespec

파일 조작의 소스를 지정합니다. 이 속성 값에는 와일드카드를 사용할 수 있습니다.

### srcqueue

전송 소스가 큐가 되도록 지정합니다. 전송하면 이 속성으로 지정한 큐에 저장된 메시지에서 데이터가 이동됩니다. **fte:filespec** 태스크가 **fte:filecopy** 태스크 내에 중첩된 경우 이 속성을 지정할 수 없습니다.

소스 에이전트가 프로토콜 브릿지 에이전트인 경우에는 srcqueue 속성이 지원되지 않습니다.

## 목적지 스펙 속성

dstdir, dstds, dstfilespace, dstfile, dstqueue 또는 dstpds 중 하나를 지정해야 합니다.

### dstdir

파일 조작의 목적지로 디렉토리를 지정합니다.

### dstds

파일 조작의 목적지로 데이터 세트를 지정합니다.

이 속성은 목적지 에이전트가 z/OS 플랫폼에서 실행 중인 경우에만 지원됩니다.

### dstfile

파일 조작의 목적지로 파일을 지정합니다.

### dstfilespace

파일 공간을 파일 조작의 목적지로 지정합니다.

이 속성은 목적지 에이전트가 웹 게이트웨이 파일 공간에 대한 액세스 권한이 있는 IBM MQ 8.0 웹 에이전트인 경우에만 적용됩니다.

### dstpds

파일 조작의 목적지로 파티션된 데이터 세트를 지정합니다.

이 속성은 목적지 에이전트가 z/OS 플랫폼에서 실행 중인 경우에만 지원됩니다.

### dstqueue

파일에서 메시지로 조작의 목적지로 큐를 지정합니다. QUEUE@QUEUEMANAGER 양식을 사용하여 선택적으로 이 스펙에 큐 관리자 이름을 포함시킬 수 있습니다. 큐 관리자 이름을 지정하지 않으면 enableClusterQueueInputOutput 에이전트 특성을 True로 설정하지 않은 경우 목적지 에이전트 큐 관리자가 사용됩니다. enableClusterQueueInputOutput 특성이 True로 설정되면 목적지 에이전트는 표준 IBM MQ 프로시저를 사용하여 큐의 위치를 판별합니다. 큐 관리자에 존재하는 올바른 큐 이름을 지정해야 합니다.

dstqueue 속성을 지정하는 경우, 이 속성이 상호 배타적이므로 srcqueue 속성을 지정할 수 없습니다.

목적지 에이전트가 프로토콜 브릿지 에이전트인 경우에는 dstqueue 속성이 지원되지 않습니다.

## 소스 옵션 속성

### srcencoding

선택사항. 전송할 파일에서 사용하는 문자 세트 인코딩입니다.

conversion 속성이 text. 값으로 설정된 경우에만 이 속성을 지정할 수 있습니다.

srcencoding 속성을 지정하지 않을 경우 텍스트 전송에 소스 시스템의 문자 세트가 사용됩니다.

### srceol

선택사항. 전송 중인 파일에서 사용하는 행 구분 기호의 끝입니다. 올바른 값은 다음과 같습니다.

- CRLF - 캐리지 리턴 문자와 줄 바꿈 문자를 줄 끝 구분 기호로 사용합니다. 이 변환은 Windows 시스템에서 일반적입니다.
- LF - 줄 바꿈 문자를 줄 끝 구분 기호로 사용합니다. 이 변환은 UNIX 시스템에서 일반적입니다.

conversion 속성이 text 값으로 설정된 경우에만 이 속성을 지정할 수 있습니다. srceol 속성을 지정하지 않을 경우 텍스트 전송에서 소스 에이전트의 운영 체제를 기반으로 자동으로 올바른 값을 판별합니다.

## z/OS

### srckeeptrailingspaces

선택사항. 텍스트 모드 전송의 일부분으로 고정 길이 양식 데이터 세트로부터 읽은 소스 레코드에서 후미 공백이 유지되는지 여부를 결정합니다. 올바른 값은 다음과 같습니다.

- true - 후미 공백이 유지됩니다.
- false - 후미 공백이 제거됩니다.

srckeeptrailingspace 속성을 지정하지 않으면 false의 기본값이 지정됩니다.

srcfilespec 속성도 지정하고 conversion 속성을 text. 값으로 설정한 경우에만 이 속성을 지정할 수 있습니다.

### srcmsgdelimbytes

선택사항. 여러 개의 메시지를 하나의 2진 파일에 추가할 때 구분 기호로 삽입할 하나 이상의 바이트값을 지정합니다. 각 값은 00-FF 범위의 16진 숫자 두 개로 지정되어야 하며, x로 접두부가 지정되어야 합니다. 다중 바이트는 쉼표로 분리해야 합니다. 예를 들어, srcmsgdelimbytes="x08,xA4"입니다. 또한 srcqueue 속성을 지정한 경우에만 srcmsgdelimbytes 속성을 지정할 수 있습니다. conversion 속성에 text 값을 지정한 경우에는 srcmsgdelimbytes 속성을 지정할 수 없습니다.

### srcmsgdelimtext

선택사항. 여러 개의 메시지를 한 텍스트 파일에 추가할 때 구분 기호로 삽입할 텍스트의 순서를 지정합니다. 구분 기호에 문자열 리터럴로 Java 이스케이프 순서를 포함시킬 수 있습니다 예를 들어, srcmsgdelimtext="\u007d\n"입니다. 소스 에이전트에 의해 각 메시지 뒤에 텍스트 구분 기호가 삽입됩니다. 텍스트 구분 기호는 전송의 소스 인코딩을 사용하여 2진 양식으로 인코딩됩니다. 각 메시지는 2진 양식으로 읽고 인코딩된 구분 문자가 2진 양식으로 메시지에 추가되며 결과는 2진 양식으로 목적지 에이전트에 전송됩니다. 소스 에이전트 코드 페이지에 SI(shift-in) 및 SO(shift-out) 상태가 포함된 경우 에이전트는 각 메시지가 메시지의 끝에서 SO 상태가 되는 것으로 간주합니다. 목적지 에이전트에서 2진 데이터는 파일 대 파일 텍스트 전송과 동일한 방식으로 변환됩니다. srcqueue 속성과 conversion 속성에 대한 text 값도 지정한 경우에만 srcmsgdelimtext 속성을 지정할 수 있습니다.

### srcmsgdelimposition

선택사항. 텍스트 또는 2진 구분 기호가 삽입되는 위치를 지정합니다. 올바른 값은 다음과 같습니다.

- prefix - 각 메시지의 데이터 앞에 구분 기호가 목적지 파일에 삽입됩니다.
- postfix - 각 메시지의 데이터 다음에 구분 기호가 목적지 파일에 삽입됩니다.

또한 srcmsgdelimbytes 또는 srcmsgdelimtext 속성 중 하나를 지정한 경우에만 srcmsgdelimposition 속성을 지정할 수 있습니다.

### srcmsggroups

선택사항. IBM MQ 그룹 ID별로 메시지가 그룹화되도록 지정합니다. 첫 번째 완료 그룹이 목적지 파일에 기록됩니다. 이 속성을 지정하지 않는 경우, 소스 큐의 모든 메시지가 목적지 파일에 기록됩니다. 또한 srcqueue 속성을 지정한 경우에만 srcmsggroups 속성을 지정할 수 있습니다.

### srcqueuetimeout

선택사항. 다음 조건 중 하나가 충족될 때까지 대기하는 시간(초)을 지정합니다.

- 새 메시지가 큐에 기록됨
- srcmsggroups 속성이 지정된 경우, 전체 그룹이 큐에 기록됨

이러한 조건 중 하나라도 srcqueuetimeout에서 지정한 시간 내에 충족되지 않으면, 소스 에이전트는 큐 읽기를 중지하고 전송을 완료합니다. srcqueuetimeout 속성을 지정하지 않는 경우, 소스 큐가 비어 있거나 srcmsggroups 속성을 지정했지만 큐에 전체 그룹이 없으면 소스 에이전트는 즉시 소스 큐 읽기를 중지합니다. 또한 srcqueue 속성을 지정한 경우에만 srcqueuetimeout 속성을 지정할 수 있습니다.

srcqueuetimeout 값 설정에 대한 정보는 [메시지 대 파일 전송에서 대기 시간 지정에 대한 지침](#)을 참조하십시오.

**srcrcdelimbytes**

선택사항. 여러 개의 레코드 중심 소스 파일을 하나의 2진 파일에 추가할 때 구분 기호로 삽입할 하나 이상의 바이트 값을 지정합니다. 각 값을 00-FF 범위의 16진 숫자 두개로 지정하고, x로 접두부를 지정해야 합니다. 다중 바이트는 쉼표로 분리해야 합니다. 예를 들면, 다음과 같습니다.

```
srcrcdelimbytes="x08,xA4"
```

전송 소스 파일이 레코드 지향이고(예: z/OS 데이터 세트) 목적지 파일이 레코드 지향이 아닌 정상 파일인 경우에만 srcrcdelimbytes 속성을 지정할 수 있습니다. conversion 속성에 text 값을 지정한 경우에는 srcrcdelimbytes 속성을 지정할 수 없습니다.

**srcrcdelimpos**

선택사항. 2진 구분 기호가 삽입되는 위치를 지정합니다. 올바른 값은 다음과 같습니다.

- prefix - 구분 기호가 각 소스 레코드 중심 파일 레코드의 데이터 앞 목적지 파일에 삽입됩니다.
- postfix - 구분 기호가 각 소스 레코드 중심 파일 레코드의 데이터 뒤 목적지 파일에 삽입됩니다.

또한 srcrcdelimbytes 속성을 지정한 경우에만 srcrcdelimpos 속성을 지정할 수 있습니다.

**목적지 옵션 속성****dstencoding**

선택사항. 전송된 파일에 사용하는 문자 세트 인코딩입니다.

conversion 속성이 text. 값으로 설정된 경우에만 이 속성을 지정할 수 있습니다.

dstencoding 속성이 지정되지 않은 경우 텍스트 전송에 목적지 시스템의 문자 세트가 사용됩니다.

**dsteol**

선택사항. 전송된 파일에 사용하는 행 구분 기호의 끝입니다. 올바른 값은 다음과 같습니다.

- CRLF - 캐리지 리턴 문자와 줄 바꿈 문자를 줄 끝 구분 기호로 사용합니다. 이 변환은 Windows 시스템에서 일반적입니다.
- LF - 줄 바꿈 문자를 줄 끝 구분 기호로 사용합니다. 이 변환은 UNIX 시스템에서 일반적입니다.

conversion 속성이 text. 값으로 설정된 경우에만 이 속성을 지정할 수 있습니다.

dsteol 속성을 지정하지 않으면 텍스트 전송에서 목적지 에이전트의 운영 체제를 기반으로 자동으로 올바른 값을 판별합니다.

**dstmsgdelimbytes**

선택사항. 2진 파일을 다중 메시지로 분할할 때 사용할 16진 구분 기호를 지정합니다. 모든 메시지에 동일한 IBM MQ 그룹 ID가 있습니다. 그룹의 마지막 메시지에는 IBM MQ LAST\_MSG\_IN\_GROUP 플래그 세트가 있습니다. 구분 기호로 16진수 바이트를 지정하기 위한 형식은 xNN이며, 여기서 N은 0-9 또는 a-f 범위의 문자입니다. 16진수 바이트의 쉼표로 구분된 목록(예: x3e, x20, x20, xbf)을 지정하여 구분 기호로 16진수 바이트 시퀀스를 지정할 수 있습니다.

또한 dstqueue 속성을 지정하고 전송이 2진 모드인 경우에만 dstmsgdelimbytes 속성을 지정할 수 있습니다. dstmsgsize, dstmsgdelimbytes 및 dstmsgdelimpattern 속성 중 하나만 지정할 수 있습니다.

**dstmsgdelimpattern**

선택사항. 텍스트 파일을 다중 메시지로 분할할 때 사용할 Java 정규식을 지정합니다. 모든 메시지에 동일한 IBM MQ 그룹 ID가 있습니다. 그룹의 마지막 메시지에는 IBM MQ LAST\_MSG\_IN\_GROUP 플래그 세트가 있습니다. 정규식을 구분 기호로 지정하기 위한 형식은 괄호((*regular\_expression*))로 묶이거나 큰따옴표("regular\_expression")로 묶인 정규식입니다. 자세한 정보는 [MFT에서 사용하는 정규식을 참조하십시오](#).

기본적으로 정규식이 일치시킬 수 있는 문자열의 길이는 목적지 에이전트에 의해 다섯자로 제한됩니다.

**maxDelimiterMatchLength** 에이전트 특성을 사용하여 이 동작을 변경할 수 있습니다. 자세한 정보는 [MFT 고급 에이전트 특성을 참조하십시오](#).

또한 dstqueue 속성을 지정하고 전송이 텍스트 모드인 경우에만 dstmsgdelimpattern 속성을 지정할 수 있습니다. dstmsgsize, dstmsgdelimbytes 및 dstmsgdelimpattern 속성 중 하나만 지정할 수 있습니다.



### **dstmsgdelimposition**

선택사항. 텍스트 또는 2진 구분 기호가 있을 것으로 예상되는 위치를 지정합니다. 올바른 값은 다음과 같습니다.

- **prefix** - 구분 기호가 각 행의 시작 부분에 있어야 합니다.
- **postfix** - 구분 기호가 각 행의 끝에 있어야 합니다.

dstmsgdelimpattern 속성도 지정한 경우에만 dstmsgdelimposition 속성을 지정할 수 있습니다.

### **dstmsgincludedelim**

선택사항. 파일을 다중 메시지로 분할하는 데 사용되는 구분 기호를 메시지에 포함시킬지 여부를 지정합니다. dstmsgincludedelim 속성을 지정하는 경우, 구분 기호 앞에 파일 데이터를 포함하는 메시지의 끝에 구분 기호가 포함됩니다. 기본적으로, 구분 기호는 메시지에 포함되지 않습니다. 또한 dstmsgdelimpattern 및 dstmsgdelimbytes 속성 중 하나를 지정한 경우에만 dstmsgincludedelim 속성을 지정할 수 있습니다.

### **dstmsgpersist**

선택사항. 목적지 큐에 기록되는 메시지가 지속 메시지인지 여부를 지정합니다. 올바른 값은 다음과 같습니다.

- **true** - 목적지 큐에 지속 메시지를 기록합니다. 이는 기본값입니다.
- **false** - 목적지 큐에 비지속 메시지를 기록합니다.
- **qdef** - 지속성 값은 목적지 큐의 DefPersistence 속성에서 가져옵니다.

또한 dstqueue 속성이 지정된 경우에만 이 속성을 지정할 수 있습니다.

### **dstmsgprops**

선택사항. 전송에 의해 목적지 큐에 기록된 첫 번째 메시지에 IBM MQ 메시지 특성이 설정되는지 여부를 지정합니다. 가능한 값은 다음과 같습니다.

- **true** - 전송에 의해 작성된 첫 번째 메시지에 메시지 특성을 설정합니다.
- **false** - 전송에 의해 작성된 첫 번째 메시지에 메시지 특성을 설정하지 않습니다. 이는 기본값입니다.

자세한 정보는 목적지 큐에 기록된 메시지에서 [MFT가 설정한 MQ 메시지 특성](#)을 참조하십시오.

또한 dstqueue 속성이 지정된 경우에만 이 속성을 지정할 수 있습니다.

### **dstmsgsize**

선택사항. 파일을 여러 개의 고정 길이 메시지로 분할할지 여부를 지정합니다. 모든 메시지에 동일한 IBM MQ 그룹 ID가 있습니다. 그룹의 마지막 메시지에는 IBM MQ LAST\_MSG\_IN\_GROUP 플래그 세트가 있습니다. 메시지의 크기는 dstmsgsize의 값으로 지정됩니다. dstmsgsize의 양식은 *lengthunits*입니다. 여기서 *length*는 양의 정수 값이고 *units*는 다음 값 중 하나입니다.

- **B** - 바이트. 허용되는 최소값은 목적지 메시지 코드 페이지의 문자당 최대 바이트 수의 두 배입니다.
- **K** - 키비바이트. 1024바이트와 동일합니다.
- **M** - 메비바이트. 1024키비바이트와 동일합니다.

파일이 텍스트 모드에서 전송되어 2바이트 문자 세트 또는 다중 바이트 문자 세트로 되어 있는 경우, 파일은 지정한 메시지 크기에 가장 근접한 문자 경계에서 메시지로 분할됩니다.

또한 dstqueue 속성을 지정한 경우에만 dstmsgsize 속성을 지정할 수 있습니다. dstmsgsize, dstmsgdelimbytes 및 dstmsgdelimpattern 속성 중 하나만 지정할 수 있습니다.

### **dstunsupportedcodepage**

선택사항. dstqueue 속성을 통해 지정한 대로 목적지 큐 관리자가 파일 데이터를 큐에 텍스트 전송으로 전송할 때 사용되는 코드 페이지를 지원하지 않는 경우 수행할 조치를 지정합니다. 이 속성의 올바른 값은 다음과 같습니다.

- **binary** - 전송을 계속하지만 전송 중인 데이터에 코드 페이지 변환을 적용하지 않습니다. 이 값을 지정하는 것은 conversion 속성을 text(으)로 설정하지 않는 것과 같습니다.
- **fail** - 전송 조작을 계속하지 않습니다. 파일이 전송에 실패한 것으로 레코드됩니다. 기본값입니다.

또한 dstqueue 속성을 지정하고 conversion 속성에 text 값을 지정한 경우에는 dstunsupportedcodepage 속성만 지정할 수 있습니다.

## z/OS **dsttruncaterecords**

선택사항. LRECL 데이터 세트 속성보다 긴 목적지 레코드가 잘리도록 지정합니다. true로 설정하면 레코드가 잘립니다. false로 설정하면 레코드가 줄 바꿈됩니다. 기본 설정은 false입니다. 이 매개변수는 목적지가 데이터 세트인 텍스트 모드 전송의 경우에만 유효합니다.

## 기타 속성

### checksum

선택사항. 전송된 파일 체크섬에 사용되는 알고리즘을 판별합니다.

- MD5 - MD5 해싱 알고리즘을 사용합니다.
- NONE - 체크섬 알고리즘을 사용하지 않습니다.

checksum 속성을 지정하지 않으면 MD5의 기본값이 사용됩니다.

### conversion

선택사항. 전송할 때 파일에 적용할 변환 유형을 지정합니다. 가능한 값은 다음과 같습니다.

- binary - 변환을 적용하지 않습니다.
- text - 소스 시스템과 목적지 시스템 간에 코드 페이지 변환을 적용합니다. 행 구분 기호 변환도 적용합니다. srcencoding, dstencoding, srceol 및 dsteol 속성이 적용되는 변환에 영향을 줍니다.

conversion 속성을 지정하지 않으면 binary의 기본값이 지정됩니다.

### overwrite

선택사항. 조작이 기존 목적지 파일 **z/OS** 또는 데이터 세트의 덮어쓰기 여부를 판별합니다. true 값을 지정하면 모든 기존 목적지 파일을 겹쳐 **z/OS** 쓰고 모든 기존 데이터 세트를 겹쳐씁니다. false 값을 지정하면 목적지에 중복 파일이 있 **z/OS** 거나 중복 데이터 세트가 있는 경우 조작이 실패합니다. 겹쳐쓰기 속성이 지정되지 않은 경우 기본값인 false이(가) 지정됩니다.

### recurse

선택사항. 파일 전송이 서브디렉토리에 반복되는지 여부를 판별합니다. true 값을 지정하면 전송이 서브디렉토리에 대해 반복됩니다. false 값을 지정하면 전송이 서브디렉토리에 대해 반복되지 않습니다. recurse 속성이 지정되지 않은 경우 기본값인 false이(가) 지정됩니다.

## 예

이 예제는 소스 파일이 file1.bin이고 대상 파일이 file2.bin.인 fte:filespec을 지정합니다.

```
<fte:filespec srcfilespec="/home/fteuser/file1.bin" dstfile="/home/fteuser/file2.bin"/>
```

## 관련 태스크

[MFT과\(와\) 함께 Apache Ant 사용](#)

## fte:메타데이터 Ant 중첩 요소

메타데이터는 추가 사용자 정의된 정보를 파일 전송 조작과 함께 전달하는 데 사용됩니다.

Managed File Transfer 에서 메타데이터를 사용하는 방법에 대한 자세한 정보는 [2032 페이지의 『MFT 사용자 엑시트의 메타데이터』](#) 의 내용을 참조하십시오.

## 중첩 주체

- [fte:filecopy](#) 태스크
- [fte:filemove](#) 태스크
- [fte:call](#) 태스크



## 중첩 요소로 지정되는 매개변수

### fte:entry

fte:metadata 중첩 요소 내에 하나 이상의 항목을 지정해야 합니다. 둘 이상의 입력 항목을 지정할 수도 있습니다. 입력 항목은 키 이름과 값을 연관시킵니다. 키는 fte:metadata 블록에서 고유해야 합니다.

### 입력 항목 속성

#### name

필수입니다. 이 입력 항목에 속하는 키의 이름입니다. 이 이름은 fte:metadata 요소 내부에 중첩된 모든 entry 매개변수에서 고유해야 합니다.

#### value

필수입니다. 이 entry에 지정할 값입니다.

### 예

이 예에서는 두 개의 항목을 포함하는 fte:metadata 정의를 표시합니다.

```
<fte:metadata>
  <fte:entry name="org.foo.partColor" value="red"/>
  <fte:entry name="org.foo.partSize" value="medium"/>
</fte:metadata>
```

### 관련 태스크

[MFT과\(와\) 함께 Apache Ant 사용](#)

## 프로그램 호출 중첩 요소

다섯 개의 중첩 요소(fte:presrc, fte:predst, fte:postdst, fte:postsrc 및 fte:command) 중 하나를 사용하여 프로그램을 시작할 수 있습니다. 이 중첩 요소는 처리의 일부로 외부 프로그램을 호출하도록 에이전트에 지시합니다. 프로그램을 시작하기 전에 명령이 명령을 실행하는 에이전트의 agent.properties 파일에 있는 commandPath 특성에 지정된 위치에 있는지 확인해야 합니다.

각 프로그램 호출 요소는 이름이 서로 다르더라도 동일한 속성과 중첩 요소 설정을 공유합니다. 프로그램은 **fte:filecopy**, **fte:filemove** 및 **fte:command** Ant 태스크로 시작할 수 있습니다.

Connect:Direct® 브릿지 에이전트에서 프로그램을 호출할 수 없습니다.

## 프로그램을 호출할 수 있는 Ant 태스크

- **fte:filecopy** 태스크는 fte:predst, fte:postdst, fte:presrc 및 fte:postsrc 중첩 요소를 사용하여 프로그램 호출 매개변수를 중첩합니다.
- **fte:filemove** 태스크는 fte:predst, fte:postdst, fte:presrc 및 fte:postsrc 중첩 요소를 사용하여 프로그램 호출 매개변수를 중첩합니다.
- **fte:call** 태스크는 fte:command 중첩 요소를 사용하여 프로그램 호출 매개변수를 중첩합니다.

## 속성

### 명령

필수입니다. 호출할 프로그램을 지정합니다. 에이전트가 명령을 실행할 수 있게 하려면 명령이 에이전트의 agent.properties 파일에 있는 commandPath 특성에 지정된 위치에 있어야 합니다. 자세한 정보는 [commandPath MFT 특성을 참조하십시오](#). command 속성에 지정된 경로 정보는 commandPath 특성에 지정된 위치에 상대적인 것으로 간주됩니다. type이(가) executable인 경우 실행 가능 프로그램이 필요하고 그렇지 않으면 호출 유형에 적합한 스크립트가 필요합니다.

### retrycount

선택사항. 프로그램이 성공 리턴 코드를 리턴하지 않은 경우 프로그램 호출을 재시도하는 횟수입니다. command 속성에 의해 이름 지정된 프로그램이 이 횟수만큼 호출됩니다. 이 속성에 지정된 값은 음수가 아니어야 합니다. retrycount 속성을 지정하지 않으면 기본값 0이 사용됩니다.

## retrywait

선택사항. 프로그램 호출을 재시도하기 전에 대기하는 시간(초)입니다. `command` 속성에 의해 이름 지정된 프로그램이 성공 리턴 코드를 리턴하지 않고 `retrycount` 속성이 0이 아닌 값을 지정하는 경우, 이 매개변수는 재시도 사이의 대기 시간을 결정합니다. 이 속성에 지정된 값은 음수가 아니어야 합니다. `retrywait` 속성을 지정하지 않으면 기본값 0이 사용됩니다.

## successrc

선택사항. 이 속성의 값은 프로그램 호출이 성공적으로 실행되는 시기를 판별하는 데 사용됩니다. 이 표현식을 사용하여 명령의 프로세스 리턴 코드를 평가합니다. 값은 부울 OR을 나타내는 세로 막대 문자(|) 또는 부울 AND를 나타내는 앰퍼샌드(&) 문자와 결합된 하나 이상의 표현식으로 구성될 수 있습니다. 각 표현식은 다음 유형의 표현식 중 하나입니다.

- 프로세스 리턴 코드와 숫자 사이에 동등 테스트를 표시하는 숫자
- 숫자와 프로세스 리턴 코드 사이에 보다 큼(greater-than) 테스트를 표시하기 위해 ">" 문자가 앞에 붙은 숫자
- 숫자와 프로세스 리턴 코드 사이에 미만 테스트를 표시하기 위해보다 "<" 문자가 접두부에 사용된 숫자.
- "!=" 문자가 접두부로 오는 숫자는 숫자와 프로세스 리턴 코드 사이의 같지 않음(not-equal-to) 테스트를 나타냅니다.

예를 들어, `>2&<7&!5|0|14`은(는) 리턴 코드 0, 3, 4, 6, 14가 성공한 것으로 해석됩니다. 다른 모든 리턴 코드는 실패로 해석됩니다. `successrc` 속성을 지정하지 않으면 기본값 0이 사용됩니다. 이것은 코드 0을 리턴하는 경우에만 명령이 성공적으로 실행된 것으로 평가됨을 의미합니다.

## 유형

선택사항. 이 속성의 값은 호출되는 프로그램 유형을 지정합니다. 다음 옵션 중 하나를 지정합니다.

### executable

태스크는 실행 가능 프로그램을 호출합니다. `arg` 중첩 요소를 사용하여 추가 인수가 지정되도록 할 수 있습니다. 프로그램은 `commandPath`에서 액세스할 수 있으며 적용 가능한 경우 실행 권한 세트를 가지고 있을 것으로 예상됩니다. UNIX 스크립트는 셸 프로그램을 지정하는 한 호출될 수 있습니다(예: 셸 스크립트 파일의 첫 번째 행은 `#!/bin/sh`임). `stderr` 또는 `stdout`에 기록되는 명령 출력은 호출에 대한 Managed File Transfer 로그로 전송됩니다. 하지만 데이터 출력량은 에이전트 구성에 의해 제한됩니다. 기본값은 10KB의 데이터이지만 에이전트 특성 `maxCommandOutput`을 사용하여 이 기본값을 대체할 수 있습니다.

### antscript

태스크는 `fteAnt` 명령을 사용하여 지정된 Ant 스크립트를 실행합니다. 특성은 `property` 중첩 요소를 사용하여 지정할 수 있습니다. Ant 대상은 `target` 중첩 요소를 사용하여 지정할 수 있습니다. Ant 스크립트는 `commandPath`에서 액세스 가능할 것으로 예상됩니다. `stderr` 또는 `stdout`에 기록되는 Ant 출력은 호출을 위해 Managed File Transfer 로그에 송신됩니다. 하지만 데이터 출력량은 에이전트 구성에 의해 제한됩니다. 기본값은 10KB의 데이터이지만 에이전트 특성 `maxCommandOutput`을 사용하여 이 기본값을 대체할 수 있습니다.

### z/OS jcl

`jcl` 값은 z/OS에서만 지원되며 지정된 z/OS JCL 스크립트를 실행합니다. JCL은 작업으로 제출되므로 작업 카드가 있어야 합니다. 작업이 정상적으로 제출된 경우, Managed File Transfer 로그에 기록되는 JCL 명령 출력에는 `JOB job_name(job_id)` 텍스트가 포함됩니다. 여기서,

- `job_name`은 JCL에서 작업 카드로 식별되는 작업의 이름입니다.
- `job_id`는 z/OS 시스템 생성 작업 ID입니다.

작업을 성공적으로 제출할 수 없는 경우에는 JCL 스크립트 명령이 실패하고 실패 이유(예: 작업 카드가 없음)를 표시하는 메시지를 로그에 기록합니다. 작업이 실행되었거나 정상적으로 완료되었는지 여부를 알려면 SDSF와 같은 시스템 서비스를 사용하십시오. Managed File Transfer는 작업을 제출하기만 하기 때문에 정보를 제공하지 않으므로, 시스템이 작업 실행 시기와 작업 출력 표시 방법을 판별합니다. JCL 스크립트는 일괄처리 작업으로 제출되므로 작업이 성공적으로 제출되었다는 것만 알고 전송이 시작되기 전에 성공적으로 실행되었는지 여부는 알지 못하기 때문에 `presrc` 또는 `predst` 중첩 요소에 대해 `jcl`을(를) 지정하는 것은 권장되지 않습니다. `jcl` 유형에 유효한 중첩 요소가 없습니다.

다음 예는 JCL 작업을 보여줍니다.

```

//MYJOB JOB
//*
//MYJOB EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=H
//SYSUT1 DD DSN=FRED.DEMO.TXT,DISP=SHR
//SYSUT2 DD DSN=BOB.DEMO.TXT,DISP=(NEW,CATLG),
// RECFM=VB,LRECL=133,BLKSIZE=2048,
// SPACE=(TRK,(30,5),RLSE)
//SYSIN DD DUMMY

```

## 중첩 요소로 지정되는 매개변수

### fte:arg

type 속성의 값이 executable인 경우에만 유효합니다. 중첩된 fte:arg 요소를 사용하여 프로그램 호출의 일부로 호출되는 프로그램에 인수를 지정하십시오. 프로그램 인수는 fte:arg 요소가 발생하는 순서로 fte:arg 요소에 지정된 값에서 빌드됩니다. 0개 이상의 fte:arg 요소를 프로그램 호출의 중첩 요소로 지정하도록 선택할 수 있습니다.

### fte:property

type 속성의 값이 antscript인 경우에만 유효합니다. 중첩된 fte:property 요소의 name 및 value 속성을 사용하여 이름-값 쌍을 Ant 스크립트에 전달하십시오. 0개 이상의 property 요소를 프로그램 호출의 중첩 요소로 지정하도록 선택할 수 있습니다.

### fte:target

type 속성의 값이 antscript인 경우에만 유효합니다. Ant 스크립트에서 호출할 대상을 지정하십시오. 0개 이상의 fte:target 요소를 프로그램 호출의 중첩 요소로 지정하도록 선택할 수 있습니다.

## Arg 속성

### 값

필수입니다. 호출 중인 프로그램에 전달할 인수의 값입니다.

## 특성 속성

### 이름

필수입니다. Ant 스크립트에 전달할 특성의 이름입니다.

### 값

필수입니다. Ant 스크립트에 전달되는 특성 이름과 연관시킬 값입니다.

## 예

이 예제는 fte:filecopy 태스크의 일부로 지정되는 fte:postsrc 프로그램 호출을 보여줍니다. 프로그램 호출은 post.sh(이)라는 프로그램에 대해 수행하며 /home/fteuser2/file.bin.의 단일 인수가 제공됩니다.

```

<fte:filecopy
cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
src="agent1@qm1" dst="agent2@qm2"
rcproperty="copy.result">
<fte:filespec srcfilespec="/home/fteuser1/file.bin" dstfile="/home/fteuser2/file.bin"/>
<fte:postsrc command="post.sh" successsrc="1" >
  <fte:arg value="/home/fteuser2/file.bin"/>
</fte:postsrc>
</fte:filecopy>

```

이 예제는 fte:call 태스크의 일부로 지정되는 fte:command 프로그램 호출을 보여줍니다. 프로그램 호출은 명령행 인수를 전달하지 않는 command.sh(이)라는 실행 파일에 대해 수행됩니다. command.sh(이)가 성공 리턴 코드 1을 리턴하지 않으면 30초 후에 명령을 다시 시도합니다.

```

<fte:call
cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
agent="agent1@qm1"
rcproperty="call.rc"

```

```
origuser="bob"
jobname="${job.id}">
<fte:command command="command.sh" successsrc="1" retrycount="5" retrywait="30"/>
</fte:call>
```

이 예제는 `fte:call` 태스크의 일부로 지정되는 `fte:command` 프로그램 호출을 보여줍니다. 프로그램 호출은 두 개의 특성이 전달되는 `script.xml`(이)라는 Ant 스크립트의 복사 및 압축 대상에 대해 수행됩니다.

```
<fte:call
cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
agent="agent1@qm1"
rcproperty="call.rc"
origuser="bob"
jobname="${job.id}">
<fte:command command="script.xml" type="antscript">
<property name="src" value="AGENT5@QM5"/>
<property name="dst" value="AGENT3@QM3"/>
<target name="copy"/>
<target name="compress"/>
</fte:command>
</fte:call>
```

### 관련 태스크

[MFT와 함께 실행할 프로그램 지정](#)

[MFT과\(와\) 함께 Apache Ant 사용](#)

## 사용자 정의 참조용 MFT 사용자 엑시트

Managed File Transfer에 대해 사용자 엑시트를 구성하는 데 도움이 되는 참조 정보입니다.

### 관련 개념

[MFT 소스 및 목적지 사용자 엑시트](#)

## MFT 사용자 엑시트의 메타데이터

Managed File Transfer의 사용자 엑시트 루틴에 제공할 수 있는 세 가지 메타데이터 유형(환경, 전송 및 파일 메타데이터)이 있습니다. 이 메타데이터는 Java 키-값 쌍의 맵으로 제공됩니다.

### 환경 메타데이터

환경 메타데이터는 사용자 엑시트 루틴 모두에 전달되며 사용자 엑시트 루틴이 호출되는 에이전트의 런타임 환경을 설명합니다. 이 메타데이터는 읽기 전용이며 사용자 엑시트 루틴으로 업데이트할 수 없습니다.

표 882. 환경 메타데이터	
키	설명
AGENT_CONFIGURATION_DIRECTORY_KEY	에이전트의 구성 정보가 들어 있는 디렉토리의 이름입니다.
AGENT_PRODUCT_DIRECTORY_KEY	에이전트 코드가 설치된 디렉토리의 이름입니다.
AGENT_VERSION_KEY	엑시트 루틴을 호출하는 에이전트 런타임의 버전 번호입니다.

표 1에 지정된 키 이름 및 값 이름은 EnvironmentMetaDataConstants 인터페이스에 정의된 상수입니다.

### 전송 메타데이터

전송 메타데이터는 사용자 엑시트 루틴 모두에 전달됩니다. 메타데이터는 시스템 제공 값 및 사용자 제공 값으로 구성됩니다. 시스템 제공 값을 변경하는 경우 해당 변경사항은 무시됩니다. 소스 전송 시작 사용자 엑시트의 초기 사용자 제공 값은 전송 정의 시 제공한 값을 기반으로 합니다. 소스 에이전트는 소스 전송 시작 사용자 엑시트 처리 과정에서 사용자 제공 값을 변경할 수 있습니다. 이 사용자 엑시트는 전체 파일 전송이 시작되기 전에 호출됩니다. 이러한 변경사항은 해당 전송과 관련한 기타 엑시트 루틴에 대한 연속 호출에 사용됩니다. 전송 메타데이터는 전체 전송에 적용됩니다.

모든 사용자 엑시트가 전송 메타데이터에서 값을 읽을 수 있지만, 소스 전송 시작의 사용자 엑시트만 전송 메타데이터를 변경할 수 있습니다.

서로 다른 파일 전송 사이에서 정보를 전달하는 경우에는 전송 메타데이터를 사용할 수 없습니다.

시스템 제공 전송 메타데이터에 대해서는 표 2에 자세히 설명되어 있습니다.

표 883. 전송 메타데이터	
키	설명
DESTINATION_AGENT_KEY	전송의 목적지인 에이전트의 이름
JOB_NAME_KEY	전송 요청과 연관된 작업 이름
MQMD_USER_KEY	전송 요청을 제출하는 데 사용되는 메시지의 MQMD 사용자 필드
ORIGINATING_HOST_KEY	전송 요청에서 생성 호스트 이름으로 지정된 호스트 이름
ORIGINATING_USER_KEY	전송 요청에서 생성 사용자 ID로 지정된 사용자 이름
SOURCE_AGENT_KEY	전송의 소스인 에이전트의 이름
TRANSFER_ID_KEY	전송 ID

표 2에 지정된 키 이름 및 값 이름은 TransferMetaDataConstants 인터페이스에 정의된 상수입니다.

## 파일 메타데이터

파일 메타데이터는 파일 스펙의 일부로 소스 전송 시작 엑시트에 전달됩니다. 소스 및 목적지 파일에 대한 개별 파일 메타데이터가 있습니다.

서로 다른 파일 전송 사이에서 정보를 전달하는 경우에는 파일 메타데이터를 사용할 수 없습니다.

표 884. 파일 메타데이터		
키	허용 값	설명
CONVERT_LINE_SEPARATORS		소스 데이터 내의 캐리지 리턴 줄 바꾸기(CRLF) 또는 줄 바꾸기(LF) 행 구분 기호 시퀀스가 대상에서 행 구분 기호 시퀀스로 변경되는지를 나타내기 위해 텍스트 전송에서 사용되는 키 값입니다.
DELIMITER_KEY		레코드 지향 데이터를 보통 파일에 전송할 때 레코드 데이터를 구분하는 데 필요한 구분 기호를 정의하는 데 사용되는 키 값입니다.  메시지에서 파일로 및 파일에서 메시지로 전송에도 사용됩니다.
DELIMITER_POSITION_KEY	DELIMITER_POSITION_PREFIX_VALUE DELIMITER_POSITION_POSTFIX_VALUE	구분 기호의 위치를 접두부 또는 접미부로 정의하려면 DELIMITER_KEY와 함께 사용하십시오.
DELIMITER_TYPE_KEY	DELIMITER_TYPE_BINARY_VALUE DELIMITER_TYPE_TEXT_VALUE DELIMITER_TYPE_SIZE_VALUE	구분 기호의 유형을 정의하려면 DELIMITER_KEY와 함께 사용하십시오.
DESTINATION_EXIST_KEY	DESTINATION_EXIST_KEY_ERROR_VALUE DESTINATION_EXIST_KEY_OVERWRITE_VALUE	목적지 파일이 존재하는 경우 파일 전송 작업을 판별합니다.
FILE_ALIAS_KEY		전송되는 파일에 대한 별명을 정의하는 데 사용되는 키 값입니다.
FILE_CHECKSUM_METHOD_KEY	FILE_CHECKSUM_METHOD_NONE_VALUE FILE_CHECKSUM_METHOD_MD5_VALUE	파일 전송 시 체크섬 메소드를 판별합니다.
FILE_CONVERSION_KEY	FILE_CONVERSION_TEXT_VALUE FILE_CONVERSION_BINARY_VALUE	파일 콘텐츠에 적용된 변환 유형을 판별합니다.

표 884. 파일 메타데이터 (계속)		
키	허용 값	설명
FILE_ENCODING_KEY		텍스트 파일에 사용되는 인코딩을 판별합니다.
FILE_END_OF_LINE_KEY	FILE_END_OF_LINE_LF_VALUE FILE_END_OF_LINE_CRLF_VALUE	행의 끝을 표시하는 문자 순서인 <LF> 또는 <CR><LF>를 판별합니다.
FILE_SPACE_ALIAS		파일 공간에 있는 파일의 알리아스를 판별합니다.  <b>참고:</b> 이 메타데이터는 FILE_TYPE_KEY가 FILE_TYPE_FILE_SPACE_VALUE인 경우에만 사용할 수 있습니다.
FILE_SPACE_NAME		파일 공간의 이름을 판별합니다.  <b>참고:</b> 이 메타데이터는 FILE_TYPE_KEY가 FILE_TYPE_FILE_SPACE_VALUE인 경우에만 사용할 수 있습니다.
FILE_TYPE_KEY	FILE_TYPE_FILE_VALUE FILE_TYPE_DIRECTORY_VALUE FILE_TYPE_DATASET_VALUE FILE_TYPE_PDS_VALUE FILE_TYPE_QUEUE_VALUE FILE_TYPE_FILE_SPACE_VALUE	목적지 파일, 큐 또는 파일 공간 스펙을 판별합니다.
GROUP_ID_KEY		메시지에서 파일로 전송에서 소스 큐에서 읽을 메시지 그룹을 결정하기 위해 사용되는 키 값입니다. 이 속성은 USE_GROUPS_KEY의 값이 USE_GROUPS_TRUE_VALUE인 경우에만 유효합니다.
INCLUDE_DELIMITER_IN_MESSAGE_KEY	INCLUDE_DELIMITER_IN_MESSAGE_TRUE_VALUE INCLUDE_DELIMITER_IN_MESSAGE_FALSE_VALUE	파일에서 메시지로 전송에서 파일을 여러 메시지로 분할할 때 사용된 구분 기호를 메시지의 끝에 포함시킬지 결정하기 위해 사용되는 키 값입니다. 이 속성은 DELIMITER_TYPE_KEY의 값이 DELIMITER_TYPE_BINARY_VALUE DELIMITER_TYPE_TEXT_VALUE인 경우에만 유효합니다.
INSERT_RECORD_LINE_SEPARATOR_KEY		레코드 지향 파일로부터의 텍스트 전송에서 각 레코드 뒤의 데이터에 행 구분 기호를 삽입할지 지정하는 데 사용되는 키 값입니다.
KEEP_TRAILING_SPACES_KEY	KEEP_TRAILING_SPACES_TRUE_VALUE KEEP_TRAILING_SPACES_FALSE_VALUE	고정 길이 형식 데이터 세트에서 읽은 레코드에서 후미 공백을 제거할지 결정하는 데 사용되는 키 값입니다.
NEW_RECORD_ON_LINE_SEPARATOR_KEY		레코드 지향 파일로의 텍스트 전송에서 데이터 내의 행 구분 기호를 레코드 데이터에 포함시킬지, 아니면 새 레코드를 작성할지 지정하는 데 사용되는 키 값입니다.
PERSISTENT_KEY	PERSISTENT_TRUE_VALUE PERSISTENT_FALSE_VALUE PERSISTENT_QDEF_VALUE	파일에서 메시지로 전송에서 메시지가 지속적인지 결정하는 데 사용되는 키 값입니다.
SET_MQ_PROPS_KEY	SET_MQ_PROPS_TRUE_VALUE SET_MQ_PROPS_FALSE_VALUE	파일에서 메시지로 전송에서 IBM MQ 메시지 특성을 파일의 첫 번째 메시지에 설정할지, 그리고 오류가 발생하면 이 큐에 메시지를 기록할지 결정하는 데 사용되는 키 값입니다.

표 884. 파일 메타데이터 (계속)		
키	허용 값	설명
UNRECOGNISED_CODE_PAGE_KEY	UNRECOGNISED_CODE_PAGE_FAIL_VALUE UNRECOGNISED_CODE_PAGE_BINARY_VALUE	파일에서 메시지로 전송 시 목적지 큐 관리자가 데이터의 코드 페이지를 인식하지 못하는 경우에 텍스트 모드 전송이 실패하는지 또는 변환이 수행되는지 판별하는 데 사용되는 키 값입니다.
USE_GROUPS_KEY	USE_GROUPS_TRUE_VALUE USE_GROUPS_FALSE_VALUE	메시지에서 파일로 전송에서, 소스 큐로부터 완료 메시지 그룹만 전송할지 결정하는 데 사용되는 키 값입니다.
WAIT_TIME_KEY		메시지에서 파일로 전송에서 다음과 같은 경우에 소스 에이전트가 대기하는 시간(초)를 결정하는 데 사용되는 키 값입니다. <ul style="list-style-type: none"> <li>소스 큐에 메시지가 나타났을 때, 큐가 비어있거나 비워졌으며 USE_GROUPS_KEY의 값이 FALSE 인 경우</li> <li>소스 큐에 완료 그룹이 나타났을 때, USE_GROUPS_KEY의 값이 TRUE 인 경우</li> </ul>

표 3에 지정된 키 이름 및 값 이름은 FileMetaDataConstants 인터페이스에 정의된 상수입니다.

#### 관련 개념

2041 페이지의 『MFT 사용자 엑시트에 대한 Java 인터페이스』

사용자 엑시트 루틴 관련 Java 인터페이스에 대한 참조 정보는 이 절의 주제를 사용하십시오.

#### 관련 태스크

사용자 엑시트를 사용하여 MFT 사용자 정의

#### 관련 참조

2035 페이지의 『MFT 자원 모니터 사용자 엑시트』

자원 모니터 사용자 엑시트를 사용하면 모니터의 트리거 조건이 충족될 때 연관된 태스크가 시작되기 전에 실행할 사용자 정의 코드를 구성할 수 있습니다.

2039 페이지의 『사용자 엑시트에 대한 MFT 에이전트 특성』

agent.properties 파일의 표준 특성 외에도 사용자 엑시트 루틴에 고유한 여러 고급 특성이 있습니다. 이 특성은 기본적으로 포함되지 않으므로 이 특성을 사용하려면 agent.properties 파일을 수동으로 편집해야 합니다. 해당 에이전트가 실행 중인 동안 agent.properties 파일을 변경하는 경우 에이전트를 중지하고 다시 시작하여 변경사항을 적용하십시오.

### MFT 자원 모니터 사용자 엑시트

자원 모니터 사용자 엑시트를 사용하면 모니터의 트리거 조건이 충족될 때 연관된 태스크가 시작되기 전에 실행할 사용자 정의 코드를 구성할 수 있습니다.

사용자 엑시트 코드에서 직접 새 전송을 호출하는 것은 좋지 않습니다. 사용자 엑시트는 에이전트 다시 시작에 대해 복원력이 없기 때문에 일부 상황에서 이는 파일의 복수 전송을 발생시킬 수 있습니다.

자원 모니터 사용자 엑시트는 사용자 엑시트에 대한 기존 인프라를 사용합니다. 모니터 사용자 엑시트는 모니터가 트리거된 후, 해당되는 태스크가 모니터 태스크에 의해 실행되기 전에 호출됩니다. 그러면 사용자 엑시트는 실행할 태스크를 수정하고 태스크를 진행해야 하는지 여부를 결정할 수 있습니다. 모니터 메타데이터를 업데이트하여 모니터 태스크를 수정할 수 있습니다. 그런 다음 원래 모니터 작성에 의해 작성된 태스크 문서에서 변수 대체에 사용됩니다. 또한 모니터 엑시트는 매개변수로 전달되는 태스크 정의 XML 문자열을 바꾸거나 업데이트할 수도 있습니다. 모니터 엑시트는 태스크에 대한 '진행' 또는 '취소'의 결과 코드를 리턴할 수 있습니다. 취소가 리턴되는 경우, 태스크는 시작되지 않고 모니터는 모니터링할 자원이 트리거 조건과 일치할 때까지 다시 시작하지 않습니다. 자원이 변경되지 않으면 트리거는 시작하지 않습니다. 다른 사용자 엑시트와 같이, 모니터 엑시트를 함께



체인으로 연결할 수 있습니다. 엑시트 중 하나가 취소 결과 코드를 리턴하는 경우 전체 결과는 취소이고 태스크는 시작하지 않습니다.

- 환경 메타데이터의 맵(다른 사용자 엑시트와 같음)
- 변하지 않는 시스템 메타데이터와 변하기 쉬운 사용자 메타데이터를 포함한 모니터 메타데이터의 맵. 변하지 않는 시스템 메타데이터는 다음과 같습니다.
  - FILENAME - 트리거 조건을 충족한 파일의 이름
  - FILEPATH - 트리거 조건을 충족한 파일의 경로
  - FILESIZE(바이트 단위 - 이 메타데이터는 존재하지 않을 수도 있음) - 트리거 조건을 충족한 파일의 크기
  - LASTMODIFIEDDATE(로컬) - 트리거 조건을 충족한 파일이 마지막으로 변경된 날짜. 이 날짜는 에이전트가 실행 중인 시간대의 로컬 날짜로 표현되고 ISO 8601 날짜로 양식화됩니다.
  - LASTMODIFIEDTIME(로컬) - 트리거 조건을 충족한 파일이 마지막으로 변경된 로컬 양식 시간. 이 시간은 에이전트가 실행 중인 시간대의 로컬 시간으로 표현되고 ISO 8601 시간으로 양식화됩니다.
  - LASTMODIFIEDDATEUTC - 트리거 조건을 충족한 파일이 마지막으로 변경된 범용 양식 날짜. 이 날짜는 UTC 시간대로 변환된 로컬 날짜로 표현되고 ISO 8601 날짜로 양식화됩니다.
  - LASTMODIFIEDTIMEUTC - 트리거 조건을 충족한 파일이 마지막으로 변경된 범용 양식 시간. 이 시간은 UTC 시간대로 변환된 로컬 시간으로 표현되고 ISO 8601 시간으로 양식화됩니다.
  - AGENTNAME - 모니터 에이전트 이름
- 모니터 트리거의 결과로 실행할 태스크를 표시하는 XML 문자열

모니터 엑시트는 다음 데이터를 리턴합니다.

- 추가 진행 여부(진행 또는 취소)를 지정하는 표시기
- 트리거 충족 로그 메시지에 삽입할 문자열

모니터 엑시트 코드의 실행 결과로, 원래 매개변수로 전달된 모니터 메타데이터 및 태스크 정의 XML 문자열도 업데이트될 수 있습니다.

에이전트 특성 `monitorExitClasses(agent.properties` 파일에 포함)의 값은 로드할 모니터 엑시트 클래스를 지정하며 각 엑시트 클래스는 심볼로 구분됩니다. 예를 들면, 다음과 같습니다.

```
monitorExitClasses=testExits.TestExit1,testExits.testExit2
```

모니터 사용자 엑시트에 대한 인터페이스는 다음과 같습니다.

```
package com.ibm.wmqfte.exitroutine.api;

import java.util.Map;

/**
 * An interface that is implemented by classes that want to be invoked as part of
 * user exit routine processing. This interface defines a method that will be
 * invoked immediately prior to starting a task as the result of a monitor trigger
 */
public interface MonitorExit {

    /**
     * Invoked immediately prior to starting a task as the result of a monitor
     * trigger.
     *
     * @param environmentMetaData
     *      meta data about the environment in which the implementation
     *      of this method is running. This information can only be read,
     *      it cannot be updated by the implementation. The constant
     *      defined in <code>EnvironmentMetaDataConstants</code> class can
     *      be used to access the data held by this map.
     *
     * @param monitorMetaData
     *      meta data to associate with the monitor. The meta data passed
     *      to this method can be altered, and the changes will be
     */
}
```



```

*           reflected in subsequent exit routine invocations. This map
*           also contains keys with IBM reserved names. These entries are
*           defined in the <code>MonitorMetaDataConstants</code> class and
*           have special semantics. The the values of the IBM reserved names
*           cannot be modified by the exit
*
* @param taskDetails
*           An XML String representing the task to be executed as a result of
*           the monitor triggering. This XML string may be modified by the
*           exit
*
* @return   a monitor exit result object which is used to determine if the
*           task should proceed, or be cancelled.
*/
MonitorExitResult onMonitor(Map<String, String> environmentMetaData,
                           Map<String, String> monitorMetaData,
                           Reference<String> taskDetails);
}

```

모니터 메타데이터에서 IBM 예약 값에 대한 상수는 다음과 같습니다.

```

package com.ibm.wmqfte.exitroutine.api;

/**
 * Constants for IBM reserved values placed into the monitor meta data
 * maps used by the monitor exit routines.
 */
public interface MonitorMetaDataConstants {

    /**
     * The value associated with this key is the name of the trigger
     * file associated with the monitor. Any modification performed
     * to this property by user exit routines will be ignored.
     */
    final String FILE_NAME_KEY = "FILENAME";

    /**
     * The value associated with this key is the path to the trigger
     * file associated with the monitor. Any modification performed
     * to this property by user exit routines will be ignored.
     */
    final String FILE_PATH_KEY = "FILEPATH";

    /**
     * The value associated with this key is the size of the trigger
     * file associated with the monitor. This will not be present in
     * the cases where the size cannot be determined. Any modification
     * performed to this property by user exit routines will be ignored.
     */
    final String FILE_SIZE_KEY = "FILESIZE";

    /**
     * The value associated with this key is the local date on which
     * the trigger file associated with the monitor was last modified.
     * Any modification performed to this property by user exit routines
     * will be ignored.
     */
    final String LAST_MODIFIED_DATE_KEY = "LASTMODIFIEDDATE";

    /**
     * The value associated with this key is the local time at which
     * the trigger file associated with the monitor was last modified.
     * Any modification performed to this property by user exit routines
     * will be ignored.
     */
    final String LAST_MODIFIED_TIME_KEY = "LASTMODIFIEDTIME";

    /**
     * The value associated with this key is the UTC date on which
     * the trigger file associated with the monitor was last modified.
     * Any modification performed to this property by user exit routines
     * will be ignored.
     */
    final String LAST_MODIFIED_DATE_KEY_UTC = "LASTMODIFIEDDATEUTC";

    /**
     * The value associated with this key is the UTC time at which
     * the trigger file associated with the monitor was last modified.
     */
}

```

```

* Any modification performed to this property by user exit routines
* will be ignored.
*/
final String LAST_MODIFIED_TIME_KEY_UTC = "LASTMODIFIEDTIMEUTC";

/**
* The value associated with this key is the name of the agent on which
* the monitor is running. Any modification performed to this property by
* user exit routines will be ignored.
*/
final String MONITOR_AGENT_KEY = "AGENTNAME";
}

```

## 모니터 사용자 엑시트 예제

이 예제 클래스는 MonitorExit 인터페이스를 구현합니다. 이 예제는 사용자 정의 대체 변수를 *REDIRECTEDAGENT*라는 모니터 메타데이터에 추가합니다. 이 변수는 시간이 홀수인 경우 PARIS 값, 짝수 시간인 경우 LONDON 값으로 채워집니다. 모니터 엑시트 결과 코드는 항상 proceed을(를) 리턴하도록 설정됩니다.

```

package com.ibm.wmqfte.monitor;

import java.util.Calendar;
import java.util.Map;

import com.ibm.wmqfte.exitroutine.api.MonitorExit;
import com.ibm.wmqfte.exitroutine.api.MonitorExitResult;
import com.ibm.wmqfte.exitroutine.api.Reference;

/**
* Example resource monitor user exit that changes the monitor mutable
* metadata value between 'LONDON' and 'PARIS' depending on the hour of the day.
*
*/
public class TestMonitorExit implements MonitorExit {

    // custom variable that will substitute destination agent
    final static String REDIRECTED_AGENT = "REDIRECTEDAGENT";

    public MonitorExitResult onMonitor(
        Map<String, String> environmentMetaData,
        Map<String, String> monitorMetaData,
        Reference<String> taskDetails) {

        // always succeed
        final MonitorExitResult result = MonitorExitResult.PROCEED_RESULT;

        final int hour = Calendar.getInstance().get(Calendar.HOUR_OF_DAY);

        if (hour%2 == 1) {
            monitorMetaData.put(REDIRECTED_AGENT, "LONDON");
        } else {
            monitorMetaData.put(REDIRECTED_AGENT, "PARIS");
        }

        return result;
    }
}

```

*REDIRECTEDAGENT* 대체 변수를 사용하는 모니터에 해당하는 태스크는 다음과 비슷하게 표시됩니다.

```

<?xml version="1.0" encoding="UTF-8"?>
<request version="4.00"
  xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="FileTransfer.xsd">
  <managedTransfer>
    <originator>
      <hostName>reportserver.com</hostName>
      <userID>USER1</userID>
    </originator>
    <sourceAgent agent="AGENT1"
      QMgr="QM1"/>
    <destinationAgent agent="{REDIRECTEDAGENT}"
      QMgr="QM2"/>
    <transferSet>
      <item mode="binary" checksumMethod="MD5">

```

```

<source recursive="false" disposition="delete">
  <file>c:\sourcefiles\reports.doc</file>
</source>
<destination type="file" exist="overwrite">
  <file>c:\destinationfiles\reports.doc</file>
</destination>
</item>
</transferSet>
</managedTransfer>
</request>

```

이 전송이 시작되기 전에 <destinationAgent> 요소의 에이전트 속성 값이 LONDON 또는 PARIS(으)로 바뀝니다. 태스크 정의 XML 및 모니터 엑시트 클래스의 대체 변수는 대문자로 지정해야 합니다.

### 관련 개념

사용자 엑시트를 사용하여 MFT 사용자 정의

2032 페이지의 『MFT 사용자 엑시트의 메타데이터』

Managed File Transfer의 사용자 엑시트 루틴에 제공할 수 있는 세 가지 메타데이터 유형(환경, 전송 및 파일 메타데이터)이 있습니다. 이 메타데이터는 Java 키-값 쌍의 맵으로 제공됩니다.

2041 페이지의 『MFT 사용자 엑시트에 대한 Java 인터페이스』

사용자 엑시트 루틴 관련 Java 인터페이스에 대한 참조 정보는 이 절의 주제를 사용하십시오.

MFT 소스 및 목적지 사용자 엑시트

### 관련 참조

2039 페이지의 『사용자 엑시트에 대한 MFT 에이전트 특성』

agent.properties 파일의 표준 특성 외에도 사용자 엑시트 루틴에 고유한 여러 고급 특성이 있습니다. 이 특성은 기본적으로 포함되지 않으므로 이 특성을 사용하려면 agent.properties 파일을 수동으로 편집해야 합니다. 해당 에이전트가 실행 중인 동안 agent.properties 파일을 변경하는 경우 에이전트를 중지하고 다시 시작하여 변경사항을 적용하십시오.

## 사용자 엑시트에 대한 MFT 에이전트 특성

agent.properties 파일의 표준 특성 외에도 사용자 엑시트 루틴에 고유한 여러 고급 특성이 있습니다. 이 특성은 기본적으로 포함되지 않으므로 이 특성을 사용하려면 agent.properties 파일을 수동으로 편집해야 합니다. 해당 에이전트가 실행 중인 동안 agent.properties 파일을 변경하는 경우 에이전트를 중지하고 다시 시작하여 변경사항을 적용하십시오.

환경 변수는 파일 또는 디렉토리 위치를 나타내는 일부 Managed File Transfer 특성에서 사용할 수 있습니다. 이 경우 제품의 부분을 실행 중인 때 사용되는 파일 또는 디렉토리의 위치가 환경 변경사항(예: 프로세스를 실행 중인 사용자)에 따라 달라질 수 있습니다. 자세한 정보는 [MFT 특성의 환경 변수](#)를 참조하십시오.

### 사용자 엑시트 루틴 특성

다음 표에 나열된 순서대로 사용자 엑시트 루틴이 호출됩니다. agent.properties 파일에 대한 자세한 정보는 [고급 에이전트 특성: 사용자 엑시트 루틴의 내용](#)을 참조하십시오.

표 885. 사용자 엑시트에 대한 에이전트 특성	
특성 이름	설명
sourceTransferEndExitClasses	소스 전송 종료 엑시트 루틴을 구현하는 쉘표로 구분된 클래스 목록을 지정합니다.
sourceTransferStartExitClasses	소스 전송 시작 엑시트 루틴을 구현하는 쉘표로 구분된 클래스 목록을 지정합니다.
destinationTransferStartExitClasses	목적지 전송 시작의 사용자 엑시트 루틴을 구현하는 쉘표로 구분된 클래스 목록을 지정합니다.
destinationTransferEndExitClasses	대상 전송 사용자 엑시트 루틴을 구현하는 쉘표로 구분된 클래스 목록을 지정합니다.

표 885. 사용자 엑시트에 대한 에이전트 특성 (계속)	
특성 이름	설명
exitClassPath	<p>사용자 엑시트 루틴에 대한 클래스 경로로 사용되는 문자로 구분된 플랫폼별 디렉토리 목록을 지정합니다.</p> <p>에이전트의 엑시트 디렉토리는 이 클래스 경로의 항목보다 먼저 검색됩니다.</p> <p>괄호, 쉼표 (,) 및 백슬래시 (\) 는 MFT 명령에서 특수 문자이며 백슬래시 (\) 문자로 이스케이프해야 합니다. <b>Windows</b> Windows의 파일 경로는 이중 백슬래시 (\\) 를 구분 기호로 사용하거나 단일 슬래시 (/) 를 사용하여 지정할 수 있습니다. 예를 들면 다음과 같습니다.</p> <pre>exitClassPath=C:/mycomp/mqft/exits/encryptFileExit.jar; C:/mycomp/mqft/exits/fileFilter.jar.</pre> <p>이 특성의 값은 환경 변수를 포함할 수 있습니다.</p>
exitNativeLibraryPath	<p>사용자 엑시트 루틴에 대한 원시 라이브러리 경로로 작동하는 플랫폼별, 문자로 구분된 디렉토리 목록을 지정합니다.</p> <p>이 특성의 값은 환경 변수를 포함할 수 있습니다.</p>
monitorExitClasses	<p>모니터 엑시트 루틴을 구현하는 쉼표로 구분된 클래스 목록을 지정합니다. 추가 정보는 <a href="#">MFT 자원 모니터 사용자 엑시트의 내용을 참조하십시오.</a></p>
protocolBridgeCredentialExitClasses	<p>프로토콜 브릿지 신임 사용자 엑시트 루틴을 구현하는 쉼표로 구분된 클래스 목록을 지정합니다. 자세한 정보는 <a href="#">엑시트 클래스를 사용하여 파일 서버에 대한 신임 정보 맵핑을 참조하십시오.</a></p>
protocolBridgePropertiesExitClasses	<p>프로토콜 브릿지 서버 특성 사용자 엑시트 루틴을 구현하는 쉼표로 구분된 클래스 목록을 지정합니다. 자세한 정보는 <a href="#">ProtocolBridgePropertiesExit2: 프로토콜 파일 서버 특성 검색을 참조하십시오.</a></p>
IOExitClasses	<p>I/O 사용자 엑시트 루틴을 구현하는 쉼표로 구분된 클래스 목록을 지정합니다. IOExit 인터페이스를 구현하는 클래스만 나열합니다. 즉, 기타 I/O 사용자 엑시트 인터페이스를 구현하는 클래스(예: IOExitResourcePath 및 IOExitChannel)는 나열하지 않습니다. 자세한 정보는 <a href="#">MFT 전송 I/O 사용자 엑시트를 참조하십시오.</a></p>

## 엑시트 호출 순서

소스 및 목적지 엑시트는 다음 순서로 호출됩니다.

1. SourceTransferStartExit
2. DestinationTransferStartExit
3. DestinationTransferEndExit
4. SourceTransferEndExit

## 소스 및 목적지 엑시트 체인

여러 개의 엑시트를 지정하는 경우, 목록에 있는 첫 번째 엑시트가 먼저 호출된 후에 순서대로 다음 엑시트가 호출됩니다. 첫 번째 엑시트에 의한 변경사항은 차례대로 호출되는 엑시트에 대한 입력으로 전달됩니다. 예를 들어, 두 개의 소스 전송 시작 엑시트가 있는 경우 첫 번째 엑시트에 의한 전송 메타데이터의 변경사항은 두 번째 엑시트에 입력됩니다. 각 엑시트는 고유 결과를 리턴합니다. 주어진 유형의 모든 엑시트가 전송 결과 코드로 PROCEED를 리턴하면 전체 결과는 PROCEED입니다. 하나 이상의 엑시트가 CANCEL\_TRANSFER를 리턴하면, 전체 결과도 CANCEL\_TRANSFER입니다. 엑시트가 리턴하는 모든 결과 코드 및 문자열은 전송 로그에 출력됩니다.

소스 전송 시작 엑시트의 전체 결과가 PROCEED이면, 엑시트에 의한 변경사항을 사용하여 전송이 진행됩니다. 전체 결과가 CANCEL\_TRANSFER이면 소스 전송 종료 엑시트가 호출된 다음 전송이 취소됩니다. 전송 로그의 완료 상태는 "취소"입니다.

목적지 전송 시작 엑시트의 전체 결과가 PROCEED이면, 엑시트에 의한 변경사항을 사용하여 전송이 진행됩니다. 전체 결과가 CANCEL\_TRANSFER이면 목적지 전송 종료 엑시트가 호출된 다음 소스 전송 종료 엑시트가 호출됩니다. 마지막으로 전송이 취소됩니다. 전송 로그의 완료 상태는 "취소"입니다.

소스 또는 목적지 엑시트가 연쇄적으로 또는 실행 순서대로 정보를 다음 엑시트에 전달해야 하는 경우, 전송 메타 데이터를 업데이트하여 수행해야 합니다. 전송 메타데이터의 사용법은 엑시트 구현과 관련됩니다. 예를 들어, 엑시트가 리턴 결과를 CANCEL\_TRANSFER로 설정했는데 전송이 취소된 다음 엑시트와 통신해야 하는 경우, 다른 엑시트가 인식할 수 있는 방법으로 전송 메타데이터 값을 설정하여 이를 수행해야 합니다.

## 예

```
sourceTransferStartExitClasses=com.ibm.wmqfte.test.MFTTestSourceTransferStartExit
sourceTransferEndExitClasses=com.ibm.wmqfte.test.MFTTestSourceTransferEndExit
destinationTransferStartExitClasses=com.ibm.wmqfte.test.MFTTestDestinationTransferStartExit
destinationTransferEndExitClasses=com.ibm.wmqfte.test.MFTTestDestinationTransferEndExit
exitClassPath=C:/mycomp/mqft/exits/encryptFileExit.jar;C:/mycomp/mqft/exits/fileFilter.jar
```

## 관련 개념

[사용자 엑시트를 사용하여 MFT 사용자 정의](#)

[2032 페이지의 『MFT 사용자 엑시트의 메타데이터』](#)

Managed File Transfer의 사용자 엑시트 루틴에 제공할 수 있는 세 가지 메타데이터 유형(환경, 전송 및 파일 메타데이터)이 있습니다. 이 메타데이터는 Java 키-값 쌍의 맵으로 제공됩니다.

[2041 페이지의 『MFT 사용자 엑시트에 대한 Java 인터페이스』](#)

사용자 엑시트 루틴 관련 Java 인터페이스에 대한 참조 정보는 이 절의 주제를 사용하십시오.

## 관련 참조

[2035 페이지의 『MFT 자원 모니터 사용자 엑시트』](#)

자원 모니터 사용자 엑시트를 사용하면 모니터의 트리거 조건이 충족될 때 연관된 태스크가 시작되기 전에 실행할 사용자 정의 코드를 구성할 수 있습니다.

MFT 특성의 환경 변수

[MFTagent.properties 파일](#)

## MFT 사용자 엑시트에 대한 Java 인터페이스

사용자 엑시트 루틴 관련 Java 인터페이스에 대한 참조 정보는 이 절의 주제를 사용하십시오.

### 관련 태스크

[사용자 엑시트를 사용하여 MFT 사용자 정의](#)

### 관련 참조

[2045 페이지의 『DestinationTransferStartExit.java 인터페이스』](#)

[2044 페이지의 『DestinationTransferEndExit.java 인터페이스』](#)

[2047 페이지의 『IOExit.java 인터페이스』](#)

[2049 페이지의 『IOExitChannel.java 인터페이스』](#)

[2051 페이지의 『IOExitLock.java 인터페이스』](#)

[2052 페이지의 『IOExitPath.java 인터페이스』](#)

[2053 페이지의 『IOExitProperties.java 인터페이스』](#)

[2056 페이지의 『IOExitRecordChannel.java 인터페이스』](#)

[2057 페이지의 『IOExitRecordResourcePath.java interface』](#)

[2059 페이지의 『IOExitResourcePath.java 인터페이스』](#)

[2063 페이지의 『IOExitWildcardPath.java 인터페이스』](#)

[2063 페이지의 『MonitorExit.java 인터페이스』](#)

[2064 페이지의 『ProtocolBridgeCredentialExit.java 인터페이스』](#)

[2065 페이지의 『ProtocolBridgeCredentialExit2.java 인터페이스』](#)

[2066 페이지의 『ProtocolBridgePropertiesExit2.java 인터페이스』](#)

[2070 페이지의 『SourceTransferStartExit.java 인터페이스』](#)

**CDCredentialExit.java 인터페이스****CDCredentialExit.java**

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * ☒ Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.util.Map;

/**
 * An interface that is implemented by classes that are invoked as part of
 * user exit routine processing. This interface defines methods that are
 * invoked by a Connect:Direct bridge agent to map the IBM MQ user ID of the transfer to credentials
 * that are used to access the Connect:Direct node.
 * There will be one instance of each implementation class per Connect:Direct bridge agent. The methods
 * can be called from different threads so the methods must be synchronized.
 */
public interface CDCredentialExit {

    /**
     * Invoked once when a Connect:Direct bridge agent is started. It is intended to initialize
     * any resources that are required by the exit
     *
     * @param bridgeProperties
     *         The values of properties defined for the Connect:Direct bridge.
     *         These values can only be read, they cannot be updated by
     *         the implementation.
     *
     * @return true if the initialisation is successful and false if unsuccessful
     *         If false is returned from an exit the Connect:Direct bridge agent does not
     *         start.
     */
    public boolean initialize(final Map<String, String> bridgeProperties);

    /**
     * Invoked once per transfer to map the IBM MQ user ID in the transfer message to the
     * credentials to be used to access the Connect:Direct node.
     *
     * @param mqUserId The IBM MQ user ID from which to map to the credentials to be used
     *                 to access the Connect:Direct node
     * @param snode    The name of the Connect:Direct SNODE specified as the cdNode in the
     *                 file path. This is used to map the correct user ID and password for the
     *                 SNODE.
     * @return        A credential exit result object that contains the result of the map and
     *                 the credentials to use to access the Connect:Direct node
     */
    public CDCredentialExitResult mapMQUserId(final String mqUserId, final String snode);

    /**
     * Invoked once when a Connect:Direct bridge agent is shutdown. This method releases
     * any resources that were allocated by the exit
     *
     * @param bridgeProperties
     *         The values of properties defined for the Connect:Direct bridge.
     *         These values can only be read, they cannot be updated by
     *         the implementation.
     *
     * @return
     */
    public void shutdown(final Map<String, String> bridgeProperties); }

```

## CredentialExitResult.java 인터페이스

### CredentialExitResult.java

```
/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * ☒ Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */

package com.ibm.wmqfte.exitroutine.api;

/**
 * The result of invoking a Credential mapMqUserId exit method. It is composed of a result
 * code, which determines whether the mapping of the user id was successful, and an optional
 * Credentials object if the mapping is successful.
 */
public class CredentialExitResult {

    private final CredentialExitResultCode resultCode;
    private final Credentials credentials;

    /**
     * Constructor. Creates a credential exit result object with a specified result
     * code and optionally credentials.
     *
     * @param resultCode
     *     The result code to associate with the exit result being created.
     *
     * @param credentials
     *     The credentials to associate with the exit result being created.
     *     A value of <code>null</code> can be specified to indicate no
     *     credentials. If the resultCode is USER_SUCCESSFULLY_MAPPED the
     *     credentials must be set to a non-null value,
     */
    public CredentialExitResult(CredentialExitResultCode resultCode, Credentials credentials) {
        this.resultCode = resultCode;
        this.credentials = credentials;
    }

    /**
     * Returns the result code associated with this credential exit result
     *
     * @return
     *     the result code associated with this exit result.
     */
    public CredentialExitResultCode getResultCode() {
        return resultCode;
    }

    /**
     * Returns the credentials associated with this credential exit result
     *
     * @return
     *     the explanation associated with this credential exit result.
     */
    public Credentials getCredentials() {
        return credentials;
    }
}
```

#### 관련 태스크

[사용자 엑시트를 사용하여 MFT 사용자 정의](#)

#### 관련 참조

[2070 페이지의 『SourceTransferStartExit.java 인터페이스』](#)

[2045 페이지의 『DestinationTransferStartExit.java 인터페이스』](#)

[2044 페이지의 『DestinationTransferEndExit.java 인터페이스』](#)

## DestinationTransferEndExit.java 인터페이스

### DestinationTransferEndExit.java

```
/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * ☒ Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitpoint.api;

/**
 * An interface that is implemented by classes that want to be invoked as part of
 * user exit routine processing. This interface defines a method that will be
 * invoked immediately after completing a transfer on the agent acting as the
 * destination of the transfer.
 */
public interface DestinationTransferEndExit {

    /**
     * Invoked immediately after the completion of a transfer on the agent acting as
     * the destination of the transfer.
     *
     * @param transferExitResult
     *        a result object reflecting whether or not the transfer completed
     *        successfully.
     *
     * @param sourceAgentName
     *        the name of the agent acting as the source of the transfer.
     *
     * @param destinationAgentName
     *        the name of the agent acting as the destination of the
     *        transfer. This is the name of the agent that the
     *        implementation of this method will be invoked from.
     *
     * @param environmentMetaData
     *        meta data about the environment in which the implementation
     *        of this method is running. This information can only be read,
     *        it cannot be updated by the implementation. The constants
     *        defined in <code>EnvironmentMetaDataConstants</code> class can
     *        be used to access the data held by this map.
     *
     * @param transferMetaData
     *        meta data to associate with the transfer. The information can
     *        only be read, it cannot be updated by the implementation. This
     *        map may also contain keys with IBM reserved names. These
     *        entries are defined in the <code>TransferMetaDataConstants</code>
     *        class and have special semantics.
     *
     * @param fileResults
     *        a list of file transfer result objects that describe the source
     *        file name, destination file name and result of each file transfer
     *        operation attempted.
     *
     * @return
     *        an optional description to enter into the log message describing
     *        transfer completion. A value of <code>null</code> can be used
     *        when no description is required.
     */
    String onDestinationTransferEnd(TransferExitResult transferExitResult,
        String sourceAgentName,
        String destinationAgentName,
        Map<String, String>environmentMetaData,
        Map<String, String>transferMetaData,
        List<FileTransferResult>fileResults);
}
```



## 관련 태스크

[사용자 엑시트를 사용하여 MFT 사용자 정의](#)

## 관련 참조

2070 페이지의 『[SourceTransferStartExit.java 인터페이스](#)』

2068 페이지의 『[SourceTransferEndExit.java 인터페이스](#)』

2045 페이지의 『[DestinationTransferStartExit.java 인터페이스](#)』

2063 페이지의 『[MonitorExit.java 인터페이스](#)』

2064 페이지의 『[ProtocolBridgeCredentialExit.java 인터페이스](#)』

## ***DestinationTransferStartExit.java*** 인터페이스

### DestinationTransferStartExit.java

```
/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * ☒ Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitpoint.api;

/**
 * An interface that is implemented by classes that want to be invoked as part of
 * user exit routine processing. This interface defines a method that will be
 * invoked immediately prior to starting a transfer on the agent acting as the
 * destination of the transfer.
 */
public interface DestinationTransferStartExit {

    /**
     * Invoked immediately prior to starting a transfer on the agent acting as
     * the destination of the transfer.
     *
     * @param sourceAgentName
     *     the name of the agent acting as the source of the transfer.
     *
     * @param destinationAgentName
     *     the name of the agent acting as the destination of the
     *     transfer. This is the name of the agent that the
     *     implementation of this method will be invoked from.
     *
     * @param environmentMetaData
     *     meta data about the environment in which the implementation
     *     of this method is running. This information can only be read,
     *     it cannot be updated by the implementation. The constants
     *     defined in EnvironmentMetaDataConstants class can
     *     be used to access the data held by this map.
     *
     * @param transferMetaData
     *     meta data to associate with the transfer. The information can
     *     only be read, it cannot be updated by the implementation. This
     *     map may also contain keys with IBM reserved names. These
     *     entries are defined in the TransferMetaDataConstants
     *     class and have special semantics.
     *
     * @param fileSpecs
     *     a list of file specifications that govern the file data to
     *     transfer. The implementation of this method can modify the
     *     entries in this list and the changes will be reflected in the
     *     files transferred. However, new entries may not be added and
     *     existing entries may not be removed.
     *
     * @return
     *     a transfer exit result object which is used to determine if the
     *     transfer should proceed, or be cancelled.
     */
}
```

```
TransferExitResult onDestinationTransferStart(String sourceAgentName,
                                             String destinationAgentName,
                                             Map<String, String> environmentMetaData,
                                             Map<String, String> transferMetaData,
                                             List<Reference<String>> fileSpecs);
```

### 관련 태스크

[사용자 엑시트를 사용하여 MFT 사용자 정의](#)

### 관련 참조

[2070 페이지의 『SourceTransferStartExit.java 인터페이스』](#)

[2068 페이지의 『SourceTransferEndExit.java 인터페이스』](#)

[2044 페이지의 『DestinationTransferEndExit.java 인터페이스』](#)

[2063 페이지의 『MonitorExit.java 인터페이스』](#)

[2064 페이지의 『ProtocolBridgeCredentialExit.java 인터페이스』](#)

## FileTransferResult.java 인터페이스

### FileTransferResult.java

```
/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * ☒ Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */

package com.ibm.wmqfte.exitroutine.api;

/**
 * Result information about a file transfer.
 */
public interface FileTransferResult {

    /** An enumeration for the <code>getCorrelatorType()</code> method. */
    public enum CorrelationInformationType {
        /** No correlation information is available for this result */
        NONE,
        /**
         * The correlation information relates to work done in
         * IBM Sterling File Gateway.
         */
        SFG
    }

    /**
     * Returns the source file specification, from which the file was transferred.
     *
     * @return the source file specification, from which the file was
     * transferred.
     */
    String getSourceFileSpecification();

    /**
     * Returns the destination file specification, to which the file was transferred.
     *
     * @return the destination file specification, to which the file was
     * transferred. A value of <code>null</code> may be returned
     * if the transfer did not complete successfully.
     */
    String getDestinationFileSpecification();

    /**
     * Returns the result of the file transfer operation.
     */
}
```

```

    * @return    the result of the file transfer operation.
    */
    FileExitResult getExitResult();

    /**
     * @return an enumerated value that identifies the product to which this correlating
     *         information relates.
     */
    CorrelationInformationType getCorrelatorType();

    /**
     * @return the first string component of the correlating identifier that relates
     *         this transfer result to work done in another product. A value of null
     *         may be returned either because the other product does not utilize a
     *         string based correlation information or because there is no correlation
     *         information.
     */
    String getString1Correlator();

    /**
     * @return the first long component of the correlating identifier that relates
     *         this transfer result to work done in another product. A value of zero
     *         is returned when there is no correlation information or the other
     *         product does not utilize long based correlation information or because
     *         the value really is zero!
     */
    long getLong1Correlator();
}

```

## 관련 태스크

[사용자 엑시트를 사용하여 MFT 사용자 정의](#)

## 관련 참조

[2070 페이지의 『SourceTransferStartExit.java 인터페이스』](#)

[2045 페이지의 『DestinationTransferStartExit.java 인터페이스』](#)

[2044 페이지의 『DestinationTransferEndExit.java 인터페이스』](#)

[2063 페이지의 『MonitorExit.java 인터페이스』](#)

[2064 페이지의 『ProtocolBridgeCredentialExit.java 인터페이스』](#)

## IOExit.java 인터페이스

### IOExit.java

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * ☒ Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.io.IOException;
import java.util.Map;

import com.ibm.wmqfte.exitroutine.api.IOExitRecordResourcePath.RecordFormat;

/**
 * An interface that is implemented by classes that you want to be invoked as
 * part of user exit routine processing. This interface defines methods that
 * will be invoked during transfers to perform the underlying file system I/O
 * work for WMQFTE transfers.
 * <p>
 * The {@link #initialize(Map)} method will be called once when the exit is
 * first installed. The WMQFTE agent properties are passed to this method, thus
 * enabling the exit to understand its environment.

```

```

* <p>
* The {@link #isSupported(String)} method will be invoked during WMQFTE
* transfers to determine whether the user exit should be used. If the
* {@link #isSupported(String)} method returns a value of {@code true}, the
* {@link #newPath(String)} method will be invoked for the paths specified for
* the transfer request. The returned {@link IOExitPath} instance from a
* {@link #newPath(String)} method invocation will then be used by the WMQFTE
* transfer to obtain information about the resource and to transfer data to or
* from the resource.
* <p>
* To obtain transfer context for an I/O exit, a {@link SourceTransferStartExit}
* or {@link DestinationTransferStartExit} as appropriate, should be installed
* to enable information to be seen by this exit. The
* {@link SourceTransferStartExit} or {@link DestinationTransferStartExit} are
* passed the transfer's environment, metadata, and a list of file
* specifications for the transfer. The paths for the file specifications are
* the paths passed to the I/O exit's {@link #newPath(String)} method.
* <p>
* Note also that the {@link #isSupported(String)} and {@link #newPath(String)}
* methods might be called at other times by a WMQFTE agent and not just during
* transfers. For example, at transfer setup time the I/O system is queried to
* resolve the full resource paths for transfer.
*/
public interface IOExit {

    /**
     * Invoked once when the I/O exit is first required for use. It is intended
     * to initialize any resources that are required by the exit.
     *
     * @param agentProperties
     *     The values of properties defined for the WMQFTE agent. These
     *     values can only be read, they cannot be updated by the
     *     implementation.
     * @return {@code true} if the initialization is successful and {@code
     *     false} if unsuccessful. If {@code false} is returned from an
     *     exit, the exit will not be used.
     */
    boolean initialize(final Map<String, String> agentProperties);

    /**
     * Indicates whether this I/O user exit supports the specified path.
     * <p>
     * This method is used by WMQFTE to determine whether the I/O user exit
     * should be used within a transfer. If no I/O user exit returns true for
     * this method, the default WMQFTE file I/O function will be used.
     *
     * @param path
     *     The path to the required I/O resource.
     * @return {@code true} if the specified path is supported by the I/O exit,
     *     {@code false} otherwise
     */
    boolean isSupported(String path);

    /**
     * Obtains a new {@link IOExitPath} instance for the specified I/O resource
     * path.
     * <p>
     * This method will be invoked by WMQFTE only if the
     * {@link #isSupported(String)} method has been called for the path and
     * returned {@code true}.
     *
     * @param path
     *     The path to the required I/O resource.
     * @return A {@link IOExitPath} instance for the specified path.
     * @throws IOException
     *     If the path cannot be created for any reason.
     */
    IOExitPath newPath(String path) throws IOException;

    /**
     * Obtains a new {@link IOExitPath} instance for the specified I/O resource
     * path and passes record format and length information required by the
     * WMQFTE transfer.
     * <p>
     * Typically this method will be called for the following cases:
     * <ul>
     * <li>A path where a call to {@link #newPath(String)} has previously
     * returned a {@link IOExitRecordResourcePath} instance and WMQFTE is
     * re-establishing a new {@link IOExitPath} instance for the path, from an
     * internally-serialized state. The passed recordFormat and recordLength
     * will be the same as those for the original
     * {@link IOExitRecordResourcePath} instance.</li>
     */

```

```

* <li>A transfer destination path where the source of the transfer is
* record oriented. The passed recordFormat and recordLength will be the
* same as those for the source.</li>
* </ul>
* The implementation can act on the record format and length information as
* deemed appropriate. For example, for a destination agent if the
* destination does not already exist and the source of the transfer is
* record oriented, the passed recordFormat and recordLength information
* could be used to create an appropriate record-oriented destination path.
* If the destination path already exists, the passed recordFormat and
* recordLength information could be used to perform a compatibility check
* and throw an {@link IOException} if the path is not compatible. A
* compatibility check could ensure that a record oriented path's record
* format is the same as the passed record format or that the record length
* is greater or equal to the passed record length.
* <p>
* This method will be invoked by WMQFTE only if the
* {@link #isSupported(String)} method has been called for the path and
* returned {@code true}.
*
* @param path
*         The path to the required I/O resource.
* @param recordFormat
*         The advised record format.
* @param recordLength
*         The advised record length.
* @return A {@link IOExitPath} instance for the specified path.
* @throws IOException
*         If the path cannot be created for any reason. For example,
*         the passed record format or length is incompatible with the
*         path's actual record format or length.
*/
IOExitPath newPath(String path, RecordFormat recordFormat, int recordLength)
    throws IOException;

```

## 관련 태스크

[MFT I/O 사용자 엑시트 사용](#)

[사용자 엑시트를 사용하여 MFT 사용자 정의](#)

## IOExitChannel.java 인터페이스

### IOExitChannel.java

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * ☒ Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.io.IOException;
import java.nio.ByteBuffer;

/**
 * Represents a channel that enables data to be read from or written to an
 * {@link IOExitResourcePath} resource.
 */
public interface IOExitChannel {

    /**
     * Obtains the data size for the associated {@link IOExitResourcePath} in
     * bytes.
     *
     * @return The data size in bytes.
     * @throws IOException
     *         If a problem occurs while attempting obtain the size.
     */
    long size() throws IOException;

```

```

/**
 * Closes the channel, flushing any buffered write data to the resource and
 * releasing any locks.
 *
 * @throws RecoverableIOException
 *         If a recoverable problem occurs while closing the resource.
 *         This means that WMQFTE can attempt to recover the transfer.
 * @throws IOException
 *         If some other I/O problem occurs. For example, the channel might
 *         already be closed.
 */
void close() throws RecoverableIOException, IOException;

/**
 * Reads data from this channel into the given buffer, starting at this
 * channel's current position, and updates the current position by the
 * amount of data read.
 * <p>
 * Data is copied into the buffer starting at its current position and up to
 * its limit. On return, the buffer's position is updated to reflect the
 * number of bytes read.
 *
 * @param buffer
 *         The buffer that the data is to be copied into.
 * @return The number of bytes read, which might be zero, or -1 if the end of
 *         data has been reached.
 * @throws RecoverableIOException
 *         If a recoverable problem occurs while reading the data. For a
 *         WMQFTE transfer this means that it will attempt to recover.
 * @throws IOException
 *         If some other I/O problem occurs. For a WMQFTE transfer this
 *         means that it will be failed.
 */
int read(ByteBuffer buffer) throws RecoverableIOException, IOException;

/**
 * Writes data to this channel from the given buffer, starting at this
 * channel's current position, and updates the current position by the
 * amount of data written. The channel's resource is grown to accommodate
 * the data, if necessary.
 * <p>
 * Data is copied from the buffer starting at its current position and up to
 * its limit. On return, the buffer's position is updated to reflect the
 * number of bytes written.
 *
 * @param buffer
 *         The buffer containing the data to be written.
 * @return The number of bytes written, which might be zero.
 * @throws RecoverableIOException
 *         If a recoverable problem occurs while writing the data. For a
 *         WMQFTE transfer this means that it will attempt to recover.
 * @throws IOException
 *         If some other I/O problem occurs. For a WMQFTE transfer this
 *         means that it will be failed.
 */
int write(ByteBuffer buffer) throws RecoverableIOException, IOException;

/**
 * Forces any updates to this channel's resource to be written to its
 * storage device.
 * <p>
 * This method is required to force changes to both the resource's content
 * and any associated metadata to be written to storage.
 *
 * @throws RecoverableIOException
 *         If a recoverable problem occurs while performing the force.
 *         For a WMQFTE transfer this means that it will attempt to
 *         recover.
 * @throws IOException
 *         If some other I/O problem occurs. For a WMQFTE transfer this
 *         means that it will be failed.
 */
void force() throws RecoverableIOException, IOException;

/**
 * Attempts to lock the entire resource associated with the channel for
 * shared or exclusive access.
 * <p>
 * The intention is for this method not to block if the lock is currently
 * unavailable.
 *
 */

```

```

* @param shared
*         {@code true} if a shared lock is required, {@code false} if an
*         exclusive lock is required.
* @return A {@link IOExitLock} instance representing the newly acquired
*         lock or null if the lock cannot be obtained.
* @throws IOException
*         If a problem occurs while attempting to acquire the lock.
*/
IOExitLock tryLock(boolean shared) throws IOException;
}

```

## 관련 태스크

[MFT I/O 사용자 엑시트 사용](#)

[사용자 엑시트를 사용하여 MFT 사용자 정의](#)

## IOExitLock.java 인터페이스

### IOExitLock.java

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 *  Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.io.IOException;

/**
 * Represents a lock on a resource for either shared or exclusive access.
 * {@link IOExitLock} instances are returned from
 * {@link IOExitChannel#tryLock(boolean)} calls and WMQFTE will request the
 * release of the lock at the appropriate time during a transfer. Additionally, when
 * a {@link IOExitChannel#close()} method is called it will be the
 * responsibility of the channel to release any associated locks.
 */
public interface IOExitLock {

    /**
     * Releases the lock.
     * <p>
     * After this method has been successfully called the lock is to be deemed as invalid.
     *
     * @throws IOException
     *         If the channel associated with the lock is not open or
     *         another problem occurs while attempting to release the lock.
     */
    void release() throws IOException;

    /**
     * Indicates whether this lock is valid.
     * <p>
     * A lock is considered valid until its @ {@link #release()} method is
     * called or the associated {@link IOExitChannel} is closed.
     *
     * @return {@code true} if this lock is valid, {@code false} otherwise.
     */
    boolean isValid();

    /**
     * @return {@code true} if this lock is for shared access, {@code false} if
     *         this lock is for exclusive access.
     */
    boolean isShared();
}

```

## 관련 태스크

MFT I/O 사용자 엑시트 사용

사용자 엑시트를 사용하여 MFT 사용자 정의

## IOExitPath.java 인터페이스

### IOExitPath.java

```
/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * ☒ Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

/**
 * Represents an abstract path that can be inspected and queried by WMQFTE for
 * transfer purposes.
 * <p>
 * There are two types of path supported:
 * <ul>
 * <li>{@link IOExitResourcePath} - Represents a path that denotes a data
 * resource. For example, a file, directory, or group of database records.</li>
 * <li>{@link IOExitWildcardPath} - Represents a wildcard path that can be
 * expanded to multiple {@link IOExitResourcePath} instances.</li>
 * </ul>
 */
public abstract interface IOExitPath {

    /**
     * Obtains the abstract path as a {@link String}.
     *
     * @return The abstract path as a {@link String}.
     */
    String getPath();

    /**
     * Obtains the name portion of this abstract path as a {@link String}.
     * <p>
     * For example, a UNIX-style file system implementation evaluates the
     * path {@code /home/fteuser/file1.txt} as having a name of {@code
     * file1.txt}.
     *
     * @return the name portion of this abstract path as a {@link String}.
     */
    String getName();

    /**
     * Obtains the parent path for this abstract path as a {@link String}.
     * <p>
     * For example, a UNIX-style file system implementation evaluates the
     * path {@code /home/fteuser/file1.txt} as having a parent path of {@code
     * /home/fteuser}.
     *
     * @return The parent portion of the path as a {@link String}.
     */
    String getParent();

    /**
     * Obtains the abstract paths that match this abstract path.
     * <p>
     * If this abstract path denotes a directory resource, a list of paths
     * for all resources within the directory are returned.
     * <p>
     * If this abstract path denotes a wildcard, a list of all paths
     * matching the wildcard are returned.
     * <p>
     * Otherwise null is returned, because this abstract path probably denotes a

```



```

* single file resource.
*
* @return An array of {@link IOExitResourcePath}s that
*         match this path, or null if this method is not applicable.
*/
IOExitResourcePath[] listPaths();
}

```

## 관련 태스크

[MFT I/O 사용자 엑시트 사용](#)

[사용자 엑시트를 사용하여 MFT 사용자 정의](#)

## IOExitProperties.java 인터페이스

### IOExitProperties.java

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * ☒ Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

/**
 * Properties that determine how WMQFTE treats an {@link IOExitPath} for certain
 * aspects of I/O. For example, whether to use intermediate files.
 */
public class IOExitProperties {

    private boolean rereadSourceOnRestart = true;
    private boolean rechecksumSourceOnRestart = true;
    private boolean rechecksumDestinationOnRestart = true;
    private boolean useIntermediateFileAtDestination = true;
    private boolean requiresSingleThreadedChannelIO = false;

    /**
     * Determines whether the I/O exit implementation expects the resource to be
     * re-read from the start if a transfer is restarted.
     *
     * @return {@code true} if, on restart, the I/O exit expects the source
     *         resource to be opened at the beginning and re-read from the
     *         beginning (the {@link IOExitPath#openForRead(long)} method is
     *         always invoked with 0L as an argument). {@code false} if, on
     *         restart, the I/O exit expects the source to be opened at the
     *         offset that the source agent intends to start reading from (the
     *         {@link IOExitPath#openForRead(long)} method can be invoked with a
     *         non-zero value as its argument).
     */
    public boolean getRereadSourceOnRestart() {
        return rereadSourceOnRestart;
    }

    /**
     * Sets the value to determine whether the I/O exit implementation expects
     * the resource to be re-read from the beginning if a transfer is restarted.
     *
     * <p>
     * The default is {@code true}. The I/O exit should call this method when
     * required to change this value.
     *
     * @param rereadSourceOnRestart
     *        {@code true} if, on restart, the I/O exit expects the source
     *        resource to be opened at the beginning and re-read from the
     *        beginning (the {@link IOExitPath#openForRead(long)} method
     *        is always invoked with 0L as an argument). {@code false}
     *        if, on restart, the I/O exit expects the source to be opened
     *        at the offset that the source agent intends to start reading
     *        from (the {@link IOExitPath#openForRead(long)} method can be
     *        invoked with a non-zero value as its argument).
     */

```

```

*/
public void setRereadSourceOnRestart(boolean rereadSourceOnRestart) {
    this.rereadSourceOnRestart = rereadSourceOnRestart;
}

/**
 * Determines whether the I/O exit implementation requires the source
 * resource to be re-checksummed if the transfer is restarted.
 * Re-checksumming takes place only if the
 * {@link #getRereadSourceOnRestart()} method returns {@code true}.
 *
 * @return {@code true} if, on restart, the I/O exit expects the already-
 *         transferred portion of the source to be re-checksummed for
 *         inconsistencies. Use this option in environments
 *         where the source could be changed during a restart. {@code
 *         false} if, on restart, the I/O exit does not require the
 *         already-transferred portion of the source to be re-checksummed.
 */
public boolean getRechecksumSourceOnRestart() {
    return rechecksumSourceOnRestart;
}

/**
 * Sets the value to determine whether the I/O exit implementation requires
 * the source resource to be re-checksummed if the transfer is restarted.
 * Re-checksumming takes place only if the
 * {@link #getRereadSourceOnRestart()} method returns {@code true}.
 * <p>
 * The default is {@code true}. The I/O exit should call this method when
 * required to change this value.
 *
 * @param rechecksumSourceOnRestart
 *         {@code true} if, on restart, the I/O exit expects the already
 *         transferred portion of the source to be re-checksummed
 *         for inconsistencies. Use this option in environments
 *         where the source could be changed during a restart.
 *         {@code false} if, on restart, the I/O exit does not
 *         require the already-transferred portion of the source to be
 *         re-checksummed.
 */
public void setRechecksumSourceOnRestart(boolean rechecksumSourceOnRestart) {
    this.rechecksumSourceOnRestart = rechecksumSourceOnRestart;
}

/**
 * Determines whether the I/O exit implementation requires the destination
 * resource to be re-checksummed if the transfer is restarted.
 *
 * @return {@code true} if, on restart, the I/O exit expects the already
 *         transferred portion of the destination to be re-checksummed to
 *         check for inconsistencies. This option should be used in
 *         environments where the destination could have been changed while
 *         a restart is occurring. {@code false} if, on restart, the I/O exit
 *         does not require the already transferred portion of the
 *         destination to be re-checksummed.
 */
public boolean getRechecksumDestinationOnRestart() {
    return rechecksumDestinationOnRestart;
}

/**
 * Sets the value to determine whether the I/O exit implementation requires
 * the destination resource to be re-checksummed if the transfer is
 * restarted.
 * <p>
 * The default is {@code true}. The I/O exit should call this method when
 * required to change this value.
 *
 * @param rechecksumDestinationOnRestart
 *         {@code true} if, on restart, the I/O exit expects the already-
 *         transferred portion of the destination to be re-checksummed
 *         for inconsistencies. Use this option in environments
 *         where the destination could have been changed during a
 *         restart. {@code false} if, on restart, the I/O exit does not
 *         require the already-transferred portion of the destination
 *         to be re-checksummed.
 */
public void setRechecksumDestinationOnRestart(
    boolean rechecksumDestinationOnRestart) {
    this.rechecksumDestinationOnRestart = rechecksumDestinationOnRestart;
}

```

```

/**
 * Determines whether the I/O exit implementation requires the use of an
 * intermediate file when writing the data at the destination. The
 * intermediate file mechanism is typically used to prevent an incomplete
 * destination resource from being processed.
 *
 * @return {@code true} if data should be written to an intermediate file at
 *         the destination and then renamed (to the requested destination
 *         path name as specified in the transfer request) after the transfer is
 *         complete. {@code false} if data should be written directly to the
 *         requested destination path name without the use of an
 *         intermediate file.
 */
public boolean getUseIntermediateFileAtDestination() {
    return useIntermediateFileAtDestination;
}

/**
 * Sets the value to determine whether the I/O exit implementation requires
 * the use of an intermediate file when writing the data at the destination.
 * The intermediate file mechanism is typically used to prevent an
 * incomplete destination resource from being processed.
 *
 * <p>
 * The default is {@code true}. The I/O exit should call this method when
 * required to change this value.
 *
 * @param useIntermediateFileAtDestination
 *        {@code true} if data should be written to an intermediate file
 *        at the destination and then renamed (to the requested
 *        destination path name as specified in the transfer request) after
 *        the transfer is complete. {@code false} if data should be written
 *        directly to the requested destination path name without the
 *        use of an intermediate file
 */
public void setUseIntermediateFileAtDestination(
    boolean useIntermediateFileAtDestination) {
    this.useIntermediateFileAtDestination = useIntermediateFileAtDestination;
}

/**
 * Determines whether the I/O exit implementation requires
 * {@link IOExitChannel} instances to be accessed by a single thread only.
 *
 * @return {@code true} if {@link IOExitChannel} instances are to be
 *         accessed by a single thread only.
 */
public boolean requiresSingleThreadedChannelIO() {
    return requiresSingleThreadedChannelIO;
}

/**
 * Sets the value to determine whether the I/O exit implementation requires
 * channel operations for a particular instance to be accessed by a
 * single thread only.
 *
 * <p>
 * For certain I/O implementations it is necessary that resource path
 * operations such as open, read, write, and close are invoked only from a
 * single execution {@link Thread}. When set {@code true}, WMQFTE ensures
 * that the following are invoked on a single thread:
 *
 * <ul>
 * <li>{@link IOExitResourcePath#openForRead(long)} method and all methods of
 * the returned {@link IOExitChannel} instance.</li>
 * <li>{@link IOExitResourcePath#openForWrite(boolean)} method and all
 * methods of the returned {@link IOExitChannel} instance.</li>
 * </ul>
 *
 * <p>
 * This has a slight performance impact, hence enable single-threaded channel
 * I/O only when absolutely necessary.
 *
 * <p>
 * The default is {@code false}. The I/O exit should call this method when
 * required to change this value.
 *
 * @param requiresSingleThreadedChannelIO
 *        {@code true} if {@link IOExitChannel} instances are to be
 *        accessed by a single thread only.
 */
public void setRequiresSingleThreadedChannelIO(boolean requiresSingleThreadedChannelIO) {
    this.requiresSingleThreadedChannelIO = requiresSingleThreadedChannelIO;
}
}

```

## 관련 태스크

MFT I/O 사용자 엑시트 사용

사용자 엑시트를 사용하여 MFT 사용자 정의

## *IOExitRecordChannel.java* 인터페이스

### IOExitRecordChannel.java

```
/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * ☒ Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.io.IOException;
import java.nio.ByteBuffer;

/**
 * Represents a channel that enables records of data to be read from or written
 * to an {@link IOExitRecordResourcePath} resource.
 * <p>
 * This is an extension of the {@link IOExitChannel} interface such that the
 * {@link #read(java.nio.ByteBuffer)} and {@link #write(java.nio.ByteBuffer)}
 * methods are expected to deal in whole records of data only. That is, the
 * {@link java.nio.ByteBuffer} returned from the read method and passed to the
 * write method is assumed to contain one or more complete records.
 */
public interface IOExitRecordChannel extends IOExitChannel {

    /**
     * Reads records from this channel into the given buffer, starting at this
     * channel's current position, and updates the current position by the
     * amount of data read.
     * <p>
     * Record data is copied into the buffer starting at its current position
     * and up to its limit. On return, the buffer's position is updated to
     * reflect the number of bytes read.
     * <p>
     * Only whole records are copied into the buffer.
     * <p>
     * For a fixed-record-format resource, this might be multiple records. The
     * amount of data in the return buffer does not necessarily need to be a
     * multiple of the record length, but the last record is still to be treated
     * as a complete record and padded as required by the caller.
     * <p>
     * For a variable-format resource, this is a single whole record of a size
     * corresponding to the amount of return data or multiple whole records with
     * all except the last being treated as records of maximum size.
     *
     * @param buffer
     *         The buffer that the record data is to be copied into.
     * @return The number of bytes read, which might be zero, or -1 if the end of
     *         data has been reached.
     * @throws RecoverableIOException
     *         If a recoverable problem occurs while reading the data. For a
     *         WMQFTE transfer this means that it will attempt to recover.
     * @throws IOException
     *         If some other I/O problem occurs, for example, if the passed
     *         buffer is insufficient to contain at least one complete
     *         record). For a WMQFTE transfer this means that it will be
     *         failed.
     */
    int read(ByteBuffer buffer) throws RecoverableIOException, IOException;

    /**
     * Writes records to this channel from the given buffer, starting at this
     * channel's current position, and updates the current position by the

```

```

* amount of data written. The channel's resource is grown to accommodate
* the data, if necessary.
* <p>
* Record data is copied from the buffer starting at its current position
* and up to its limit. On return, the buffer's position is updated to
* reflect the number of bytes written.
* <p>
* The buffer is expected to contain only whole records.
* <p>
* For a fixed-record-format resource, this might be multiple records and if
* there is insufficient data in the buffer for a complete record, the
* record is to be padded as required to complete the record.
* <p>
* For a variable-record format resource the buffer is normally expected to
* contain a single record of length corresponding to the amount of data
* within the buffer. However, if the amount of data within the buffer
* exceeds the maximum record length, the implementation can either:
* <ol>
* <li>throw an {@link IOException} indicating that it cannot handle the
* situation.</li>
* <li>Consume a record's worth of data from the buffer, leaving the remaining
* data within the buffer.</li>
* <li>Consume all the buffer data and just write what it can to the current
* record. This effectively truncates the data.</li>
* <li>Consume all the buffer data and write to multiple records.</li>
* </ol>
*
* @param buffer
*         The buffer containing the data to be written.
* @return The number of bytes written, which might be zero.
* @throws RecoverableIOException
*         If a recoverable problem occurs while writing the data. For a
*         WMQFTE transfer this means that it will attempt to recover.
* @throws IOException
*         If some other I/O problem occurs. For a WMQFTE transfer this
*         means that it will be failed.
*/
int write(ByteBuffer buffer) throws RecoverableIOException, IOException;
}

```

## 관련 태스크

[MFT I/O 사용자 엑시트 사용](#)

[사용자 엑시트를 사용하여 MFT 사용자 정의](#)

## **IOExitRecordResourcePath.java interface**

### IOExitRecordResourcePath.java

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 *  Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.io.IOException;

/**
 * Represents a path that denotes a record-oriented data resource (for example,
 * a z/OS data set). It allows the data to be located, the record format to be
 * understood, and {@link IOExitRecordChannel} instances to be created for read
 * or write operations.
 */
public interface IOExitRecordResourcePath extends IOExitResourcePath {

    /**
     * Record formats for record-oriented resources.
     */
}

```

```

*/
public enum RecordFormat {
    FIXED, VARIABLE
}

/**
 * Obtains the record length for records that are maintained by the resource
 * denoted by this abstract path.
 * <p>
 * For a resource with fixed-length records, the data for each record read
 * and written is assumed to be this length.
 * <p>
 * For a resource with variable-length records, this is the maximum length
 * for a record's data.
 * <p>
 * This method should return a value greater than zero, otherwise it can
 * result in the failure of a WMQFTE transfer that involves this abstract
 * path.
 *
 * @return The record length, in bytes, for records maintained by the
 *         resource.
 */
int getRecordLength();

/**
 * Obtains record format, as a {@link RecordFormat} instance, for records
 * that are maintained by the resource denoted by this abstract path.
 *
 * @return A {@link RecordFormat} instance for the record format for records
 *         that are maintained by the resource denoted by this abstract
 *         path.
 */
RecordFormat getRecordFormat();

/**
 * Opens a {@link IOExitRecordChannel} instance for reading data from the
 * resource denoted by this abstract path. The current data byte position
 * for the resource is expected to be the passed position value, such that
 * when {@link IOExitRecordChannel#read(java.nio.ByteBuffer)} is called,
 * data starting from that position is read.
 * <p>
 * Note that the data byte read position will be on a record boundary.
 *
 * @param position
 *        The required data byte read position.
 * @return A new {@link IOExitRecordChannel} instance allowing data to be
 *         read from the resource denoted by this abstract path.
 * @throws RecoverableIOException
 *         If a recoverable problem occurs while attempting to open the
 *         resource for reading. This means that WMQFTE can attempt to
 *         recover the transfer.
 * @throws IOException
 *         If some other I/O problem occurs.
 */
IOExitRecordChannel openForRead(long position)
    throws RecoverableIOException, IOException;

/**
 * Opens a {@link IOExitRecordChannel} instance for writing data to the
 * resource denoted by this abstract path. Writing of data, using the
 * {@link IOExitRecordChannel#write(java.nio.ByteBuffer)} method, starts at
 * either the beginning of the resource or end of the current data for the
 * resource, depending on the specified append parameter.
 *
 * @param append
 *        When {@code true} indicates that data written to the resource
 *        should be appended to the end of the current data. When
 *        {@code false} indicates that writing of data is to start at
 *        the beginning of the resource; any existing data is lost.
 * @return A new {@link IOExitRecordChannel} instance allowing data to be
 *         written to the resource denoted by this abstract path.
 * @throws RecoverableIOException
 *         If a recoverable problem occurs while attempting to open the
 *         resource for writing. This means that WMQFTE can attempt to
 *         recover the transfer.
 * @throws IOException
 *         If some other I/O problem occurs.
 */
IOExitRecordChannel openForWrite(boolean append)
    throws RecoverableIOException, IOException;
}

```

## Related tasks

[Using MFT transfer I/O user exits](#)

[Customizing MFT with user exits](#)

## ***IOExitResourcePath.java*** 인터페이스

### **IOExitResourcePath.java**

```
/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * ☒ Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.io.IOException;

/**
 * Represents a path that denotes a data resource (for example, a file,
 * directory, or group of database records). It allows the data to be located
 * and {@link IOExitChannel} instances to be created for read or write
 * operations.
 * <p>
 * There are two types of data resources as follows:
 * <ul>
 * <li>Directory - a container for other data resources. The
 * {@link #isDirectory()} method returns {@code true} for these.</li>
 * <li>File - a data container. This allows data to be read from or written to
 * it. The {@link #isFile()} method returns {@code true} for these.</li>
 * </ul>
 */
public interface IOExitResourcePath extends IOExitPath {

    /**
     * Creates a new {@link IOExitResourcePath} instance for a child path of the
     * resource denoted by this abstract path.
     * <p>
     * For example, with a UNIX-style path, {@code
     * IOExitResourcePath("/home/fteuser/test").newPath("subtest")} could be
     * equivalent to: {@code IOExitResourcePath("/home/fteuser/test/subtest")}
     *
     * @param child
     *         The child path name.
     * @return A new {@link IOExitResourcePath} instance that represents a child
     *         of this path.
     */
    IOExitResourcePath newPath(final String child);

    /**
     * Creates the directory path for the resource denoted by this abstract
     * path, including any necessary but nonexistent parent directories. If the
     * directory path already exists, this method has no effect.
     * <p>
     * If this operation fails, it might have succeeded in creating some of the
     * necessary parent directories.
     *
     * @throws IOException
     *         If the directory path cannot be fully created, when it does
     *         not already exist.
     */
    void makePath() throws IOException;

    /**
     * Obtains the canonical path of the abstract path as a {@link String}.
     * <p>
     * A canonical path is defined as being absolute and unique. For example,
     * the path can be represented as UNIX-style relative path: {@code
     * test/file.txt} but the absolute and unique canonical path representation
     */
}
```

```

* is: {@code /home/fteuser/test/file.txt}
*
* @return The canonical path as a {@link String}.
* @throws IOException
*         If the canonical path cannot be determined for any reason.
*/
String getCanonicalPath() throws IOException;

/**
 * Tests if this abstract path is an absolute path.
 * <p>
 * For example, a UNIX-style path, {@code /home/fteuser/test} is an absolute
 * path, whereas {@code fteuser/test} is not.
 *
 * @return {@code true} if this abstract path is an absolute path, {@code
 *         false} otherwise.
 */
boolean isAbsolute();

/**
 * Tests if the resource denoted by this abstract path exists.
 *
 * @return {@code true} if the resource denoted by this abstract path
 *         exists, {@code false} otherwise.
 * @throws IOException
 *         If the existence of the resource cannot be determined for any
 *         reason.
 */
boolean exists() throws IOException;

/**
 * Tests whether the calling application can read the resource denoted by
 * this abstract path.
 *
 * @return {@code true} if the resource for this path exists and can be
 *         read, {@code false} otherwise.
 * @throws IOException
 *         If a problem occurs while attempting to determine if the
 *         resource can be read.
 */
boolean canRead() throws IOException;

/**
 * Tests whether the calling application can modify the resource denoted by
 * this abstract path.
 *
 * @return {@code true} if the resource for this path exists and can be
 *         modified, {@code false} otherwise.
 * @throws IOException
 *         If a problem occurs while attempting to determine if the
 *         resource can be modified.
 */
boolean canWrite() throws IOException;

/**
 * Tests whether the specified user is permitted to read the resource
 * denoted by this abstract path.
 * <p>
 * When WMQFTE invokes this method, the user identifier is the MQMD user
 * identifier for the requesting transfer.
 *
 * @param userId
 *         User identifier to test for access.
 * @return {@code true} if the resource for this abstract path exists and is
 *         permitted to be read by the specified user, {@code false}
 *         otherwise.
 * @throws IOException
 *         If a problem occurs while attempting to determine if the user
 *         is permitted to read the resource.
 */
boolean readPermitted(String userId) throws IOException;

/**
 * Tests whether the specified user is permitted to modify the resource
 * denoted by this abstract path.
 * <p>
 * When WMQFTE invokes this method, the user identifier is the MQMD user
 * identifier for the requesting transfer.
 *
 * @param userId
 *         User identifier to test for access.
 * @return {@code true} if the resource for this abstract path exists and is

```



```

*         permitted to be modified by the specified user, {@code false}
*         otherwise.
* @throws IOException
*         If a problem occurs while attempting to determine if the user
*         is permitted to modify the resource.
*/
boolean writePermitted(String userId) throws IOException;

/**
 * Tests if the resource denoted by this abstract path is a directory-type
 * resource.
 *
 * @return {@code true} if the resource denoted by this abstract path is a
 *         directory type resource, {@code false} otherwise.
 */
boolean isDirectory();

/**
 * Creates the resource denoted by this abstract path, if it does not
 * already exist.
 *
 * @return {@code true} if the resource does not exist and was successfully
 *         created, {@code false} if the resource already existed.
 * @throws RecoverableIOException
 *         If a recoverable problem occurs while attempting to create
 *         the resource. This means that WMQFTE can attempt to recover
 *         the transfer.
 * @throws IOException
 *         If some other I/O problem occurs.
 */
boolean createNewPath() throws RecoverableIOException, IOException;

/**
 * Tests if the resource denoted by this abstract path is a file-type
 * resource.
 *
 * @return {@code true} if the resource denoted by this abstract path is a
 *         file type resource, {@code false} otherwise.
 */
boolean isFile();

/**
 * Obtains the last modified time for the resource denoted by this abstract
 * path.
 * <p>
 * This time is measured in milliseconds since the epoch (00:00:00 GMT,
 * January 1, 1970).
 *
 * @return The last modified time for the resource denoted by this abstract
 *         path, or a value of 0L if the resource does not exist or a
 *         problem occurs.
 */
long lastModified();

/**
 * Deletes the resource denoted by this abstract path.
 * <p>
 * If the resource is a directory, it must be empty for the delete to work.
 *
 * @throws IOException
 *         If the delete of the resource fails for any reason.
 */
void delete() throws IOException;

/**
 * Renames the resource denoted by this abstract path to the specified
 * destination abstract path.
 * <p>
 * The rename should still be successful if the resource for the specified
 * destination abstract path already exists and it is possible to replace
 * it.
 *
 * @param destination
 *         The new abstract path for the resource denoted by this
 *         abstract path.
 * @throws IOException
 *         If the rename of the resource fails for any reason.
 */
void renameTo(IOExceptionResourcePath destination) throws IOException;

/**
 * Creates a new path to use for writing to a temporary resource that did

```

```

* not previously exist.
* <p>
* The implementation can choose the abstract path name for the temporary
* resource. However, for clarity and problem diagnosis, the abstract path
* name for the temporary resource should be based on this abstract path
* name with the specified suffix appended and additional characters to make
* the path unique (for example, sequence numbers), as required.
* <p>
* When WMQFTE transfers data to a destination it normally attempts to first
* write to a temporary resource then on transfer completion renames the
* temporary resource to the required destination. This method is called by
* WMQFTE to create a new temporary resource path. The returned path should
* be new and the resource should not previously exist.
*
* @param suffix
*         Recommended suffix to use for the generated temporary path.
*
* @return A new {@link IOExitResourcePath} instance for the temporary
*         resource path, that did not previously exist.
* @throws RecoverableIOException
*         If a recoverable problem occurs whilst attempting to create
*         the temporary resource. This means that WMQFTE can attempt to
*         recover the transfer.
* @throws IOException
*         If some other I/O problem occurs.
*/
IOExitResourcePath createTempPath(String suffix)
    throws RecoverableIOException, IOException;

/**
* Opens a {@link IOExitChannel} instance for reading data from the resource
* denoted by this abstract path. The current data byte position for the
* resource is expected to be the passed position value, such that when
* {@link IOExitChannel#read(java.nio.ByteBuffer)} is called, data starting
* from that position is read.
*
* @param position
*         The required data byte read position.
* @return A new {@link IOExitChannel} instance allowing data to be read
*         from the resource denoted by this abstract path.
* @throws RecoverableIOException
*         If a recoverable problem occurs while attempting to open the
*         resource for reading. This means that WMQFTE can attempt to
*         recover the transfer.
* @throws IOException
*         If some other I/O problem occurs.
*/
IOExitChannel openForRead(long position) throws RecoverableIOException,
    IOException;

/**
* Opens a {@link IOExitChannel} instance for writing data to the resource
* denoted by this abstract path. Writing of data, using the
* {@link IOExitChannel#write(java.nio.ByteBuffer)} method, starts at either
* the beginning of the resource or end of the current data for the
* resource, depending on the specified append parameter.
*
* @param append
*         When {@code true} indicates that data written to the resource
*         should be appended to the end of the current data. When
*         {@code false} indicates that writing of data is to start at
*         the beginning of the resource; any existing data is lost.
* @return A new {@link IOExitChannel} instance allowing data to be written
*         to the resource denoted by this abstract path.
* @throws RecoverableIOException
*         If a recoverable problem occurs whilst attempting to open the
*         resource for writing. This means that WMQFTE can attempt to
*         recover the transfer.
* @throws IOException
*         If some other I/O problem occurs.
*/
IOExitChannel openForWrite(boolean append) throws RecoverableIOException,
    IOException;

/**
* Tests if the resource denoted by this abstract path is in use by another
* application. Typically, this is because another application has a lock on
* the resource either for shared or exclusive access.
*
* @return {@code true} if resource denoted by this abstract path is in use
*         by another application, {@code false} otherwise.
*/

```

```

boolean inUse();

/**
 * Obtains a {@link IOExitProperties} instance for properties associated
 * with the resource denoted by this abstract path.
 * <p>
 * WMQFTE will read these properties to govern how a transfer behaves when
 * interacting with the resource.
 *
 * @return A {@link IOExitProperties} instance for properties associated
 *         with the resource denoted by this abstract path.
 */
IOExitProperties getProperties();
}

```

### 관련 태스크

[MFT I/O 사용자 엑시트 사용](#)

[사용자 엑시트를 사용하여 MFT 사용자 정의](#)

## ***IOExitWildcardPath.java*** 인터페이스

### **IOExitWildcardPath.java**

```

/**
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

/**
 * Represents a path that denotes a wildcard. This can be used to match multiple
 * resource paths.
 */
public interface IOExitWildcardPath extends IOExitPath {

```

### 관련 태스크

[MFT I/O 사용자 엑시트 사용](#)

[사용자 엑시트를 사용하여 MFT 사용자 정의](#)

## ***MonitorExit.java*** 인터페이스

### **MonitorExit.java**

```

/**
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * Copyright IBM Corp. 2009, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.util.Map;

/**

```

```

* An interface that is implemented by classes that want to be invoked as part of
* user exit routine processing. This interface defines a method that will be
* invoked immediately prior to starting a task as the result of a monitor trigger
*/
public interface MonitorExit {

    /**
     * Invoked immediately prior to starting a task as the result of a monitor
     * trigger.
     *
     * @param environmentMetaData
     *      meta data about the environment in which the implementation
     *      of this method is running. This information can only be read,
     *      it cannot be updated by the implementation. The constant
     *      defined in <code>EnvironmentMetaDataConstants</code> class can
     *      be used to access the data held by this map.
     *
     * @param monitorMetaData
     *      meta data to associate with the monitor. The meta data passed
     *      to this method can be altered, and the changes will be
     *      reflected in subsequent exit routine invocations. This map
     *      also contains keys with IBM reserved names. These entries are
     *      defined in the <code>MonitorMetaDataConstants</code> class and
     *      have special semantics. The the values of the IBM reserved names
     *      cannot be modified by the exit
     *
     * @param taskDetails
     *      An XML String representing the task to be executed as a result of
     *      the monitor triggering. This XML string may be modified by the
     *      exit
     *
     * @return
     *      a monitor exit result object which is used to determine if the
     *      task should proceed, or be cancelled.
     */
    MonitorExitResult onMonitor(Map<String, String> environmentMetaData,
                               Map<String, String> monitorMetaData,
                               Reference<String> taskDetails);
}

```

## 관련 태스크

### MFT 자원 모니터링

[사용자 엑시트를 사용하여 MFT 사용자 정의](#)

## 관련 참조

2070 페이지의 『SourceTransferStartExit.java 인터페이스』

2068 페이지의 『SourceTransferEndExit.java 인터페이스』

2045 페이지의 『DestinationTransferStartExit.java 인터페이스』

2044 페이지의 『DestinationTransferEndExit.java 인터페이스』

2064 페이지의 『ProtocolBridgeCredentialExit.java 인터페이스』

## **ProtocolBridgeCredentialExit.java 인터페이스**

### ProtocolBridgeCredentialExit.java

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * ☒ Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.util.Map;

/**

```

```

* An interface that is implemented by classes that are to be invoked as part of
* user exit routine processing. This interface defines methods that will
* be invoked by a protocol bridge agent to map the MQ user ID of the transfer to credentials
* that are to be used to access the protocol server.
* There will be one instance of each implementation class per protocol bridge agent. The methods
* can be called from different threads so the methods must be synchronized.
*/
public interface ProtocolBridgeCredentialExit {

    /**
     * Invoked once when a protocol bridge agent is started. It is intended to initialize
     * any resources that are required by the exit
     *
     * @param bridgeProperties
     *         The values of properties defined for the protocol bridge.
     *         These values can only be read, they cannot be updated by
     *         the implementation.
     *
     * @return true if the initialization is successful and false if unsuccessful
     *         If false is returned from an exit the protocol bridge agent will not
     *         start
     */
    public boolean initialize(final Map<String> bridgeProperties);

    /**
     * Invoked once for each transfer to map the MQ user ID in the transfer message to the
     * credentials to be used to access the protocol server
     *
     * @param mqUserId The MQ user ID from which to map to the credentials to be used
     *                 access the protocol server
     * @return A credential exit result object that contains the result of the map and
     *         the credentials to use to access the protocol server
     */
    public CredentialExitResult mapMQUserId(final String mqUserId);

    /**
     * Invoked once when a protocol bridge agent is shutdown. It is intended to release
     * any resources that were allocated by the exit
     *
     * @param bridgeProperties
     *         The values of properties defined for the protocol bridge.
     *         These values can only be read, they cannot be updated by
     *         the implementation.
     *
     * @return
     */
    public void shutdown(final Map<String> bridgeProperties);
}

```

## 관련 태스크

[사용자 엑시트를 사용하여 MFT 사용자 정의 엑시트 클래스를 사용하여 파일 서버에 대한 신임 정보 맵핑](#)

## ProtocolBridgeCredentialExit2.java 인터페이스

### ProtocolBridgeCredentialExit2.java

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 *  Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

```

```

/**
 * An interface that is implemented by classes that are invoked as part of user
 * exit routine processing. This interface defines methods that are invoked by a
 * protocol bridge agent to map the MQ user ID of the transfer to credentials
 * used to access a specified protocol bridge server. There will be one instance
 * of each implementation class for each protocol bridge agent. The methods can
 * be called from different threads so the methods must be synchronized.
 */
public interface ProtocolBridgeCredentialExit2 extends
    ProtocolBridgeCredentialExit {

    /**
     * Invoked once for each transfer to map the MQ user ID in the transfer
     * message to the credentials used to access a specified protocol server.
     *
     * @param endPoint
     *         Information that describes the protocol server to be accessed.
     * @param mqUserId
     *         The MQ user ID from which to map the credentials used to
     *         access the protocol server.
     * @return A {@link CredentialExitResult} instance that contains the result
     *         of the map and the credentials to use to access the protocol
     *         server.
     */
    public CredentialExitResult mapMQUserId(
        final ProtocolServerEndPoint endPoint, final String mqUserId);
}

```

## 관련 태스크

[사용자 엑시트를 사용하여 MFT 사용자 정의](#)

[엑시트 클래스를 사용하여 파일 서버에 대한 신임 정보 매핑](#)

## **ProtocolBridgePropertiesExit2.java** 인터페이스

### **ProtocolBridgePropertiesExit2.java**

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * ☒ Copyright IBM Corp. 2011, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.util.Map;
import java.util.Properties;

/**
 * An interface that is implemented by classes that are to be invoked as part of
 * user exit routine processing. This interface defines methods that will be
 * invoked by a protocol bridge agent to look up properties for protocol servers
 * that are referenced in transfers.
 * <p>
 * There will be one instance of each implementation class for each protocol
 * bridge agent. The methods can be called from different threads so the methods
 * must be synchronised.
 */
public interface ProtocolBridgePropertiesExit2 {

    /**
     * Invoked once when a protocol bridge agent is started. It is intended to
     * initialize any resources that are required by the exit.
     *
     * @param bridgeProperties
     *         The values of properties defined for the protocol bridge.
     *         These values can only be read, they cannot be updated by the
     *         implementation.
     * @return {@code true} if the initialization is successful and {@code

```

```

*         false} if unsuccessful. If {@code false} is returned from an exit
*         the protocol bridge agent will not start.
*/
public boolean initialize(final Map<String, String> bridgeProperties);

/**
 * Invoked when the Protocol Bridge needs to access the protocol bridge credentials XML file.
 *
 * @return a {@link String} object giving the location of the ProtocolBridgeCredentials.xml
 */
public String getCredentialLocation ();

/**
 * Obtains a set of properties for the specified protocol server name.
 * <p>
 * The returned {@link Properties} must contain entries with key names
 * corresponding to the constants defined in
 * {@link ProtocolServerPropertyConstants} and in particular must include an
 * entry for all appropriate constants described as required.
 *
 * @param protocolServerName
 *         The name of the protocol server whose properties are to be
 *         returned. If a null or a blank value is specified, properties
 *         for the default protocol server are to be returned.
 * @return The {@link Properties} for the specified protocol server, or null
 *         if the server cannot be found.
 */
public Properties getProtocolServerProperties(
    final String protocolServerName);

/**
 * Invoked once when a protocol bridge agent is shut down. It is intended to
 * release any resources that were allocated by the exit.
 *
 * @param bridgeProperties
 *         The values of properties defined for the protocol bridge.
 *         These values can only be read, they cannot be updated by the
 *         implementation.
 */
public void shutdown(final Map<String, String> bridgeProperties);
}

```

## 관련 태스크

[ProtocolBridgePropertiesExit: 프로토콜 파일 서버 특성 검색](#)

[사용자 엑시트를 사용하여 MFT 사용자 정의](#)

[엑시트 클래스를 사용하여 파일 서버에 대한 신임 정보 맵핑](#)

## SourceFileExitFileSpecification.java 클래스

### SourceFileExitFileSpecification.java

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * ☒ Copyright IBM Corp. 2012, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitroutine.api;

import java.util.Map;

/**
 * A specification of the file names to use for a file transfer, as evaluated by the
 * agent acting as the source of the transfer.
 */
public final class SourceFileExitFileSpecification {

```

```

private final String sourceFileSpecification;
private final String destinationFileSpecification;
private final Map<String, String> sourceFileMetaData;
private final Map<String, String> destinationFileMetaData;

/**
 * Constructor. Creates a source file exit file specification.
 *
 * @param sourceFileSpecification
 *         the source file specification to associate with the source file
 *         exit file specification.
 *
 * @param destinationFileSpecification
 *         the destination file specification to associate with the
 *         source file exit file specification.
 *
 * @param sourceFileMetaData
 *         the source file meta data.
 *
 * @param destinationFileMetaData
 *         the destination file meta data .
 */
public SourceFileExitFileSpecification(final String sourceFileSpecification,
                                       final String destinationFileSpecification,
                                       final Map<String, String> sourceFileMetaData,
                                       final Map<String, String> destinationFileMetaData) {
    this.sourceFileSpecification = sourceFileSpecification;
    this.destinationFileSpecification = destinationFileSpecification;
    this.sourceFileMetaData = sourceFileMetaData;
    this.destinationFileMetaData = destinationFileMetaData;
}

/**
 * Returns the destination file specification.
 *
 * @return the destination file specification. This represents the location,
 *         on the agent acting as the destination for the transfer, where the
 *         file should be written. Exit routines installed into the agent
 *         acting as the destination for the transfer may override this value.
 */
public String getDestination() {
    return destinationFileSpecification;
}

/**
 * Returns the source file specification.
 *
 * @return the source file specification. This represents the location where
 *         the file data will be read from.
 */
public String getSource() {
    return sourceFileSpecification;
}

/**
 * Returns the file meta data that relates to the source file specification.
 *
 * @return the file meta data that relates to the source file specification.
 */
public Map<String, String> getSourceFileMetaData() {
    return sourceFileMetaData;
}

/**
 * Returns the file meta data that relates to the destination file specification.
 *
 * @return the file meta data that relates to the destination file specification.
 */
public Map<String, String> getDestinationFileMetaData() {
    return destinationFileMetaData;
}
}

```

## 관련 개념

2032 페이지의 『MFT 사용자 엑시트의 메타데이터』

Managed File Transfer의 사용자 엑시트 루틴에 제공할 수 있는 세 가지 메타데이터 유형(환경, 전송 및 파일 메타데이터)이 있습니다. 이 메타데이터는 Java 키-값 쌍의 맵으로 제공됩니다.

## SourceTransferEndExit.java 인터페이스



## SourceTransferEndExit.java

```
/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * ☒ Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitpoint.api;

/**
 * An interface that is implemented by classes that want to be invoked as part of
 * user exit routine processing. This interface defines a method that will be
 * invoked immediately after completing a transfer on the agent acting as the
 * source of the transfer.
 */
public interface SourceTransferEndExit {

    /**
     * Invoked immediately after the completion of a transfer on the agent acting as
     * the source of the transfer.
     *
     * @param transferExitResult
     *     a result object reflecting whether or not the transfer completed
     *     successfully.
     *
     * @param sourceAgentName
     *     the name of the agent acting as the source of the transfer.
     *     This is the name of the agent that the implementation of this
     *     method will be invoked from.
     *
     * @param destinationAgentName
     *     the name of the agent acting as the destination of the
     *     transfer.
     *
     * @param environmentMetaData
     *     meta data about the environment in which the implementation
     *     of this method is running. This information can only be read,
     *     it cannot be updated by the implementation. The constants
     *     defined in EnvironmentMetaDataConstants class can
     *     be used to access the data held by this map.
     *
     * @param transferMetaData
     *     meta data to associate with the transfer. The information can
     *     only be read, it cannot be updated by the implementation. This
     *     map may also contain keys with IBM reserved names. These
     *     entries are defined in the TransferMetaDataConstants
     *     class and have special semantics.
     *
     * @param fileResults
     *     a list of file transfer result objects that describe the source
     *     file name, destination file name and result of each file transfer
     *     operation attempted.
     *
     * @return
     *     an optional description to enter into the log message describing
     *     transfer completion. A value of null can be used
     *     when no description is required.
     */
    String onSourceTransferEnd(TransferExitResult transferExitResult,
                               String sourceAgentName,
                               String destinationAgentName,
                               Map<String, String>environmentMetaData,
                               Map<String, String>transferMetaData,
                               List<FileTransferResult>fileResults);

}

```

### 관련 태스크

[사용자 엑시트를 사용하여 MFT 사용자 정의](#)

## 관련 참조

[2070 페이지의 『SourceTransferStartExit.java 인터페이스』](#)

[2045 페이지의 『DestinationTransferStartExit.java 인터페이스』](#)

[2044 페이지의 『DestinationTransferEndExit.java 인터페이스』](#)

[2063 페이지의 『MonitorExit.java 인터페이스』](#)

[2064 페이지의 『ProtocolBridgeCredentialExit.java 인터페이스』](#)

## SourceTransferStartExit.java 인터페이스

### SourceTransferStartExit.java

```
/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * ☒ Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */
package com.ibm.wmqfte.exitpoint.api;

import java.util.List;
import java.util.Map;

/**
 * An interface that is implemented by classes that want to be invoked as part of
 * user exit routine processing. This interface defines a method that will be
 * invoked immediately prior to starting a transfer on the agent acting as the
 * source of the transfer.
 */
public interface SourceTransferStartExit {

    /**
     * Invoked immediately prior to starting a transfer on the agent acting as
     * the source of the transfer.
     *
     * @param sourceAgentName
     *     the name of the agent acting as the source of the transfer.
     *     This is the name of the agent that the implementation of this
     *     method will be invoked from.
     *
     * @param destinationAgentName
     *     the name of the agent acting as the destination of the
     *     transfer.
     *
     * @param environmentMetaData
     *     meta data about the environment in which the implementation
     *     of this method is running. This information can only be read,
     *     it cannot be updated by the implementation. The constants
     *     defined in EnvironmentMetaDataConstants class can
     *     be used to access the data held by this map.
     *
     * @param transferMetaData
     *     meta data to associate with the transfer. The meta data passed
     *     to this method can be altered, and the changes to will be
     *     reflected in subsequent exit routine invocations. This map may
     *     also contain keys with IBM reserved names. These entries are
     *     defined in the TransferMetaDataConstants class and
     *     have special semantics.
     *
     * @param fileSpecs
     *     a list of file specifications that govern the file data to
     *     transfer. The implementation of this method can add entries,
     *     remove entries, or modify entries in this list and the changes
     *     will be reflected in the files transferred.
     *
     * @return
     *     a transfer exit result object which is used to determine if the
     *     transfer should proceed, or be cancelled.
     */
}
```

```

*/
TransferExitResult onSourceTransferStart(String sourceAgentName,
String destinationAgentName,
Map<String, String> environmentMetaData,
Map<String, String>transferMetaData,
List<SourceFileExitFileSpecification>fileSpecs);
}

```

## 관련 태스크

[사용자 엑시트를 사용하여 MFT 사용자 정의](#)

## 관련 참조

[2067 페이지의 『SourceFileExitFileSpecification.java 클래스』](#)

[2068 페이지의 『SourceTransferEndExit.java 인터페이스』](#)

[2045 페이지의 『DestinationTransferStartExit.java 인터페이스』](#)

[2044 페이지의 『DestinationTransferEndExit.java 인터페이스』](#)

[2063 페이지의 『MonitorExit.java 인터페이스』](#)

[2064 페이지의 『ProtocolBridgeCredentialExit.java 인터페이스』](#)

## **TransferExitResult.java** 인터페이스

### TransferExitResult.java

```

/*
 * Licensed Materials - Property of IBM
 *
 * "Restricted Materials of IBM"
 *
 * 5724-H72
 *
 * ☒ Copyright IBM Corp. 2008, 2024. All Rights Reserved.
 *
 * US Government Users Restricted Rights - Use, duplication or
 * disclosure restricted by GSA ADP Schedule Contract with
 * IBM Corp.
 */

package com.ibm.wmqfte.exitroutine.api;

/**
 * The result of invoking a transfer exit routine. It is composed of a result
 * code, which determines if the transfer should proceed, and an optional explanatory
 * message. The explanation, if present, is entered into the log message.
 */
public class TransferExitResult {

    private final TransferExitResultCode resultCode;
    private final String explanation;

    /**
     * For convenience, a static "proceed" result with no associated explanation
     * message.
     */
    public static final TransferExitResult PROCEED_RESULT =
        new TransferExitResult(TransferExitResultCode.PROCEED, null);

    /**
     * Constructor. Creates a transfer exit result object with a specified result
     * code and explanation.
     *
     * @param resultCode
     *     The result code to associate with the exit result being created.
     *
     * @param explanation
     *     The explanation to associate with the exit result being created.
     *     A value of <code>null</code> can be specified to indicate no
     *     explanation.
     */
    public TransferExitResult(TransferExitResultCode resultCode, String explanation) {
        this.resultCode = resultCode;
        this.explanation = explanation;
    }
}

```

```

}

/**
 * Returns the explanation associated with this transfer exit result.
 *
 * @return the explanation associated with this exit result.
 */
public String getExplanation() {
    return explanation;
}

/**
 * Returns the result code associated with this transfer exit result.
 *
 * @return the result code associated with this exit result.
 */
public TransferExitResultCode getResultCode() {
    return resultCode;
}
}

```

### 관련 태스크

사용자 엑시트를 사용하여 MFT 사용자 정의

### 관련 참조

2070 페이지의 『SourceTransferStartExit.java 인터페이스』

2045 페이지의 『DestinationTransferStartExit.java 인터페이스』

2044 페이지의 『DestinationTransferEndExit.java 인터페이스』

2063 페이지의 『MonitorExit.java 인터페이스』

2064 페이지의 『ProtocolBridgeCredentialExit.java 인터페이스』

## MFT 에이전트 명령 큐에 넣을 수 있는 메시지의 메시지 형식

이러한 XML 스키마는 에이전트가 조치를 수행하도록 요청하기 위해 에이전트 명령 큐에 추가할 수 있는 메시지의 형식을 정의합니다. XML 메시지는 명령행 명령 또는 애플리케이션을 사용하여 에이전트 명령 큐에 추가할 수 있습니다.

- 파일 전송 요청 메시지 형식
- MFT 모니터 요청 메시지 형식
- Ping MFT 에이전트 요청 메시지 형식
- MFT 에이전트 응답 메시지 형식

## 메시징 REST API 참조

messaging REST API에 대한 참조 정보입니다.

messaging REST API 사용에 대한 자세한 정보는 [REST API를 사용하여 메시지 관리를 참조하십시오](#).

### REST API 자원

이 토픽 컬렉션에서는 각 messaging REST API 자원에 대한 참조 정보를 제공합니다.

messaging REST API 사용에 대한 자세한 정보는 [REST API를 사용하여 메시지 관리를 참조하십시오](#).

#### /messaging/qmgr/{qmgrName}/queue/{queueName}/message

메시징 REST API를 사용하면 /messaging/qmgr/{qmgrName}/queue/{queueName}/message 자원을 사용하여 메시지를 큐에 넣거나 메시지를 찾아보거나 큐에서 파괴적으로 가져올 수 있습니다.

## POST/messaging/qmgr/{qmgrName}/queue/{queueName}/message

HTTP POST 메소드를 /messaging/qmgr/{qmgrName}/queue/{queueName}/message 자원과 함께 사용하여 지정된 큐 관리자의 지정된 큐에 메시지를 넣을 수 있습니다.

HTTP 요청 본문을 포함하는 IBM MQ 메시지를 지정된 큐 관리자 및 큐에 넣습니다. 큐 관리자는 mqweb 서버와 같은 시스템에 있어야 합니다. 메소드는 텍스트 기반의 HTTP 요청 본문만 지원합니다. 메시지는 MQSTR 또는 JMS TextMessage 형식의 메시지로 전송되며 현재 사용자 컨텍스트를 사용하여 넣습니다.

REST API V3는 사용자 정의 메시지 특성을 지정하고 메시지 우선순위를 포함하는 기능을 추가합니다. ibm-mq-md-priority 및 ibm-mq-usr 요청 헤더는 REST API V3에서만 사용할 수 있습니다. REST API V3에서 ibm-mq-md-correlationId 요청 헤더의 형식은 다릅니다. 헤더는 애플리케이션 특정 ID이거나 인코딩된 문자열일 경우에는 ID: 접두부가 필요합니다. POST 요청에 사용자 정의 메시지 또는 애플리케이션 특정 상관 ID가 포함된 경우 메시지는 JMS TextMessage(으)로 형식화됩니다.

- [2073 페이지의 『자원 URL』](#)
- [2074 페이지의 『요청 헤더』](#)
- [2076 페이지의 『요청 본문 형식』](#)
- [2076 페이지의 『보안 요구사항』](#)
- [2076 페이지의 『응답 상태 코드』](#)
- [2077 페이지의 『응답 헤더』](#)
- [2078 페이지의 『응답 본문 형식』](#)
- [2078 페이지의 『예』](#)

## 자원 URL

`https://host:port/ibmmq/rest/v1/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

`https://host:port/ibmmq/rest/v2/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

`https://host:port/ibmmq/rest/v3/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

## qmgrName

메시징에 연결할 큐 관리자의 이름을 지정합니다. 큐 관리자는 mqweb 서버와 같은 시스템에 있어야 합니다.

**V 9.4.0** IBM MQ 9.3.3에서 로컬 큐 관리자 또는 리모트 큐 관리자에 연결할 수 있습니다. **qmgrName**에 지정하는 이름은 mqweb 서버가 구성되는 방법에 따라 다릅니다.

- mqweb 서버가 로컬 큐 관리자에만 연결하도록 구성된 경우에는 큐 관리자의 이름을 사용하십시오.
- mqweb 서버가 리모트 큐 관리자를 연결하도록 구성된 경우에는 리모트 큐 관리자 연결 정보에 지정된 대로 큐 관리자의 고유 이름을 사용하십시오.

**dspmweb properties** 명령을 사용하고 **mqRestMessagingConnectionMode** 특성을 보고 mqweb 서버가 로컬 큐 관리자 또는 리모트 큐 관리자에 연결하도록 구성되었는지 여부를 판별할 수 있습니다.

큐 관리자 이름은 대소문자를 구분합니다.

큐 관리자 이름에 슬래시, 마침표 또는 퍼센트 부호가 포함되는 경우 해당 문자는 인코딩된 URL이어야 합니다.

- 슬래시는 %2F(으)로 인코딩되어야 합니다.
- 마침표는 %2E(으)로 인코딩되어야 합니다.
- 백분율 기호는 %25(으)로 인코딩되어야 합니다.

## queueName

메시지를 넣을 큐의 이름을 지정합니다.

큐는 지정된 큐 관리자에 로컬, 리모트 또는 알리어스로서 정의되어야 하며, 클러스터 큐를 참조할 수도 있습니다.

큐 이름은 대소문자를 구분합니다.

큐 이름이 슬래시 또는 퍼센트 부호를 포함하는 경우 해당 문자는 인코딩된 URL이어야 합니다.

- 슬래시(/)는 %2F(으)로 인코딩되어야 합니다.
- 퍼센트 부호(%)는 %25(으)로 인코딩되어야 합니다.

HTTP 연결을 사용으로 설정하는 경우 HTTPS 대신 HTTP를 사용할 수 있습니다. HTTP 사용에 대한 자세한 정보는 [HTTP 및 HTTPS 포트 구성을 참조하십시오](#).

## 요청 헤더

다음 헤더를 요청과 함께 전송해야 합니다.

### 권한 부여

기본 인증을 사용하는 경우 이 헤더를 송신해야 합니다. 자세한 정보는 [REST API로 HTTP 기본 인증 사용](#)을 참조하십시오.

### Content-Type

이 헤더는 다음 값 중 하나로 송신해야 합니다.

- text/plain;charset=utf-8
- text/html;charset=utf-8
- text/xml;charset=utf-8
- application/json;charset=utf-8
- application/xml;charset=utf-8

### ibm-mq-rest-csrf-token

이 헤더가 설정되어야 하지만 값은 공백을 포함한 어떤 값도 될 수 있습니다.

다음 헤더를 요청과 함께 선택적으로 전송할 수 있습니다.

### Accept-Language

이 헤더는 응답 메시지 본문에서 리턴되는 예외 또는 오류 메시지의 필수 언어를 지정합니다.

#### REST API V1 > REST API V2 **ibm-mq-md-correlationId**

이 헤더는 작성된 메시지의 상관 ID를 설정합니다. 헤더는 24바이트를 나타내는 48자의 16진인코딩 문자열로 지정됩니다. "ID:"(으)로 값의 접두부를 지정하지 마십시오. REST API는 해당 문자열을 자동으로 추가합니다.

예를 들면 다음과 같습니다. `ibm-mq-md-correlationId: 414d5120514d4144455620202020202067d8bf5923582e02`

#### REST API V3 **ibm-mq-md-correlationId**

이 헤더는 작성된 메시지의 상관 ID를 설정합니다. 상관 ID는 다음 양식 중 하나를 사용할 수 있습니다.

- 앞에 "ID:" 문자열이 접두어로 붙는 24바이트를 나타내는 48자의 16진 인코딩 문자열. 예: `ibm-mq-md-correlationId: ID:414d5120514d4144455620202020202067d8bf5923582e02`
- 애플리케이션 특정 값입니다. 값은 애플리케이션 특정 문자열입니다. `ibm-mq-md-correlationId: My-Custom-CorrelId`

이 상관 ID 양식을 지정하면 메시지 대상이 WMQ\_CLIENT\_JMS\_COMPLIANT로 지정되므로 MQRFH2 헤더가 통합됩니다.

### ibm-mq-md-expiry

이 헤더는 작성된 메시지의 만기 지속 기간을 설정합니다. 메시지의 만기는 메시지가 큐에 도착하는 시간부터 시작됩니다. 결과 네트워크 지연이 무시됩니다. 헤더는 다음 값 중 하나로 지정해야 합니다.

#### 무제한

메시지가 만료되지 않습니다.

이 값이 기본값입니다.

#### 정수 값

메시지 만기까지 시간(밀리초).

0 - 99999999900 범위로 제한됩니다.

#### ibm-mq-md-persistence

이 헤더는 작성된 메시지의 지속성을 설정합니다. 헤더는 다음 값 중 하나로 지정됩니다.

##### nonPersistent

시스템 장애 또는 큐 관리자 재시작 시 메시지가 지속되지 않습니다.

이 값이 기본값입니다.

##### 지속적

메시지는 시스템 실패 또는 큐 관리자 재시작에서 지속됩니다.

#### REST API V3 **ibm-mq-md-priority**

REST API V3의 경우 이 헤더는 작성된 메시지의 우선순위를 설정합니다. 헤더는 다음 값 중 하나로 지정됩니다.

##### asDestination

메시지는 기본 IBM MQ 큐 오브젝트의 DEFPRTY 속성에 지정된 우선순위를 사용합니다.

이 값이 기본값입니다.

#### 정수 값

0 - 9 범위의 정수로 실제 우선순위를 지정하십시오.

예를 들면 다음과 같습니다. `ibm-mq-md-priority: asDestination`

REST API V1 및 REST API V2의 경우 POST의 메시지 우선순위는 항상 4입니다.

#### ibm-mq-md-replyTo

이 헤더는 작성된 메시지의 응답 대상을 설정합니다. 헤더 형식은 응답 대상 큐 및 선택적 큐 관리자 제공의 표준 표기법을 사용합니다. `replyQueue@[replyQmgr]`

예를 들면 다음과 같습니다. `ibm-mq-md-replyTo: myReplyQueue@myReplyQmgr`

#### REST API V3 **ibm-mq-usr**

이 헤더는 요청 메시지 사용자 정의 특성을 설정합니다.

특성의 구문은 다음과 같습니다.

`ibm-mq-usr: property_name; user_value; user_type`

##### property\_name

지정되는 사용자 특성의 이름입니다. 올바른 JMS 특성 이름이어야 합니다.

##### user\_value

특성의 값입니다.

##### user\_type

특성의 유형은 다음과 같습니다.

- boolean(true/false, MQBOOL)
- byte(8비트 정수, MQINT8)
- short(16비트 정수, MQINT16)
- integer(32비트 정수, MQINT32)
- long(64비트 정수, MQINT64)
- float(32비트 실수, MQFLOAT32)
- double(64비트 실수, MQFLOAT64)
- string(따옴표로 묶인 문자열)

메시지에 여러 특성을 설정할 수 있습니다. 단일 `ibm-mq-usr` 요청 헤더에 여러 개의 쉼표로 구분된 특성을 지정하거나 `ibm-mq-usr` 요청 헤더의 두 개 이상의 개별 인스턴스를 사용할 수 있습니다.



예를 들어, 단일 헤더에서 여러 사용자 정의 특성을 설정할 수 있습니다. `ibm-mq-usr: userPropertyA;5;byte,userPropertyB;-10;integer`

또는 헤더의 개별 인스턴스에서 여러 사용자 정의 특성을 설정할 수 있습니다. `ibm-mq-usr: userPropertyA;5;byte ibm-mq-usr: userPropertyB;-10;integer`

## 요청 본문 형식

요청 본문은 텍스트여야 하며 UTF-8 인코딩을 사용합니다. 특정 텍스트가 필요하지 않습니다. 요청 본문 텍스트를 포함하는 MQSTR 형식화된 메시지가 작성되어 지정된 큐에 배치됩니다.

**REST API V3** REST API V3 사용자 정의 특성 또는 애플리케이션 특정 상관 ID 기능이 사용되는 경우, 요청 본문 텍스트를 포함하는 JMS TextMessage 형식화된 메시지가 작성되어 지정된 큐에 넣습니다.

자세한 정보는 [examples](#)를 참조하십시오.

## 보안 요구사항

호출자를 mqweb 서버에 대해 인증해야 합니다. MQWebAdmin 및 MQWebAdminRO 역할은 messaging REST API에 적용할 수 없습니다. REST API의 보안에 대한 자세한 정보는 [IBM MQ Console 및 REST API 보안을 참조](#) 하십시오.

일단 mqweb 서버에 인증되면 사용자는 messaging REST API 및 administrative REST API 모두를 사용할 수 있습니다.

지정된 큐에 메시지를 넣기 위한 기능을 호출자의 보안 프린시펄에 부여해야 합니다.

- 자원 URL의 `{queueName}` 부분에 지정된 큐는 PUT 사용 가능해야 합니다.
- **MQ Appliance** **ALW** 자원 URL의 `{queueName}` 부분에 의해 지정되는 큐의 경우, 호출자의 보안 프린시펄에 +PUT 권한을 부여해야 합니다.
- **z/OS** 자원 URL의 `{queueName}` 부분에 의해 지정되는 큐의 경우, 호출자의 보안 프린시펄에 UPDATE 액세스 권한을 부여해야 합니다.

이 보안 프린시펄은 mqweb 서버를 시작한 사용자 또는 mqweb 서버에 로그인한 사용자일 수 있습니다. 연결하는 큐 관리자가 리모트 큐 관리자인 경우, 보안 프린시펄은 대신 리모트 큐 관리자 연결 정보의 신임 정보에 의해 제공되는 사용자이거나 채널 보안 규칙에 의해 판별되는 다른 사용자일 수 있습니다. 자세한 정보는 [messaging REST API](#)에서 사용되는 보안 프린시펄 판별을 참조하십시오.

**ALW** AIX, Linux, and Windows에서 `setmqaut` 명령을 사용하여 IBM MQ 자원을 사용하도록 보안 프린시펄에 권한을 부여할 수 있습니다. 자세한 정보는 [setmqaut\(권한 부여 또는 취소\)](#)를 참조하십시오.

z/OS의 경우 [z/OS](#)에서 보안 설정을 참조하십시오.

messaging REST API에서 AMS(Advanced Message Security)를 사용하는 경우에는 모든 메시지가 메시지를 게시하는 사용자의 컨텍스트가 아닌 mqweb 서버의 컨텍스트를 사용하여 암호화됨을 유념하십시오.

## 응답 상태 코드

### 201

메시지가 작성되고 송신되었습니다.

### 400

올바르지 않은 데이터가 제공되었습니다.

예를 들어, 올바른 요청 헤더 값이 지정되었습니다.

### 401

인증되지 않았습니다.

호출자는 mqweb 서버에 대해 인증되어야 하며 MQWebAdmin, MQWebAdminRO 또는 MQWebUser 역할 중 하나 이상의 구성원이어야 합니다. `ibm-mq-rest-csrf-token` 헤더도 지정해야 합니다. 자세한 정보는 [2076 페이지의 『보안 요구사항』](#)의 내용을 참조하십시오.



403

권한이 부여되지 않았습니다.

호출자가 mqweb 서버에 대해 인증되었거나 올바른 프린시פל과 연관되어 있습니다. 그러나 프린시פל에 필수 IBM MQ 자원의 전체 또는 서브세트에 대한 액세스 권한이 없거나 MQWebUser 역할에 없습니다. 필수인 액세스에 대한 자세한 정보는 2076 페이지의 『보안 요구사항』의 내용을 참조하십시오.

404

큐가 없습니다.

405

큐에서 PUT이 금지됩니다.

415

메시지 헤더 또는 본문은 지원되지 않는 매체 유형입니다.

예를 들어, Content-Type 헤더가 지원되지 않는 매체 유형으로 설정되어 있습니다.

500

IBM MQ의 서버 문제 또는 오류 코드입니다.

502

메시징 제공자가 필수 함수를 지원하지 않으므로 현재 보안 프린시פל에서 메시지를 송신할 수 없습니다. 예를 들어 mqweb 서버 클래스 경로가 올바르지 않은 경우입니다.

503

큐 관리자가 실행 중이지 않습니다.

## 응답 헤더

다음 헤더가 응답과 함께 리턴됩니다.

### Content-Language

오류 또는 예외의 경우 응답 메시지의 언어 ID를 지정합니다. 오류 또는 예외 조건에 필요한 언어를 나타내기 위해 Accept-Language 요청 헤더와 함께 사용됩니다. 요청된 언어가 지원되지 않는 경우 mqweb 서버 기본값이 사용됩니다.

### Content-Length

컨텐츠가 없는 경우라도 HTTP 응답 본문의 길이를 지정합니다. 성공 시 값은 0(영)입니다.

### Content-Type

응답의 본문을 지정합니다. 성공하면 값은 text/plain;charset=utf-8입니다. 오류 또는 예외가 발생할 경우 값은 application/json;charset=utf-8입니다.

### REST API V1 ▶ REST API V2 **ibm-mq-md-messageId**

IBM MQ가 이 메시지에 할당하는 메시지 ID를 지정합니다. ibm-mq-md-correlationId 요청 헤더와 마찬가지로, 24바이트를 나타내는 48자의 16진 인코딩 문자열로 표시됩니다.

예를 들면, 다음과 같습니다.

```
ibm-mq-md-messageId: 414d5120514d4144455620202020202067d8ce5923582f07
```

### REST API V3 **ibm-mq-md-messageId**

IBM MQ가 이 메시지에 할당하는 메시지 ID를 지정합니다. ibm-mq-md-correlationId 요청 헤더와 마찬가지로 문자열 ID: 이(가) 앞에 붙은 24바이트를 나타내는 48자의 16진 인코딩 문자열로 표시됩니다.

예를 들면, 다음과 같습니다.

```
ibm-mq-md-messageId: ID:414d5120514d4144455620202020202067d8ce5923582f07
```

### ▶ V 9.4.0 **ibm-mq-resolved-qmgr**

요청에 사용된 큐 관리자의 이름을 지정합니다. 자원 URL에서 고유 이름이 사용된 경우 이 헤더는 해당 고유 이름과 연관된 큐 관리자의 이름을 식별합니다. 자원 URL에 사용된 고유 이름이 큐 관리자 그룹을 참조하는 경우 이 헤더는 그룹 내에서 사용된 큐 관리자를 식별합니다.

## 응답 본문 형식

메시지가 송신되면 응답 본문은 비어 있습니다. 오류가 발생하면 응답 본문이 오류 메시지를 포함합니다. 자세한 정보는 [REST API 오류 처리](#)를 참조하십시오.

## 예

다음 예제는 v2 자원 URL을 사용합니다. IBM MQ 9.1.5 이전의 IBM MQ 버전을 사용하는 경우 대신 v1 자원 URL을 사용해야 합니다. 즉, 리소스 URL에서 예제 URL이 v2을(를) 사용하는 v1을(를) 대체합니다.

다음 예제는 비밀번호가 mquser인 사용자 mquser에 로그인합니다. cURL에서 요청의 로그는 다음 Windows 예제와 같을 수 있습니다. LTPA 토큰은 -c 플래그를 사용하여 cookiejar.txt 파일에 저장됩니다.

```
curl -k "https://localhost:9443/ibmmq/rest/v2/login" -X POST
-H "Content-Type: application/json" --data "{\"username\":\"mquser\", \"password\":\"mquser\"}"
-c c:\cookiejar.txt
```

사용자가 로그인되면 LTPA 토큰 및 ibm-mq-rest-csrf-token HTTP 헤더가 추가 요청을 인증하는 데 사용됩니다. ibm-mq-rest-csrf-token token\_value은(는) 공백을 포함하여 임의의 값이 될 수 있습니다.

- 다음 Windows cURL 예제는 기본 옵션을 사용하여 큐 관리자 QM1의 큐 Q1에 메시지를 보냅니다. 메시지에는 "Hello World!" 텍스트가 들어 있습니다.

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/Q1/message"
-X POST -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token_value"
-H "Content-Type: text/plain; charset=utf-8" --data "Hello World!"
```

- 다음 Windows cURL 예제는 큐 관리자 QM1의 큐 Q1에 지속 메시지를 전송하며, 만기는 2분입니다. 메시지는 "Hello World!" 텍스트가 들어 있습니다.

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/Q1/message"
-X POST -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token_value"
-H "Content-Type: text/plain; charset=utf-8" -H "ibm-mq-md-persistence: persistent"
-H "ibm-mq-md-expiry: 120000" --data "Hello World!"
```

- 다음 Windows cURL에는 만기 및 정의된 상관 ID가 없는 큐 관리자 QM1의 큐 Q1에 비지속 메시지를 보냅니다. 메시지는 "Hello World!" 텍스트가 들어 있습니다.

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/Q1/message"
-X POST -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token_value"
-H "Content-Type: text/plain; charset=utf-8" -H "ibm-mq-md-persistence: nonPersistent"
-H "ibm-mq-md-expiry: unlimited" -H "ibm-mq-md-correlationId:
414d5120514d41444455620202020202067d8b
f5923582e02" --data "Hello World!"
```

## **GET/messaging/qmgr/{qmgrName}/queue/{queueName}/message**

/messaging/qmgr/{qmgrName}/queue/{queueName}/message 자원과 HTTP GET 메소드를 함께 사용하여 연관된 큐 관리자 및 큐에서 메시지를 찾아볼 수 있습니다.

지정된 큐 관리자 및 큐에서 첫 번째 사용 가능한 메시지를 찾아봅니다. 큐 관리자는 mqweb 서버와 같은 시스템에 있어야 합니다. 메시지 본문이 HTTP 응답 본문에 리턴됩니다. 메시지의 형식은 MQSTR 또는 JMS TextMessage여야 하며 현재 사용자 컨텍스트를 사용하여 수신됩니다.

모든 메시지는 큐에 그대로 남아 있으며 부적절한 메시지에 대한 적절한 상태 코드가 호출자에 리턴됩니다. 예를 들어, MQSTR 또는 JMS TextMessage 형식이 아닌 메시지가 있습니다.

REST API V3는 사용자 정의 메시지 특성을 지정하고 메시지에 메시지 우선순위를 포함하는 기능을 추가합니다. ibm-mq-md-priority 및 ibm-mq-usr 응답 헤더는 REST API V3에서만 사용할 수 있습니다. REST API V3에서 ibm-mq-md-correlationId 요청 헤더의 형식은 다릅니다. 헤더는 애플리케이션 특정 ID이거나 인코딩된 문자열인 경우 ID: 접두부를 유지합니다. ibm-mq-md-messageId 응답 헤더 및 조회 매개변수는 REST API V3에서 형식이 다르며, ID: 접두부를 유지합니다.

- [2079 페이지의 『자원 URL』](#)
- [2079 페이지의 『선택적 조회 매개변수』](#)

- 2080 페이지의 『요청 헤더』
- 2081 페이지의 『요청 본문 형식』
- 2081 페이지의 『보안 요구사항』
- 2081 페이지의 『응답 상태 코드』
- 2082 페이지의 『응답 헤더』
- 2084 페이지의 『응답 본문 형식』
- 2084 페이지의 『예』

## 자원 URL

`https://host:port/ibmmq/rest/v1/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

`https://host:port/ibmmq/rest/v2/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

`https://host:port/ibmmq/rest/v3/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

### qmgrName

메시징에 연결할 큐 관리자의 이름을 지정합니다. 큐 관리자는 mqweb 서버와 같은 시스템에 있어야 합니다.

**V 9.4.0** IBM MQ 9.3.3에서 로컬 큐 관리자 또는 리모트 큐 관리자에 연결할 수 있습니다. **qmgrName**에 지정하는 이름은 mqweb 서버가 구성되는 방법에 따라 다릅니다.

- mqweb 서버가 로컬 큐 관리자에만 연결하도록 구성된 경우에는 큐 관리자의 이름을 사용하십시오.
- mqweb 서버가 리모트 큐 관리자를 연결하도록 구성된 경우에는 리모트 큐 관리자 연결 정보에 지정된 대로 큐 관리자의 고유 이름을 사용하십시오.

**dspmweb properties** 명령을 사용하고 **mqRestMessagingConnectionMode** 특성을 보고 mqweb 서버가 로컬 큐 관리자 또는 리모트 큐 관리자에 연결하도록 구성되었는지 여부를 판별할 수 있습니다.

큐 관리자 이름은 대소문자를 구분합니다.

큐 관리자 이름에 슬래시, 마침표 또는 퍼센트 부호가 포함되는 경우 해당 문자는 인코딩된 URL이어야 합니다.

- 슬래시(/)는 %2F(으)로 인코딩되어야 합니다.
- 백분율 기호(%)는 %25(으)로 인코딩되어야 합니다.

### queueName

메시지를 찾을 큐의 이름을 지정합니다.

큐는 로컬 또는 로컬 큐를 나타내는 알리어스로 정의되어야 합니다.

큐 이름은 대소문자를 구분합니다.

큐 이름이 슬래시 또는 퍼센트 부호를 포함하는 경우 해당 문자는 인코딩된 URL이어야 합니다.

- 슬래시(/)는 %2F(으)로 인코딩되어야 합니다.
- 퍼센트 부호(%)는 %25(으)로 인코딩되어야 합니다.

HTTP 연결을 사용으로 설정하는 경우 HTTPS 대신 HTTP를 사용할 수 있습니다. HTTP 사용에 대한 자세한 정보는 [HTTP 및 HTTPS 포트 구성을 참조하십시오](#).

## 선택적 조회 매개변수

**REST API V1** **REST API V2** **correlationId=hexValue**

HTTP 메소드가 해당하는 상관 ID를 가진 다음 메시지를 리턴하도록 지정합니다.

### hexValue

조회 매개변수는 48자의 16진 인코딩 문자열로서 지정되어야 하고, 이는 24바이트를 나타냅니다.

예를 들면, 다음과 같습니다.

```
../message?correlationId=414d5120514d4144455620202020202067d8bf5923582e02
```

### REST API V3 **correlationId=ID:hexValue** 또는 **correlationId=application\_specific\_value**

HTTP 메소드가 해당하는 상관 ID를 가진 다음 메시지를 리턴하도록 지정합니다.

#### hexValue

조회 매개변수는 24바이트를 나타내는 48자의 16진 인코딩 문자열로 지정해야 하며, 앞에 "ID:" 문자열이 접두어로 붙습니다.

예를 들면, 다음과 같습니다.

```
../message?correlationId=ID:414d5120514d4144455620202020202067d8bf5923582e02
```

#### application\_specific\_value

조회 매개변수는 애플리케이션 특정 문자열로 지정될 수 있습니다.

예를 들면, 다음과 같습니다.

```
../message?correlationId=My-Custom-CorrelId
```

### REST API V1 > REST API V2 **messageId=hexValue**

HTTP 메소드가 해당하는 메시지 ID를 가진 다음 메시지를 리턴하도록 지정합니다.

#### hexValue

조회 매개변수는 48자의 16진 인코딩 문자열로서 지정되어야 하고, 이는 24바이트를 나타냅니다.

예를 들면, 다음과 같습니다.

```
../message?messageId=414d5120514d4144455620202020202067d8ce5923582f07
```

### REST API V3 **messageId=ID:hexValue**

HTTP 메소드가 해당하는 메시지 ID를 가진 다음 메시지를 리턴하도록 지정합니다.

#### hexValue

조회 매개변수는 24바이트를 나타내는 48자의 16진 인코딩 문자열로 지정해야 하며, 앞에 "ID:" 문자열이 접두어로 붙습니다.

예를 들면, 다음과 같습니다.

```
../message?messageId=ID:414d5120514d4144455620202020202067d8ce5923582f07
```

## 요청 헤더

다음 헤더를 요청과 함께 전송해야 합니다.

### 권한 부여

기본 인증을 사용하는 경우 이 헤더를 송신해야 합니다. 자세한 정보는 [REST API로 HTTP 기본 인증 사용](#)을 참조하십시오.

### ibm-mq-rest-csrf-token

이 헤더가 설정되어야 하지만 값은 공백을 포함한 어떤 값도 될 수 있습니다.

다음 헤더를 요청과 함께 선택적으로 전송할 수 있습니다.

### Accept-Charset

이 헤더를 사용하여 응답에 수용할 수 있는 문자 세트를 표시할 수 있습니다. 지정된 경우 이 헤더는 UTF-8(으)로 설정되어야 합니다.

### Accept-Language

이 헤더는 응답 메시지 본문에서 리턴되는 예외 또는 오류 메시지의 필수 언어를 지정합니다.

## 요청 본문 형식




없음

## 보안 요구사항

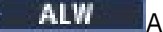
호출자를 mqweb 서버에 대해 인증해야 합니다. MQWebAdmin 및 MQWebAdminRO 역할은 messaging REST API에 적용할 수 없습니다. REST API의 보안에 대한 자세한 정보는 [IBM MQ Console 및 REST API 보안을 참조하십시오](#).

일단 mqweb 서버에 인증되면 사용자는 messaging REST API 및 administrative REST API 모두를 사용할 수 있습니다.

지정된 큐에서 메시지를 찾아볼 수 있는 기능을 호출자의 보안 프린시펄에 부여해야 합니다.

- 자원 URL의 `{queueName}` 부분에 지정된 큐는 BROWSE 사용 가능해야 합니다.
-   자원 URL의 `{queueName}` 부분에 의해 지정되는 큐의 경우, 호출자의 보안 프린시펄에 +GET, +INQ 및 +BROWSE 권한이 부여되어야 합니다.
-  자원 URL UPDATE의 `{queueName}` 부분에 의해 지정되는 큐의 경우, 호출자의 보안 프린시펄에 액세스가 부여되어야 합니다.

이 보안 프린시펄은 mqweb 서버를 시작한 사용자 또는 mqweb 서버에 로그인한 사용자일 수 있습니다. 연결하는 큐 관리자가 리모트 큐 관리자인 경우, 보안 프린시펄은 대신 리모트 큐 관리자 연결 정보의 신임 정보에 의해 제공되는 사용자이거나 채널 보안 규칙에 의해 판별되는 다른 사용자일 수 있습니다. 자세한 정보는 [messaging REST API에서 사용되는 보안 프린시펄 판별을 참조하십시오](#).

 에서 `setmqaut` 명령을 사용하여 IBM MQ 자원을 사용하도록 보안 프린시펄에 권한을 부여할 수 있습니다. 자세한 정보는 [setmqaut\(권한 부여 또는 취소\)](#)를 참조하십시오.

z/OS의 경우 [z/OS에서 보안 설정을 참조하십시오](#).

## 응답 상태 코드

### 200

메시지가 수신되었습니다.

### 204

사용 가능한 메시지가 없습니다.

### 400

올바르지 않은 데이터가 제공되었습니다.

예를 들어, 올바른 조회 매개변수 값이 지정되었습니다.

### 401

인증되지 않았습니다.

호출자는 mqweb 서버에 대해 인증되어야 하며 MQWebAdmin, MQWebAdminRO 또는 MQWebUser 역할 중 하나 이상의 구성원이어야 합니다. `ibm-mq-rest-csrf-token` 헤더도 지정해야 합니다. 자세한 정보는 [2081 페이지의 『보안 요구사항』](#)의 내용을 참조하십시오.

### 403

권한이 부여되지 않았습니다.

호출자가 mqweb 서버에 대해 인증되었거나 올바른 프린시펄과 연관되어 있습니다. 그러나 프린시펄에 필수 IBM MQ 자원의 전체 또는 서브세트에 대한 액세스 권한이 없거나 MQWebUser 역할에 없습니다. 필수인 액세스에 대한 자세한 정보는 [2081 페이지의 『보안 요구사항』](#)의 내용을 참조하십시오.

### 404

큐가 없습니다.

### 500

IBM MQ의 서버 문제 또는 오류 코드입니다.

### 501

HTTP 응답을 구성할 수 없습니다.

예를 들어 수신된 메시지의 유형이 올바르지 않거나 유형이 올바르지만 본문을 처리할 수 없습니다.

### 502

메시징 제공자가 필수 함수를 지원하지 않으므로 현재 보안 프린시펄에서 메시징을 수신할 수 없습니다. 예를 들어 mqweb 서버 클래스 경로가 올바르지 않은 경우입니다.

### 503

큐 관리자가 실행 중이지 않습니다.

## 응답 헤더

다음 헤더가 응답과 함께 리턴됩니다.

### Content-Language

오류 또는 예외의 경우 응답 메시지의 언어 ID를 지정합니다. 오류 또는 예외 조건에 필요한 언어를 나타내기 위해 Accept-Language 요청 헤더와 함께 사용됩니다. 요청된 언어가 지원되지 않는 경우 mqweb 서버 기본값이 사용됩니다.

### Content-Length

컨텐츠가 없는 경우라도 HTTP 응답 본문의 길이를 지정합니다. 값에는 메시지 데이터의 길이(바이트)가 들어 있습니다.

### Content-Type

수신된 메시지의 응답 본문에서 리턴되는 컨텐츠 유형을 지정합니다. 성공하면 값은 text/plain;charset=utf-8입니다. 오류 또는 예외가 발생할 경우 값은 application/json;charset=utf-8입니다.

### REST API V1 > REST API V2 **ibm-mq-md-correlationId**

수신된 메시지의 상관 ID를 지정합니다. 수신된 메시지에 올바른 상관 ID가 들어 있는 경우 헤더가 리턴됩니다. 이는 48자 16진 인코딩 문자열로 표시되며, 24바이트를 나타냅니다.

예를 들면, 다음과 같습니다.

```
ibm-mq-md-correlationId: 414d5120514d414445562020202020202067d8bf5923582e02
```

### REST API V3 **ibm-mq-md-correlationId**

수신된 메시지의 상관 ID를 지정합니다. 수신된 메시지에 올바른 상관 ID가 들어 있는 경우 헤더가 리턴됩니다. 상관 ID는 다음 양식 중 하나를 사용할 수 있습니다.

- 앞에 "ID:" 문자열이 접두어로 붙는 24바이트를 나타내는 48자의 16진 인코딩 문자열. 예를 들면, 다음과 같습니다.

```
ibm-mq-md-correlationId: ID:414d5120514d414445562020202020202067d8bf5923582e02
```

- 애플리케이션 특정 값입니다. 값은 애플리케이션 특정 문자열입니다.

```
ibm-mq-md-correlationId: My-Custom-CorrelId
```

### ibm-mq-md-expiry

수신된 메시지의 남아 있는 만기 지속 기간을 지정합니다. 헤더는 다음 중 하나입니다.

#### 무제한

메시지가 만료되지 않습니다.

#### 정수 값

메시지 만기까지 남아 있는 시간(밀리초).

### REST API V1 > REST API V2 **ibm-mq-md-messageId**

IBM MQ가 이 메시지에 할당하는 메시지 ID를 지정합니다. ibm-mq-md-correlationId 헤더와 마찬가지로, 24바이트를 나타내는 48자의 16진 인코딩 문자열로 표시됩니다.

예를 들면, 다음과 같습니다.

```
ibm-mq-md-messageId: 414d5120514d4144455620202020202067d8ce5923582f07
```

### REST API V3 **ibm-mq-md-messageId**

IBM MQ가 이 메시지에 할당하는 메시지 ID를 지정합니다. `ibm-mq-md-correlationId` 헤더와 마찬가지로, 앞에 "ID:" 문자열이 접두어로 붙는 24바이트를 나타내는 48자의 16진 인코딩 문자열로 표시됩니다.

예를 들면, 다음과 같습니다.

```
ibm-mq-md-messageId: ID:414d5120514d4144455620202020202067d8ce5923582f07
```

### **ibm-mq-md-persistence**

수신된 메시지의 지속성을 지정합니다. 헤더는 다음 중 하나입니다.

#### **nonPersistent**

시스템 장애 또는 큐 관리자 재시작 시 메시지가 지속되지 않습니다.

#### **지속적**

메시지는 시스템 실패 또는 큐 관리자 재시작에서 지속됩니다.

### REST API V3 **ibm-mq-md-priority**

메시지 우선순위의 설정을 리턴합니다. 예를 들면, 다음과 같습니다.

```
ibm-mq-md-priority: 3
```

### **ibm-mq-md-replyTo**

수신된 메시지의 응답 대상을 지정합니다. 헤더 형식은 응답 대상 큐 및 큐 관리자 `replyQueue@replyQmgr`의 표준 표기법을 사용합니다.

예를 들면, 다음과 같습니다.

```
ibm-mq-md-replyTo: myReplyQueue@myReplyQMGr
```

### V 9.4.0 **ibm-mq-resolved-qmgr**

요청에 사용된 큐 관리자의 이름을 지정합니다. 자원 URL에서 고유 이름이 사용된 경우 이 헤더는 해당 고유 이름과 연관된 큐 관리자의 이름을 식별합니다. 자원 URL에 사용된 고유 이름이 큐 관리자 그룹을 참조하는 경우 이 헤더는 그룹 내에서 사용된 큐 관리자를 식별합니다.

### REST API V3 **ibm-mq-usr**

메시지 사용자 정의 특성을 리턴합니다. 메시지에 여러 개의 특성을 설정할 수 있으며, 이 경우 `ibm-mq-usr` 응답 헤더에 대한 두 개 이상의 개별 인스턴스가 있습니다.

예를 들면, 다음과 같습니다.

```
ibm-mq-usr: myIPProp;5;short
ibm-mq-usr: mySProp;"hi";string
ibm-mq-usr: myBProp>true;boolean
```

특성의 구문은 다음과 같습니다.

```
ibm-mq-usr: property_name; user_value; user_type
```

#### **property\_name**

지정되는 사용자 특성의 이름입니다. 올바른 JMS 특성 이름이어야 합니다.

#### **user\_value**

특성의 값입니다.

#### **user\_type**

특성의 유형은 다음과 같습니다.







```
-X GET -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token-value"
-H "Accept: text/plain"
```

## **/messaging/qmgr/{qmgrName}/queue/{queueName}/message 삭제**

/messaging/qmgr/{qmgrName}/queue/{queueName}/message 자원과 함께 HTTP DELETE 메소드를 사용하여 연관된 큐 관리자 및 큐에서 파괴적으로 메시지를 가져올 수 있습니다.

HTTP 응답 본문에 있는 메시지 본문을 리턴하며, 지정된 큐 관리자 및 큐에서 다음으로 사용 가능한 메시지를 비공개로 가져옵니다. 큐 관리자는 mqweb 서버와 같은 시스템에 있어야 합니다. 메시지의 형식은 MQSTR 또는 JMS TextMessage이어야 하며 현재 사용자 컨텍스트를 사용하여 수신됩니다.

호환되지 않는 메시지는 큐에 남겨지며 적절한 상태 코드가 호출자에게 리턴됩니다. 예를 들어, MQSTR 또는 JMS TextMessage 형식이 아닌 메시지가 있습니다.

REST API V3는 사용자 정의 메시지 특성을 지정하고 메시지에 메시지 우선순위를 포함하는 기능을 추가합니다. ibm-mq-md-priority 및 ibm-mq-usr 응답 헤더는 REST API V3에서만 사용할 수 있습니다. REST API V3에서 ibm-mq-md-correlationId 요청 헤더의 형식은 다릅니다. 헤더는 애플리케이션 특정 ID이거나 인코딩된 문자열인 경우 ID: 접두부를 유지합니다. ibm-mq-md-messageId 응답 헤더 및 조회 매개변수는 REST API V3에서 형식이 다르며, ID: 접두부를 유지합니다.

- [2085 페이지의 『자원 URL』](#)
- [2086 페이지의 『선택적 조회 매개변수』](#)
- [2087 페이지의 『요청 헤더』](#)
- [2087 페이지의 『요청 본문 형식』](#)
- [2087 페이지의 『보안 요구사항』](#)
- [2088 페이지의 『응답 상태 코드』](#)
- [2088 페이지의 『응답 헤더』](#)
- [2090 페이지의 『응답 본문 형식』](#)
- [2091 페이지의 『예』](#)

## **자원 URL**

https://host:port/ibmmq/rest/v1/messaging/qmgr/{qmgrName}/queue/{queueName}/message

https://host:port/ibmmq/rest/v2/messaging/qmgr/{qmgrName}/queue/{queueName}/message

https://host:port/ibmmq/rest/v3/messaging/qmgr/{qmgrName}/queue/{queueName}/message

## **qmgrName**

메시징에 연결할 큐 관리자의 이름을 지정합니다. 큐 관리자는 mqweb 서버와 같은 시스템에 있어야 합니다.

**V 9.4.0** IBM MQ 9.3.3에서 로컬 큐 관리자 또는 리모트 큐 관리자에 연결할 수 있습니다. **qmgrName**에 지정하는 이름은 mqweb 서버가 구성되는 방법에 따라 다릅니다.

- mqweb 서버가 로컬 큐 관리자에만 연결하도록 구성된 경우에는 큐 관리자의 이름을 사용하십시오.
- mqweb 서버가 리모트 큐 관리자를 연결하도록 구성된 경우에는 리모트 큐 관리자 연결 정보에 지정된 대로 큐 관리자의 고유 이름을 사용하십시오.

**dspmqweb properties** 명령을 사용하고 **mqRestMessagingConnectionMode** 특성을 보고 mqweb 서버가 로컬 큐 관리자 또는 리모트 큐 관리자에 연결하도록 구성되었는지 여부를 판별할 수 있습니다.

큐 관리자 이름은 대소문자를 구분합니다.

큐 관리자 이름에 슬래시, 마침표 또는 퍼센트 부호가 포함되는 경우 해당 문자는 인코딩된 URL이어야 합니다.

- 슬래시(/)는 %2F(으)로 인코딩되어야 합니다.

- 백분율 기호(%)는 %25(으)로 인코딩되어야 합니다.

### queueName

다음 대상물 가져올 큐의 이름을 지정합니다.

큐는 로컬 또는 로컬 큐를 나타내는 알리어스인 것으로 정의되어야 합니다.

큐 이름은 대소문자를 구분합니다.

큐 이름이 슬래시 또는 퍼센트 부호를 포함하는 경우 해당 문자는 인코딩된 URL이어야 합니다.

- 슬래시(/)는 %2F(으)로 인코딩되어야 합니다.
- 퍼센트 부호(%)는 %25(으)로 인코딩되어야 합니다.

HTTP 연결을 사용으로 설정하는 경우 HTTPS 대신 HTTP를 사용할 수 있습니다. HTTP 사용에 대한 자세한 정보는 [HTTP 및 HTTPS 포트 구성을 참조하십시오](#).

## 선택적 조회 매개변수

### REST API V1 > REST API V2 correlationId=hexValue

HTTP 메소드가 해당하는 상관 ID를 가진 다음 메시지를 리턴하도록 지정합니다.

#### hexValue

조회 매개변수는 48자의 16진 인코딩 문자열로서 지정되어야 하고, 이는 24바이트를 나타냅니다.

예를 들면, 다음과 같습니다.

```
../message?correlationId=414d5120514d4144455620202020202067d8bf5923582e02
```

### REST API V3 correlationId=ID:hexValue 또는 correlationId=application\_specific\_value

HTTP 메소드가 해당하는 상관 ID를 가진 다음 메시지를 리턴하도록 지정합니다.

#### hexValue

조회 매개변수는 24바이트를 나타내는 48자의 16진 인코딩 문자열로 지정해야 하며, 앞에 "ID:" 문자열이 접두어로 붙습니다.

예를 들면, 다음과 같습니다.

```
../message?correlationId=ID:414d5120514d4144455620202020202067d8bf5923582e02
```

#### application\_specific\_value

조회 매개변수는 애플리케이션 특정 문자열로 지정될 수 있습니다.

예를 들면, 다음과 같습니다.

```
../message?correlationId=My-Custom-CorrelId
```

### REST API V1 > REST API V2 messageId=hexValue

HTTP 메소드가 해당하는 메시지 ID를 가진 다음 메시지를 리턴하도록 지정합니다.

#### hexValue

조회 매개변수는 48자의 16진 인코딩 문자열로서 지정되어야 하고, 이는 24바이트를 나타냅니다.

예를 들면, 다음과 같습니다.

```
../message?messageId=414d5120514d4144455620202020202067d8ce5923582f07
```

### REST API V3 messageId=ID:hexValue

HTTP 메소드가 해당하는 메시지 ID를 가진 다음 메시지를 리턴하도록 지정합니다.

#### hexValue

조회 매개변수는 24바이트를 나타내는 48자의 16진 인코딩 문자열로 지정해야 하며, 앞에 "ID:" 문자열이 접두어로 붙습니다.

예를 들면, 다음과 같습니다.

```
../message?messageId=ID:414d5120514d41444556202020202067d8ce5923582f07
```

### **wait=integerValue**

HTTP 메소드가 다음 메시지가 사용 가능하게 될 때까지 *integerValue* 밀리초 동안 대기하도록 지정합니다.

### **integerValue**

조회 매개변수는 밀리초 기간을 나타내는 정수 값으로 지정되어야 합니다. 최대값은 2147483647입니다.

예를 들면, 다음과 같습니다.

```
../message?wait=120000
```

## 요청 헤더

다음 헤더를 요청과 함께 전송해야 합니다.

### 권한 부여

기본 인증을 사용하는 경우 이 헤더를 송신해야 합니다. 자세한 정보는 [REST API로 HTTP 기본 인증 사용](#)을 참조하십시오.

### **ibm-mq-rest-csrf-token**

이 헤더가 설정되어야 하지만 값은 공백을 포함한 어떤 값도 될 수 있습니다.

다음 헤더를 요청과 함께 선택적으로 전송할 수 있습니다.

### **Accept-Charset**

이 헤더를 사용하여 응답에 수용할 수 있는 문자 세트를 표시할 수 있습니다. 지정된 경우 이 헤더는 UTF-8(으)로 설정되어야 합니다.

### **Accept-Language**

이 헤더는 응답 메시지 본문에서 리턴되는 예외 또는 오류 메시지의 필수 언어를 지정합니다.

## 요청 본문 형식




없음

## 보안 요구사항

호출자를 mqweb 서버에 대해 인증해야 합니다. MQWebAdmin 및 MQWebAdminRO 역할은 messaging REST API에 적용할 수 없습니다. REST API의 보안에 대한 자세한 정보는 [IBM MQ Console 및 REST API 보안](#)을 참조하십시오.

일단 mqweb 서버에 인증되면 사용자는 messaging REST API 및 administrative REST API 모두를 사용할 수 있습니다.

지정된 큐에서 메시지를 가져오기 위한 기능을 호출자의 보안 프린시펄에 부여해야 합니다.

- 자원 URL의 *{queueName}* 부분에 지정된 큐는 GET 사용 가능해야 합니다.
-   자원 URL의 *{queueName}* 부분에 의해 지정되는 큐의 경우, 호출자의 보안 프린시펄에 +GET, +INQ 및 +BROWSE 권한이 부여되어야 합니다.
-  자원 URL UPDATE의 *{queueName}* 부분에 의해 지정되는 큐의 경우, 호출자의 보안 프린시펄에 액세스가 부여되어야 합니다.

이 보안 프린시펄은 mqweb 서버를 시작한 사용자 또는 mqweb 서버에 로그인한 사용자일 수 있습니다. 연결하는 큐 관리자가 리모트 큐 관리자인 경우, 보안 프린시펄은 대신 리모트 큐 관리자 연결 정보의 신임 정보에 의해 제공되는 사용자이거나 채널 보안 규칙에 의해 판별되는 다른 사용자일 수 있습니다. 자세한 정보는 [messaging REST API](#)에서 사용되는 보안 프린시펄 판별을 참조하십시오.

**ALW** AIX, Linux, and Windows에서 **setmqaut** 명령을 사용하여 IBM MQ 자원을 사용하도록 보안 프린시펄에 권한을 부여할 수 있습니다. 자세한 정보는 [setmqaut\(권한 부여 또는 취소\)](#)를 참조하십시오.  
z/OS의 경우 [z/OS에서 보안 설정을 참조하십시오](#).

## 응답 상태 코드

### 200

메시지가 수신되었습니다.

### 204

사용 가능한 메시지가 없습니다.

### 400

올바르지 않은 데이터가 제공되었습니다.

예를 들어, 올바르지 않은 조회 매개변수 값이 지정되었습니다.

### 401

인증되지 않았습니다.

호출자는 mqweb 서버에 대해 인증되어야 하며 MQWebAdmin, MQWebAdminRO 또는 MQWebUser 역할 중 하나 이상의 구성원이어야 합니다. `ibm-mq-rest-csrf-token` 헤더도 지정해야 합니다. 자세한 정보는 [2087 페이지의 『보안 요구사항』](#)의 내용을 참조하십시오.

### 403

권한이 부여되지 않았습니다.

호출자가 mqweb 서버에 대해 인증되었거나 올바른 프린시펄과 연관되어 있습니다. 그러나 프린시펄에 필수 IBM MQ 자원의 전체 또는 서브세트에 대한 액세스 권한이 없거나 MQWebUser 역할에 없습니다. 필수인 액세스에 대한 자세한 정보는 [2087 페이지의 『보안 요구사항』](#)의 내용을 참조하십시오.

### 404

큐가 없습니다.

### 405

큐에서 GET이 금지됩니다.

### 500

IBM MQ의 서버 문제 또는 오류 코드입니다.

### 501

HTTP 응답을 구성할 수 없습니다.

예를 들어 수신된 메시지의 유형이 올바르지 않거나 유형이 올바르지만 본문을 처리할 수 없습니다.

### 502

메시징 제공자가 필수 함수를 지원하지 않으므로 현재 보안 프린시펄에서 메시지를 수신할 수 없습니다. 예를 들어 mqweb 서버 클래스 경로가 올바르지 않은 경우입니다.

### 503

큐 관리자가 실행 중이지 않습니다.

## 응답 헤더

다음 헤더가 응답과 함께 리턴됩니다.

### Content-Language

오류 또는 예외의 경우 응답 메시지의 언어 ID를 지정합니다. 오류 또는 예외 조건에 필요한 언어를 나타내기 위해 Accept-Language 요청 헤더와 함께 사용됩니다. 요청된 언어가 지원되지 않는 경우 mqweb 서버 기본값이 사용됩니다.

### Content-Length

컨텐츠가 없는 경우라도 HTTP 응답 본문의 길이를 지정합니다. 값에는 메시지 데이터의 길이(바이트)가 들어 있습니다.

## Content-Type

수신된 메시지의 응답 본문에서 리턴되는 콘텐츠 유형을 지정합니다. 성공하면 값은 `text/plain; charset=utf-8`입니다. 오류 또는 예외가 발생할 경우 값은 `application/json; charset=utf-8`입니다.

### REST API V1 > REST API V2 **ibm-mq-md-correlationId**

수신된 메시지의 상관 ID를 지정합니다. 수신된 메시지에 올바른 상관 ID가 들어 있는 경우 헤더가 리턴됩니다. 이는 48자 16진 인코딩 문자열로 표시되며, 24바이트를 나타냅니다.

예를 들면, 다음과 같습니다.

```
ibm-mq-md-correlationId: 414d5120514d4144455620202020202067d8bf5923582e02
```

### REST API V3 **ibm-mq-md-correlationId**

수신된 메시지의 상관 ID를 지정합니다. 수신된 메시지에 올바른 상관 ID가 들어 있는 경우 헤더가 리턴됩니다. 상관 ID는 다음 양식 중 하나를 사용할 수 있습니다.

- 앞에 "ID:" 문자열이 접두어로 붙는 24바이트를 나타내는 48자의 16진 인코딩 문자열. 예를 들면, 다음과 같습니다.

```
ibm-mq-md-correlationId: ID:414d5120514d4144455620202020202067d8bf5923582e02
```

- 애플리케이션 특정 값입니다. 값은 애플리케이션 특정 문자열입니다.

```
ibm-mq-md-correlationId: My-Custom-CorrelId
```

## ibm-mq-md-expiry

수신된 메시지의 남아 있는 만기 지속 기간을 지정합니다. 헤더는 다음 중 하나입니다.

### 무제한

메시지가 만료되지 않습니다.

### 정수 값

메시지 만기까지 남아 있는 시간(밀리초).

### REST API V1 > REST API V2 **ibm-mq-md-messageId**

IBM MQ가 이 메시지에 할당하는 메시지 ID를 지정합니다. `ibm-mq-md-correlationId` 헤더와 마찬가지로, 24바이트를 나타내는 48자의 16진 인코딩 문자열로 표시됩니다.

예를 들면, 다음과 같습니다.

```
ibm-mq-md-messageId: 414d5120514d4144455620202020202067d8ce5923582f07
```

### REST API V3 **ibm-mq-md-messageId**

IBM MQ가 이 메시지에 할당하는 메시지 ID를 지정합니다. `ibm-mq-md-correlationId` 헤더와 마찬가지로, 앞에 "ID:" 문자열이 접두어로 붙는 24바이트를 나타내는 48자의 16진 인코딩 문자열로 표시됩니다.

예를 들면, 다음과 같습니다.

```
ibm-mq-md-messageId: ID:414d5120514d4144455620202020202067d8ce5923582f07
```

## ibm-mq-md-persistence

수신된 메시지의 지속성을 지정합니다. 헤더는 다음 중 하나입니다.

### nonPersistent

시스템 장애 또는 큐 관리자 재시작 시 메시지가 지속되지 않습니다.

### 지속적

메시지는 시스템 실패 또는 큐 관리자 재시작에서 지속됩니다.

### REST API V3 **ibm-mq-md-priority**

메시지 우선순위의 설정을 리턴합니다. 예를 들면, 다음과 같습니다.

```
ibm-mq-md-priority: 3
```

### **ibm-mq-md-replyTo**

수신된 메시지의 응답 대상을 지정합니다. 헤더 형식은 응답 대상 큐 및 큐 관리자 `replyQueue@replyQmgr`의 표준 표기법을 사용합니다.

예를 들면, 다음과 같습니다.

```
ibm-mq-md-replyTo: myReplyQueue@myReplyQMGR
```

### V 9.4.0 **ibm-mq-resolved-qmgr**

요청에 사용된 큐 관리자의 이름을 지정합니다. 자원 URL에서 고유 이름이 사용된 경우 이 헤더는 해당 고유 이름과 연관된 큐 관리자의 이름을 식별합니다. 자원 URL에 사용된 고유 이름이 큐 관리자 그룹을 참조하는 경우 이 헤더는 그룹 내에서 사용된 큐 관리자를 식별합니다.

### REST API V3 **ibm-mq-usr**

메시지 사용자 정의 특성을 리턴합니다. 메시지에 여러 개의 특성을 설정할 수 있으며, 이 경우 `ibm-mq-usr` 응답 헤더에 대한 두 개 이상의 개별 인스턴스가 있습니다.

예를 들면, 다음과 같습니다.

```
ibm-mq-usr: myIProp;5;short
ibm-mq-usr: mySProp;"hi";string
ibm-mq-usr: myBProp;true;boolean
```

특성의 구문은 다음과 같습니다.

```
ibm-mq-usr: property_name; user_value; user_type
```

#### **property\_name**

지정되는 사용자 특성의 이름입니다. 올바른 JMS 특성 이름이어야 합니다.

#### **user\_value**

특성의 값입니다.

#### **user\_type**

특성의 유형은 다음과 같습니다.

- `boolean`(true/false, MQBOOL)
- `byte`(8비트 정수, MQINT8)
- `short`(16비트 정수, MQINT16)
- `integer`(32비트 정수, MQINT32)
- `long`(64비트 정수, MQINT64)
- `float`(32비트 실수, MQFLOAT32)
- `double`(64비트 실수, MQFLOAT64)
- `string`(따옴표로 묶인 문자열)

## 응답 본문 형식

성공 시 응답 본문에는 수신된 메시지의 메시지 본문이 들어 있습니다. 오류가 발생하면, 응답 본문이 JSON 형식의 오류 메시지를 포함합니다. 두 응답 모두 UTF-8 인코딩됩니다. 자세한 정보는 [REST API 오류 처리](#)를 참조하십시오.

메시지 수신 시 IBM MQ MQSTR 및 JMS `TextMessage` 형식화된 메시지만 지원된다는 점에 유의하십시오. 나중에 모든 메시지는 동기점 아래에서 수신되며 핸들링하지 않은 메시지는 큐에 남아 있습니다. 이러한 변조 메시





- 2094 페이지의 『보안 요구사항』
- 2094 페이지의 『응답 상태 코드』
- 2095 페이지의 『응답 헤더』
- 2095 페이지의 『응답 본문 형식』
- 2096 페이지의 『예』

## 자원 URL

`https://host:port/ibmmq/rest/v1/messaging/qmgr/{qmgrName}/queue/{queueName}/messagelist`

`https://host:port/ibmmq/rest/v2/messaging/qmgr/{qmgrName}/queue/{queueName}/messagelist`

`https://host:port/ibmmq/rest/v3/messaging/qmgr/{qmgrName}/queue/{queueName}/messagelist`

### qmgrName

메시징에 연결할 큐 관리자의 이름을 지정합니다. 큐 관리자는 mqweb 서버와 같은 시스템에 있어야 합니다.

**V 9.4.0** IBM MQ 9.3.3에서 로컬 큐 관리자 또는 리모트 큐 관리자에 연결할 수 있습니다. **qmgrName**에 지정하는 이름은 mqweb 서버가 구성되는 방법에 따라 다릅니다.

- mqweb 서버가 로컬 큐 관리자에만 연결하도록 구성된 경우에는 큐 관리자의 이름을 사용하십시오.
- mqweb 서버가 리모트 큐 관리자를 연결하도록 구성된 경우에는 리모트 큐 관리자 연결 정보에 지정된 대로 큐 관리자의 고유 이름을 사용하십시오.

**dspmqweb properties** 명령을 사용하고 **mqRestMessagingConnectionMode** 특성을 보고 mqweb 서버가 로컬 큐 관리자 또는 리모트 큐 관리자에 연결하도록 구성되었는지 여부를 판별할 수 있습니다.

큐 관리자 이름은 대소문자를 구분합니다.

큐 관리자 이름에 슬래시, 마침표 또는 퍼센트 부호가 포함되는 경우 해당 문자는 인코딩된 URL이어야 합니다.

- 슬래시(/)는 %2F(으)로 인코딩되어야 합니다.
- 백분율 기호(%)는 %25(으)로 인코딩되어야 합니다.

### queueName

메시지를 찾을 큐의 이름을 지정합니다.

큐는 로컬 또는 로컬 큐를 나타내는 알리어스로 정의되어야 합니다.

큐 이름은 대소문자를 구분합니다.

큐 이름이 슬래시 또는 퍼센트 부호를 포함하는 경우 해당 문자는 인코딩된 URL이어야 합니다.

- 슬래시(/)는 %2F(으)로 인코딩되어야 합니다.
- 퍼센트 부호(%)는 %25(으)로 인코딩되어야 합니다.

HTTP 연결을 사용으로 설정하는 경우 HTTPS 대신 HTTP를 사용할 수 있습니다. HTTP 사용에 대한 자세한 정보는 [HTTP 및 HTTPS 포트 구성](#)을 참조하십시오.

## 선택적 조회 매개변수

### **REST API V2** `correlationId=hexValue`

HTTP 메소드가 해당하는 상관 ID를 가진 다음 메시지를 리턴하도록 지정합니다.

#### hexValue

조회 매개변수는 48자의 16진 인코딩 문자열로서 지정되어야 하고, 이는 24바이트를 나타냅니다.

예를 들면, 다음과 같습니다.



```
../messagelist?correlationId=414d5120514d41444556202020202067d8bf5923582e02
```

### REST API V3 **correlationId=ID:hexValue** 또는 **correlationId=application\_specific\_value**

HTTP 메소드가 해당 상관 ID가 있는 메시지 목록을 리턴하도록 지정합니다.

#### **hexValue**

조회 매개변수는 24바이트를 나타내는 48자의 16진 인코딩 문자열로 지정해야 하며, 앞에 "ID:" 문자열이 접두어로 붙습니다.

예를 들면, 다음과 같습니다.

```
../message?correlationId=ID:414d5120514d41444556202020202067d8bf5923582e02
```

#### **application\_specific\_value**

조회 매개변수는 애플리케이션 특정 문자열로 지정될 수 있습니다.

예를 들면, 다음과 같습니다.

```
../message?correlationId=My-Custom-CorrelId
```

### REST API V1 > REST API V2 **messageId=hexValue**

HTTP 메소드가 해당하는 메시지 ID를 가진 다음 메시지를 리턴하도록 지정합니다.

#### **hexValue**

조회 매개변수는 48자의 16진 인코딩 문자열로서 지정되어야 하고, 이는 24바이트를 나타냅니다.

예를 들면, 다음과 같습니다.

```
../message?messageId=414d5120514d41444556202020202067d8ce5923582f07
```

### REST API V3 **messageId=ID:hexValue**

HTTP 메소드가 해당하는 메시지 ID를 가진 다음 메시지를 리턴하도록 지정합니다.

#### **hexValue**

조회 매개변수는 24바이트를 나타내는 48자의 16진 인코딩 문자열로 지정해야 하며, 앞에 "ID:" 문자열이 접두어로 붙습니다.

예를 들면, 다음과 같습니다.

```
../message?messageId=ID:414d5120514d41444556202020202067d8ce5923582f07
```

### **limit=integerValue**

HTTP 메소드 응답 본문이 *integerValue* JSON 요소로 제한되도록 지정합니다.

#### **integerValue**

조회 매개변수는 JSON 응답 본문에 포함된 최대 요소 수를 나타내는 정수 값으로 지정되어야 합니다.

기본값은 10이고 최대값은 2147483647입니다.

예를 들면, 다음과 같습니다.

```
../messagelist?limit=250
```

## 요청 헤더

다음 헤더를 요청과 함께 전송해야 합니다.

### 권한 부여

기본 인증을 사용하는 경우 이 헤더를 송신해야 합니다. 자세한 정보는 [REST API로 HTTP 기본 인증 사용](#)을 참조하십시오.

## ibm-mq-rest-csrf-token

이 헤더가 설정되어야 하지만 값은 공백을 포함한 어떤 값도 될 수 있습니다.

다음 헤더를 요청과 함께 선택적으로 전송할 수 있습니다.

### Accept-Charset

이 헤더를 사용하여 응답에 수용할 수 있는 문자 세트를 표시할 수 있습니다. 지정된 경우 이 헤더는 UTF-8(으)로 설정되어야 합니다.

### Accept-Language

이 헤더는 응답 메시지 본문에서 리턴되는 예외 또는 오류 메시지의 필수 언어를 지정합니다.

## 요청 본문 형식




없음

## 보안 요구사항

호출자를 mqweb 서버에 대해 인증해야 합니다. MQWebAdmin 및 MQWebAdminRO 역할은 messaging REST API에 적용할 수 없습니다. REST API의 보안에 대한 자세한 정보는 [IBM MQ Console 및 REST API 보안을 참조](#) 하십시오.

일단 mqweb 서버에 인증되면 사용자는 messaging REST API 및 administrative REST API 모두를 사용할 수 있습니다.

지정된 큐에서 메시지를 찾아볼 수 있는 기능을 호출자의 보안 프린시펄에 부여해야 합니다.

- 자원 URL의 `{queueName}` 부분에 지정된 큐는 BROWSE 사용 가능해야 합니다.
-   자원 URL의 `{queueName}` 부분에 의해 지정되는 큐의 경우, 호출자의 보안 프린시펄에 +GET, +INQ 및 +BROWSE 권한이 부여되어야 합니다.
-  자원 URL UPDATE의 `{queueName}` 부분에 의해 지정되는 큐의 경우, 호출자의 보안 프린시펄에 액세스가 부여되어야 합니다.

이 보안 프린시펄은 mqweb 서버를 시작한 사용자 또는 mqweb 서버에 로그인한 사용자일 수 있습니다. 연결하는 큐 관리자가 리모트 큐 관리자인 경우, 보안 프린시펄은 대신 리모트 큐 관리자 연결 정보의 신임 정보에 의해 제공되는 사용자이거나 채널 보안 규칙에 의해 판별되는 다른 사용자일 수 있습니다. 자세한 정보는 [messaging REST API에서 사용되는 보안 프린시펄 판별](#)을 참조하십시오.

 AIX, Linux, and Windows에서 `setmqaut` 명령을 사용하여 IBM MQ 자원을 사용하도록 보안 프린시펄에 권한을 부여할 수 있습니다. 자세한 정보는 [setmqaut\(권한 부여 또는 취소\)](#)를 참조하십시오.

z/OS의 경우 [z/OS에서 보안 설정](#)을 참조하십시오.

## 응답 상태 코드

### 200

메시지 목록이 수신되었습니다.

### 400

올바르지 않은 데이터가 제공되었습니다.

예를 들어, 올바른지 않은 조회 매개변수 값이 지정되었습니다.

### 401

인증되지 않았습니다.

호출자는 mqweb 서버에 대해 인증되어야 하며 MQWebAdmin, MQWebAdminRO 또는 MQWebUser 역할 중 하나 이상의 구성원이어야 합니다. `ibm-mq-rest-csrf-token` 헤더도 지정해야 합니다. 자세한 정보는 [2094 페이지의 『보안 요구사항』](#)의 내용을 참조하십시오.

### 403

권한이 부여되지 않았습니다.

호출자가 mqweb 서버에 대해 인증되었거나 올바른 프린시펄과 연관되어 있습니다. 그러나 프린시펄에 필수 IBM MQ 자원의 전체 또는 서브세트에 대한 액세스 권한이 없거나 MQWebUser 역할에 없습니다. 필수인 액세스에 대한 자세한 정보는 [2094 페이지의 『보안 요구사항』](#)의 내용을 참조하십시오.

**404**

큐가 없습니다.

**500**

IBM MQ의 서버 문제 또는 오류 코드입니다.

**501**

HTTP 응답을 구성할 수 없습니다.

예를 들어 수신된 메시지의 유형이 올바르지 않거나 유형이 올바르지만 본문을 처리할 수 없습니다.

**502**

메시징 제공자가 필수 함수를 지원하지 않으므로 현재 보안 프린시펄에서 메시지를 수신할 수 없습니다. 예를 들어 mqweb 서버 클래스 경로가 올바르지 않은 경우입니다.

**503**

큐 관리자가 실행 중이지 않습니다.

## 응답 헤더

### Content-Language

오류 또는 예외의 경우 응답 메시지의 언어 ID를 지정합니다. 오류 또는 예외 조건에 필요한 언어를 나타내기 위해 Accept-Language 요청 헤더와 함께 사용됩니다. 요청된 언어가 지원되지 않는 경우 mqweb 서버 기본값이 사용됩니다.

### Content-Length

컨텐츠가 없는 경우라도 HTTP 응답 본문의 길이를 지정합니다. 값에는 메시지 데이터의 길이(바이트)가 포함됩니다.

### Content-Type

응답의 본문을 지정합니다. 값은 application/json;charset=utf-8입니다.

**V 9.4.0**

### ibm-mq-resolved-qmgr

요청에 사용된 큐 관리자의 이름을 지정합니다. 자원 URL에서 고유 이름이 사용된 경우 이 헤더는 해당 고유 이름과 연관된 큐 관리자의 이름을 식별합니다. 자원 URL에 사용된 고유 이름이 큐 관리자 그룹을 참조하는 경우 이 헤더는 그룹 내에서 사용된 큐 관리자를 식별합니다.

## 응답 본문 형식

성공 시 응답 본문은 UTF-8 인코딩 응답입니다. 응답은 messages라고 하는 하나의 JSON 배열을 포함하는 외부 JSON 오브젝트를 포함합니다. 배열의 각 요소는 큐의 메시지에 대한 정보가 포함된 JSON 오브젝트입니다. 각 요소는 다음 속성을 포함합니다.

### REST API V1 > REST API V2 correlationId

메시지의 상관 ID를 지정합니다. 메시지에 올바른 상관 ID가 포함된 경우 값이 리턴됩니다.

### REST API V3 correlationId

메시지의 상관 ID를 지정합니다. 메시지에 올바른 상관 ID가 포함된 경우 값이 리턴됩니다. 상관 ID는 앞에 "ID:" 문자열이 나오거나 애플리케이션 특정 값일 수 있습니다.

### REST API V1 > REST API V2 messageId

IBM MQ가 이 메시지에 할당한 메시지 ID를 지정합니다. 이는 48자 16진 인코딩 문자열로 표시되며, 24바이트를 나타냅니다.

### REST API V3 messageId

IBM MQ가 이 메시지에 할당한 메시지 ID를 지정합니다. 이는 48자 16진 인코딩 문자열로 표시되며, 24바이트를 나타냅니다. 메시지 ID는 앞에 "ID:" 문자열이 나옵니다.

### 형식

MQMD 형식 필드를 지정합니다. 일반적인 상황에서 텍스트 메시지에 IBM MQ MQSTR 값이 포함됩니다.



HTTP 요청 본문의 텍스트 기반을 지정된 큐 관리자 및 토픽에 넣습니다. 큐 관리자는 mqweb 서버와 같은 시스템에 있어야 하고 텍스트 기반 메시지만 지원됩니다. 메시지는 현재 사용자 컨텍스트를 사용하여 MQSTR 또는 JMS TextMessage 형식화된 메시지로 발행되며 기본 메시지 우선순위는 4 (REST API V2의 경우) 및 asDestination (REST API V3의 경우) 입니다.

REST API V3는 사용자 정의 메시지 특성을 지정하고 메시지 우선순위를 포함하는 기능을 추가합니다. ibm-mq-md-priority 및 ibm-mq-usr 요청 헤더는 REST API V3에서만 사용할 수 있습니다. REST API V3에서 ibm-mq-md-correlationId 요청 헤더의 형식은 다릅니다. 헤더는 애플리케이션 특정 ID이거나 인코딩된 문자열일 경우에는 ID: 접두부가 필요합니다. POST 요청에 사용자 정의 메시지 또는 애플리케이션 특정 상관 ID가 포함된 경우 메시지는 JMS TextMessage(으)로 형식화됩니다.

- [2097 페이지의 『자원 URL』](#)
- [2098 페이지의 『요청 헤더』](#)
- [2099 페이지의 『요청 본문 형식』](#)
- [2100 페이지의 『보안 요구사항』](#)
- [2100 페이지의 『응답 상태 코드』](#)
- [2101 페이지의 『응답 헤더』](#)
- [2101 페이지의 『응답 본문 형식』](#)
- [2101 페이지의 『예』](#)


## 자원 URL

`https://host:port/ibmmq/rest/v2/messaging/qmgr/{qmgrName}/topic/{topicString}/message`

`https://host:port/ibmmq/rest/v3/messaging/qmgr/{qmgrName}/topic/{topicString}/message`

### qmgrName

메시징을 위해 연결할 큐 관리자의 이름을 지정합니다.

 IBM MQ 9.3.3에서 로컬 큐 관리자 또는 리모트 큐 관리자에 연결할 수 있습니다. **qmgrName**에 지정하는 이름은 mqweb 서버가 구성되는 방법에 따라 다릅니다.

- mqweb 서버가 로컬 큐 관리자에만 연결하도록 구성된 경우에는 큐 관리자의 이름을 사용하십시오.
- mqweb 서버가 리모트 큐 관리자를 연결하도록 구성된 경우에는 리모트 큐 관리자 연결 정보에 지정된 대로 큐 관리자의 고유 이름을 사용하십시오.

**dspmqweb properties** 명령을 사용하고 **mqRestMessagingConnectionMode** 특성을 보고 mqweb 서버가 로컬 큐 관리자 또는 리모트 큐 관리자에 연결하도록 구성되었는지 여부를 판별할 수 있습니다.

이름은 대소문자를 구분합니다.

이름에 슬래시, 마침표 또는 퍼센트 부호가 포함된 경우 다음 문자는 URL 인코딩되어야 합니다.

- 슬래시는 %2F(으)로 인코딩되어야 합니다.
- 마침표는 %2E(으)로 인코딩되어야 합니다.
- 백분율 기호는 %25(으)로 인코딩되어야 합니다.

### topicString

메시지를 넣을 토픽 문자열을 지정합니다.

토픽 문자열은 대소문자를 구분합니다. 토픽 문자열에는 슬래시 구분 기호로 구분된 여러 토픽 레벨이 포함될 수 있습니다.

토픽 문자열에 백분율 기호, 마침표 또는 물음표가 포함된 경우 이러한 문자는 URL로 인코딩되어야 합니다.

- 백분율 기호는 %25(으)로 인코딩되어야 합니다.
- 마침표는 %2E(으)로 인코딩되어야 합니다.
- 물음표는 %3F(으)로 인코딩되어야 합니다.

토픽 문자열이 슬래시로 시작하거나 끝나는 경우 %2F(으)로 인코딩되어야 합니다.

예를 들어, 토픽 문자열을 발행하려면 다음을 수행합니다.

- MY.QMGR 큐 관리자의 sport/football에서는 다음 URL을 사용합니다.

```
https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/MY%2EQMGR/topic/sport/football/message
```

- MY.QMGR 큐 관리자의 /sport/football에서는 다음 URL을 사용합니다.

```
https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/MY%2EQMGR/topic/%2Fsport/football/message
```

HTTP 연결을 사용으로 설정하는 경우 HTTPS 대신 HTTP를 사용할 수 있습니다. HTTP 사용에 대한 자세한 정보는 [HTTP 및 HTTPS 포트 구성을 참조하십시오](#).

## 요청 헤더

다음 헤더를 요청과 함께 전송해야 합니다.

### 권한 부여

기본 인증을 사용하는 경우 이 헤더를 송신해야 합니다. 자세한 정보는 [REST API로 HTTP 기본 인증 사용](#)을 참조하십시오.

### Content-Type

이 헤더는 다음 값 중 하나로 송신해야 합니다.

- text/plain;charset=utf-8
- text/html;charset=utf-8
- text/xml;charset=utf-8
- application/json;charset=utf-8
- application/xml;charset=utf-8

### ibm-mq-rest-csrf-token

이 헤더가 설정되어야 하지만 값은 공백을 포함한 어떤 값도 될 수 있습니다.

다음 헤더를 요청과 함께 선택적으로 전송할 수 있습니다.

### Accept-Language

이 헤더는 응답 메시지 본문에서 리턴되는 예외 또는 오류 메시지의 필수 언어를 지정합니다.

### ibm-mq-md-expiry

이 헤더는 작성된 메시지의 만기 지속 기간을 설정합니다. 메시지의 만기는 메시지가 큐 관리자에 도착하는 시간부터 시작됩니다. 결과 네트워크 지연이 무시됩니다. 헤더는 다음 값 중 하나로 지정해야 합니다.

#### 무제한

메시지가 만료되지 않습니다.

이 값이 기본값입니다.

#### 정수 값

메시지 만기까지 시간(밀리초).

0 - 99999999900 범위로 제한됩니다.

### ibm-mq-md-persistence

이 헤더는 작성된 메시지의 지속성을 설정합니다. 헤더는 다음 값 중 하나로 지정해야 합니다.

#### nonPersistent

시스템 장애 또는 큐 관리자 재시작 시 메시지가 지속되지 않습니다.

이 값이 기본값입니다.

#### 지속적

메시지는 시스템 실패 또는 큐 관리자 재시작에서 지속됩니다.

### REST API V3 **ibm-mq-md-priority**

이 헤더는 작성된 메시지의 우선순위를 설정합니다. 헤더는 다음 값 중 하나로 지정해야 합니다.

#### **asDestination**

메시지는 기본 IBM MQ 큐 오브젝트의 DEFPRTY 속성에 지정된 우선순위를 사용합니다.

#### **정수 값**

0 - 9 범위의 정수로 실제 우선순위를 지정하십시오.

예를 들면, 다음과 같습니다.

```
ibm-mq-md-priority: asDestination
```

### **ibm-mq-md-replyTo**

이 헤더는 작성된 메시지의 응답 대상을 설정합니다. 헤더 형식은 응답 대상 큐 및 선택적 큐 관리자 제공의 표준 표기법을 사용합니다. `replyQueue[@replyQmgr]`

예를 들면, 다음과 같습니다.

```
ibm-mq-md-replyTo: myReplyQueue@myReplyQmgr
```

### REST API V3 **ibm-mq-usr**

요청 메시지 사용자 정의 특성을 설정하십시오. 메시지에서 여러 특성을 설정할 수 있습니다. 단일 `ibm-mq-usr` 요청 헤더에 쉼표로 구분된 여러 개의 특성을 지정하거나 `ibm-mq-usr` 요청 헤더에 대한 두 개 이상의 개별 인스턴스를 사용할 수 있습니다.

예를 들면, 다음과 같습니다.

```
ibm-mq-usr: myIPprop;5;short
ibm-mq-usr: mySPprop;"hi";string
ibm-mq-usr: myBProp>true;boolean
ibm-mq-usr: myA;5;byte,myB;-10;integer
```

특성의 구문은 다음과 같습니다.

```
ibm-mq-usr: property_name; user_value; user_type
```

#### **property\_name**

지정되는 사용자 특성의 이름입니다. 올바른 JMS 특성 이름이어야 합니다.

#### **user\_value**

특성의 값입니다.

#### **user\_type**

특성의 유형은 다음과 같습니다.

- `boolean`(true/false, MQBOOL)
- `byte`(8비트 정수, MQINT8)
- `short`(16비트 정수, MQINT16)
- `integer`(32비트 정수, MQINT32)
- `long`(64비트 정수, MQINT64)
- `float`(32비트 실수, MQFLOAT32)
- `double`(64비트 실수, MQFLOAT64)
- `string`(따옴표로 묶인 문자열)

### **요청 본문 형식**

요청 본문은 텍스트여야 하며 UTF-8 인코딩을 사용합니다. 특정 텍스트가 필요하지 않습니다. 요청 본문 텍스트를 포함하는 MQSTR 형식화된 메시지가 작성되어 지정된 토픽에 발행됩니다.



**REST API V3** REST API V3 사용자 정의 특성 또는 애플리케이션 특정 상관 ID 기능이 사용되는 경우, 요청 본문 텍스트를 포함하는 JMS TextMessage 형식화된 메시지가 작성되어 지정된 큐에 배치됩니다.

자세한 정보는 [examples](#)를 참조하십시오.

## 보안 요구사항

호출자를 mqweb 서버에 대해 인증해야 합니다. MQWebAdmin 및 MQWebAdminRO 역할은 messaging REST API에 적용할 수 없습니다. REST API의 보안에 대한 자세한 정보는 [IBM MQ Console 및 REST API 보안을 참조하십시오](#).

일단 mqweb 서버에 인증되면 사용자는 messaging REST API 및 administrative REST API 모두를 사용할 수 있습니다.

지정된 토픽에 메시지를 넣기 위한 기능을 호출자의 보안 프린시펄에 부여해야 합니다.

- 자원 URL의 `{topicString}` 부분에 지정된 토픽은 PUBLISH 사용 가능해야 합니다.
- **MQ Appliance** **ALW** 자원 URL의 `{topicString}` 부분에 의해 지정되는 토픽의 경우, 호출자의 보안 프린시펄에 +PUB 권한을 부여해야 합니다.
- **z/OS** 자원 URL의 `{topicString}` 부분에 의해 지정되는 토픽의 경우, 호출자의 보안 프린시펄에 UPDATE 액세스 권한을 부여해야 합니다.

이 보안 프린시펄은 mqweb 서버를 시작한 사용자 또는 mqweb 서버에 로그인한 사용자일 수 있습니다. 연결하는 큐 관리자가 리모트 큐 관리자인 경우, 보안 프린시펄은 대신 리모트 큐 관리자 연결 정보의 신임 정보에 의해 제공되는 사용자이거나 채널 보안 규칙에 의해 판별되는 다른 사용자일 수 있습니다. 자세한 정보는 [messaging REST API에서 사용되는 보안 프린시펄 판별](#)을 참조하십시오.

**ALW** AIX, Linux, and Windows에서 **setmqaut** 명령을 사용하여 IBM MQ 자원을 사용하도록 보안 프린시펄에 권한을 부여할 수 있습니다. 자세한 정보는 [setmqaut\(권한 부여 또는 취소\)](#)를 참조하십시오.

z/OS의 경우 [z/OS에서 보안 설정](#)을 참조하십시오.

messaging REST API에서 AMS(Advanced Message Security)를 사용하는 경우에는 모든 메시지가 메시지를 게시하는 사용자의 컨텍스트가 아닌 mqweb 서버의 컨텍스트를 사용하여 암호화됨을 유념하십시오.

## 응답 상태 코드

### 201

메시지가 작성되고 공개되었습니다.

### 400

올바르지 않은 데이터가 제공되었습니다.

예를 들어, 올바른 요청 헤더 값이 지정되었습니다.

### 401

인증되지 않았습니다.

호출자는 mqweb 서버에 대해 인증되어야 하며 MQWebAdmin, MQWebAdminRO 또는 MQWebUser 역할 중 하나 이상의 구성원이어야 합니다. `ibm-mq-rest-csrf-token` 헤더도 지정해야 합니다. 자세한 정보는 [2100 페이지의 『보안 요구사항』](#)의 내용을 참조하십시오.

### 403

권한이 부여되지 않았습니다.

호출자가 mqweb 서버에 대해 인증되었거나 올바른 프린시펄과 연관되어 있습니다. 그러나 프린시펄에 필수 IBM MQ 자원의 전체 또는 서브세트에 대한 액세스 권한이 없거나 MQWebUser 역할에 없습니다. 필수인 액세스에 대한 자세한 정보는 [2100 페이지의 『보안 요구사항』](#)의 내용을 참조하십시오.

### 404

큐 관리자가 존재하지 않습니다.

### 405

토픽에서 PUBLISH가 금지됩니다.



415

메시지 헤더 또는 본문은 지원되지 않는 매체 유형입니다.

예를 들어, Content-Type 헤더가 지원되지 않는 매체 유형으로 설정되어 있습니다.

500

IBM MQ의 서버 문제 또는 오류 코드입니다.

502

메시징 제공자가 필수 함수를 지원하지 않으므로 현재 보안 프린시플에서 메시지를 공개할 수 없습니다. 예를 들어 mqweb 서버 클래스 경로가 올바르지 않은 경우입니다.

503

큐 관리자가 실행 중이지 않습니다.

## 응답 헤더

다음 헤더가 응답과 함께 리턴됩니다.

### Content-Language

오류 또는 예외의 경우 응답 메시지의 언어 ID를 지정합니다. 오류 또는 예외 조건에 필요한 언어를 나타내기 위해 Accept-Language 요청 헤더와 함께 사용됩니다. 요청된 언어가 지원되지 않는 경우 mqweb 서버 기본값이 사용됩니다.

### Content-Length

컨텐츠가 없는 경우라도 HTTP 응답 본문의 길이를 지정합니다. 성공 시 값은 0(영)입니다.

### Content-Type

응답의 본문을 지정합니다. 성공하면 값은 text/plain;charset=utf-8입니다. 오류 또는 예외가 발생할 경우 값은 application/json;charset=utf-8입니다.

V 9.4.0

### ibm-mq-resolved-qmgr

요청에 사용된 큐 관리자의 이름을 지정합니다. 자원 URL에서 고유 이름이 사용된 경우 이 헤더는 해당 고유 이름과 연관된 큐 관리자의 이름을 식별합니다. 자원 URL에 사용된 고유 이름이 큐 관리자 그룹을 참조하는 경우 이 헤더는 그룹 내에서 사용된 큐 관리자를 식별합니다.

## 응답 본문 형식

메시지가 공개되면 응답 본문은 비어 있습니다. 오류가 발생하면 응답 본문이 오류 메시지를 포함합니다. 자세한 정보는 [REST API 오류 처리](#)를 참조하십시오.

## 예

다음 예제는 비밀번호가 mquser인 사용자 mquser에 로그인합니다. cURL에서 요청의 로그는 다음 Windows 예제와 같을 수 있습니다. LTPA 토큰은 -c 플래그를 사용하여 cookiejar.txt 파일에 저장됩니다.

```
curl -k "https://localhost:9443/ibmmq/rest/v1/login" -X POST
-H "Content-Type: application/json" --data "{\"username\":\"mquser\", \"password\":\"mquser\"}"
-c c:\cookiejar.txt
```

사용자가 로그인되면 LTPA 토큰 및 ibm-mq-rest-csrf-token HTTP 헤더가 추가 요청을 인증하는 데 사용됩니다. ibm-mq-rest-csrf-token token\_value은(는) 공백을 포함하여 임의의 값이 될 수 있습니다.

- 다음 Windows cURL 예제는 기본 옵션을 사용하여 큐 관리자 QM1의 토픽 문자열 myTopic에 메시지를 발행합니다. 메시지에는 "Hello World!" 텍스트가 들어 있습니다.

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/topic/myTopic/message"
-X POST -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token_value"
-H "Content-Type: text/plain;charset=utf-8" --data "Hello World!"
```

- 다음 Windows cURL 예제는 큐 관리자 QM1의 토픽 문자열 myTopic/thisTopic에 지속 메시지를 발행하며, 만기는 2분입니다. 메시지에는 "Hello World!" 텍스트가 들어 있습니다.

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/topic/myTopic%2FthisTopic/message"
```

```
-X POST -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token_value"  
-H "Content-Type: text/plain;charset=utf-8" -H "ibm-mq-md-persistence: persistent"  
-H "ibm-mq-md-expiry: 120000" --data "Hello World!"
```

## 주의사항

이 정보는 미국에서 제공되는 제품 및 서비스용으로 작성된 것입니다.

IBM은 다른 국가에서 이 책에 기술된 제품, 서비스 또는 기능을 제공하지 않을 수도 있습니다. 현재 사용할 수 있는 제품 및 서비스에 대한 정보는 한국 IBM 담당자에게 문의하십시오. IBM 제품, 프로그램 또는 서비스를 언급했다고 해서 해당 IBM 제품, 프로그램 또는 서비스만을 사용할 수 있다는 것을 의미하지는 않습니다. IBM의 지적 재산권을 침해하지 않는 한, 기능상으로 동등한 제품, 프로그램 또는 서비스를 대신 사용할 수도 있습니다. 그러나 비IBM 제품, 프로그램 또는 서비스의 운영에 대한 평가 및 검증은 사용자의 책임입니다.

IBM은 이 책에서 다루고 있는 특정 내용에 대해 특허를 보유하고 있거나 현재 특허 출원 중일 수 있습니다. 이 책을 제공한다고 해서 특허에 대한 라이선스까지 부여하는 것은 아닙니다. 라이선스에 대한 의문사항은 다음으로 문의하십시오.

07326

서울특별시 영등포구  
국제금융로 10, 3IFC  
한국 아이.비.엠 주식회사  
U.S.A.

2바이트(DBCS) 정보에 관한 라이선스 문의는 한국 IBM에 문의하거나 다음 주소로 서면 문의하시기 바랍니다.

Intellectual Property Licensing  
2-31 Roppongi 3-chome, Minato-Ku  
IBM Japan, Ltd.  
19-21, Nihonbashi-Hakozakicho, Chuo-ku  
Tokyo 103-8510, Japan

**다음 단락은 현지법과 상충하는 영국이나 기타 국가에서는 적용되지 않습니다.** IBM은 타인의 권리 비침해, 상품성 및 특정 목적에의 적합성에 대한 묵시적 보증을 포함하여(단, 이에 한하지 않음) 명시적 또는 묵시적인 일체의 보증 없이 이 책을 "현상태대로" 제공합니다. 일부 국가에서는 특정 거래에서 명시적 또는 묵시적 보증의 면책사항을 허용하지 않으므로, 이 사항이 적용되지 않을 수도 있습니다.

이 정보에는 기술적으로 부정확한 내용이나 인쇄상의 오류가 있을 수 있습니다. 이 정보는 주기적으로 변경되며, 변경된 사항은 최신판에 통합됩니다. IBM은 이 책에서 설명한 제품 및/또는 프로그램을 사전 통지 없이 언제든지 개선 및/또는 변경할 수 있습니다.

이 정보에서 언급되는 비IBM의 웹 사이트는 단지 편의상 제공된 것으로, 어떤 방식으로든 이들 웹 사이트를 옹호하고자 하는 것은 아닙니다. 해당 웹 사이트의 자료는 본 IBM 제품 자료의 일부가 아니므로 해당 웹 사이트 사용으로 인한 위험은 사용자 본인이 감수해야 합니다.

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

(i) 독립적으로 작성된 프로그램과 기타 프로그램(본 프로그램 포함) 간의 정보 교환 및 (ii) 교환된 정보의 상호 이용을 목적으로 본 프로그램에 관한 정보를 얻고자 하는 라이선스 사용자는 다음 주소로 문의하십시오.

서울특별시 영등포구  
서울특별시 강남구 도곡동 467-12,  
군인공제회관빌딩  
한국 아이.비.엠 주식회사  
U.S.A.

이러한 정보는 해당 조건(예를 들면, 사용료 지불 등)하에서 사용될 수 있습니다.

이 정보에 기술된 라이선스가 부여된 프로그램 및 프로그램에 대해 사용 가능한 모든 라이선스가 부여된 자료는 IBM이 IBM 기본 계약, IBM 프로그램 라이선스 계약(IPLA) 또는 이와 동등한 계약에 따라 제공한 것입니다.

본 문서에 포함된 모든 성능 데이터는 제한된 환경에서 산출된 것입니다. 따라서 다른 운영 환경에서 얻어진 결과는 상당히 다를 수 있습니다. 일부 성능은 개발 단계의 시스템에서 측정되었을 수 있으므로 이러한 측정치가 일반적으로 사용되고 있는 시스템에서도 동일하게 나타날 것이라고는 보증할 수 없습니다. 또한 일부 성능은 추정

통해 추측되었을 수도 있으므로 실제 결과는 다를 수 있습니다. 이 책의 사용자는 해당 데이터를 본인의 특정 환경에서 검증해야 합니다.

비IBM 제품에 관한 정보는 해당 제품의 공급업체, 공개 자료 또는 기타 범용 소스에서 얻은 것입니다. IBM에서는 이러한 제품들을 테스트하지 않았으므로, 비IBM 제품과 관련된 성능의 정확성, 호환성 또는 기타 청구에 대해서는 확신할 수 없습니다. 비IBM 제품의 성능에 대한 의문사항은 해당 제품의 공급업체에 문의하십시오.

IBM이 제시하는 방향 또는 의도에 관한 모든 언급은 특별한 통지 없이 변경될 수 있습니다.

이 정보에는 일상의 비즈니스 운영에서 사용되는 자료 및 보고서에 대한 예제가 들어 있습니다. 이들 예제에는 개념을 가능한 완벽하게 설명하기 위하여 개인, 회사, 상표 및 제품의 이름이 사용될 수 있습니다. 이들 이름은 모두 가공의 것이며 실제 기업의 이름 및 주소와 유사하더라도 이는 전적으로 우연입니다.

저작권 라이선스:

이 정보에는 여러 운영 플랫폼에서의 프로그래밍 기법을 보여주는 원어로 된 샘플 응용프로그램이 들어 있습니다. 귀하는 이러한 샘플 프로그램의 작성 기준이 된 운영 플랫폼의 애플리케이션 프로그래밍 인터페이스(API)에 부합하는 애플리케이션을 개발, 사용, 판매 또는 배포할 목적으로 IBM에 추가 비용을 지불하지 않고 이들 샘플 프로그램을 어떠한 형태로든 복사, 수정 및 배포할 수 있습니다. 이러한 샘플 프로그램은 모든 조건하에서 완전히 테스트된 것은 아닙니다. 따라서 IBM은 이들 샘플 프로그램의 신뢰성, 서비스 가능성 또는 기능을 보증하거나 진술하지 않습니다.

이 정보를 소프트웨어로 확인하는 경우에는 사진과 컬러 삽화가 제대로 나타나지 않을 수도 있습니다.

## 프로그래밍 인터페이스 정보

---

프로그래밍 인터페이스 정보는 본 프로그램과 함께 사용하기 위한 응용프로그램 소프트웨어 작성을 돕기 위해 제공됩니다.

이 책에는 고객이 IBM MQ의 서비스를 얻기 위해 프로그램을 작성할 수 있도록 하는 의도된 프로그래밍 인터페이스에 대한 정보가 들어 있습니다.

그러나 본 정보에는 진단, 수정 및 성능 조정 정보도 포함되어 있습니다. 진단, 수정 및 성능 조정 정보는 응용프로그램 소프트웨어의 디버거를 돕기 위해 제공된 것입니다.

**중요사항:** 이 진단, 수정 및 튜닝 정보는 변경될 수 있으므로 프로그래밍 인터페이스로 사용하지 마십시오.

## 상표

---

IBM, IBM 로고, [ibm.com](http://ibm.com)®는 전세계 여러 국가에 등록된 IBM Corporation의 상표입니다. 현재 IBM 상표 목록은 웹 "저작권 및 상표 정보"([www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml))에 있습니다. 기타 제품 및 서비스 이름은 IBM 또는 타사의 상표입니다.

Microsoft 및 Windows는 미국 또는 기타 국가에서 사용되는 Microsoft Corporation의 상표입니다.

UNIX는 미국 또는 기타 국가에서 사용되는 The Open Group의 등록상표입니다.

Linux는 미국 또는 기타 국가에서 사용되는 Linus Torvalds의 등록상표입니다.

이 제품에는 Eclipse 프로젝트 (<https://www.eclipse.org/>)에서 개발한 소프트웨어가 포함되어 있습니다.

Java 및 모든 Java 기반 상표와 로고는 Oracle 및/또는 그 계열사의 상표 또는 등록상표입니다.





부품 번호:

(1P) P/N: