7.5

IBM WebSphere MQ 開發應用程式參考手冊

附註

使用本資訊及其支援的產品之前，請先閱讀第 1331 頁的『注意事項』中的資訊。

除非新版中另有指示，否則此版本適用於 IBM® WebSphere MQ 7.5 版及所有後續版次與修訂。

當您將資訊傳送至 IBM 時，您授與 IBM 非專屬權利，以任何其認為適當的方式使用或散佈資訊，而無需對您負責。

© Copyright International Business Machines Corporation 2007, 2024.

目錄

開發應用程式參照 7
MQI 應用程式參照 7
程式碼範例 7
常數 49
MQI 中使用的資料類型 214
函數呼叫 545
物件的屬性 700
回覆碼 764
MQI 選項的驗證規則 764
發佈/訂閱指令訊息 767
機器編碼 785
報告選項及訊息旗標 788
資料轉換 791
指定為 MQRFH2 元素的內容 811
字碼頁轉換 817
64 位元平台上的編碼標準 846
SOAP 參照 850
amqSOAPNETListener: IBM Websphere MQ SOAP Listener for .NET Framework 1 或 2 850
amqswsdl: 產生 WSDL for .NET 服務 852
amqwclientconfig: 建立用戶端部署描述子 853
amqwdeployWMQService: 部署 Web 服務公用程式 853
amqwRegisterdotNet: 登錄 IBM WebSphere MQ 傳輸 861
Apache 軟體授權 861
MQMD SOAP 設定 865
MQRFH2 SOAP 設定 870
runivt: 安裝驗證測試 871
安全 IBM WebSphere MQ Web 服務 874
SimpleJava 接聽器: IBM Websphere MQ SOAP Listener for Axis 1.4 877
SOAP 接聽器 879
SOAP 寄件者 884
交易 885
URI 參數 885
W3C SOAP over JMS URI 891
IBM WebSphere MQ 傳輸 896
IBM WebSphere MQ 傳輸 899
使用者結束程式、API 結束程式及可安裝的服務參照 901
MQIEP 介面進入點結構 902
資料轉換結束程式參照 905
MQ_PUBLISH_EXIT-發佈結束程式 908
通道結束程式呼叫和資料結構 915
API 結束程式參照 972
可安裝的服務介面參照資訊 1028
適用於 HTTP 參照的 IBM WebSphere MQ 橋接器 1087
HTTP DELETE 1087
HTTP GET 1090
HTTP POST 1093
HTTP 標頭 1096
HTTP 回覆碼 1111
支援的訊息類型 1118
URI 格式 1120
IBM WebSphere MQ .NET 類別和介面 1120
MQAsyncStatus 1121

MQAuthenticationInformation 記錄 1122
MQDestination 1123
MQEnvironment 1125
MQException 1127
MQGetMessage 選項 1128
MQManagedObject 1130
MQMessage 1132
MQProcess 1143
MQPropertyDescriptor 1145
MQPutMessage 選項 1146
MQQUEUE 1149
MQQueueManager 1155
MQSubscription 1167
MQTopic 1168
IMQObjectTrigger 1173
MQC 1174
.NET 應用程式的字集 ID 1174
IBM WebSphere MQ C++ 類別 1176
MQI 交互參照 1178
ImqAuthentication 記錄 1191
ImqBinary 1193
ImqCache 1195
ImqChannel 1198
ImqCICSBridge 標頭 1203
ImqDeadLetterHeader 1208
ImqDistribution 清單 1211
ImqError 1212
ImqGetMessageOptions 1213
ImqHeader 1216
ImqIMSBridge 標頭 1218
ImqItem 1220
ImqMessage 1222
ImqMessage 追蹤器 1228
ImqNamelist 1231
ImqObject 1233
ImqProcess 1238
ImqPutMessageOptions 1239
ImqQueue 1241
ImqQueue 管理程式 1251
ImqReference 標頭 1266
ImqString 1269
ImqTrigger 1273
ImqWork 標頭 1276
適用於 Java 程式庫的 IBM WebSphere MQ 類別 1277
適用於 JMS 物件的 IBM WebSphere MQ 類別內容 1278
APPLICATIONNAME 1282
ASYNCEXCEPTION 1282
BROKERCCDURSUBQ 1283
BROKERCCSUBQ 1284
BROKERCONQ 1284
BROKERDURSUBQ 1285
BROKERPUBQ 1285
BROKERPUBQMGR 1286
BROKERQMGR 1286
BROKERSUBQ 1286
BROKERVER 1287
CCDTURL 1287
CCSID 1288

CHANNEL..... 1288
CLEANUP..... 1289
CLEANUPINT..... 1289
ConnectionNameList..... 1290
CLIENTRECONNECTOPTIONS..... 1290
CLIENTRECONNECTTIMEOUT..... 1291
CLIENTID..... 1291
CLONESUPP..... 1292
COMPHDR..... 1292
COMPMSG..... 1293
CONNOPT..... 1293
CONNTAG..... 1294
說明..... 1294
DIRECTAUTH..... 1295
ENCODING..... 1295
EXPIRY..... 1296
FAILIFQUIESCE..... 1297
HOSTNAME..... 1297
LOCALADDRESS..... 1298
MAPNAMESTYLE..... 1299
MAXBUFFSIZE..... 1299
MDREAD..... 1299
MDWRITE..... 1300
MDMSGCTX..... 1300
MSGBATCHSZ..... 1301
MSGBODY..... 1301
MSGRETENTION..... 1302
MSGSELECTION..... 1303
MULTICAST..... 1303
OPTIMISTICPUBLICATION..... 1304
OUTCOMENOTIFICATION..... 1304
PERSISTENCE..... 1305
POLLINGINT..... 1305
PORT..... 1306
PRIORITY..... 1306
PROCESSDURATION..... 1307
PROVIDERVERSION..... 1307
PROXYHOSTNAME..... 1308
PROXYPORT..... 1309
PUBACKINT..... 1309
PUTASYNCALLOWED..... 1310
QMANAGER..... 1310
佇列..... 1311
READAHEADALLOWED..... 1311
READAHEADCLOSEPOLICY..... 1312
RECEIVECCSID..... 1312
RECEIVECONVERSION..... 1313
RECEIVEISOLATION..... 1313
RECEXIT..... 1313
RECEXITINIT..... 1314
REPLYTOSTYLE..... 1314
RESCANINT..... 1315
SECEXIT..... 1315
SECEXITINIT..... 1316
SENDCHECKCOUNT..... 1316
SENDEXIT..... 1317
SENDEXITINIT..... 1317
SHARECONVALLOWED..... 1318

SPARSESUBS....1318
SSLCIPHERSUITE....1319
SSLCRL....1319
SSLFIPSREQUIRED....1320
SSLPEERNAME....1320
SSLRESETCOUNT....1320
STATREFRESHINT....1321
SUBSTORE....1321
SYNCPOINTALLGETS....1322
TARGCLIENT....1322
TARGCLIENTMATCHING....1323
TEMPMODEL....1323
TEMPQPREFIX....1324
TEMPTOPICPREFIX....1324
TOPIC....1325
TRANSPORT....1325
WILDCARDFORMAT....1326
內容相依關係....1326
ENCODING 內容....1327
SSL 內容....1328

注意事項....1331
程式設計介面資訊....1332
商標....1332

開發應用程式參照

請使用本節中的資訊來協助您開發 IBM WebSphere MQ 應用程式:

相關工作

開發應用程式

MQI 應用程式參照

使用本節中提供的鏈結，可協助您開發 MQI 應用程式:

- 第 7 頁的『程式碼範例』
- 第 49 頁的『常數』
- 第 214 頁的『MQI 中使用的資料類型』
- 第 545 頁的『函數呼叫』
- 第 700 頁的『物件的屬性』
- 第 764 頁的『回覆碼』
- 第 764 頁的『MQI 選項的驗證規則』
- 第 785 頁的『機器編碼』
- 第 788 頁的『報告選項及訊息旗標』
- 第 791 頁的『資料轉換結束程式』
- 第 811 頁的『指定為 MQRFH2 元素的內容』
- 第 817 頁的『字碼頁轉換』

相關概念

第 901 頁的『使用者結束程式、API 結束程式及可安裝的服務參照』
使用本節中提供的鏈結，可協助您開發使用者結束程式、API 結束程式及可安裝的服務應用程式:

第 1277 頁的『Java 程式庫的 IBM WebSphere MQ 類別』
Java 程式庫的 IBM WebSphere MQ 類別位置會因平台而異。 當您啟動應用程式時，請指定此位置。

相關工作

開發應用程式

相關參考

第 850 頁的『SOAP 參照』
WebSphere MQ SOAP 參照資訊的傳輸依字母順序排列。

第 1087 頁的『HTTP IBM WebSphere MQ 橋接器的參考資料』
HTTP IBM WebSphere MQ 橋接器的參考主題，按字母順序排列

第 1120 頁的『IBM WebSphere MQ .NET 類別和介面』
IBM WebSphere MQ .NET 類別和介面會按字母順序列出。 說明內容、方法和建構子。

第 1176 頁的『IBM WebSphere MQ C++ 類別』
IBM WebSphere MQ C++ 類別會封裝「 IBM WebSphere MQ 訊息佇列介面 (MQI)」。 有一個單一 C++ 標頭檔 **imqi.hpp**，它涵蓋所有這些類別。

相關資訊

../com.ibm.mq.javadoc.doc/WMQJMSClasses/index.html

程式碼範例

使用本節中的參照資訊，可完成作業以解決您的商業需要。

© Copyright IBM Corp. 2007, 2024

C 語言範例

這個主題集合大部分取自 WebSphere MQ for z/OS 範例應用程式。 除非另有說明，否則它們適用於所有平台。

連接至佇列管理程式

此範例示範如何使用 MQCONN 呼叫將程式連接至 z/OS 批次中的佇列管理程式。

此擷取內容取自 WebSphere MQ for z/OS 所提供的「瀏覽」範例應用程式 (程式 CSQ4BCA1)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台)。

```
#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;
      }
.
.
.
   }
```

切斷佇列管理程式的連線

此範例示範如何使用 MQDISC 呼叫，在 z/OS 批次中中斷程式與佇列管理程式的連線。

在此程式碼擷取中使用的變數是在 第 8 頁的『連接至佇列管理程式』 中設定的變數。 此擷取內容取自 WebSphere MQ for z/OS 所提供的「瀏覽」範例應用程式 (程式 CSQ4BCA1)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台) 。

```
.
.
.
   /*                                                          */
   /*   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;
      }
⋮
```

建立動態佇列

此範例示範如何使用 MQOPEN 呼叫來建立動態佇列。

此摘錄取自 WebSphere MQ for z/OS 所提供的「郵件管理程式」範例應用程式 (程式 CSQ4TCD1)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台) 。

```
⋮
MQLONG  HCONN = 0;   /* Connection handle      */
MQHOBJ  HOBJ;        /* MailQ Object handle    */
MQHOBJ  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;
}
⋮
```

開啟現有佇列

此範例示範如何使用 MQOPEN 呼叫來開啟已定義的佇列。

此擷取內容取自 WebSphere MQ for z/OS 所提供的「瀏覽」範例應用程式 (程式 CSQ4BCA1)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台)。

```
#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 };
                              /* Object descriptor       */
   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 */
```

關閉佇列

此範例示範如何使用 MQCLOSE 呼叫來關閉佇列。

此擷取內容取自 WebSphere MQ for z/OS 所提供的「瀏覽」範例應用程式 (程式 CSQ4BCA1)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台)。

```
.
   /*                                                    */
   /*   Close the queue.                                 */
   /*   Test the output of the close call.  If the 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;
      }
⋮
```

使用 *MQPUT* 放置訊息

此範例示範如何使用 MQPUT 呼叫將訊息放置在佇列上。

此擷取並非取自 WebSphere MQ 所提供的範例應用程式。 如需範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台)。

```
⋮
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 */
        }
}
```

使用 *MQPUT1* 放置訊息

此範例示範如何使用 MQPUT1 呼叫來開啟佇列，將單一訊息放置在佇列上，然後關閉佇列。

此擷取內容取自 WebSphere MQ for z/OS 所提供的 Credit Check 範例應用程式 (程式 CSQ4CCB5)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台) 。

```
⋮
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;
   }
.
```

取得訊息

此範例示範如何使用 MQGET 呼叫從佇列中移除訊息。

此擷取內容取自 WebSphere MQ for z/OS 所提供的「瀏覽」範例應用程式 (程式 CSQ4BCA1)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台)。

```
#include "cmqc.h"
.
#define BUFFERLENGTH 80
.
int  main(int argc, char *argv[] )
   {
   /*                                                   */
   /*     Variables for MQ calls                        */
   /*                                                   */
   MQHCONN Hconn ;            /* Connection handle      */
   MQLONG  CompCode;          /* Completion code        */
   MQLONG  Reason;            /* Qualifying reason      */
   MQHOBJ  Hobj;              /* Object handle          */
   MQMD    MsgDesc  = { MQMD_DEFAULT };
                              /* Message descriptor     */
   MQLONG  DataLength ;       /* Length of the message  */
   MQCHAR  Buffer[BUFFERLENGTH+1];
                              /* Area for message data  */
   MQGMO   GetMsgOpts = { MQGMO_DEFAULT };
                              /* Options which control  */
                              /*  the MQGET call        */
   MQLONG  BufferLength = BUFFERLENGTH ;
                              /* 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).                            */
   /*                                                   */
   GetMsgOpts.Options = 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 */
```

使用等待選項取得訊息

此範例示範如何使用 MQGET 呼叫的等待選項。

此程式碼接受截斷的訊息。 此擷取內容取自 WebSphere MQ for z/OS 所提供的 Credit Check 範例應用程式 (程式 CSQ4CCB5)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台)。

```
⋮
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        */
```

```
⋮
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;
      MsgBuffLen = 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();
         }
⋮
```

使用信號取得訊息

信號僅適用於 *WebSphere MQ for z/OS*。

此範例示範如何使用 MQGET 呼叫來設定信號，以便在適當的訊息到達佇列時通知您。 此擷取並非取自 WebSphere MQ 所提供的範例應用程式。

```
⋮
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;
}
```

查詢物件的屬性

此範例示範如何使用 MQINQ 呼叫來查詢佇列的屬性。

此摘錄取自 WebSphere MQ for z/OS 所提供的「佇列屬性」範例應用程式 (程式 CSQ4CCC1)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台) 。

```
#include <cmqc.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,
            SelectorsTable,
            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 */
```

設定佇列的屬性

此範例示範如何使用 MQSET 呼叫來變更佇列的屬性。

此摘錄取自 WebSphere MQ for z/OS 所提供的「佇列屬性」範例應用程式 (程式 CSQ4CCC1)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台) 。

```
#include <cmqc.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  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 set call:  */
      /*    - Set SelectorsTable to the attributes to be */
      /*      set                                      */
      /*    - Set IntAttrsTable to the required status */
      /*    - All other variables are already set      */
      /*                                               */
      SelectorsTable[0] = MQIA_INHIBIT_GET;
      SelectorsTable[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 */
```

使用 *MQSTAT* 擷取狀態資訊

此範例示範如何使用 MQSTAT 發出非同步 MQPUT 並擷取狀態資訊。

此擷取資料取自「呼叫 MQSTAT」範例應用程式 (程式 amqsapt0) 隨附於 WebSphere MQ for Windows 系統。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台) 。

```
/**************************************************************************/
/*                                                                        */
/* 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   O_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,
            &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,                   /* connection handle            */
            &CompCode,               /* completion code              */
            &Reason);                /* reason code                  */

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

   /******************************************************************/
   /*                                                                  */
   /* END OF AMQSAPT0                                                  */
   /*                                                                  */
   /******************************************************************/
   printf("Sample AMQSAPT0 end\n");
   return(0);
 }
```

COBOL 範例

此主題集合取自 WebSphere MQ for z/OS 範例應用程式。 除非另有說明，否則它們適用於所有平台。

連接至佇列管理程式

此範例示範如何使用 MQCONN 呼叫將程式連接至 z/OS 批次中的佇列管理程式。

此擷取內容取自 WebSphere MQ for z/OS 所提供的「瀏覽」範例應用程式 (程式 CSQ4BVA1)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台)。

```
* -------------------------------------------------------*
 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.
⋮
```

切斷佇列管理程式的連線

此範例示範如何使用 MQDISC 呼叫，在 z/OS 批次中中斷程式與佇列管理程式的連線。

在此程式碼擷取中使用的變數是在 第 23 頁的『連接至佇列管理程式』中設定的變數。 此擷取內容取自 WebSphere MQ for z/OS 所提供的「瀏覽」範例應用程式 (程式 CSQ4BVA1)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台)。

```
⋮
*
* 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.
⋮
```

建立動態佇列

此範例示範如何使用 MQOPEN 呼叫來建立動態佇列。

此摘錄取自 WebSphere MQ for z/OS 所提供的 Credit Check 範例應用程式 (程式 CSQ4CVB1)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台)。

```
⋮
* -------------------------------------------------------*
 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 = MQOO-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
*
```

開啟現有佇列

此範例示範如何使用 MQOPEN 呼叫來開啟現有佇列。

此擷取內容取自 WebSphere MQ for z/OS 所提供的「瀏覽」範例應用程式 (程式 CSQ4BVA1)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台)。

```
⋮
* ------------------------------------------------------*
 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 = MQOO-INQUIRE + MQOO-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 'MQOPEN'      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
```

關閉佇列

此範例示範如何使用 MQCLOSE 呼叫。

在此程式碼擷取中使用的變數是在 第 23 頁的『連接至佇列管理程式』中設定的變數。 此擷取內容取自 WebSphere MQ for z/OS 所提供的「瀏覽」範例應用程式 (程式 CSQ4BVA1)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台) 。

```
⋮
*
*    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.
*
```

使用 *MQPUT* 放置訊息

此範例示範如何使用環境定義來使用 MQPUT 呼叫。

此摘錄取自 WebSphere MQ for z/OS 所提供的 Credit Check 範例應用程式 (程式 CSQ4CVB1)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台) 。

```
⋮
* -------------------------------------------------------*
 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.
```

使用 *MQPUT1* 放置訊息

此範例示範如何使用 MQPUT1 呼叫。

此摘錄取自 WebSphere MQ for z/OS 所提供的「信用檢查」範例應用程式 (程式 CSQ4CVB5)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台) 。

```
⋮
* ----------------------------------------------------------*
 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.
*
```

取得訊息

此範例示範如何使用 MQGET 呼叫從佇列中移除訊息。

此摘錄取自 WebSphere MQ for z/OS 所提供的 Credit Check 範例應用程式 (程式 CSQ4CVB1)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台) 。

```
⋮
* ----------------------------------------------------------*
 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 CMQGMOV.
*
*    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.
```

使用等待選項取得訊息

此範例示範如何搭配使用 MQGET 呼叫與等待選項，並接受截斷的訊息。

此摘錄取自 WebSphere MQ for z/OS 所提供的「信用檢查」範例應用程式 (程式 CSQ4CVB5)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台) 。

```
⋮
* ------------------------------------------------------------*
 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-SYNCPOINT.
  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 MQOD-OBJECTNAME   TO M02-OBJECTNAME
              PERFORM RECORD-CALL-ERROR
     END-IF.
⋮
```

使用信號取得訊息

此範例示範如何搭配使用 MQGET 呼叫與信號。 此摘錄取自 WebSphere MQ for z/OS 所提供的「信用檢查」範例應用程式 (程式 CSQ4CVB2)。

信號僅適用於 *WebSphere MQ for z/OS*。

```
⋮
* -------------------------------------------------------*
 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 CMQGMOV.
⋮
*    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 MQOD-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
*
⋮
```

查詢物件的屬性

此範例示範如何使用 MQINQ 呼叫來查詢佇列的屬性。

此擷取資料取自 WebSphere MQ for z/OS 所提供的「佇列屬性」範例應用程式 (程式 CSQ4CVC1)。 如需其他平台上範例應用程式的名稱和位置，請參閱 範例程式 (z/OS 以外的平台) 。

```
⋮
* ------------------------------------------------------------*
 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 for the inquiry call:
*     - Set W02-SELECTORS-TABLE to the attributes whose
*    status is required
*     - All other variables are already set
*
     MOVE MQIA-INHIBIT-GET TO W02-SELECTORS(1).
     MOVE MQIA-INHIBIT-PUT TO W02-SELECTORS(2).
```

```
*
*    Inquire about the attributes.
*
     CALL 'MQINQ' 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 inquiry:
*
* - If the completion code is not OK, display an error
*   message showing the completion and reason codes
*
* - Otherwise, move the correct attribute status into
*   the relevant screen map fields
*
     IF W02-COMPCODE NOT = MQCC-OK
        MOVE 'MQINQ'       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.
⋮
```

設定佇列的屬性

此範例示範如何使用 MQSET 呼叫來變更佇列的屬性。

此擷取資料取自 WebSphere MQ for z/OS 所提供的「佇列屬性」範例應用程式 (程式 CSQ4CVC1)。 如需其他平台上範例應用程式的名稱及位置，請參閱 範例程式 (z/OS 以外的平台)

```
⋮
* ------------------------------------------------------*
 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.
```

System/390 組譯語言範例

這個主題集合大部分取自 WebSphere MQ for z/OS 範例應用程式。

連接至佇列管理程式

此範例示範如何使用 MQCONN 呼叫將程式連接至 z/OS 批次中的佇列管理程式。

此擷取內容取自 WebSphere MQ for z/OS 所提供的「瀏覽」範例程式 (CSQ4BAA1)。

```
⋮
WORKAREA DSECT
*
PARMLIST CALL ,(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
*
         LA    R0,MQCC_OK        Expected compcode
         C     R0,COMPCODE       As expected?
         BER   R6                Yes .. return to caller
*
         MVC   INF4_TYP,=CL10'CONNECT   '
         BAL   R7,ERRCODE             Translate error
         LA    R0,8                   Set exit code
         ST    R0,EXITCODE            to 8
```

```
         B     ENDPROG                End the program
*
```

切斷佇列管理程式的連線

此範例示範如何使用 MQDISC 呼叫，在 z/OS 批次中中斷程式與佇列管理程式的連線。

此擷取並非取自 WebSphere MQ 所提供的範例應用程式。

```
⋮
*
*        ISSUE MQI DISC REQUEST USING REENTRANT FORM
*        OF CALL MACRO
*
*        HCONN WAS SET BY A PREVIOUS MQCONN REQUEST
*        R5 = WORK REGISTER
*
DISC     DS    0H
         CALL  MQDISC,                       X
               (HCONN,                       X
               COMPCODE,                     X
               REASON),                      X
               VL,MF=(E,CALLLST)
*
         LA    R5,MQCC_OK
         C     R5,COMPCODE
         BNE   BADCALL
⋮
```

```
BADCALL  DS    0H
⋮
*                    CONSTANTS
*
         CMQA
*
*        WORKING STORAGE (RE-ENTRANT)
*
WEG3     DSECT
*
CALLLST  CALL ,(0,0,0,0,0,0,0,0,0,0,0),VL,MF=L
*
HCONN    DS   F
COMPCODE DS   F
REASON   DS   F
*
*
LEG3     EQU  *-WKEG3
         END
```

建立動態佇列

此範例示範如何使用 MQOPEN 呼叫來建立動態佇列。

此擷取並非取自 WebSphere MQ 所提供的範例應用程式。

```
⋮
*
*    R5 = WORK REGISTER.
*
OPEN     DS   0H
*
         MVC  WOD_AREA,MQOD_AREA INITIALIZE WORKING VERSION OF
*                                     MQOD WITH DEFAULTS
         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,CALLLST)
*
         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
       CMQA                      MQI VALUE EQUATES
*
*   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
*
CALLLST  CALL ,(0,0,0,0,0,0,0,0,0,0,0),VL,MF=L  LIST FORM
                                                 OF CALL
*                                                 MACRO
⋮
         END
```

開啟現有佇列

此範例示範如何使用 MQOPEN 呼叫來開啟已定義的佇列。

它顯示如何指定兩個選項。 此擷取並非取自 WebSphere MQ 所提供的範例應用程式。

```
⋮
*
*    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
*
* 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,CALLLST)
*
         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
*
⋮
BADCALL  DS   0H
⋮
*
*
*   CONSTANTS:
*
Q_NAME   DC   CL48'REQUEST.QUEUE'  NAME OF QUEUE TO OPEN
*
         CMQODA DSECT=NO,LIST=YES  CONSTANT VERSION OF MQOD
         CMQA                      MQI VALUE EQUATES
*
*   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
*
WOD  CMQODA DSECT=NO,LIST=YES  WORKING VERSION OF MQOD
*
CALLLST  CALL ,(0,0,0,0,0,0,0,0,0,0,0),VL,MF=L  LIST FORM
                                                 OF CALL
*                                                 MACRO
⋮
         END
```

關閉佇列

此範例示範如何使用 MQCLOSE 呼叫來關閉佇列。

此擷取並非取自 WebSphere MQ 所提供的範例應用程式。

```
⋮
*
* ISSUE MQI CLOSE REQUEST USING REENTRANT FROM OF
* CALL MACRO
*
*        HCONN WAS SET BY A PREVIOUS MQCONN REQUEST
*        HOBJ  WAS SET BY A PREVIOUS MQOPEN REQUEST
*        R5 = WORK REGISTER
*
CLOSE    DS    0H
         LA    R5,MQCO_NONE        NO SPECIAL CLOSE OPTIONS
         ST    R5,OPTIONS              ARE REQUIRED.
*
         CALL  MQCLOSE,                        X
               (HCONN,                         X
               HOBJ,                           X
               OPTIONS,                        X
               COMPCODE,                       X
               REASON),                        X
               VL,MF=(E,CALLLST)
*
         LA    R5,MQCC_OK
         C     R5,COMPCODE
         BNE   BADCALL
*
⋮
BADCALL  DS    0H
⋮
*                   CONSTANTS
*
         CMQA
*
*        WORKING STORAGE (REENTRANT)
```

```
*
WEG4     DSECT
*
CALLLST  CALL ,(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
```

使用 *MQPUT* 放置訊息

此範例示範如何使用 MQPUT 呼叫將訊息放置在佇列上。

此擷取並非取自 WebSphere MQ 所提供的範例應用程式。

```
⋮
*     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 ,(0,0,0,0,0,0,0,0,0,0,0),VL,MF=L
*
⋮
         END
```

使用 *MQPUT1* 放置訊息

此範例示範如何使用 MQPUT1 呼叫來開啟佇列，將單一訊息放置在佇列上，然後關閉佇列。

此擷取並非取自 WebSphere MQ 所提供的範例應用程式。

```
⋮
*
*     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,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
  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
```

取得訊息

此範例示範如何使用 MQGET 呼叫從佇列中移除訊息。

此擷取並非取自 WebSphere MQ 所提供的範例應用程式。

```
⋮
*
*     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),VL,MF=L
*
.
         END
```

使用等待選項取得訊息

此範例示範如何使用 MQGET 呼叫的等待選項。

此程式碼接受截斷的訊息。 此擷取並非取自 WebSphere MQ 所提供的範例應用程式。

```
.
*     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),VL,MF=L
*
⋮
         END
```

使用信號取得訊息

此範例示範如何使用 MQGET 呼叫來設定信號，以便在適當的訊息到達佇列時通知您。

此擷取並非取自 WebSphere MQ 所提供的範例應用程式。

```
⋮
*
*     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
               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.
         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
*
```

```
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),VL,MF=L
*
⋮
         END
```

查詢及設定佇列的屬性

此範例示範如何使用 MQINQ 呼叫來查詢佇列的屬性，以及使用 MQSET 呼叫來變更佇列的屬性。

此摘錄取自 WebSphere MQ for z/OS 所提供的「佇列屬性」範例應用程式 (程式 CSQ4CAC1)。

```
⋮
DFHEISTG DSECT
⋮
OBJDESC  CMQODA LIST=YES   Working object descriptor
*
SELECTORCOUNT   DS F       Number of selectors
INTATTRCOUNT    DS F       Number of integer attributes
CHARATTRLENGTH  DS F       char attributes length
CHARATTRS       DS C       Area for char attributes
*
OPTIONS  DS   F            Command options
HCONN    DS   F            Handle of connection
HOBJ     DS   F            Handle of object
COMPCODE DS   F            Completion code
REASON   DS   F            Reason code
SELECTOR DS   2F           Array of selectors
INTATTRS DS   2F           Array of integer attributes
⋮
OBJECT   DS   CL(MQ_Q_NAME_LENGTH)   Name of queue
⋮
CALLLIST CALL ,(0,0,0,0,0,0,0,0,0,0,0),VL,MF=L
************************************************************
*                   PROGRAM EXECUTION STARTS HERE        *
⋮
CSQ4CAC1 DFHEIENT CODEREG=(R3),DATAREG=(R13)
⋮
*        Initialize the variables for the set 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 q attribute selector
         ST   R0,SELECTOR+0       Place in field
         LA   R0,MQIA_INHIBIT_PUT Load q attribute selector
         ST   R0,SELECTOR+4       Place in field
*
UPDTEST  DS   0H
         CLC  ACTION,CINHIB       Are we inhibiting?
         BE   UPDINHBT            Yes branch to section
*
         CLC  ACTION,CALLOW       Are we allowing?
         BE   UPDALLOW            Yes branch to section
*
         MVC  M00_MSG,M01_MSG1    Invalid request
         BR   R6                  Return to caller
*
```

```
UPDINHBT DS   0H
         MVC  UPDTYPE,CINHIBIT      Indicate action type
         LA   R0,MQQA_GET_INHIBITED Load attribute value
         ST   R0,INTATTRS+0         Place in field
         LA   R0,MQQA_PUT_INHIBITED Load attribute value
         ST   R0,INTATTRS+4         Place in field
         B    UPDCALL               Go and do call
*
UPDALLOW DS   0H
         MVC  UPDTYPE,CALLOWED      Indicate action type
         LA   R0,MQQA_GET_ALLOWED   Load attribute value
```

```
         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?
⋮
```

常數

使用本節中的參照資訊，可完成作業以解決您的商業需要。

IBM WebSphere MQ COPY、標頭、併入和模組檔案

此資訊是一般用途程式設計介面資訊。

本節包含的資訊可協助您將 MQI 用於各種程式設計語言，如下所示。

C 標頭檔

標頭檔可協助您撰寫使用 MQI 的 C 應用程式。 這些標頭檔在表格中彙總:

表 *1: C* 標頭檔-呼叫原型、資料類型、回覆碼、常數及結構					
檔名	**說明**	**IBM i**	**UNIX 和 Linux® 系統**	**Windows**	**z/OS**
呼叫原型、資料類型、回覆碼、常數及結構					
CMQC	MQI 定義	C	C	C	C
CMQBC	MQAI 定義	C	C	C	
CMQEC	介面進入點定義 (包括 CMQC、CMQXC 及 CMQZC)		C	C	
CMQCFC	PCF 定義	C	C	C	C
CMQPSC	發佈/訂閱定義	C	C	C	C
CMQXC	通道和結束程式定義	C	C	C	C
CMQZC	可安裝的服務定義	C	C	C	
索引鍵: C= 提供的檔案					

COBOL COPY 檔案

提供各種 COPY 檔案來協助您撰寫使用 MQI 的 COBOL 應用程式。 這些檔案彙總在表格中:

表 *2: COBOL* 複製檔案-回覆碼、常數及結構					
檔名	**說明**	**IBM i**	**UNIX 系統**	**Windows**	**z/OS**
回覆碼和常數					
CMQx	MQI 定義	V	V	V	V
CMQCFx	PCF 定義	V	V	V	V
CMQPSx	發佈/訂閱定義	V	V	V	V
CMQXx	通道和結束程式定義	V	V	V	V
結構					
CMQAIRx	MQAIR-鑑別資訊記錄		V L	V L	
CMQBOx	MQBO-開始選項	V L	V L	V L	
CMQCDx	MQCD-通道定義	V L	V L	V L	V L
CMQCFBFx	MQCFBF-PCF 位元組字串過濾器參數	V L	V L	V L	V L
CMQCFBSx	MQCFBS-PCF 位元組字串參數	V L	V L	V L	V L
CMQCFGRx	MQCFGR-PCF 群組參數	V L	V L	V L	V L
CMQCFHx	MQCFH-PCF 標頭	V L	V L	V L	V L
CMQCFIFx	MQCFIF-PCF 整數過濾器參數	V L	V L	V L	V L
CMQCFILx	MQCFIL-PCF 整數清單參數	V L	V L	V L	V L
CMQCFINx	MQCFIN-PCF 整數參數	V L	V L	V L	V L
CMQCFSFx	MQCFSF-PCF 字串過濾器參數	V L	V L	V L	V L
CMQCFSLx	MQCFSL-PCF 字串清單參數	V L	V L	V L	V L

表 *2: COBOL* 複製檔案-回覆碼、常數及結構 *(*繼續*)*

檔名	說明	IBM i	UNIX 系統	Windows	z/OS
CMQCFSTx	MQCFST-PCF 字串參數	V L	V L	V L	V L
CMQCFXLx	MQCFIL64 -PCF 64 位元整數清單參數	V L	V L	V L	V L
CMQCFXNx	MQCFIN64 -PCF 64 位元整數參數	V L	V L	V L	V L
CMQCHRVx	MQCHARV-可變長度字串	V L	V L	V L	V L
CMQCIHx	MQCIH- CICS 橋接器標頭	V L	V L	V L	V L
CMQCNOx	MQCNO-連接選項	V L	V L	V L	V L
CMQCSPx	MQCSP-安全參數	V L	V L	V L	V L
CMQCXPx	MQCXP-通道結束程式參數	V L			V L
CMQDHx	MQDH-Distribution 標頭	V L	V L	V L	V L
CMQDLHx	MQDLH-無法傳送郵件的標頭	V L	V L	V L	V L
CMQDXPx	MQDXP-資料轉換結束程式參數	V L		V L	
CMQEPHx	MQEPH-內嵌 PCF 標頭	V L	V L	V L	V L
CMQGMOx	MQGMO-取得訊息選項	V L	V L	V L	V L
CMQIIHx	MQIIH- IMS 資訊標頭	V L	V L	V L	V L
CMQMDx	MQMD-訊息描述子	V L	V L	V L	V L
CMQMD1x	MQMD1 -訊息描述子第 1 版	V L	V L	V L	V L
CMQMD2x	MQMD2 -訊息描述子第 2 版	V L	V L	V L	V L
CMQMDEx	MQMDE-延伸的訊息描述子	V L	V L	V L	V L
CMQODx	MQOD-物件描述子	V L	V L	V L	V L
CMQORx	MQOR-物件記錄	V L	V L	V L	V L
CMQPMOx	MQPMO-放置訊息選項	V L	V L	V L	V L
CMQRFHx	MQRFH-規則和格式化標頭	V L	V L	V L	V L
CMQRFH2x	MQRFH2 -規則和格式化標頭 2	V L	V L	V L	V L
CMQRMHx	MQRMH-參照訊息標頭	V L	V L	V L	V L
CMQRRx	MQRR-回應記錄	V L	V L	V L	
CMQSCOx	MQSCO-SSL 配置選項		V L	V L	
CMQTMx	MQTM-觸發訊息	V L		V L	V L
CMQTMCx	MQTMC-觸發訊息字元	V L	V L		
CMQTMC2x	MQTMC2 -觸發訊息 2 個字元	V L	V L	V L	V L
CMQWIHx	MQWIH-工作資訊標頭	V L	V L	V L	V L
CMQXQHx	MQXQH-傳輸佇列標頭	V L	V L	V L	V L

索引鍵:

- 已提供起始值的檔案， x = V
- 未提供起始值的檔案， x = L

PL/I 併入檔

下列 INCLUDE 檔案是針對 PL/I 程式設計語言所提供。 這些檔案僅適用於 z/OS 。

表 *3: PL/I* 併入檔-資料類型、回覆碼、常數及結構					
檔名	**說明**	**IBM i**	**UNIX 系統**	**Windows**	**z/OS**
資料類型、回覆碼、常數及結構					
CMQP	MQI 定義				P
CMQCFP	PCF 定義				P
CMQEPP	進入點定義				P
CMQPSP	發佈/訂閱定義				P
CMQXP	通道和結束程式定義				P
索引鍵: P= 提供的檔案					

RPG 複製檔案

下列 COPY 檔案是針對 RPG 程式設計語言所提供。 這些檔案只能在 IBM i 上使用。

表 *4: RPG* 複製檔案-回覆碼、常數及結構					
檔名	**說明**	**IBM i**	**UNIX 系統**	**Windows**	**z/OS**
回覆碼和常數					
CMQx	MQI 定義	G R			
CMQCFx	PCF 定義	G			
CMQPSx	發佈/訂閱定義	G			
CMQXx	通道和結束程式定義	G R			
結構					
CMQBOx	MQBO-開始選項	G H			
CMQCDx	MQCD-通道定義	G H R			
CMQCFBFx	MQCFBF-PCF 位元組字串過濾器參數	G H			
CMQCFBSx	MQCFBS-PCF 位元組字串參數	G H			
CMQCFGRx	MQCFGR-PCF 群組參數	G H			
CMQCFHx	MQCFH-PCF 標頭	G H			
CMQCFIFx	MQCFIF-PCF 整數過濾器參數	G H			
CMQCFILx	MQCFIL-PCF 整數清單參數	G H			
CMQCFINx	MQCFIN-PCF 整數參數	G H			
CMQCFSFx	MQCFSF-PCF 字串過濾器參數	G H			
CMQCFSLx	MQCFSL-PCF 字串清單參數	G H			
CMQCFSTx	MQCFST-PCF 字串參數	G H			

表 *4: RPG* 複製檔案-回覆碼、常數及結構 *(*繼續*)*					
檔名	**說明**	**IBM i**	**UNIX 系統**	**Windows**	**z/OS**
CMQCFXLx	MQCFIL64 -PCF 64 位元整數清單參數	G H			
CMQCFXNx	MQCFIN64 -PCF 64 位元整數參數	G H			
CMQCHARVx	MQCHARV-可變長度字串	G H			
CMQCIHx	MQCIH- CICS 橋接器標頭	G H			
CMQCNOx	MQCNO-連接選項	G H			
CMQCSPx	MQCSP-安全參數	G H			
CMQCXPx	MQCXP-通道結束程式參數	G H R			
CMQDHx	MQDH-Distribution 標頭	G H R			
CMQDLHx	MQDLH-無法傳送郵件的標頭	G H R			
CMQDXPx	MQDXP-資料轉換結束程式參數	G H R			
CMQEPHx	MQEPH-內嵌 PCF 標頭	G H			
CMQGMOx	MQGMO-取得訊息選項	G H R			
CMQIIHx	MQIIH- IMS 資訊標頭	G H R			
CMQMDx	MQMD-訊息描述子	G H R			
CMQMD1x	MQMD1 -訊息描述子第 1 版	G H R			
CMQMD2x	MQMD2 -訊息描述子第 2 版	G H			
CMQMDEx	MQMDE-延伸的訊息描述子	G H R			
CMQODx	MQOD-物件描述子	G H R			
CMQORx	MQOR-物件記錄	G H R			
CMQPMOx	MQPMO-放置訊息選項	G H R			
CMQPXPx	MQPXP-發佈/訂閱遞送結束程式參數	G H			
CMQRFHx	MQRFH-規則和格式化標頭	G H			
CMQRFH2x	MQRFH2 -規則和格式化標頭 2	G H			
CMQRMHx	MQRMH-參照訊息標頭	G H R			
CMQRRx	MQRR-回應記錄	G H R			
CMQTMx	MQTM-觸發訊息	G H R			
CMQTMCx	MQTMC-觸發訊息字元	G H R			
CMQTMC2x	MQTMC2 -觸發訊息 2 個字元	G H R			
CMQWIHx	MQWIH-工作資訊標頭	G H			
CMQXQHx	MQXQH-傳輸佇列標頭	G H R			

表 *4: RPG* 複製檔案-回覆碼、常數及結構 *(*繼續*)*					
檔名	**說明**	**IBM i**	**UNIX 系統**	**Windows**	**z/OS**
索引鍵: • 靜態鏈結的檔案，已起始設定，提供的 x = G • 靜態鏈結的檔案，未起始設定，提供 x = H • 動態鏈結的檔案，已起始設定，提供， x = R					

Visual Basic 模組檔案

提供的標頭 (或表單) 檔案可協助您撰寫使用 MQI 的 Visual Basic 應用程式。 這些標頭檔僅以 32 位元版本提供，並在表格中彙總:

表 *5: Visual Basic* 模組檔案-呼叫宣告、資料類型、回覆碼、常數及結構					
檔名	**說明**	**IBM i**	**UNIX and Linux 系統**	**Windows**	**z/OS**
呼叫宣告、資料類型、回覆碼、常數及結構					
CMQB	MQI 定義			B	
CMQBB	MQAI 定義			B	
CMQCFB	PCF 定義			B	
CMQXB	通道和結束程式定義			B	
索引鍵: B= 提供的檔案					

MQ_ * (字串長度)

表 *6:* 常數值		
名稱	**小數值**	**十六進位值**
MQ_ABEND_CODE_LENGTH	4	X'00000004'
MQ_ACCOUNTING_TOKEN_LENGTH	32	X'00000020'
MQ_APPL_IDENTITY_DATA_LENGTH	32	X'00000020'
MQ_APPL_NAME_LENGTH	28	X'0000001C'
MQ_APPL_ORIGIN_DATA_LENGTH	4	X'00000004'
MQ 應用程式標籤長度	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'

表 6: 常數值 (繼續)

名稱	小數值	十六進位值
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'
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 資料長度	12	X'0000000C'
MQ_DISTINGUISHED_NAME_LENGTH	1024	X'00000400'
MQ_DNS_GROUP_NAME_LENGTH	18	X'00000012'
MQ 結束程式資料長度	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 格式長度	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'

表 6: 常數值 (繼續)

名稱	小數值	十六進位值
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'
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)	
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_OBJECT_INSTANCE_ID_LENGTH	24	X'00000018'
MQ_OBJECT_NAME_LENGTH	48	X'00000030'
MQ_PASS_TICKET_APPL_LENGTH	8	X'00000008'
MQ 密碼長度	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 選取器長度	10240	X'00002800'
MQ 服務 _ 引數長度	255	X'000000FF'
MQ_SERVICE_COMMAND_LENGTH	255	X'000000FF'
MQ_SERVICE_DESC_LENGTH	64	X'00000040'

表 6: 常數值 (繼續)

名稱	小數值	十六進位值
MQ_SERVICE_NAME_LENGTH	32	X'00000020'
MQ 服務 _ 路徑長度	255	X'000000FF'
MQ 服務 _ 步驟長度	8	X'00000008'
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_CRYPTO_HARDWARE_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 時間長度	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 版本長度	8	X'00000008'

表 6: 常數值 (繼續)		
名稱	**小數值**	**十六進位值**
MQ_XCF_GROUP_NAME_LENGTH	8	X'00000008'
MQ_XCF_MEMBER_NAME_LENGTH	16	X'00000010'

MQ_ * (指令格式字串長度)

表 7: 常數值		
名稱	**小數值**	**十六進位值**
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	64	X'00000040'
MQ 環境 _ 資訊 _ 長度	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 日誌路徑長度	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	12	X'0000000C'
MQ 安全 _ 設定檔長度	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 結束程式鏈結區域標頭結構)

表 8: 常數的結構	
名稱	**結構**
MQACH_STRUC_ID	"ACH¬"
MQACH_STRUC_ID_ARRAY	'A','C','H','¬'

註: 符號 ¬ 代表單一空白字元。

表 9: 常數值		
名稱	**小數值**	**十六進位值**
MQACH_VERSION_1	1	X'00000001'
MQACH_CURRENT_VERSION	1	X'00000001'
MQACH_LENGTH_1	(value differs by platform or version)	
MQACH_CURRENT_LENGTH	(value differs by platform or version)	

MQACT_ * (帳戶記號)

表 10: 常數名稱和值	
名稱	**值**
MQACT_NONE	X'00...00' (32 個空值)
MQACT_NONE_ARRAY	'\0','\0',... (32 個空值)

MQACT_ * (指令格式動作選項)

表 11: 常數值		
名稱	**小數值**	**十六進位值**
MQACT_FORCE_REMOVE	1	X'00000001'
MQACT_ADVANCE_LOG	2	X'00000002'
MQACT_COLLECT_STATISTICS	3	X'00000003'
MQ 動作 _ 發佈訂閱	4	X'00000004'

MQACTP_ * (動作)

表 12: 常數值		
名稱	**小數值**	**十六進位值**
MQACTP_NEW	0	X'00000000'
MQACTP_FORWARD	1	X'00000001'
MQACTP_REPLY	2	X'00000002'
MQACTP_REPORT	3	X'00000003'

MQACTT_ * (帳戶記號類型)

表 13: 常數值	
名稱	**十六進位值**
MQACTT_UNKNOWN	X'00'
MQACTT_CICS_LUOW_ID	X'01'

表 13: 常數值 (繼續)	
名稱	**十六進位值**
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'
MQACTT_NT_SECURITY_ID	X'0B'
MQACTT_USER	X'19'

MQADOPT_ * (採用新的 MCA 檢查並採用新的 MCA 類型)

採用新的 MCA 檢查

表 14: 常數值		
名稱	**小數值**	**十六進位值**
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'

採用新的 MCA 類型

表 15: 常數值		
名稱	**小數值**	**十六進位值**
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_ * (鑑別資訊記錄結構)

表 16: 常數的結構	
名稱	**結構**
MQAIR_STRUC_ID	"AIR¬"
MQAIR_STRUC_ID_ARRAY	'A','I','R','¬'

註: 符號 ¬ 代表單一空白字元。

表 17: 常數值		
名稱	**小數值**	**十六進位值**
MQAIR_VERSION_1	1	X'00000001'
MQAIR_VERSION_2	2	X'00000002'
MQAIR_CURRENT_VERSION	2	X'00000002'

MQAIT_ * (鑑別資訊類型)

表 *18:* 常數值		
名稱	**小數值**	**十六進位值**
MQAIT_ALL	0	X'00000000'
MQAIT_CRL_LDAP	1	X'00000001'
MQAIT_OCSP	2	X'00000002'

MQAS_ * (指令格式非同步狀態值)

表 *19:* 常數值		
名稱	**小數值**	**十六進位值**
MQAS_NONE	0	X'00000000'
已啟動 MQAS_STARTED	1	X'00000001'
MQAS_START_WAIT	2	X'00000002'
已停止 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_ * (放置應用程式類型)

表 *20:* 常數值		
名稱	**小數值**	**十六進位值**
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'
MQ 視窗	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'

表 20: 常數值 (繼續)

名稱	小數值	十六進位值
MQAT_VOS	14	X'0000000E'
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'
MQ 屬性分配管理系統	26	X'0000001A'
MQAT_QMGR_PUBLISH	26	X'0000001A'
MQAT_JAVA	28	X'0000001C'
MQAT_DQM	29	X'0000001D'
MQ 通道起始程式	30	X'0000001E'
MQAT_WLM	31	X'0000001F'
MQ 批次	32	X'00000020'
MQ 屬性 _ rrS_BATCH	33	X'00000021'
MQAT_SIB	34	X'00000022'
MQAT_DEFAULT	(value differs by platform or version)	
MQAT_USER_FIRST	65536	X'00010000'
MQAT_USER_LAST	999999999	X'3B9AC9FF'

MQAUTH_ * (指令格式「權限值」)

表 21: 常數值

名稱	小數值	十六進位值
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'

表 *21:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
MQAUTH_SET	14	X'0000000E'
MQAUTH_SET_ALL_CONTEXT	15	X'0000000F'
MQAUTH_SET_IDENTITY_CONTEXT	16	X'00000010'
MQAUTH_CONTROL	17	X'00000011'
已延伸 MQAUTH_CONTROL_EXTENDED	18	X'00000012'
MQAUTH_PUBLISH	19	X'00000013'
MQAUTH_SUBSCRIBE	20	X'00000014'
MQ 授權 _ 回復	21	X'00000015'
MQAUTH_SYSTEM	22	X'00000016'

MQAUTHOPT_ * (指令格式「權限選項」)

表 *22:* 常數值		
名稱	**小數值**	**十六進位值**
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 結束程式環境定義結構)

表 *23:* 常數的結構	
名稱	**結構**
MQAXC_STRUC_ID	"AXC¬"
MQAXC_STRUC_ID_ARRAY	'A','X','C','¬'

註: 符號 ¬ 代表單一空白字元。

表 *24:* 常數值		
名稱	**小數值**	**十六進位值**
MQAXC_VERSION_1	1	X'00000001'
MQAXC_CURRENT_VERSION	1	X'00000001'

MQAXP_ * (API 結束程式參數結構)

表 *25:* 常數的結構	
名稱	**結構**
MQAXP_STRUC_ID	"AXP¬"
MQAXP_STRUC_ID_ARRAY	'A','X','P','¬'

註: 符號 ¬ 代表單一空白字元。

表 *26:* 常數值		
名稱	**小數值**	**十六進位值**
MQAXP_VERSION_1	1	X'00000001'
MQAXP_VERSION_2	2	X'00000002'
MQAXP_CURRENT_VERSION	2	X'00000002'

MQBA_ * (位元組屬性選取元)

表 *27:* 常數值		
名稱	**小數值**	**十六進位值**
MQBA_FIRST	6001	X'00001771'
MQBA_LAST	8000	X'00001F40'

MQBACF_ * (指令格式位元組參數類型)

表 *28:* 常數值		
名稱	**小數值**	**十六進位值**
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_ * (mqAdd 字串和 mqSet 字串的緩衝區長度)

表 *29:* 常數值		
名稱	**小數值**	**十六進位值**
已終止 MQBL_NULL_TERMINATED	-1	X'FFFFFFFF'

MQBMHO_ * (緩衝區至訊息處理選項及結構)

緩衝區至訊息控點選項結構

表 *30:* 常數的結構	
名稱	**結構**
MQBMHO_STRUC_ID	"BMHO"
MQBMHO_STRUC_ID_ARRAY	'B','M','H','O'

註: 符號 ¬ 代表單一空白字元。

表 *31:* 常數值		
名稱	**小數值**	**十六進位值**
MQBMHO_VERSION_1	1	X'00000001'
MQBMHO_CURRENT_VERSION	1	X'00000001'

緩衝區至訊息控點選項

表 *32:* 常數值		
名稱	**小數值**	**十六進位值**
MQBMHO_NONE	0	X'00000000'
MQBHO_DELETE_PROPERTIES	1	X'00000001'

MQBND_ * (預設連結)

表 *33:* 常數值		
名稱	**小數值**	**十六進位值**
MQBND_BIND_ON_OPEN	0	X'00000000'
MQBND_BIND_NOT_FIXED	1	X'00000001'
MQBND_BIND_ON_GROUP	2	X'00000002'

MQBO_ * (開始選項及結構)

開始選項結構

表 *34:* 常數的結構	
名稱	**結構**
MQBO_STRUC_ID	"BO¬¬"
MQBO_STRUC_ID_ARRAY	'B','O','¬','¬'

註: 符號 ¬ 代表單一空白字元。

表 *35:* 常數值		
名稱	**小數值**	**十六進位值**
MQBO_VERSION_1	1	X'00000001'
MQBO_CURRENT_VERSION	1	X'00000001'

開始選項

表 *36:* 常數值		
名稱	**小數值**	**十六進位值**
MQBO_NONE	0	X'00000000'

MQBT_ * (指令格式橋接器類型)

表 *37:* 常數值		
名稱	**小數值**	**十六進位值**
MQBT_OTMA	1	X'00000001'

MQCA_ * (字元屬性選取元)

表 *38:* 常數值		
名稱	**小數值**	**十六進位值**
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'

表 38: 常數值 (繼續)

名稱	小數值	十六進位值
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'

表 *38:* 常數值 *(*繼續*)*

名稱	小數值	十六進位值
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_ * (指令格式字元參數類型)

表 *39:* 常數值

名稱	小數值	十六進位值
MQCACF_FIRST	3001	X'00000BB9'

表 39: 常數值 (繼續)

名稱	小數值	十六進位值
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'
MQ CACF_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'

表 39: 常數值 (繼續)

名稱	小數值	十六進位值
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'

表 39: 常數值 (繼續)

名稱	小數值	十六進位值
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'
MQ cACF_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'
MQ CACF_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'

表 39: 常數值 (繼續)

名稱	小數值	十六進位值
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'
MQ CACF_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'
MQ CACF_SERVICE_START_DATE	3144	X'00000C48'
MQ CACF_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'

表 *39:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
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'
已使用 MQCACF_LAST_USED	3172	X'00000C64'

MQCACH_ * (指令格式「字元通道參數類型」)

表 *40:* 常數值		
名稱	**小數值**	**十六進位值**
第一個 MQCACH_FIRST	3501	X'00000DAD'
MQ 快取 _ 通道名稱	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'
MQ 快取 _ 連線名稱	3506	X'00000DB2'
MQCACH_MCA_NAME	3507	X'00000DB3'
MQCACH_SEC_EXIT_NAME	3508	X'00000DB4'
MQCACH_MSG_EXIT_NAME	3509	X'00000DB5'
MQ 快取 _ 傳送 _ 結束程式名稱	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'
MQ 快取 _ 本端名稱	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'

表 *40:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
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'
已使用 MQCACH_LAST_USED	3559	X'00000DE7'

MQCADSD_ * (CICS 資訊標頭「ADS 描述子」)

表 *41:* 常數值		
名稱	**小數值**	**十六進位值**
MQCADSD_NONE	0	X'00000000'
MQCADSD_SEND	1	X'00000001'
MQCADSD_RECV	16	X'00000010'
MQCADSD_MSGFORMAT	256	X'00000100'

MQCAFTY_ * (連線親緣性值)

表 *42:* 常數值		
名稱	**小數值**	**十六進位值**
MQCAFTY_NONE	0	X'00000000'
MQCAFTY_PREFERRED	1	X'00000001'

MQCAMO_ * (指令格式字元監視參數類型)

表 *43:* 常數值		
名稱	**小數值**	**十六進位值**
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'
MQCAOM_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'
已使用 MQCAMO_LAST_USED	2712	X'00000A98'

MQCBC_ * (MQCBC 常數結構)

表 *44:* 常數的結構	
名稱	**結構**
MQCBC_STRUC_ID	"CBC¬"
MQCBC_STRUC_ID_ARRAY	'C','B','C','¬'

註: 符號 ¬ 代表單一空白字元。

表 *45:* 常數值		
名稱	**小數值**	**十六進位值**
MQCBC_VERSION_1	1	X'00000001'
MQCBC_CURRENT_VERSION	1	X'00000001'

MQCBCF_ * (MQCBC 常數旗標)

表 *46:* 常數值		
名稱	**小數值**	**十六進位值**
MQCBCF_NONE	0	X'00000000'
MQCBCF_READA_BUFFER_EMPTY	1	X'00000001'

MQCBCT_ * (MQCBC 常數回呼類型)

表 *47:* 常數值		
名稱	**小數值**	**十六進位值**
MQCBCT_START_CALL	1	X'00000001'
MQCBCT_STOP_CALL	2	X'00000002'
MQCBCT_REGISTER_CALL	3	X'00000003'

表 47: 常數值 (繼續)		
名稱	**小數值**	**十六進位值**
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 常數結構)

表 48: 常數的結構	
名稱	**結構**
MQCBD_STRUC_ID	"CBD¬"
MQCBD_STRUC_ID_ARRAY	'C','B','D','¬'

註: 符號 ¬ 代表單一空白字元。

表 49: 常數值		
名稱	**小數值**	**十六進位值**
MQCBD_VERSION_1	1	X'00000001'
MQCBD_CURRENT_VERSION	1	X'00000001'

MQCBDO_ * (MQCBD 常數回呼選項)

表 50: 常數值		
名稱	**小數值**	**十六進位值**
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_ * (mqCreate 工具袋的建立工具袋選項)

表 51: 常數值		
名稱	**小數值**	**十六進位值**
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'
容許 MQCBO_LIST_FORM_ALLOWED	2	X'00000002'
MQCBO_LIST_FORM_INHIBITED	0	X'00000000'
MQCBO_REORDER_AS_REQUIRED	4	X'00000004'
MQCBO_DO_NOT_REORDER	0	X'00000000'

表 51: 常數值 (繼續)		
名稱	**小數值**	**十六進位值**
MQCBO_CHECK_SELECTORS	8	X'00000008'
MQCBO_DO_NOT_CHECK_SELECTORS	0	X'00000000'

MQCBT_ * (MQCBD 常數這是回呼函數的類型)

表 52: 常數值		
名稱	**小數值**	**十六進位值**
MQCBT_MESSAGE_CONSUMER	1	X'00000001'
MQCBT_EVENT_HANDLER	2	X'00000002'

MQCC_ * (完成碼)

表 53: 常數值		
名稱	**小數值**	**十六進位值**
MQCC_OK	0	X'00000000'
MQCC_WARNING	1	X'00000001'
MQCC_FAILED	2	X'00000002'
MQCC_UNKNOWN	-1	X'FFFFFFFF'

MQCCSI_ * (編碼字集 ID)

表 54: 常數值		
名稱	**小數值**	**十六進位值**
未定義 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'FFFFFFFF'
MQCCSI_APPL	-3	X'FFFFFFFD'

MQCCT_ * (CICS 資訊標頭交談式作業選項)

表 55: 常數值		
名稱	**小數值**	**十六進位值**
MQCCT_YES	1	X'00000001'
MQCCT_NO	0	X'00000000'

MQCD_ * (通道定義結構)

表 56: 常數值		
名稱	**小數值**	**十六進位值**
MQCD_VERSION_1	1	X'00000001'
MQCD_VERSION_2	2	X'00000002'
MQCD_VERSION_3	3	X'00000003'
MQCD_VERSION_4	4	X'00000004'

表 56: 常數值 (繼續)		
名稱	**小數值**	**十六進位值**
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_CURRENT_VERSION	9	X'00000009'
MQCD_LENGTH_4	(value differs by platform or version)	
MQCD_LENGTH_5	(value differs by platform or version)	
MQCD_LENGTH_6	(value differs by platform or version)	
MQCD_LENGTH_7	(value differs by platform or version)	
MQCD_LENGTH_8	(value differs by platform or version)	
MQCD_LENGTH_9	(value differs by platform or version)	
MQCD_CURRENT_LENGTH	(value differs by platform or version)	

MQCDC_ * (通道資料轉換)

表 57: 常數值		
名稱	**小數值**	**十六進位值**
MQCDC_SENDER_CONVERSION	1	X'00000001'
MQCDC_NO_SENDER_CONVERSION	0	X'00000000'

MQCERT_ * (憑證驗證原則類型)

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

MQCF_ * (功能旗標)

表 58: 常數值		
名稱	**小數值**	**十六進位值**
MQCF_NONE	0	X'00000000'
MQCF_DIST_LISTS	1	X'00000001'

MQCFAC_ * (CICS 資訊標頭「機能」)

表 59: 常數名稱和值	
名稱	**十六進位值**
MQCFAC_NONE	X'00...00' (8 個空值)

表 59: 常數名稱和值 (繼續)	
名稱	**十六進位值**
MQCFAC_NONE_ARRAY	'\0','\0',... (8 個空值)

MQCFBF_ * (指令格式位元組字串過濾器參數結構)

表 60: 常數值		
名稱	**小數值**	**十六進位值**
MQCFBF_STRUC_LENGTH_FIXED	20	X'00000014'

MQCFBS_ * (指令格式位元組字串參數結構)

表 61: 常數值		
名稱	**小數值**	**十六進位值**
MQCFBS_STRUC_LENGTH_FIXED	16	X'00000010'

MQCFC_ * (指令格式標頭控制項選項)

表 62: 常數值		
名稱	**小數值**	**十六進位值**
MQCFC_LAST	1	X'00000001'
MQCFC_NOT_LAST	0	X'00000000'

MQCFGR_ * (指令格式群組參數結構)

表 63: 常數值		
名稱	**小數值**	**十六進位值**
MQCFGR_STRUC_LENGTH	16	X'00000010'

MQCFH_ * (指令格式標頭結構)

表 64: 常數值		
名稱	**小數值**	**十六進位值**
MQCFH_STRUC_LENGTH	36	X'00000024'
MQCFH_VERSION_1	1	X'00000001'
MQCFH_VERSION_2	2	X'00000002'
MQCFH_VERSION_3	3	X'00000003'
MQCFH_CURRENT_VERSION	3	X'00000003'

MQCFIF_ * (指令格式整數過濾器參數結構)

表 65: 常數值		
名稱	**小數值**	**十六進位值**
MQCFIF_STRUC_LENGTH	20	X'00000014'

MQCFIL_ * (指令格式整數清單參數結構)

表 66: 常數值		
名稱	**小數值**	**十六進位值**
MQCFIL_STRUC_LENGTH_FIXED	16	X'00000010'

MQCFIL64_* (指令格式 64 位元整數清單參數結構)

表 67: 常數值		
名稱	**小數值**	**十六進位值**
MQCFIL64_STRUC_LENGTH_FIXED	16	X'00000010'

MQCFIN_ * (指令格式整數參數結構)

表 68: 常數值		
名稱	**小數值**	**十六進位值**
MQCFIN_STRUC_LENGTH	16	X'00000010'

MQCFIN64_* (指令格式 64 位元整數參數結構)

表 69: 常數值		
名稱	**小數值**	**十六進位值**
MQCFIN64_STRUC_LENGTH	24	X'00000018'

MQCFO_ * (指令格式「重新整理儲存庫選項」及指令格式「移除佇列選項」)

指令格式重新整理儲存庫選項

表 70: 常數值		
名稱	**小數值**	**十六進位值**
MQCFO_REFRESH_REPOSITORY_YES	1	X'00000001'
MQCFO_REFRESH_REPOSITORY_NO	0	X'00000000'

指令格式移除佇列選項

表 71: 常數值		
名稱	**小數值**	**十六進位值**
MQCFO_REMOVE_QUEUES_YES	1	X'00000001'
MQCFO_REMOVE_QUEUES_NO	0	X'00000000'

MQCFOP_ * (指令格式過濾器運算子)

表 72: 常數值		
名稱	**小數值**	**十六進位值**
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'

表 72: 常數值 (繼續)		
名稱	**小數值**	**十六進位值**
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 可回復性)

表 73: 常數值		
名稱	**小數值**	**十六進位值**
MQCFR_YES	1	X'00000001'
MQCFR_NO	0	X'00000000'

MQCFSF_ * (指令格式字串過濾器參數結構)

表 74: 常數值		
名稱	**小數值**	**十六進位值**
MQCFSF_STRUC_LENGTH_FIXED	24	X'00000018'

MQCFSL_ * (指令格式字串清單參數結構)

表 75: 常數值		
名稱	**小數值**	**十六進位值**
MQCFSL_STRUC_LENGTH_FIXED	24	X'00000018'

MQCFST_ * (指令格式字串參數結構)

表 76: 常數值		
名稱	**小數值**	**十六進位值**
MQCFST_STRUC_LENGTH_FIXED	20	X'00000014'

MQCFSTATUS_ * (指令格式 CF 狀態)

表 77: 常數值		
名稱	**小數值**	**十六進位值**
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'

表 77: 常數值 (繼續)		
名稱	**小數值**	**十六進位值**
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_ * (指令格式「結構類型」)

表 78: 常數值		
名稱	**小數值**	**十六進位值**
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'
MQ 指令 _XR	16	X'00000010'
MQCFT_XR_MSG	17	X'00000011'
MQ CFT_XR_ITEM	18	X'00000012'
MQCFT_XR_SUMMARY	19	X'00000013'
MQCFT_GROUP	20	X'00000014'
MQCFT_STATISTICS	21	X'00000015'
MQ CFT_ACCOUNTING	22	X'00000016'
MQCFT_INTEGER64	23	X'00000017'
MQCFT_INTEGER64_LIST	25	X'00000019'

MQCFTYPE_ * (指令格式 CF 類型)

表 79: 常數值		
名稱	**小數值**	**十六進位值**
MQCFTYPE_APPL	0	X'00000000'
MQ 配置類型 _ 管理	1	X'00000001'

MQCFUNC_ * (CICS 資訊標頭「函數」)

表 *80:* 常數的結構	
名稱	**結構**
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	'¬','¬','¬','¬'

註: 符號 ¬ 代表單一空白字元。

MQCGWI_ * (CICS 資訊標頭「取得等待間隔」)

表 *81:* 常數值		
名稱	**小數值**	**十六進位值**
MQCGWI_DEFAULT	-2	X'FFFFFFFE'

MQCHAD_ * (通道自動定義)

表 *82:* 常數值		
名稱	**小數值**	**十六進位值**
MQCHAD_DISABLED	0	X'00000000'
已啟用 MQCHAD_ENABLED	1	X'00000001'

MQCHIDS_ * (指令格式「不確定狀態」)

表 *83:* 常數值		
名稱	**小數值**	**十六進位值**
MQCHIDS_NOT_INDOUBT	0	X'00000000'
MQCHIDS_INDOUBT	1	X'00000001'

MQCHLD_ * (指令格式「通道處置」)

表 *84:* 常數值		
名稱	**小數值**	**十六進位值**
MQCHLD_ALL	-1	X'FFFFFFFF'
MQCHLD_DEFAULT	1	X'00000001'

表 84: 常數值 (繼續)		
名稱	**小數值**	**十六進位值**
MQCHLD_SHARED	2	X'00000002'
MQCHLD_PRIVATE	4	X'00000004'
MQCHLD_FIXSHARED	5	X'00000005'

MQCHS_ * (指令格式「通道狀態」)

表 85: 常數值		
名稱	**小數值**	**十六進位值**
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_ * (指令格式「通道共用重新啟動選項」)

表 86: 常數值		
名稱	**小數值**	**十六進位值**
MQCHSH_RESTART_NO	0	X'00000000'
MQCHSH_RESTART_YES	1	X'00000001'

MQCHSR_ * (指令格式「通道停止選項」)

表 87: 常數值		
名稱	**小數值**	**十六進位值**
MQCHSR_STOP_NOT_REQUESTED	0	X'00000000'
已要求 MQCHSR_STOP_REQUESTED	1	X'00000001'

MQCHSSTATE_ * (指令格式「通道子狀態」)

表 88: 常數值		
名稱	**小數值**	**十六進位值**
MQCHSSTATE_OTHER	0	X'00000000'
MQCHSSTATE_END_OF_BATCH	100	X'00000064'
MQ 變更狀態 _ 傳送中	200	X'000000C8'
MQCHSSTATE_RECEIVING	300	X'0000012C'
MQCHSSTATE_SERIALIZING	400	X'00000190'
MQCHSSTATE_RESYN 程	500	X'000001F4'

表 *88:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
MQ 變更狀態 _ 活動訊號	600	X'00000258'
MQCHSSTATE_IN_SCYEXIT	700	X'000002BC'
MQCHSSTATE_IN_RCVEXIT	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'
MQ 變更狀態 _ SSL _ 信號交換	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_ * (通道類型)

表 *89:* 常數值		
名稱	**小數值**	**十六進位值**
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_ * (指令格式通道表格類型)

表 *90:* 常數值		
名稱	**小數值**	**十六進位值**
MQCHTAB_Q_MGR	1	X'00000001'
MQCHTAB_CLNTCONN	2	X'00000002'

MQCI_ * (相關性 ID)

表 *91:* 常數名稱和值	
名稱	**值**
MQCI_NONE	X'00...00' (24 個空值)
MQ 配置項目 _ 非 _ 陣列	'\0','\0',... (24 個空值)
MQCI_NEW_SESSION	X'414D5121...'

表 *91:* 常數名稱和值 (繼續)	
名稱	**值**
MQCI_NEW_SESSION_ARRAY	'\x41','\x4D','\51','\x21',...

MQCIH_ * (CICS 資訊標頭結構及旗標)

CICS 資訊標頭結構

表 *92:* 常數的結構	
名稱	**結構**
MQCIH_STRUC_ID	"CIH¬"
MQCIH_STRUC_ID_ARRAY	'C','I','H','¬'

註: 符號 ¬ 代表單一空白字元。

表 *93:* 常數值		
名稱	**小數值**	**十六進位值**
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 資訊標頭旗標

表 *94:* 常數值		
名稱	**小數值**	**十六進位值**
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_XX_ENCODE_CASE_ONE return	4	X'00000004'
MQCIH_NO_SYNC_ON_XX_ENCODE_CASE_ONE return	0	X'00000000'

MQCLCT_ * (叢集快取類型)

表 *95:* 常數值		
名稱	**小數值**	**十六進位值**
MQCLCT_STATIC	0	X'00000000'
MQCLCT_DYNAMIC	1	X'00000001'

MQCLRS_ * (指令格式「清除主題字串範圍」)

表 *96:* 常數值		
名稱	**小數值**	**十六進位值**
本端 MQCLRS_LOCAL	1	X'00000001'
MQCLRS_GLOBAL	2	X'00000002'

MQCLRT_ * (指令格式「清除主題字串類型」)

表 *97:* 常數值		
名稱	**小數值**	**十六進位值**
MQCLRT_RETAINED	1	X'00000001'

MQCLT_ * (CICS 資訊標頭「鏈結類型」)

表 *98:* 常數值		
名稱	**小數值**	**十六進位值**
MQCLT_PROGRAM	1	X'00000001'
MQCLT_TRANSACTION	2	X'00000002'

MQCLWL_ * (叢集工作量)

表 *99:* 常數值		
名稱	**小數值**	**十六進位值**
MQCLWL_USEQ_LOCAL	0	X'00000000'
MQCLWL_USEQ_ANY	1	X'00000001'
MQCLWL_USEQ_AS_Q_MGR	-3	X'FFFFFFFD'

MQCLXQ_ * (叢集傳輸佇列類型)

MQCLXQ_* 是您可以在 DEFCLXQ 佇列管理程式屬性中設定的值。 DEFCLXQ 屬性 可控制叢集傳送端通道預設會選取要從中取得訊息，以將訊息傳送至叢集接收端通道的傳輸佇列。

表 *100:* 常數值		
名稱	**小數值**	**十六進位值**
MQCLXQ_SCTQ	0	X'00000000'
MQCLXQ_CHANNEL	1	X'00000001'

相關參考

第 715 頁的『DefClusterXmitQueue 類型 (MQLONG)』
DefClusterXmitQueue **類型** 屬性 可控制叢集傳送端通道預設會選取要從中取得訊息，以將訊息傳送至叢集接收端通道的傳輸佇列。

變更佇列管理程式

查詢佇列管理程式

查詢佇列管理程式 (回應)

第 618 頁的『MQINQ-查詢物件屬性』
MQINQ 呼叫會傳回整數陣列及一組包含物件屬性的字串。

MQCMD_ * (指令碼)

表 *101:* 常數值

名稱	小數值	十六進位值
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'
MQCMD_INQUIRE_NAMELIST	36	X'00000024'
MQCMD_INQUIRE_NAMELIST_NAMES	37	X'00000025'
MQCMD_ESCAPE	38	X'00000026'
MQCMD_RESOLVE_CHANNEL	39	X'00000027'

表 *101:* 常數值 *(*繼續*)*

名稱	小數值	十六進位值
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'
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'

表 *101:* 常數值 *(*繼續*)*

名稱	小數值	十六進位值
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_BSDS	128	X'00000080'
MQCMD_RECOVER_CF_STRUC	129	X'00000081'
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'

表 101: 常數值 (繼續)

名稱	小數值	十六進位值
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'
MQCMD_COPY_TOPIC	171	X'000000AB'
MQCMD_CREATE_Topic	172	X'000000AC'
MQCMD_DELETE_TOPIC	173	X'000000AD'
MQCMD_INQUIRE_Topic	174	X'000000AE'

表 *101:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
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_ * (指令格式指令資訊值)

表 *102:* 常數值		
名稱	**小數值**	**十六進位值**
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'
已接受 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'
已完成 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_ * (指令層次)

表 *103:* 常數名稱和值	
名稱	**值**
MQCMDL_LEVEL_1	100
MQCMDL_LEVEL_101	101
MQCMDL_LEVEL_110	110

表 *103:* 常數名稱和值 (繼續)	
名稱	**值**
MQCMDL_LEVEL_114	114
MQCMDL_LEVEL_120	120
MQCMDL_LEVEL_200	200
MQCMDL_LEVEL_201	201
MQCMDL_LEVEL_210	210
MQCMDL_LEVEL_211	211
MQCMDL_LEVEL_220	220
MQCMDL_LEVEL_221	221
MQCMDL_LEVEL_230	230
MQCMDL_LEVEL_320	320
MQCMDL_LEVEL_420	420
MQCMDL_LEVEL_500	500
MQCMDL_LEVEL_510	510
MQCMDL_LEVEL_520	520
MQCMDL_LEVEL_530	530
MQCMDL_LEVEL_531	531
MQCMDL_LEVEL_600	600
MQCMDL_LEVEL_700	700
MQCMDL_LEVEL_701	701
MQCMDL_LEVEL_710	710
MQCMDL_LEVEL_711	711
MQCMDL_LEVEL_750	750

MQCMHO_ * (建立訊息控點選項和結構)

建立訊息控點選項結構

表 *104:* 常數的結構	
名稱	**結構**
MQCMHO_STRUC_ID	"CMHO"
MQCMHO_STRUC_ID_ARRAY	'C','M','H','O'

註: 符號 ¬ 代表單一空白字元。

表 *105:* 常數值		
名稱	**小數值**	**十六進位值**
MQCMHO_VERSION_1	1	X'00000001'
MQCMHO_CURRENT_VERSION	1	X'00000001'

建立訊息控點選項

表 *106:* 常數值		
名稱	**小數值**	**十六進位值**
MQCMHO_DEFAULT_VALIDATION	0	X'00000000'
MQCMHO_NO_VALIDATION	1	X'00000001'
MQ 指令 _ 驗證	2	X'00000002'
MQCMHO_NONE	0	X'00000000'

MQCNO_ * (連接選項及結構)

連接選項結構

表 *107:* 常數的結構	
名稱	**結構**
MQCNO_STRUC_ID	"CNO¬"
MQCNO_STRUC_ID_ARRAY	'C','N','O','¬'

註: 符號 ¬ 代表單一空白字元。

表 *108:* 常數值		
名稱	**小數值**	**十六進位值**
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'

連接選項

表 *109:* 常數值		
名稱	**小數值**	**十六進位值**
MQCN_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'

表 *109:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
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_AS_DEF	0	X'00000000'
已停用 MQCNO_RECONNECT_DISABLED	33554432	X'02000000'
MQCNO_RECONNECT_Q_MGR	67108864	X'04000000'
MQCNO_ACTIVITY_TRACE_ENABLED	134217728	X'08000000'
已停用 MQCNO_ACTIVITY_TRACE_DISABLED	268435456	X'10000000'
MQCNO_NONE	0	X'00000000'

MQCO_ * (關閉選項)

表 *110:* 常數值		
名稱	**小數值**	**十六進位值**
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 資訊標頭輸出資料長度)

表 *111:* 常數值		
名稱	**小數值**	**十六進位值**
MQCODL_AS_INPUT	-1	X'FFFFFFFF'

MQCOMPRESS_ * (通道壓縮)

表 *112:* 常數值		
名稱	**小數值**	**十六進位值**
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'
MQCOMPRESS_SYSTEM	8	X'00000008'

表 *112:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
MQCOMPRESS_ANY	268435455	X'0FFFFFFF'

MQCONNID_ * (連線 ID)

表 *113:* 常數名稱和值	
名稱	**值**
MQCONNID_NONE	X'00...00' (24 個空值)
MQCONNID_NONE_ARRAY	'\0','\0',... (24 個空值)

MQCOPY_ * (內容複製選項)

表 *114:* 常數值		
名稱	**小數值**	**十六進位值**
MQCOPY_NONE	0	X'00000000'
MQCOPY_ALL	1	X'00000001'
MQ COPY_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_ * (叢集佇列類型)

表 *115:* 常數值		
名稱	**小數值**	**十六進位值**
MQCQT_LOCAL_Q	1	X'00000001'
MQCQT_ALIAS_Q	2	X'00000002'
MQCQT_REMOTE_Q	3	X'00000003'
MQCQT_Q_MGR_XX_ENCODE_CASE_ONE alias	4	X'00000004'

MQCRC_ * (CICS 資訊標頭回覆碼)

表 *116:* 常數值		
名稱	**小數值**	**十六進位值**
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'
MQ CRC_BRIDGE_TIMEOUT	8	X'00000008'
MQCRC_TRANSID_NOT_AVAILABLE	9	X'00000009'

MQCS_ * (MQCBC 常數消費者狀態)

表 *117:* 常數值		
名稱	**小數值**	**十六進位值**
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 資訊標頭「起始碼」)

表 *118:* 常數的結構	
名稱	**結構**
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	'¬','¬','¬','¬'

註: 符號 ¬ 代表單一空白字元。

MQCSP_ * (連線安全參數結構及鑑別類型)

連線安全參數結構

表 *119:* 常數的結構	
名稱	**結構**
MQCSP_STRUC_ID	"CSP¬"
MQCSP_STRUC_ID_ARRAY	'C','S','P','¬'

註: 符號 ¬ 代表單一空白字元。

表 *120:* 常數值		
名稱	**小數值**	**十六進位值**
MQCSP_VERSION_1	1	X'00000001'
MQCSP_CURRENT_VERSION	1	X'00000001'

連線安全參數鑑別類型

表 *121:* 常數值		
名稱	**小數值**	**十六進位值**
MQCSP_AUTH_NONE	0	X'00000000'
MQCSP_AUTH_USER_ID_AND_PWD	1	X'00000001'

MQCSRV_ * (指令伺服器選項)

表 *122:* 常數值		
名稱	**小數值**	**十六進位值**
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_ * (佇列管理程式連線標籤)

表 *123:* 常數名稱和值	
名稱	**值**
MQCT_NONE	X'00...00' (128 個空值)
MQCT_NONE_ARRAY	'\0','\0',... (128 個空值)

MQCTES_ * (CICS 資訊標頭作業結束狀態)

表 *124:* 常數值		
名稱	**小數值**	**十六進位值**
MQCTES_NOSYNC	0	X'00000000'
MQCTES_COMMIT	256	X'00000100'
MQCTES_BACKOUT	4352	X'00001100'
MQCTES_ENDTASK	65536	X'00010000'

MQCTLO_ * (MQCTL 選項結構及消費者控制選項)

MQCTL 選項結構

表 *125:* 常數的結構	
名稱	**結構**
MQCTLO_STRUC_ID	"CTLO"
MQCTLO_STRUC_ID_ARRAY	'C','T','L','O'

註: 符號 ¬ 代表單一空白字元。

表 *126:* 常數值		
名稱	**小數值**	**十六進位值**
MQCTLO_VERSION_1	1	X'00000001'
MQ CTLO_CURRENT_VERSION	1	X'00000001'

MQCTL 選項消費者控制選項

表 *127:* 常數值		
名稱	**小數值**	**十六進位值**
MQCTLO_NONE	0	X'00000000'
MQ CTLO_THREAD_AFFINITY	1	X'00000001'
MQCTLO_FAIL_IF_QUIESCING	8192	X'00002000'

MQCUOWC_ * (CICS 資訊標頭「工作單元」控制項)

表 *128:* 常數值		
名稱	**小數值**	**十六進位值**
僅 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_ * (通道結束程式參數結構)

表 *129:* 常數的結構	
名稱	**結構**
MQCXP_STRUC_ID	"CXP¬"
MQCXP_STRUC_ID_ARRAY	'C','X','P','¬'

註: 符號 ¬ 代表單一空白字元。

表 *130:* 常數值		
名稱	**小數值**	**十六進位值**
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_CURRENT_VERSION	8	X'00000008'

MQDC_ * (目的地類別)

表 *131:* 常數值		
名稱	**小數值**	**十六進位值**
MQDC_MANAGED	1	X'00000001'
MQDC_PROVIDED	2	X'00000002'

MQDCC_ * (轉換選項，以及遮罩和因素)

轉換選項

表 *132:* 常數值		
名稱	**小數值**	**十六進位值**
MQDCC_DEFAULT_CONVERSION	1	X'00000001'

表 *132:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
MQDCC_FILL_TARGET_BUFFER	2	X'00000002'
MQDCC_INT_DEFAULT_CONVERSION	4	X'00000004'
MQDCC_SOURCE_ENC_NATIVE	(value differs by platform or version)	
MQ DCC_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)	
MQDCC_TARGET_ENC_NORMAL	256	X'00000100'
MQDCC_TARGET_ENC_REVERSED	512	X'00000200'
未定義 MQDCC_TARGET_ENC_UNDEFINED	0	X'00000000'
MQDCC_NONE	0	X'00000000'

轉換選項遮罩及因素

表 *133:* 常數值		
名稱	**小數值**	**十六進位值**
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_ * (發佈/訂閱刪除選項)

表 *134:* 常數值		
名稱	**小數值**	**十六進位值**
MQDELO_NONE	0	X'00000000'
MQDELO_LOCAL	4	X'00000004'

MQDH_ * (配送標頭結構)

表 *135:* 常數的結構	
名稱	**結構**
MQDH_STRUC_ID	"DH¬¬"
MQDH_STRUC_ID_ARRAY	'D','H','¬','¬'

註: 符號 ¬ 代表單一空白字元。

表 *136:* 常數值		
名稱	**小數值**	**十六進位值**
MQDH_VERSION_1	1	X'00000001'
MQDH_CURRENT_VERSION	1	X'00000001'

MQDHF_ * (配送標頭旗標)

表 *137:* 常數值		
名稱	**小數值**	**十六進位值**
MQDHHF_NEW_MSG_IDS	1	X'00000001'
MQDHHF_NONE	0	X'00000000'

MQDISCONNECT_ * (指令格式「斷線類型」)

表 *138:* 常數值		
名稱	**小數值**	**十六進位值**
MQDISCONNECT_NORMAL	0	X'00000000'
MQDISCONNECT_IMPLICIT	1	X'00000001'
MQDISCONNECT_Q_MGR	2	X'00000002'

MQDL_ * (配送清單)

表 *139:* 常數值		
名稱	**小數值**	**十六進位值**
支援 MQDL_SUPPORTED	1	X'00000001'
不支援 MQDL_NOT_SUPPORTED	0	X'00000000'

MQDLH_ * (無法傳送郵件的標頭結構)

表 *140:* 常數的結構	
名稱	**結構**
MQDLH_STRUC_ID	"DLH¬"
MQDLH_STRUC_ID_ARRAY	'D','L','H','¬'

註: 符號 ¬ 代表單一空白字元。

表 *141:* 常數值		
名稱	**小數值**	**十六進位值**
MQDLH_VERSION_1	1	X'00000001'
MQDLH_CURRENT_VERSION	1	X'00000001'

MQDLV_ * (持續性/非持續性訊息遞送)

表 *142:* 常數值		
名稱	**小數值**	**十六進位值**
MQDLV_AS_PARENT	0	X'00000000'
MQDLV_ALL	1	X'00000001'
MQDLV_ALL_DUR	2	X'00000002'
MQDLV_ALL_AVAIL	3	X'00000003'

MQDMHO_ * (刪除訊息控點選項及結構)

刪除訊息控點選項結構

表 *143:* 常數的結構	
名稱	**結構**
MQDMHO_STRUC_ID	"DMHO"
MQDMHO_STRUC_ID_ARRAY	'D','M','H','O'

註: 符號 ¬ 代表單一空白字元。

表 *144:* 常數值		
名稱	**小數值**	**十六進位值**
MQDMHO_VERSION_1	1	X'00000001'
MQDMHO_CURRENT_VERSION	1	X'00000001'

刪除訊息控點選項

表 *145:* 常數值		
名稱	**小數值**	**十六進位值**
MQDMHO_NONE	0	X'00000000'

MQDMPO_ * (刪除訊息內容選項及結構)

刪除訊息內容選項結構

表 *146:* 常數的結構	
名稱	**結構**
MQDMPO_STRUC_ID	"DMPO"
MQDMPO_STRUC_ID_ARRAY	'D','M','P','O'

註: 符號 ¬ 代表單一空白字元。

表 *147:* 常數值		
名稱	**小數值**	**十六進位值**
MQDMPO_VERSION_1	1	X'00000001'
MQDMPO_CURRENT_VERSION	1	X'00000001'

刪除訊息內容選項

表 *148:* 常數值		
名稱	**小數值**	**十六進位值**
MQDMPO_DEL_FIRST	0	X'00000000'
MQDMPO_DEL_PROP_UNDER_CURSOR	1	X'00000001'
MQDMPO_NONE	0	X'00000000'

MQDNSWLM_ * (DNS WLM)

表 *149:* 常數值		
名稱	**小數值**	**十六進位值**
MQDNSWLM_NO	0	X'00000000'
MQDNSWLM_YES	1	X'00000001'

MQDT_ * (目的地類型)

表 *150:* 常數值		
名稱	**小數值**	**十六進位值**
MQDT_APPL	1	X'00000001'
MQDT_BROKER	2	X'00000002'

MQDXP_ * (轉換結束程式參數結構)

表 *151:* 常數的結構	
名稱	**結構**
MQDXP_STRUC_ID	"DXP¬"
MQDXP_STRUC_ID_ARRAY	'D','X','P','¬'

註: 符號 ¬ 代表單一空白字元。

表 *152:* 常數值		
名稱	**小數值**	**十六進位值**
MQDXP_VERSION_1	1	X'00000001'
MQDXP_VERSION_2	2	X'00000002'
MQDXP_CURRENT_VERSION	2	X'00000002'

MQEC_ * (信號值)

表 *153:* 常數值		
名稱	**小數值**	**十六進位值**
MQEC_MSG_REALED	2	X'00000002'
MQ ec_WAIT_INTERVAL_EXPIRED	3	X'00000003'
已取消 MQEC_WAIT_CANCELED	4	X'00000004'
MQEC_Q_MGR_QUIESCING	5	X'00000005'
MQEC_CONNECTION_QUIESCING	6	X'00000006'

品質管理 _ * (期限)

表 *154:* 常數值		
名稱	**小數值**	**十六進位值**
MQEI_UNLIMITED	-1	X'FFFFFFFF'

MQENC_ * (編碼)

MQENC_ * (編碼)

表 *155:* 依平台列出的常數值			
名稱	**平台**	**小數值**	**十六進位值**
MQENC_NATIVE	IBM i	273	X'00000111'
	Linux	546	X'00000222'
	SPARC 上的 Linux	273	X'00000111'
	Linux on x86	546	X'00000222'
	SPARC 上的 Solaris	273	X'00000111'
	UNIX 系統	273	X'00000111'
	Windows	546	X'00000222'
	Windows 上的 Micro Focus COBOL	17	X'00000011'
	z/OS	785	X'00000311'

MQENC_ * (編碼遮罩)

表 *156:* 常數值		
名稱	**小數值**	**十六進位值**
MQENC_INTEGER_MASK	15	X'0000000F'
MQENC_DECIMAL_MASK	240	X'000000F0'
MQENC_FLOAT_MASK	3840	X'00000F00'
MQ 保留遮罩	-4096	X'FFFFF000'

MQENC_ * (二進位整數的編碼)

表 *157:* 常數值		
名稱	**小數值**	**十六進位值**
未定義 MQENC_INTEGER_UNDEFINED	0	X'00000000'
MQENC_INTEGER_NORMAL	1	X'00000001'
已反轉 MQENC_INTEGER_REVERSED	2	X'00000002'

MQENC_ * (聚集十進位數整數的編碼)

表 *158:* 常數值		
名稱	**小數值**	**十六進位值**
未定義 MQENC_DECIMAL_UNDEFINED	0	X'00000000'
MQENC_DECIMAL_NORMAL	16	X'00000010'
MQENC_DECIMAL_REVERED	32	X'00000020'

MQENC_ * (浮點數字的編碼)

表 *159:* 常數值		
名稱	**小數值**	**十六進位值**
未定義 MQENC_FLOAT_UNDEFINED	0	X'00000000'

表 159: 常數值 (繼續)		
名稱	**小數值**	**十六進位值**
MQENC_FLOAT_IEE_NORMAL	256	X'00000100'
MQENC_FLOAT_IEE_REVERSED	512	X'00000200'
MQENC_FLOAT_S390	768	X'00000300'
MQENC_FLOAT_TNS	1024	X'00000400'

MQEPH_ * (內嵌指令格式標頭結構和旗標)

內嵌指令格式標頭結構

表 160: 常數的結構	
名稱	**結構**
MQEPH_STRUC_ID	"EPH¬"
MQEPH_STRUC_ID_ARRAY	'E','P','H','¬'

註: 符號 ¬ 代表單一空白字元。

表 161: 常數值		
名稱	**小數值**	**十六進位值**
MQEPH_STRUC_LENGTH_FIXED	68	X'00000044'
MQEPH_VERSION_1	1	X'00000001'
MQEPH_CURRENT_VERSION	1	X'00000001'

內嵌指令格式標頭旗標

表 162: 常數值		
名稱	**小數值**	**十六進位值**
MQEPH_NONE	0	X'00000000'
MQEPH_CCSID_EMBEDDED	1	X'00000001'

MQET_ * (指令格式 Escape Types)

表 163: 常數值		
名稱	**小數值**	**十六進位值**
MQET_MQSC	1	X'00000001'

LQEVO_ * (指令格式事件來源)

表 164: 常數值		
名稱	**小數值**	**十六進位值**
AQEVO_OTHER	0	X'00000000'
MQEVO_CONSOLE	1	X'00000001'
MCEVO_INIT	2	X'00000002'
MQEVO_MSG	3	X'00000003'
MQEVO_MQSET	4	X'00000004'
MCEVO_INTERNAL	5	X'00000005'

MQEVR_ * (指令格式「事件記錄」)

表 *165:* 常數值		
名稱	**小數值**	**十六進位值**
已停用 MQEVR_DISABLED	0	X'00000000'
已啟用 MQEVR_ENABLED	1	X'00000001'
MQ 事件 _ 異常狀況	2	X'00000002'
MQEVR_NO_DISPLAY	3	X'00000003'

MQEXPI_ * (到期掃描間隔)

表 *166:* 常數值		
名稱	**小數值**	**十六進位值**
MQEXPI_OFF	0	X'00000000'

MQFB_ * (回饋值)

表 *167:* 常數值		
名稱	**小數值**	**十六進位值**
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'
已啟動 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'
MQ fb_STOPPED_BY_CHA_EXIT	277	X'00000115'
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'
MQ fb_not_delivered	284	X'0000011C'
MQFB_UNSUPPORTED_FORWARDING	285	X'0000011D'

表 *167:* 常數值 *(*繼續*)*

名稱	小數值	十六進位值
MQ fb_UNSUPPORTED_Delivery	286	X'0000011E'
MQFB_DATA_LENGTH_ZERO	291	X'00000123'
MQFB_DATA_LENGTH_NEGATIVE	292	X'00000124'
MQ fb_data_length_too_big	293	X'00000125'
MQ fb_BUFFER_溢位	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_CCSID_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_XX_ENCODE_CASE_ONE 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'
MQ fb_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'
MQFB_APPL_LAST	999999999	X'3B9AC9FF'

MQFC_ * (指令格式「強制選項」)

表 *168:* 常數值

名稱	小數值	十六進位值
MQFC_YES	1	X'00000001'
MQFC_NO	0	X'00000000'

MQFMT_ * (格式)

表 *169:* 常數名稱和值

名稱	值
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','¬','¬'
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','¬','¬','¬'

表 *169:* 常數名稱和值 (繼續)	
名稱	**值**
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','¬','¬'

註: 符號 ¬ 代表單一空白字元。

MQGA_ * (群組屬性選取器)

表 *170:* 常數值		
名稱	**小數值**	**十六進位值**
MQGA_FIRST	8001	X'00001F41'
MQGA_LAST	9000	X'00002328'

MQGACF_ * (指令格式群組參數類型)

表 *171:* 常數值		
名稱	**小數值**	**十六進位值**
MQGACF_FIRST	8001	X'00001F41'
MQGACF_COMMAND_CONTEXT	8001	X'00001F41'
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'
已使用 MQGACF_LAST_USED	8012	X'00001F4C'

MQGI_ * (群組 ID)

表 *172:* 常數名稱和值	
名稱	**值**
MQGI_NONE	X'00...00' (24 個空值)
MQGI_NONE_ARRAY	'\0','\0',... (24 個空值)

MQGMO_ * (取得訊息選項及結構)

取得訊息選項結構

表 *173:* 常數的結構	
名稱	**結構**
MQGMO_STRUC_ID	"GMO¬"
MQGMO_STRUC_ID_ARRAY	'G','M','O','¬'

註: 符號 ¬ 代表單一空白字元。

表 *174:* 常數值		
名稱	**小數值**	**十六進位值**
MQGMO_VERSION_1	1	X'00000001'
MQGMO_VERSION_2	2	X'00000002'
MQGMO_VERSION_3	3	X'00000003'
MQGMO_VERSION_4	4	X'00000004'
MQ GMO_CURRENT_VERSION	4	X'00000004'

取得訊息選項

表 *175:* 常數值		
名稱	**小數值**	**十六進位值**
MQGMO_WAIT	1	X'00000001'
MQGMO_NO_WAIT	0	X'00000000'
MQGMO_SET_SIGNAL	8	X'00000008'
MQGMO_FAIL_IF_QUIESCING	8192	X'00002000'
MQ GMO_同步點	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'00000800'
MQ GMO_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'
MQ GMO_CONVERT	16384	X'00004000'
MQ GMO_LOGICAL_ORDER	32768	X'00008000'
MQ GMO_COMPLETE_MSG	65536	X'00010000'
MQGMO_ALL_MSGS_AVAILABLE	131072	X'00020000'
MQGMO_ALL_SEGMENTS_AVAILABLE	262144	X'00040000'

表 *175:* 常數值 *(繼續)*		
名稱	**小數值**	**十六進位值**
MQ GMO_MARK_BROWSE_HANDLE	1048576	X'00100000'
MQGMO_MARK_BROWSE_CO_OP	2097152	X'00200000'
MQGMO_UNMARK_BROWSE_CO_OP	4194304	X'00400000'
MQ GMO_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_ * (群組狀態)

表 *176:* 常數名稱和值	
名稱	**值**
MQGS_NOT_IN_GROUP	'¬'
MQGS_MSG_IN_GROUP	'G'
MQGS_LAST_MSG_IN_GROUP	'L'

註: 符號 ¬ 代表單一空白字元。

MQHA_ * (控點選取器)

表 *177:* 常數值		
名稱	**小數值**	**十六進位值**
第一個 MQHA_FIRST	4001	X'00000FA1'
MQHA_BAG_HANDLE	4001	X'00000FA1'
已使用 MQHA_LAST_UTED	4001	X'00000FA1'
MQHA_LAST	6000	X'00001770'

MQHB_ * (工具袋控點)

表 *178:* 常數值		
名稱	**小數值**	**十六進位值**
MQHB_UNUSABLE_HBAG	-1	X'FFFFFFFF'
MQHB_NONE	-2	X'FFFFFFFE'

MQHC_ * (連線控點)

表 *179:* 常數值		
名稱	**小數值**	**十六進位值**
MQHC_DEF_HCONN	0	X'00000000'
MQHC_UNUSABLE_HCONN	-1	X'FFFFFFFF'

表 179: 常數值 (繼續)		
名稱	**小數值**	**十六進位值**
MQHC_UNASSOCITED_HCONN	-3	X'FFFFFFFD'

MQHM_ * (訊息控點)

表 180: 常數值		
名稱	**小數值**	**十六進位值**
MQHM_UNUSABLE_HMSG	-1	X'FFFFFFFF'
MQHM_NONE	0	X'00000000'

MQHO_ * (物件控點)

表 181: 常數值		
名稱	**小數值**	**十六進位值**
MQHO_UNUSABLE_HOBJ	-1	X'FFFFFFFF'
MQHO_NONE	0	X'00000000'

MQHSTATE_ * (指令格式 Handle States)

表 182: 常數值		
名稱	**小數值**	**十六進位值**
MQHSTATE_INACTIVE	0	X'00000000'
MQHSTATE_ACTIVE	1	X'00000001'

MQIA_ * (整數屬性選取器)

表 183: 常數值		
名稱	**小數值**	**十六進位值**
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_TYPE	103	X'00000067'
MQIA_ADOPTNEWMCA_INTERVAL	104	X'00000068'
MQIA_APPL_TYPE	1	X'00000001'
MQIA_ARCHIVE	60	X'0000003C'
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'

表 183: 常數值 (繼續)

名稱	小數值	十六進位值
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_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'
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'

表 183: 常數值 (繼續)

名稱	小數值	十六進位值
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_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'
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_LAST	2000	X'000007D0'
MQIA_LAST_USED	233	X'000000E9'
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'

表 183: 常數值 (繼續)

名稱	小數值	十六進位值
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'
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_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_QMOPT_CONS_COMMS_MSGS	155	X'0000009B'
MQIA_QMOPT_CONS_CRITICAL_MSGS	154	X'0000009A'

表 183: 常數值 (繼續)

名稱	小數值	十六進位值
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'
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_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'

表 183: 常數值 (繼續)

名稱	小數值	十六進位值
MQIA_STATISTICS_INTERVAL	131	X'00000083'
MQIA_STATISTICS_MQI	127	X'0000007F'
MQIA_STATISTICS_Q	128	X'00000080'
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_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_* (指令格式「整數參數類型」)

表 184: 常數值

名稱	小數值	十六進位值
第一個 MQIACF_FIRST	1001	X'000003E9'
MQ IACF_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'

表 *184:* 常數值 *(*繼續*)*

名稱	小數值	十六進位值
MQIACF_ALL	1009	X'000003F1'
MQIACF_EVENT_APPL_TYPE	1010	X'000003F2'
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'
MQ IACF_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'

表 184: 常數值 (繼續)

名稱	小數值	十六進位值
MQIACF_BROKER_COUNT	1088	X'00000440'
MQIACF_APPL_COUNT	1089	X'00000441'
MQIACF_ANONYMOUS_COUNT	1090	X'00000442'
MQIACF_REG_REG_OPTIONS	1091	X'00000443'
MQIACF_DELETE_OPTIONS	1092	X'00000444'
MQ IACF_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'

表 *184:* 常數值 *(*繼續*)*

名稱	小數值	十六進位值
MQIACF_UOW_TYPE	1132	X'0000046C'
MQIACF_CF_STRUC_ATTRS	1133	X'0000046D'
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'

表 *184:* 常數值 *(*繼續*)*

名稱	小數值	十六進位值
MQIACF_MOVE_COUNT	1171	X'00000493'
MQIACF_EXPIRY_Q_COUNT	1172	X'00000494'
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'
MQ IACF_SYSP_MAX_ARCHIVE	1177	X'00000499'
MQIACF_SYSP_MAX_READ_TAPES	1178	X'0000049A'
MQ IACF_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'
MQ IACF_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'
MQ IACF_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'

表 *184:* 常數值 *(*繼續*)*

名稱	小數值	十六進位值
MQIACF_SYSP_ALLOC_SECONDARY	1210	X'000004BA'
MQIACF_SYSP_PROTECT	1211	X'000004BB'
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'
MQ IACF_SYSP_TOTAL_LOGS	1220	X'000004C4'
MQIACF_SYSP_FULL_LOGS	1221	X'000004C5'
MQIACF_LISTENER_ATTRS	1222	X'000004C6'
MQ IACF_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'
MQ IACF_Q_MGR_STATUS_ATTRS	1229	X'000004CD'
MQIACF_CONNECTION_COUNT	1230	X'000004CE'
MQIACF_Q_MGR_XX_ENCODE_CASE_ONE facility	1231	X'000004CF'
MQIACF_CHINIT_STATUS	1232	X'000004D0'
MQIACF_CMD_SERVER_STATUS	1233	X'000004D1'
MQIACF_ROUTE_DETAIL	1234	X'000004D2'
MQIACF_RECORD_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_ 回饋	1245	X'000004DD'
MQIACF_MSG_FLAGS	1247	X'000004DF'
MQIACF_MSG_LENGTH	1248	X'000004E0'
MQIACF_MSG_TYPE	1249	X'000004E1'

表 184: 常數值 (繼續)

名稱	小數值	十六進位值
MQIACF_OFFSET	1250	X'000004E2'
MQIACF_ORIGINAL_LENGTH	1251	X'000004E3'
MQIACF_持續性	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_UNRECORD_ACTIVITIES	1257	X'000004E9'
MQIACF_MONITORING	1258	X'000004EA'
MQIACF_ROUTE_FORWARDING	1259	X'000004EB'
MQ IACF_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 優先順序	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'

表 *184:* 常數值 *(*繼續*)*

名稱	小數值	十六進位值
MQIACF_TOPIC_STATUS_TYPE	1302	X'00000516'
MQIACF_SUB_OPTIONS	1303	X'00000517'
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'
已使用 MQIACF_LAST_USED	1351	X'00000547'

MQIACH_ * (指令格式「整數通道類型」)

表 *185:* 常數值

名稱	小數值	十六進位值
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'
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'

表 185: 常數值 (繼續)

名稱	小數值	十六進位值
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'
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'

表 185: 常數值 (繼續)

名稱	小數值	十六進位值
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'
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'

表 185: 常數值 (繼續)

名稱	小數值	十六進位值
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_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'
MQIACH_CHANNEL_SUMMARY_ATTRS	1642	X'0000066A'
MQIACH_LAST_USED	1642	X'0000066A'

MQIAMO_ * (指令格式「整數監視參數類型」)

表 186: 常數值

名稱	小數值	十六進位值
第一個 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'

表 *186:* 常數值 *(*繼續*)*

名稱	小數值	十六進位值
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_DIS 碟	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'
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_PUT	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'

表 186: 常數值 (繼續)

名稱	小數值	十六進位值
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_PLEARDED	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_HIGHW 水	775	X'00000307'
MQIAMO_SUB_DUR_LOWW 水	776	X'00000308'
MQIAMO_SUB_NDUR_HIGHW 水	777	X'00000309'
MQIAMO_SUB_NDUR_LOWW 水	778	X'0000030A'
MQIAMO_TOPIC_PUT	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'
已傳送 MQIAMO_MSGS_SENT	790	X'00000316'
已傳送 MQIAMO_BYTES_SENT	791	X'00000317'
MQIAMO_REPAIR_BYTES	792	X'00000318'
MQIAMO_FEEDBACK_MODE	793	X'00000319'
MQIAMO_RELIABILITY_TYPE	794	X'0000031A'

表 *186:* 常數值 *(*繼續*)*

名稱	小數值	十六進位值
MQIAMO_LATE_JOIN_MARK	795	X'0000031B'
MQIAMO_NACKS_RCVD	796	X'0000031C'
MQIAMO_REPAIR_PKTS	797	X'0000031D'
MQIAMO_HSTORY_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'
已傳送 MQIAMO_PKTS_SENT	808	X'00000328'
MQIAMO_TOTAL_REPAIR_PKTS	809	X'00000329'
已傳送 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'
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'
已傳送 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'

表 *186:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
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_* (指令格式為 64 位元整數監視參數類型)

表 *187:* 常數值		
名稱	**小數值**	**十六進位值**
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_ * (整數系統選取器)

表 *188:* 常數值		
名稱	**小數值**	**十六進位值**
第 MQIASY_FIRST	-1	X'FFFFFFFF'
MQIASY_CODED_CHAR_SET_ID	-1	X'FFFFFFFF'
MQIASY_TYPE	-2	X'FFFFFFFE'
MQIASY_COMMAND	-3	X'FFFFFFFD'
MQIASY_MSG_SEQ_NUMBER	-4	X'FFFFFFFC'
MQIASY_CONTROL	-5	X'FFFFFFFB'
MQIASY_COMP_CODE	-6	X'FFFFFFFA'
MQIASY_REASON	-7	X'FFFFFFF9'
MQIASY_bAG_OPTIONS	-8	X'FFFFFFF8'
MQIASY_VERSION	-9	X'FFFFFFF7'
MQIASY_LAST_USED	-9	X'FFFFFFF7'
MQIASY_LAST	-2000	X'FFFFF830'

MQIAUT_ * (IMS 資訊標頭 Authenticator)

表 *189:* 常數名稱和值	
名稱	**值**
無 MQIAUT_NONE	"¬¬¬¬¬¬¬¬"
MQIAUT_NONE_ARRAY	'¬','¬','¬','¬','¬','¬','¬','¬'

註: 符號 ¬ 代表單一空白字元。

MQIIV_ * (整數屬性值)

表 *190:* 常數值		
名稱	**小數值**	**十六進位值**
MQIAV_NOT_APPLICABLE	-1	X'FFFFFFFF'
未定義 MQIAV_UNDEFINED	-2	X'FFFFFFFE'

MQICM_ * (IMS 資訊標頭「確定模式」)

表 *191:* 常數名稱和值	
名稱	**值**
MQICM_COMMIT_THEN_SEND	'0'
MQICM_SEND_THEN_COMMIT	'1'

MQIDO_ * (指令格式「不確定選項」)

表 *192:* 常數值		
名稱	**小數值**	**十六進位值**
MQIDO_COMMIT	1	X'00000001'
MQIDO_BACKOUT	2	X'00000002'

MQIEP_ * (介面進入點)

連線安全參數結構

表 *193:* 常數的結構	
名稱	**結構**
MQIEP_STRUC_ID	"IEP¬"
MQIEP_STRUC_ID_ARRAY	'I','E','P','¬'

註: 符號 ¬ 代表單一空白字元。

表 *194:* 常數值		
名稱	**小數值**	**十六進位值**
MQIEP_VERSION_1	1	X'00000001'
MQDXP_CURRENT_VERSION	1	X'00000001'

MQIGQ_ * (內部群組佇列作業)

表 *195:* 常數值		
名稱	**小數值**	**十六進位值**
已停用 MQIGQ_DISABLED	0	X'00000000'
已啟用 MQIGQ_ENABLED	1	X'00000001'

MQIGQPA_ * (內部群組佇列作業放置權限)

表 *196:* 常數值		
名稱	**小數值**	**十六進位值**
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 資訊標頭結構和旗標)

IMS 資訊標頭結構

表 *197:* 常數的結構	
名稱	**結構**
MQIIH_STRUC_ID	"IIH¬"
MQIH_STRUC_ID_ARRAY	'I','I','H','¬'

註: 符號 ¬ 代表單一空白字元。

表 *198:* 常數值		
名稱	**小數值**	**十六進位值**
MQIIH_VERSION_1	1	X'00000001'
MQIIH_CURRENT_VERSION	1	X'00000001'
MQIIH_LENGTH_1	84	X'00000054'

IMS 資訊標頭旗標

表 *199:* 常數值		
名稱	**小數值**	**十六進位值**
MQIIH_NONE	0	X'00000000'
MQIH_PASS_EXPIRATION	1	X'00000001'
MQIH_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_ * (查詢訊息內容選項及結構)

查詢訊息內容選項結構

表 *200:* 常數的結構	
名稱	**結構**
MQIMPO_STRUC_ID	"IMPO"
MQIMPO_STRUC_ID_ARRAY	'I','M','P','O'

註: 符號 ¬ 代表單一空白字元。

表 *201:* 常數值		
名稱	**小數值**	**十六進位值**
MQIMPO_VERSION_1	1	X'00000001'
MQ IMPO_CURRENT_VERSION	1	X'00000001'

查詢訊息內容選項

表 *202:* 常數值		
名稱	**小數值**	**十六進位值**
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_ * (指令格式「入埠處置」)

表 *203:* 常數值		
名稱	**小數值**	**十六進位值**
MQINBD_Q_MGR	0	X'00000000'
MQINBD_GROUP	3	X'00000003'

MQIND_ * (特殊索引值)

表 *204:* 常數值		
名稱	**小數值**	**十六進位值**
無 MQIND_NONE	-1	X'FFFFFFFF'
MQIND_ALL	-2	X'FFFFFFFE'

MQIPADDR_ * (IP 位址版本)

表 *205:* 常數值		
名稱	**小數值**	**十六進位值**
MQIPADDR_IPV4	0	X'00000000'
MQIPADDR_IPV6	1	X'00000001'

MQISS_ * (IMS 資訊標頭安全範圍)

表 *206:* 常數名稱和值	
名稱	**值**
MQISS_CHECK	'C'
MQISS_FULL	'F'

MQIT_ * (索引類型)

表 *207:* 常數值		
名稱	**小數值**	**十六進位值**
無 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_ * (mqInquireItemInfo 的項目類型)

表 *208:* 常數值		
名稱	**小數值**	**十六進位值**
MQITEM_INTEGER	1	X'00000001'
MQITEM_STRING	2	X'00000002'
MQ 項目 _ 工具袋	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 資訊標頭「交易實例 ID」)

表 *209:* 常數名稱和值	
名稱	**值**
MQITII_NONE	X'00...00' (16 個空值)
MQITII_NONE_ARRAY	'\0','\0',... (16 個空值)

MQITS_ * (IMS 資訊標頭「交易狀態」)

表 *210:* 常數名稱和值	
名稱	**值**
MQITS_IN_CONVERSATION	'C'
MQITS_NOT_IN_CONVERSATION	'¬'
MQITS_ARCHITECTED	'A'

註: 符號 ¬ 代表單一空白字元。

MQKAI_ * (KeepAlive 間隔)

表 *211:* 常數值		
名稱	**小數值**	**十六進位值**
MQ 開 _ 自動	-1	X'FFFFFFFF'

MQMASTER_ * (主要管理)

表 *212:* 常數值		
名稱	**小數值**	**十六進位值**
MQMASTER_NO	0	X'00000000'
MQMASTER_YES	1	X'00000001'

MQMCAS_ * (指令格式「訊息通道代理程式狀態」)

表 *213:* 常數值		
名稱	**小數值**	**十六進位值**
已停止 MQMCAS_STOPPED	0	X'00000000'
MQMCAS_RUNNING	3	X'00000003'

MQMCAT_ * (MCA 類型)

表 *214:* 常數值		
名稱	**小數值**	**十六進位值**
MQMCAT_PROCESS	1	X'00000001'
MQMCAT_THREAD	2	X'00000002'

MQMCD_ * (發佈/訂閱選項標籤資訊)

發佈/訂閱選項標籤訊息內容描述子 (mcd) 標籤

表 *215:* 常數值		
名稱	**小數值**	**十六進位值**
MQMCD_FOLDER_VERSION	1	X'00000001'

發佈/訂閱選項標籤標籤名稱

表 *216:* 常數名稱和值	
名稱	**值**
MQMCD_MSG_DOMAIN	"Msd"
MQMCD_MSG_SET	"Set"
MQMCD_MSG_TYPE	"Type"
MQMCD_MSG_FORMAT	"Fmt"

發佈/訂閱選項標籤 XML 標籤名稱

表 *217:* 常數名稱和值	
名稱	**值**
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>"

發佈/訂閱選項標籤標籤值

表 *218:* 常數名稱和值	
名稱	**值**
MQMCT_DOMAN_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_ * (訊息描述子結構)

表 *219:* 常數的結構	
名稱	**結構**
MQMD_STRUC_ID	"MD¬¬"
MQMD_STRUC_ID_ARRAY	'M','D','¬','¬'

註: 符號 ¬ 代表單一空白字元。

表 *220:* 常數值		
名稱	**小數值**	**十六進位值**
MQMD_VERSION_1	1	X'00000001'
MQMD_VERSION_2	2	X'00000002'
MQMD_CURRENT_VERSION	2	X'00000002'

MQMDE_ * (訊息描述子延伸結構)

表 *221:* 常數的結構	
名稱	**結構**
MQMDE_STRUC_ID	"MDE¬"
MQMDE_STRUC_ID_ARRAY	'M','D','E','¬'

註: 符號 ¬ 代表單一空白字元。

表 *222:* 常數值		
名稱	**小數值**	**十六進位值**
MQMDE_VERSION_2	2	X'00000002'
MQMDE_CURRENT_VERSION	2	X'00000002'
MQMDE_LENGTH_2	72	X'00000048'

MQMDEF_ * (訊息描述子延伸旗標)

表 *223:* 常數值		
名稱	**小數值**	**十六進位值**
MQMDEF_NONE	0	X'00000000'

MQMDS_ * (訊息遞送順序)

表 *224:* 常數值		
名稱	**小數值**	**十六進位值**
MQMDS_PRIORITY	0	X'00000000'
MQMDS_FIFO	1	X'00000001'

MQMF_ * (訊息旗標)

表 *225:* 常數值		
名稱	**小數值**	**十六進位值**
MQMF_SEGMENTATION_INHIBITED	0	X'00000000'
容許 MQMF_SEGMENTATION_ALLOWED	1	X'00000001'
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'
無 MQMF_NONE	0	X'00000000'

MQMHBO_ * (緩衝區選項及結構的訊息控點)

訊息控點至緩衝區選項結構

表 *226:* 常數的結構	
名稱	**結構**
MQMHBO_STRUC_ID	"MHBO"
MQMHBO_STRUC_ID_ARRAY	'M','H','B','O'

註: 符號 ¬ 代表單一空白字元。

表 *227:* 常數值

名稱	小數值	十六進位值
MQMHBO_VERSION_1	1	X'00000001'
MQMHBO_CURRENT_VERSION	1	X'00000001'

訊息控點至緩衝區選項

表 *228:* 常數值

名稱	小數值	十六進位值
MQMHBO_PROPERTIES_IN_MQRFH2	1	X'00000001'
MQMHBO_DELETE_PROPERTIES	2	X'00000002'
MQMHBO_NONE	0	X'00000000'

MQMI_ * (訊息 ID)

表 *229:* 常數名稱和值

名稱	值
MQMI_NONE	X'00...00' (24 個空值)
MQMI_NONE_ARRAY	'\0','\0',... (24 個空值)

MQMMBI_ * (訊息標示-瀏覽間隔)

表 *230:* 常數值

名稱	小數值	十六進位值
MQMMBI_UNLIMITED	-1	X'FFFFFFFF'

MQMO_ * (符合選項)

表 *231:* 常數值

名稱	小數值	十六進位值
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_ * (指令格式模式選項)

表 *232:* 常數值

名稱	小數值	十六進位值
MQMODE_FORCE	0	X'00000000'
MQMODE_QUIESCE	1	X'00000001'
MQMODE_TERMINATE	2	X'00000002'

MQMON_ * (監視值)

表 *233:* 常數值

名稱	小數值	十六進位值
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'
已停用 MQMON_DISABLED	0	X'00000000'
已啟用 MQMON_ENABLED	1	X'00000001'
MQMON_LOW	17	X'00000011'
MQ mon_MEDIT	33	X'00000021'
MQMON_HIGH	65	X'00000041'

MQMT_ * (訊息類型)

表 *234:* 常數值

名稱	小數值	十六進位值
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'
MQMT_MQE_FIELDS	113	X'00000071'
MQMT_SYSTEM_LAST	65535	X'0000FFFF'
MQMT_APPL_FIRST	65536	X'00010000'
MQMT_APPL_LAST	999999999	X'3B9AC9FF'

MQMTOK_ * (訊息記號)

表 *235:* 常數名稱和值

名稱	值
MQMTOK_NONE	X'00...00' (16 個空值)
MQMTOK_NONE_ARRAY	'\0','\0',... (16 個空值)

MQNC_ * (名稱計數)

表 *236:* 常數值

名稱	小數值	十六進位值
MQNC_MAX_NAMELIST_NAME_COUNT	256	X'00000100'

MQNPM_ * (非持續性訊息類別)

表 *237:* 常數值		
名稱	**小數值**	**十六進位值**
MQNPM_CLASS_NORMAL	0	X'00000000'
MQNPM_CLASS_HIGH	10	X'0000000A'

MQNPMS_ * (NonPersistent-訊息速度)

表 *238:* 常數值		
名稱	**小數值**	**十六進位值**
MQNPMS_NORMAL	1	X'00000001'
MQNPMS_FAST	2	X'00000002'

MQNT_ * (名單類型)

表 *239:* 常數值		
名稱	**小數值**	**十六進位值**
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_ * (名稱/值字串的名稱)

表 *240:* 常數名稱和值	
名稱	**值**
MQNVS_APPL_TYPE	"OPT_APP_GRP¬"
MQNVS_MSG_TYPE	"OPT_MSG_TYPE¬"

註: 符號 ¬ 代表單一空白字元。

MQOA_ * (物件屬性的選取器限制)

表 *241:* 常數值		
名稱	**小數值**	**十六進位值**
MQOA_FIRST	1	X'00000001'
MQOA_LAST	9000	X'00002328'

MQOD_ * (物件描述子結構)

表 *242:* 常數的結構	
名稱	**結構**
MQOD_STRUC_ID	"OD¬¬"
MQOD_STRUC_ID_ARRAY	'O','D','¬','¬'

註: 符號 ¬ 代表單一空白字元。

表 *243:* 常數值		
名稱	**小數值**	**十六進位值**
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)	

MQOII_ * (物件實例 ID)

表 *244:* 常數名稱和值	
名稱	**值**
MQOII_NONE	X'00...00' (24 個空值)
MQOII_NONE_ARRAY	'\0','\0',... (24 個空值)

MQOL_ * (原始長度)

表 *245:* 常數值		
名稱	**小數值**	**十六進位值**
未定義 MQOL_UNDEFINED	-1	X'FFFFFFFF'

MQOM_ * (查詢群組的已作廢 Db2 訊息選項)

表 *246:* 常數值		
名稱	**小數值**	**十六進位值**
MQOM_NO	0	X'00000000'
MQOM_YES	1	X'00000001'

MQOO_ * (開啟選項)

表 *247:* 常數值		
名稱	**小數值**	**十六進位值**
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'
MQ 瀏覽	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'

表 *247:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
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'
MQ OO_BIND_ON_GROUP	4194304	X'00400000'

MQOO_ * (下列僅在 C++ 中使用)

表 *248:* 常數值		
名稱	**小數值**	**十六進位值**
MQOO_RESOLVE_NAMES	65536	X'00010000'
MQOO_RESOLVE_LOCAL_Q	262144	X'00040000'

MQOP_ * (MQCTL 及 MQCB 的作業碼)

MQCTL 的作業碼

表 *249:* 常數值		
名稱	**小數值**	**十六進位值**
MQOP_START	1	X'00000001'
MQOP_START_WAIT	2	X'00000002'
MQ 停止	4	X'00000004'

MQCB 的作業碼

表 *250:* 常數值		
名稱	**小數值**	**十六進位值**
MQOP_REGISTER	256	X'00000100'
MQ 作業 _ 取消登錄	512	X'00000200'

MQCTL 及 MQCB 的作業碼

表 *251:* 常數值		
名稱	**小數值**	**十六進位值**
MQOP_SUSPEND	65536	X'00010000'
MQ 作業 _ 回復	131072	X'00020000'

MQOPEN_ * (與 MQOPEN_PRIV 結構相關的值)

表 *252:* 常數值		
名稱	**小數值**	**十六進位值**
MQOPEN_PRIV_VERSION_1	1	X'00000001'
MQOPEN_PRIV_CURRENT_VERSION	1	X'00000001'

MQOPER_ * (活動作業)

表 *253:* 常數值		
名稱	**小數值**	**十六進位值**
MQ 作業系統 _FIRST	0	X'00000000'
MQOPER_UNKNOWN	0	X'00000000'
MQ 作業系統 _ 瀏覽	1	X'00000001'
MQ 作業 _ 捨棄	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'
MQ 作業系統 _ 轉換	9	X'00000009'
MQOPER_PUBLISH	10	X'0000000A'
MQOPER_EXCLUDED_Publish	11	X'0000000B'
MQ 作業系統 _ 捨棄 _ 發佈	12	X'0000000C'
MQOPER_SYSTEM_LAST	65535	X'0000FFFF'
第一個 MQOPER_APPL_FIRST	65536	X'00010000'
MQOPER_APPL_LAST	999999999	X'3B9AC9FF'

MQOT_ * (物件類型和延伸物件類型)

物件類型

表 *254:* 常數值		
名稱	**小數值**	**十六進位值**
MQOT_NONE	0	X'00000000'
MQOT_Q	1	X'00000001'
MQOT_NAMELIST	2	X'00000002'
MQ 處理程序	3	X'00000003'
MQOT_STORAGE_CLASS	4	X'00000004'
MQOT_Q_MGR	5	X'00000005'
MQ 通道	6	X'00000006'
MQOT_AUTH_INFO	7	X'00000007'
MQOT_TOPIC	8	X'00000008'

表 *254:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
MQOT_CF_STRUC	10	X'0000000A'
MQ 接聽器	11	X'0000000B'
MQ 服務	12	X'0000000C'
MQOT_RESERVED_1	999	X'000003E7'

延伸物件類型

表 *255:* 常數值		
名稱	**小數值**	**十六進位值**
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'
MQ 接收端通道	1010	X'000003F2'
MQOT_CURRENT_CHANNEL	1011	X'000003F3'
MQOT_SAVED_CHANNEL	1012	X'000003F4'
MQOT_SVRCONN_CHANNEL	1013	X'000003F5'
MQOT_CLNTCONN_CHANNEL	1014	X'000003F6'
MQOT_SHORT_CHANNEL	1015	X'000003F7'

MQPA_ * (放置權限)

表 *256:* 常數值		
名稱	**小數值**	**十六進位值**
MQPA_DEFAULT	1	X'00000001'
MQPA_CONTEXT	2	X'00000002'
MQPA_ONLY_MCA	3	X'00000003'
MQPA_ALTERNATE_OR_MCA	4	X'00000004'

MQPD_ * (內容描述子、支援及環境定義)

內容描述子結構

表 *257:* 常數的結構	
名稱	**結構**
MQPD_STRUC_ID	"PD¬¬"
MQPD_STRUC_ID_ARRAY	'P','D','¬','¬'

註: 符號 ¬ 代表單一空白字元。

表 *258:* 常數值		
名稱	**小數值**	**十六進位值**
MQPD_VERSION_1	1	X'00000001'
MQPD_CURRENT_VERSION	1	X'00000001'

註: 符號 ¬ 代表單一空白字元。

內容描述子選項

表 *259:* 常數值		
名稱	**小數值**	**十六進位值**
MQPD_NONE	0	X'00000000'

內容支援選項

表 *260:* 常數值		
名稱	**小數值**	**十六進位值**
MQPD_SUPPORT_OPTIONAL	1	X'00000001'
MQPD_SUPPORT_REQUIRED	1048576	X'00100000'
MQPD_SUPPORT_REQUIRED_IF_LOCAL	1024	X'00000400'
MQPD_REJECT_UNSUPP_MASK	-1048576	X'FFF00000'
MQPD_ACCEPT_UNSUPP_IF_XMIT_MASK	1047552	X'000FFC00'
MQPD_ACCEPT_UNSUPP_MASK	1023	X'000003FF'

內容環境定義

表 *261:* 常數值		
名稱	**小數值**	**十六進位值**
MQPD_NO_CONTEXT	0	X'00000000'
MQPD_USER_CONTEXT	1	X'00000001'

MQPER_ * (持續性值)

表 *262:* 常數值		
名稱	**小數值**	**十六進位值**
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_ * (平台)

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'
MQ 視窗	5	X'00000005'
MQPL_WINDOWS_NT	11	X'0000000B'
MQPL_VMS	12	X'0000000C'
MQPL_NSK	13	X'0000000D'
MQPL_NSS	13	X'0000000D'
MQPL_OPEN_TP1	15	X'0000000F'
MQPL_VM	18	X'00000012'
MQPL_TPF	23	X'00000017'
MQPL_VSE	27	X'0000001B'
MQPL_NATIVE	1	X'00000001'

MQPMO_ * (發佈遮罩的放置訊息選項及結構)

放置訊息選項結構

表 *263:* 常數的結構

名稱	結構
MQPMO_STRUC_ID	"PMO¬"
MQPMO_STRUC_ID_ARRAY	'P','M','O','¬'

註: 符號 ¬ 代表單一空白字元。

表 *264:* 常數值

名稱	小數值	十六進位值
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)

放置訊息選項

表 *265:* 常數值

名稱	小數值	十六進位值
MQPMO_SYNCPOINT	2	X'00000002'
MQPMO_NO_SYNCPOINT	4	X'00000004'
MQPMO_DEFAULT_CONTEXT	32	X'00000020'
MQPMO_NEW_MSG_ID	64	X'00000040'

表 265: 常數值 (繼續)		
名稱	**小數值**	**十六進位值**
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'
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'

發佈遮罩的放置訊息選項

表 266: 常數值		
名稱	**小數值**	**十六進位值**
MQPMO_PUB_OPTIONS_MASK	2097152	X'00200000'

MQPMRF_ * (放置訊息記錄欄位)

表 267: 常數值		
名稱	**小數值**	**十六進位值**
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_ * (指令格式清除選項)

表 *268:* 常數值		
名稱	**小數值**	**十六進位值**
MQPO_YES	1	X'00000001'
MQPO_NO	0	X'00000000'

MQPRI_ * (優先順序)

表 *269:* 常數值		
名稱	**小數值**	**十六進位值**
MQPRI_PRIORITY_AS_Q_DEF	-1	X'FFFFFFFF'
MQPRI_PRIORITY_AS_PARENT	-2	X'FFFFFFFE'
已發佈 MQPRI_PRIORITY_AS_PUBLISHED	-3	X'FFFFFFFD'
MQPRI_PRIORITY_AS_TOPIC_DEF	-1	X'FFFFFFFF'

MQPROP_ * (佇列和通道內容控制值及內容長度上限)

佇列及通道內容控制值

表 *270:* 常數值		
名稱	**小數值**	**十六進位值**
MQPROP_COMPATIBILITY	0	X'00000000'
MQPROP_NONE	1	X'00000001'
MQPROP_ALL	2	X'00000002'
MQPROP_FORCE_MQRFH2	3	X'00000003'

內容長度上限

表 *271:* 常數值		
名稱	**小數值**	**十六進位值**
MQPROP_UNRESTRICTED_LENGTH	-1	X'FFFFFFFF'

MQPRT_ * (放置回應值)

表 *272:* 常數值		
名稱	**小數值**	**十六進位值**
MQPRT_RESPONSE_AS_PARENT	0	X'00000000'
MQPRT_SYNC_response	1	X'00000001'
MQPRT_ASYNC_RESPONSE	2	X'00000002'

MQPS_ * (發佈/訂閱)

指令格式發佈/訂閱狀態

表 *273:* 常數值		
名稱	**小數值**	**十六進位值**
MQPS_STATUS_INACTIVE	0	X'00000000'

表 273: 常數值 (繼續)		
名稱	**小數值**	**十六進位值**
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'

以字串形式發佈/訂閱標籤

表 274: 常數值		
名稱	**小數值**	**十六進位值**
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	"MQPSParmId"	
MQPS_PUBLICATION_OPTIONS	"MQPSPubOpts"	
MQPS_PUBLISH_TIMESTAMP	"MQPSPubTime"	
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"	

發佈/訂閱標籤為以空白括住的字串

表 275: 常數值		
名稱	**小數值**	**十六進位值**
MQPS_COMMAND_B	"bMQPSCommandb"	
MQPS_COMP_CODE_B	"bMQPSCompCodeb"	

表 *275:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
MQPS_CORREL_ID_B	"bMQPSCorrelIdb"	
MQPS_DELETE_OPTIONS_B	"bMQPSDelOptsb"	
MQPS_ERROR_ID_B	"bMQPSErrorIdb"	
MQPS_ERROR_POS_B	"bMQPSErrorPosb"	
MQ PS_INTEGER_DATA_B	"bMQPSIntDatab"	
MQPS_PARAMETER_ID_B	"bMQPSParmIdb"	
MQPS_PUBLICATION_OPTIONS_B	"bMQPSPubOptsb"	
MQPS_PUBLISH_TIMESTAMP_B	"bMQPSPubTimeb"	
MQPS_Q_MGR_NAME_B	"bMQPSQMgrNameb"	
MQPS_Q_NAME_B	"bMQPSQNameb"	
MQPS_REASON_B	"bMQPSReasonb"	
MQPS_REASON_TEXT_B	"bMQPSReasonTextb"	
MQPS_REGISTRATION_OPTIONS_B	"bMQPSRegOptsb"	
MQPS_SEQUENCE_NUMBER_B	"bMQPSSeqNumb"	
MQPS_STREAM_NAME_B	"bMQPSStreamNameb"	
MQPS_STRING_DATA_B	"bMQPSStringDatab"	
MQPS_SUBSCRIPTION_IDENTITY_B	"bMQPSSubIdentityb"	
MQPS_SUBSCRIPTION_NAME_B	"bMQPSSubNameb"	
MQPS_SUBSCRIPTION_USER_DATA_B	"bMQPSSubUserDatab"	
MQPS_TOPIC_B	"bMQPSTopicb"	
MQPS_USER_ID_B	"bMQPSUserIdb"	

以字串形式發佈/訂閱指令標籤值

表 *276:* 常數值		
名稱	**小數值**	**十六進位值**
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"	

發佈/訂閱指令標籤值為以空白括住的字串

表 *277:* 常數值		
名稱	**小數值**	**十六進位值**
MQPS_DELETE_PUBLICATION_B	"bDeletePubb"	
MQPS_DEREGISTER_PUBLISHER_B	"bDeregPubb"	
MQPS_DEREGISTER_SUBSCRIBER_B	"bDeregSubb"	

表 277: 常數值 (繼續)		
名稱	**小數值**	**十六進位值**
MQPS_PUBLISH_B	"bPublishb"	
MQPS_REGISTER_PUBLISHER_B	"bRegPubb"	
MQPS_REGISTER_SUBSCRIBER_B	"bRegSubb"	
MQPS_REQUEST_UPDATE_B	"bReqUpdateb"	

以字串形式發佈/訂閱選項標籤值

表 278: 常數值		
名稱	**小數值**	**十六進位值**
MQPS_ADD_NAME	"AddName"	
MQPS_ANONYMOUS	"Anon"	
MQPS_CORREL_ID_AS_IDENTITY	"CorrelAsId"	
MQ PS_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"	
僅 MQPS_LEAVE_ONLY	"LeaveOnly"	
本端 MQPS_LOCAL	"Local"	
MQPS_LOCKED	"Locked"	
MQPS_NEW_PUBLICATIONS_ONLY	"NewPubsOnly"	
MQPS_NO_ALTERATION	"NoAlter"	
MQPS_NO_XX_ENCODE_CASE_ONE 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"	

發佈/訂閱選項標籤值為以空白括住的字串

表 *279:* 常數值		
名稱	**小數值**	**十六進位值**
MQPS_ADD_NAME_B	"bAddNameb"	
MQPS_ANONYMOUS_B	"bAnonb"	
MQPS_CORREL_ID_AS_IDENTITY_B	"bCorrelAsIdb"	
MQPS_DEREGISTER_ALL_B	"bDeregAllb"	
MQPS_DIRECT_REQUESTS_B	"bDirectReqb"	
MQPS_DUPLICATES_OK_B	"bDupsOKb"	
MQPS_FULL_RESPONSE_B	"bFullRespb"	
MQPS_INCLUDE_STREAM_NAME_B	"bInclStreamNameb"	
MQPS_INFORM_IF_RETAINED_B	"bInformIfRetb"	
MQPS_IS_RETAINED_PUBLICATION_B	"bIsRetainedPubb"	
MQPS_JOIN_EXCLUSIVE_B	"bJoinExclb"	
MQPS_JOIN_SHARED_B	"bJoinSharedb"	
MQPS_LEAVE_ONLY_B	"bLeaveOnlyb"	
MQPS_LOCAL_B	"bLocalb"	
MQPS_LOCKED_B	"bLockedb"	
MQPS_NEW_PUBLICATIONS_ONLY_B	"bNewPubsOnlyb"	
MQPS_NO_ALTERATION_B	"bNoAlterb"	
MQPS_NO_REGISTRATION_B	"bNoRegb"	
MQPS_NON_PERSISTENT_B	"bNonPersb"	
MQPS_NONE_B	"bNoneb"	
MQPS_OTHER_SUBSCRIBERS_ONLY_B	"bOtherSubsOnlyb"	
MQPS_PERSISTENT_B	"bPersb"	
MQPS_PERSISTENT_AS_PUBLISH_B	"bPersAsPubb"	
MQPS_PERSISTENT_AS_Q_B	"bPersAsQueueb"	
MQPS_PUBLISH_ON_REQUEST_ONLY_B	"bPubOnReqOnlyb"	
MQPS_RETAIN_PUBLICATION_B	"bRetainPubb"	
MQPS_VARIABLE_USER_ID_B	"bVariableUserIdb"	

MQPSC_ * (發佈/訂閱選項標籤發佈/訂閱指令資料夾 (psc) 標籤)

表 *280:* 常數值		
名稱	**小數值**	**十六進位值**
MQPSC_FOLDER_VERSION	1	X'00000001'

MQPSC_ * (發佈/訂閱選項標籤標籤名稱)

表 *281:* 常數值		
名稱	**小數值**	**十六進位值**
MQPSC_COMMAND	"Command"	
MQPSC_REGISTRATION_OPTION	"RegOpt"	

表 *281:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
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_ * (發佈/訂閱選項標籤 XML 標籤名稱)

表 *282:* 常數值		
名稱	**小數值**	**十六進位值**
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>"	

表 *282:* 常數值 *(*繼續*)*

名稱	小數值	十六進位值
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_ * (發佈/訂閱選項標籤值為字串)

表 *283:* 常數值

名稱	小數值	十六進位值
MqpSC_DELETE_出版品	"DeletePub"	
MQ PSC_DEREGISTER_SUBSCRIBER	"DeregSub"	
MQPSC_PUBLISH	"Publish"	
MQ PSC_REGISTER_SUBSCRIBER	"RegSub"	
MQPSC_REQUEST_UPDATE	"ReqUpdate"	

MQPSC_ * (發佈/訂閱選項標籤值為字串)

表 *284:* 常數值

名稱	小數值	十六進位值
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"	
僅 MQPSC_LEAVE_ONLY	"LeaveOnly"	
本端 MQPSC_LOCAL	"Local"	
MQPSC_LOCKED	"Locked"	
僅 MQPSC_NEW_PUBS_ONLY	"NewPubsOnly"	
MQPSC_NO_變更	"NoAlter"	
MQPSC_NON_PERSISTENT	"NonPers"	
僅 MQPSC_OTHER_SUBS_ONLY	"OtherSubsOnly"	
MQPSC_PERSISTENT	"Pers"	
MQPSC_PERSISTENT_AS_PUBLISH	"PersAsPub"	
MQPSC_PERSISTENT_AS_Q	"PersAsQueue"	

表 *284:* 常數值 *(繼續)*		
名稱	**小數值**	**十六進位值**
MQPSC_NONE	"None"	
MQPSC_PUB_ON_REQUEST_ONLY	"PubOnReqOnly"	
MQPSC_RETAIN_PUB	"RetainPub"	
MQPSC_VARIABLE_USER_ID	"VariableUserId"	

MQPSCR_ * (發佈/訂閱選項)

發佈/訂閱選項標籤發佈/訂閱回應資料夾 (pscr) 標籤

表 *285:* 常數值		
名稱	**小數值**	**十六進位值**
MQPSCR_FOLDER_VERSION	1	X'00000001'

發佈/訂閱選項標籤標籤名稱

表 *286:* 常數值		
名稱	**小數值**	**十六進位值**
MQPSCR_COMPLETION	"Completion"	
MQPSCR_response	"Response"	
MQPSCR_REASON	"Reason"	

發佈/訂閱選項標籤 XML 標籤名稱

表 *287:* 常數值		
名稱	**小數值**	**十六進位值**
MQPSCR_COMPLETION_B	"<Completion>"	
MQPSCR_COMPLETION_E	"</Completion>"	
MQPSCR_RESPONSE_B	"<Response>"	
MQPSCR_RESPONSE_E	"</Response>"	
MQPSCR_REASON_B	"<Reason>"	
MQPSCR_REASON_E	"</Reason>"	

發佈/訂閱選項標籤標籤值

表 *288:* 常數值		
名稱	**小數值**	**十六進位值**
MQPSCR_OK	"ok"	
MQPSCR_WARNING	"warning"	
MQPSCR_ERROR	"error"	

MQPSM_ * (發佈/訂閱模式)

表 *289:* 常數值		
名稱	**小數值**	**十六進位值**
已停用 MQPSM_DISABLED	0	X'00000000'

表 *289:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
MQPSM_COMPAT	1	X'00000001'
已啟用 MQPSM_ENABLED	2	X'00000002'

MQPSPROP_ * (發佈/訂閱訊息內容)

表 *290:* 常數值		
名稱	**小數值**	**十六進位值**
MQPSPROP_NONE	0	X'00000000'
MQPSPROP_COMPAT	1	X'00000001'
MQPSPROP_RFH2	2	X'00000002'
MQPSPROP_MSGPROP	3	X'00000003'

MQPSST_ * (指令格式「發佈/訂閱狀態類型」)

表 *291:* 常數值		
名稱	**小數值**	**十六進位值**
MQPSST_ALL	0	X'00000000'
本端 MQPSST_LOCAL	1	X'00000001'
MQ PSST_母項	2	X'00000002'
MQ PSST _ 子項	3	X'00000003'

MQPUBO_ * (發佈/訂閱發佈選項)

表 *292:* 常數值		
名稱	**小數值**	**十六進位值**
MQPUBO_NONE	0	X'00000000'
MQPUBO_CORREL_ID_AS_IDENTITY	1	X'00000001'
MQPUBO_RETAIN_PUBLICATION	2	X'00000002'
MQPUBO_OTHER_SUBSCRIBERS_ONLY	4	X'00000004'
MQPUBO_NO_XX_ENCODE_CASE_CAPS_LOCK_ON registration	8	X'00000008'
MQPUBO_IS_RETAINED_PUBLICATION	16	X'00000010'

MQPXP_ * (發佈/訂閱遞送結束程式參數結構)

表 *293:* 常數的結構	
名稱	**結構**
MQPXP_STRUC_ID	"PXP¬"
MQPXP_STRUC_ID_ARRAY	'P','X','P','¬'

註: 符號 ¬ 代表單一空白字元。

表 *294:* 常數值		
名稱	**小數值**	**十六進位值**
MQPXP_VERSION_1	1	X'00000001'
MQPXP_CURRENT_VERSION	1	X'00000001'

MQQA_ * (佇列屬性)

禁止取得值

表 *295:* 常數值

名稱	小數值	十六進位值
MQQA_GET_INHIBITED	1	X'00000001'
MQQA_GET_ALLOWED	0	X'00000000'

禁止放置值

表 *296:* 常數值

名稱	小數值	十六進位值
MQQA_PUT_INHIBITED	1	X'00000001'
容許 MQQA_PUT_ALLOWED	0	X'00000000'

佇列可共用

表 *297:* 常數值

名稱	小數值	十六進位值
MQQA_SHAREABLE	1	X'00000001'
MQQA_NOT_SHAREABLE	0	X'00000000'

取消強化

表 *298:* 常數值

名稱	小數值	十六進位值
MQQA_BACKOUT_HARDENED	1	X'00000001'
MQQA_BACKOUT_NOT_HARDENED	0	X'00000000'

MQQDT_ * (佇列定義類型)

表 *299:* 常數值

名稱	小數值	十六進位值
MQQDT_PREDEFINED	1	X'00000001'
MQQDT_PERMANENT_DYNAMIC	2	X'00000002'
MQQDT_TEMPORARY_DYNAMIC	3	X'00000003'
MQQDT_SHARED_DYNAMIC	4	X'00000004'

MQQF_ * (佇列旗標)

表 *300:* 常數值

名稱	小數值	十六進位值
MQQF_LOCAL_Q	1	X'00000001'
MQQF_CLWL_USEQ_ANY	64	X'00000040'
MQQF_CLWL_USEQ_LOCAL	128	X'00000080'

MQQMDT_ * (指令格式佇列管理程式定義類型)

表 *301:* 常數值		
名稱	**小數值**	**十六進位值**
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_ * (佇列管理程式旗標)

表 *302:* 常數值		
名稱	**小數值**	**十六進位值**
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'

MQQMFAC_ * (指令格式佇列管理程式機能)

表 *303:* 常數值		
名稱	**小數值**	**十六進位值**
MQQMFAC_IMS_BRIDGE	1	X'00000001'
MQQMFAC_DB2	2	X'00000002'

MQQMSTA_ * (指令格式「佇列管理程式狀態」)

表 *304:* 常數值		
名稱	**小數值**	**十六進位值**
MQQMSTA_STARTING	1	X'00000001'
MQQMSTA_RUNNING	2	X'00000002'
MQQMSTA_QUIESCING	3	X'00000003'

MQQMT_ * (指令格式佇列管理程式類型)

表 *305:* 常數值		
名稱	**小數值**	**十六進位值**
MQQMT_NORMAL	0	X'00000000'
MQQMT_REPOSITORY	1	X'00000001'

MQQO_ * (指令格式靜止選項)

表 *306:* 常數值		
名稱	**小數值**	**十六進位值**
MQQO_YES	1	X'00000001'
MQQO_NO	0	X'00000000'

MQQSGD_ * (佇列共用群組處置)

表 *307:* 常數值

名稱	小數值	十六進位值
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_ * (指令格式 QSG 狀態)

表 *308:* 常數值

名稱	小數值	十六進位值
MQQSGS_UNKNOWN	0	X'00000000'
已建立 MQQSGS_CREATED	1	X'00000001'
MQQSGS_ACTIVE	2	X'00000002'
MQQSGS_INACTIVE	3	X'00000003'
MQQSGS_FAILED	4	X'00000004'
MQQSGS_PENDING	5	X'00000005'

MQQSIE_ * (指令格式「佇列服務-間隔事件」)

表 *309:* 常數值

名稱	小數值	十六進位值
MQQSIE_NONE	0	X'00000000'
MQQSIE_高	1	X'00000001'
MQQSIE_OK	2	X'00000002'

MQQSO_ * (SET、BROWSE、INPUT 的指令格式「佇列狀態開啟選項」)

表 *310:* 常數值

名稱	小數值	十六進位值
MQQSO_NO	0	X'00000000'
MQQSO_YES	1	X'00000001'
MQQSO_SHARED	1	X'00000001'
MQQSO_EXCLUSIVE	2	X'00000002'

MQQSOT_ * (指令格式「佇列狀態開啟類型」)

表 *311:* 常數值

名稱	小數值	十六進位值
MQQSOT_ALL	1	X'00000001'
MQQSOT_INPUT	2	X'00000002'

表 311: 常數值 (繼續)		
名稱	**小數值**	**十六進位值**
MQQSOT_OUTPUT	3	X'00000003'

MQQSUM_ * (指令格式「佇列狀態未確定的訊息」)

表 312: 常數值		
名稱	**小數值**	**十六進位值**
MQQSUM_YES	1	X'00000001'
MQQSUM_NO	0	X'00000000'

MQQT_ * (佇列類型及延伸佇列類型)

佇列類型

表 313: 常數值		
名稱	**小數值**	**十六進位值**
本端 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'

延伸佇列類型

表 314: 常數值		
名稱	**小數值**	**十六進位值**
MQQT_ALL	1001	X'000003E9'

MQRC_ * (原因碼)

表 315: 常數值		
名稱	**小數值**	**十六進位值**
MQRC_NONE	0	X'00000000'
第一個 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'

表 315: 常數值 (繼續)

名稱	小數值	十六進位值
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'
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'
已變更 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'

表 315: 常數值 (繼續)

名稱	小數值	十六進位值
MQRC_Q_FULL	2053	X'00000805'
MQRC_Q_NOT_EMPTY	2055	X'00000807'
無法使用 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'
不接受 MQRC_SECOND_MARK_NOT_ALLOWED	2062	X'0000080E'
MQRC_SECURITY_ERROR	2063	X'0000080F'
MQRC_SELECTOR_COUNT_ERROR	2065	X'00000811'
已超出 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'
已接受 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'
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'

表 *315:* 常數值 *(*繼續*)*

名稱	小數值	十六進位值
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'
已取消 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'
MQ RC_CONVERTED_MSG_TOO_Big	2120	X'00000848'
MQRC_TRUNCATED	2120	X'00000848'
MQ RC_NO_EXTERNAL_PARATENTS	2121	X'00000849'
MQRC_PARTICIPANT_NOT_AVAILABLE	2122	X'0000084A'
MQRC_OUTCOME_MIXED	2123	X'0000084B'
MQRC_OUTCOME_PENDING	2124	X'0000084C'
已啟動 MQRC_BRIDGE_STARTED	2125	X'0000084D'
MQRC_BRIDGE_STOPPED	2126	X'0000084E'
MQRC_ADAPTER_STORAGE_短缺	2127	X'0000084F'
MQ RC_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'

表 315: 常數值 (繼續)

名稱	小數值	十六進位值
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_ASDID_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'
MQRC_Q_MGR_STOPPING	2162	X'00000872'
MQRC_DUPLICATE_RECOV_COORD	2163	X'00000873'
MQRC_PMO_ERROR	2173	X'0000087D'
找不到 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'

表 315: 常數值 (繼續)

名稱	小數值	十六進位值
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'
MQ RC_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'
MQ RC_FILE_NOT_AUDIT	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'
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'
MQ RC_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'

表 315: 常數值 (繼續)

名稱	小數值	十六進位值
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'
MQ RC_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'
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'
MQ RC_CHANNEL_STOPPED_BY_USER	2279	X'000008E7'
MQRC_HCONFIG_ERROR	2280	X'000008E8'
MQRC_FUNCTION_ERROR	2281	X'000008E9'
已啟動 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'

表 315: 常數值 (繼續)

名稱	小數值	十六進位值
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'
已啟動 MQRC_CHANNEL_ACTIVATED	2295	X'000008F7'
已啟動 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'
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'
不支援 MQRC_FORMAT_NOT_SUPPORTED	2317	X'0000090D'
不支援 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'

表 315: 常數值 (繼續)

名稱	小數值	十六進位值
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'
遺漏 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'
未釋放 MQRC_CONN_TAG_NOT_RELEARED	2344	X'00000928'
無法使用 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'
不支援 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'
已達到 MQRC_BACKOUT_THRESHOLD_REALED	2362	X'0000093A'
MQRC_MSG_NOT_MATCHED	2363	X'0000093B'

表 315: 常數值 (繼續)

名稱	小數值	十六進位值
MQRC_JMS_FORMAT_ERROR	2364	X'0000093C'
不支援 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'
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'

表 315: 常數值 (繼續)

名稱	小數值	十六進位值
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'
MQ RC_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'
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'

表 315: 常數值 (繼續)

名稱	小數值	十六進位值
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'
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'
找不到 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'

表 315: 常數值 (繼續)

名稱	小數值	十六進位值
MQRC_SUB_INHIBITED	2503	X'000009C7'
MQ RC_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'
MQ RC_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'
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_ALREADY_ALREADY 已結合	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'

表 *315:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
MQRC_REOPEN_EXCEL_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'
MQ RC_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'
MQ RC_NEGATIVE_偏移	6118	X'000017E6'
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_ * (指令格式標頭原因碼)

表 *316:* 常數值		
名稱	**小數值**	**十六進位值**
MQRCCF_CFH_TYPE_ERROR	3001	X'00000BB9'
MQRCF_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'

表 316: 常數值 (繼續)

名稱	小數值	十六進位值
MQRCCF_CFH_PARM_COUNT_ERROR	3006	X'00000BBE'
MQRCCF_CFH_COMMAND_ERROR	3007	X'00000BBF'
MQRCF_COMMAND_FAILED	3008	X'00000BC0'
MQRCCF_CFIN_LENGTH_ERROR	3009	X'00000BC1'
MQRCF_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'
MQRCF_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'
MQRCF_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_CFILE_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'
MQRCF_BATCH_SIZE_ERROR	3037	X'00000BDD'
MQRCCF_DISC_INT_ERROR	3038	X'00000BDE'
MQRCCF_SHORT_RETRY_ERROR	3039	X'00000BDF'
MQRCF_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'

表 316: 常數值 (繼續)

名稱	小數值	十六進位值
MQRCF_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'
MQRCF_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'
找不到 MQRCCF_CHL_STATUS_NOT_FOUND	3065	X'00000BF9'
MQRCCF_CFSL_DUPLICATE_PARM	3066	X'00000BFA'
MQRCF_CFSL_TOTAL_LENGTH_ERROR	3067	X'00000BFB'
MQRCCF_CFSL_COUNT_ERROR	3068	X'00000BFC'
MQRCCF_CFSL_STRING_LENGTH_ERR	3069	X'00000BFD'
已刪除 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'

表 316: 常數值 (繼續)

名稱	小數值	十六進位值
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_COMBED	3157	X'00000C55'
MQ RCCF_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'
遺漏 MQRCCF_OBJECT_TYPE_MISSING	3173	X'00000C65'
MQRCF_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'
找到 MQRCCF_NONE_FOUND	3200	X'00000C80'
MQRCCF_SECURITY_SWITCH_OFF	3201	X'00000C81'
MQRCCF_SECURITY_REFRESH_FAILED	3202	X'00000C82'

表 316: 常數值 (繼續)

名稱	小數值	十六進位值
MQRCF_PARM_CONFLICT	3203	X'00000C83'
MQRCF_COMMAND_INHIBITED	3204	X'00000C84'
已刪除 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'
MQ RCCF_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'
MQRCF_COMMAND_LEVEL_CONFLICT	3222	X'00000C96'
MQRCF_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'
MQRCCF_FUNCTION_RESTRICTED	3227	X'00000C9B'
遺漏 MQRCCF_PARM_MISSING	3228	X'00000C9C'
MQRCCF_PARM_VALUE_ERROR	3229	X'00000C9D'
MQRCF_COMMAND_LENGTH_ERROR	3230	X'00000C9E'
MQRCCF_COMMAND_ORIGIN_ERROR	3231	X'00000C9F'
MQRCF_LISTENER_CONFLICT	3232	X'00000CA0'
MQ RCCF_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'

表 316: 常數值 (繼續)

名稱	小數值	十六進位值
MQRCCF_TOO_MANY_FILTERS	3248	X'00000CB0'
MQRCCF_LISTENER_RUNNING	3249	X'00000CB1'
找不到 MQRCCF_LSTR_STATUS_NOT_FOUND	3250	X'00000CB2'
MQ RCCF_SERVICE_RUNNING	3251	X'00000CB3'
找不到 MQRCCF_SERV_STATUS_NOT_FOUND	3252	X'00000CB4'
MQRCF_SERVICE_STOPPED	3253	X'00000CB5'
MQRCCF_CFBS_DUPLICATE_PARM	3254	X'00000CB6'
MQRCF_CFBS_LENGTH_ERROR	3255	X'00000CB7'
MQRCCF_CFBS_PARM_ID_ERROR	3256	X'00000CB8'
MQRCCF_CFBS_STRING_LENGTH_ERR	3257	X'00000CB9'
MQRCF_CFGR_LENGTH_ERROR	3258	X'00000CBA'
MQRCCF_CFGR_PARM_COUNT_ERROR	3259	X'00000CBB'
MQRCCF_CONN_NOT_STOPPED	3260	X'00000CBC'
MQ RCF_SERVICE_REQUEST_PENDING	3261	X'00000CBD'
MQRCCF_NO_START_CMD	3262	X'00000CBE'
MQRCCF_NO_STOP_CMD	3263	X'00000CBF'
MQRCF_CFBF_LENGTH_ERROR	3264	X'00000CC0'
MQRCCF_CFBF_PARM_ID_ERROR	3265	X'00000CC1'
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'0000CC5'
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'
MQRCF_SUB_ALREADY_EXISTS	3311	X'00000CEF'
MQRCCF_UNKNOWN_OBJECT_NAME	3312	X'00000CF0'
MQRCF_REMOTE_Q_NAME_ERROR	3313	X'00000CF1'
MQRCCF_DURABILITY_NOT_ALLOWED	3314	X'00000CF2'
MQRCCF_HOBJ_ERROR	3315	X'00000CF3'
MQRCF_DEST_NAME_ERROR	3316	X'00000CF4'
MQRCCF_INVALID_DESTINATION	3317	X'00000CF5'
MQRCF_PUBSUB_INHIBITED	3318	X'00000CF6'

表 316: 常數值 (繼續)

名稱	小數值	十六進位值
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'
MQRCF_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'
MQRCF_CHLAUTH_NAME_ERROR	3349	X'00000D15'
MQRCCF_SUITE_B_ERROR	3353	X'00000D19'
MQRCCF_PSCLUS_DISABLED_TOPDEF	3359	X'00000D1F'
MQRCCF_PSCLUS_TOPIC_EXISTS	3360	X'00000D20'
MQRCF_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'
無法使用 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'
MQRCF_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'

表 316: 常數值 (繼續)

名稱	小數值	十六進位值
MQ RCCF_LISTENER_NOT_STARTED	4020	X'00000FB4'
MQRCCF_BIND_FAILED	4024	X'00000FB8'
MQRCF_CHANNEL_INDOUBT	4025	X'00000FB9'
MQRCCF_MQCONN_FAILED	4026	X'00000FBA'
MQRCF_MQOPEN_FAILED	4027	X'00000FBB'
MQRCCF_MQGET_FAILED	4028	X'00000FBC'
MQRCF_MQPUT_FAILED	4029	X'00000FBD'
MQRCCF_PING_ERROR	4030	X'00000FBE'
MQRCCF_CHANNEL_IN_USE	4031	X'00000FBF'
找不到 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'
MQRCCF_USER_EXIT_NOT_AVAILABLE	4039	X'00000FC7'
MQRCCF_COMMIT_FAILED	4040	X'00000FC8'
MQRCCF_WRONG_CHANNEL_TYPE	4041	X'00000FC9'
MQRCF_CHANNEL_ALREADY_EXISTS	4042	X'00000FCA'
MQRCCF_DATA_TOO_LARGE	4043	X'00000FCB'
MQRCCF_CHANNEL_NAME_ERROR	4044	X'00000FCC'
MQRCF_XMIT_Q_NAME_ERROR	4045	X'00000FCD'
MQRCCF_MCA_NAME_ERROR	4047	X'00000FCF'
MQRCF_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'

表 316: 常數值 (繼續)

名稱	小數值	十六進位值
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'
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'
找不到 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'

MQRCN_ * (用戶端重新連接常數)

表 317: 常數值

名稱	小數值	十六進位值
MQRCN_NO	0	X'00000000'
MQRCN_YES	1	X'00000001'
MQRCN_Q_MGR	2	X'00000002'

表 *317:* 常數值 *(繼續)*		
名稱	**小數值**	**十六進位值**
MQRCN_DISABLED	3	X'00000003'

MQRCVTIME_ * (接收逾時類型)

表 *318:* 常數值		
名稱	**小數值**	**十六進位值**
MQRCVTIME_MULTIPLY	0	X'00000000'
MQRCVTIME_ADD	1	X'00000001'
MQRCVTIME_EQUAL	2	X'00000002'

MQREADA_ * (先讀值)

表 *319:* 常數值		
名稱	**小數值**	**十六進位值**
MQREADA_NO	0	X'00000000'
MQREADA_YES	1	X'00000001'
MQREADA_DISABLED	2	X'00000002'
MQREADA_INHIBITED	3	X'00000003'
MQREADA_BACKLOG	4	X'00000004'

MQRECORDING_ * (錄製選項)

表 *320:* 常數值		
名稱	**小數值**	**十六進位值**
已停用 MQRECORDING_DISABLED	0	X'00000000'
MQRECORDING_Q	1	X'00000001'
MQRECORDING_MSG	2	X'00000002'

MQREGO_ * (發佈/訂閱登錄選項)

表 *321:* 常數值		
名稱	**小數值**	**十六進位值**
MQREGO_NONE	0	X'00000000'
MQREGO_CORREL_ID_AS_IDENTITY	1	X'00000001'
MQREGO_ANONYMOUS	2	X'00000002'
本端 MQREGO_LOCAL	4	X'00000004'
MQREGO_DIRECT_REQUESTS	8	X'00000008'
僅 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'

表 *321:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
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'
僅 MQREGO_LEAVE_ONLY	524288	X'00080000'
MQREGO_VARIABLE_USER_ID	1048576	X'00100000'
MQREGO_LOCKED	2097152	X'00200000'

MQRFH_ * (規則及格式化標頭結構和旗標)

規則和格式化標頭結構

表 *322:* 常數的結構	
名稱	**結構**
MQRFH_STRUC_ID	"RFH¬"
MQRFH_STRUC_ID_ARRAY	'R','F','H','¬'

註: 符號 ¬ 代表單一空白字元。

表 *323:* 常數值		
名稱	**小數值**	**十六進位值**
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'

規則和格式化標頭旗標

表 *324:* 常數值		
名稱	**小數值**	**十六進位值**
MQRFH_NONE	0	X'00000000'
MQRFH_NO_FLAGS	0	X'00000000'

MQRFH2_* (發佈/訂閱選項標籤 RFH2 最上層資料夾標籤)

表 *325:* 常數值		
名稱	**小數值**	**十六進位值**
MQRFH2_NAME_VALUE_VERSION	1	X'00000001'

MQRFH2_* (發佈/訂閱選項標籤標籤名稱)

表 *326:* 常數值		
名稱	**小數值**	**十六進位值**
MQRFH2_PUBSUB_CMD_FOLDER	"psc"	
MQRFH2_PUBSUB_RESP_FOLDER	"pscr"	
MQRFH2_MSG_CONTENT_FOLDER	"mcd"	
MQRFH2_USER_FOLDER	"usr"	

MQRFH2_* (發佈/訂閱選項標籤 XML 標籤名稱)

表 *327:* 常數值		
名稱	**小數值**	**十六進位值**
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_ * (傳回長度)

表 *328:* 常數值		
名稱	**小數值**	**十六進位值**
MQRL_UNDEFINED	-1	X'FFFFFFFF'

MQRMH_ * (參照訊息標頭結構)

表 *329:* 常數的結構	
名稱	**結構**
MQRMH_STRUC_ID	"RMH¬"
MQRMH_STRUC_ID_ARRAY	'R','M','H','¬'

註: 符號 ¬ 代表單一空白字元。

表 *330:* 常數值		
名稱	**小數值**	**十六進位值**
MQRMH_VERSION_1	1	X'00000001'
MQRMH_CURRENT_VERSION	1	X'00000001'

MQRMHF_ * (參照訊息標頭旗標)

表 *331:* 常數值		
名稱	**小數值**	**十六進位值**
MQRMHF_LAST	1	X'00000001'
MQRMHF_NOT_LAST	0	X'00000000'

MQRO_ * (報告選項)

表 *332:* 常數值

名稱	小數值	十六進位值
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'
MQ Ro_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'
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_ * (報告選項遮罩)

表 *333:* 常數值

名稱	小數值	十六進位值
MQRO_REJECT_UNSUPP_MASK	270270464	X'101C0000'
MQRO_ACCEPT_UNSUPP_MASK	-270532353	X'EFE000FF'
MQRO_ACCEPT_UNSUPP_IF_XMIT_MASK	261888	X'0003FF00'

MQROUTE_ * (追蹤路徑)

追蹤路徑活動數上限 (MQIACF_MAX_XX_ENCODE_CASE_ONE activities)

表 *334:* 常數值

名稱	小數值	十六進位值
MQROUTE_UNLIMITED_ACTIVITIES	0	X'00000000'

追蹤路徑明細 (MQIACF_ROUTE_DETAIL)

表 *335:* 常數值

名稱	小數值	十六進位值
MQROUTE_DETAIL_LOW	2	X'00000002'
MQROUTE_DETAIL_MEDIUM	8	X'00000008'
MQROUTE_DETAIL_HIGH	32	X'00000020'

追蹤路徑轉遞 (MQIACF_ROUTE_FORWARDING)

表 *336:* 常數值

名稱	小數值	十六進位值
MQROUTE_FORWARD_ALL	256	X'00000100'
MQROUTE_FORWARD_IF_SUPPORTED	512	X'00000200'
MQROUTE_FORWARD_REJ_UNSUPP_MASK	-65536	X'FFFF0000'

追蹤路徑遞送 (MQIACF_ROUTE_DELIVERY)

表 *337:* 常數值

名稱	小數值	十六進位值
MQROUTE_DELIVER_YES	4096	X'00001000'
MQROUTE_DELIVER_NO	8192	X'00002000'
MQROUTE_DELIVER_REJ_UNSUPP_MASK	-65536	X'FFFF0000'

追蹤路徑累積 (MQIACF_ROUTE_ACCUMULATION)

表 *338:* 常數值

名稱	小數值	十六進位值
MQROUTE_ACCUMULATE_NONE	65539	X'00010003'
MQROUTE_ACCUMULATE_IN_MSG	65540	X'00010004'
MQROUTE_ACCUMULATE_AND_REPLY	65541	X'00010005'

MQRP_ * (指令格式「取代選項」)

表 *339:* 常數值

名稱	小數值	十六進位值
MQRP_YES	1	X'00000001'
MQRP_NO	0	X'00000000'

MQRQ_ * (指令格式原因限定元)

表 *340:* 常數值

名稱	小數值	十六進位值
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'

表 *340:* 常數值 *(繼續)*		
名稱	**小數值**	**十六進位值**
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'
MQRQ_SSL_UNKNOWN_REVOCATION	19	X'00000013'

MQRT_ * (指令格式「重新整理類型」)

表 *341:* 常數值		
名稱	**小數值**	**十六進位值**
MQRT_CONFIGURATION	1	X'00000001'
MQRT_EXPIRY	2	X'00000002'
MQRT_NSPROC	3	X'00000003'
MQRT_PROXYSUB	4	X'00000004'

MQRU_ * (僅限要求)

表 *342:* 常數值		
名稱	**小數值**	**十六進位值**
MQRU_Publish_ON_REQUEST	1	X'00000001'
MQRU_PUBLISH_ALL	2	X'00000002'

MQSCA_ * (SSL 用戶端鑑別)

表 *343:* 常數值		
名稱	**小數值**	**十六進位值**
需要 MQSCA_REQUIRED	0	X'00000000'
MQSCA_OPTIONAL	1	X'00000001'

MQSCO_ * (SSL 配置選項)

SSL 配置選項結構

表 *344:* 常數的結構	
名稱	**結構**
MQSCO_STRUC_ID	"SCO¬"
MQSCO_STRUC_ID_ARRAY	'S','C','O','¬'

註: 符號 ¬ 代表單一空白字元。

表 *345:* 常數值		
名稱	**小數值**	**十六進位值**
MQSCO_VERSION_1	1	X'00000001'
MQSCO_VERSION_2	2	X'00000002'
MQSCO_VERSION_3	3	X'00000003'
MQSCO_VERSION_4	4	X'00000004'
MQSO_CURRENT_VERSION	4	X'00000004'

註: 符號 ¬ 代表單一空白字元。

SSL 配置選項金鑰重設計數

表 *346:* 常數值		
名稱	**小數值**	**十六進位值**
MQSCO_RESET_COUNT_DEFAULT	0	X'00000000'

指令格式佇列定義範圍

表 *347:* 常數值		
名稱	**小數值**	**十六進位值**
MQSCO_Q_MGR	1	X'00000001'
MQSCO_CELL	2	X'00000002'

MQSCOPE_ * (發佈範圍)

表 *348:* 常數值		
名稱	**小數值**	**十六進位值**
MQSCOPE_ALL	0	X'00000000'
MQ 範圍_AS_母項	1	X'00000001'
MQSCOPE_QMGR	4	X'00000004'

MQSCYC_ * (安全案例)

表 *349:* 常數值		
名稱	**小數值**	**十六進位值**
MQ 循環 _ 上層	0	X'00000000'
MQ SCYC_MIXED	1	X'00000001'

MQSD_ * (物件描述子結構)

表 *350:* 常數名稱和結構	
名稱	**結構**
MQSD_STRUC_ID	"SD¬¬"
MQSD_STRUC_ID_ARRAY	'S','D','¬','¬'

註: 符號 ¬ 代表單一空白字元。

表 *351:* 常數值		
名稱	**小數值**	**十六進位值**
MQSD_VERSION_1	1	X'00000001'
MqsD_CURRENT_VERSION	1	X'00000001'

MQSECITEM_ * (指令格式安全項目)

表 *352:* 常數值		
名稱	**小數值**	**十六進位值**
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'

MQSECSW_ * (指令格式安全交換器和交換器狀態)

指令格式安全交換器

表 *353:* 常數值		
名稱	**小數值**	**十六進位值**
MQSECSW_PROCESS	1	X'00000001'
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'

表 *353:* 常數值 *(繼續)*		
名稱	**小數值**	**十六進位值**
MQSECSW_COMMAND_RESOURCES	11	X'0000000B'
MQSECSW_Q_MGR	15	X'0000000F'
MQSECSW_QSG	16	X'00000010'

指令格式安全切換狀態

表 *354:* 常數值		
名稱	**小數值**	**十六進位值**
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_ * (指令格式安全類型)

表 *355:* 常數值		
名稱	**小數值**	**十六進位值**
MQSECTYPE_AUTHSERV	1	X'00000001'
MQSECTYPE_SSL	2	X'00000002'
MQ 安全類型 _ 類別	3	X'00000003'

MQSEG_ * (分段)

表 *356:* 常數名稱和值	
名稱	**值**
MQSEG_INHIBITED	'¬'
容許 MQSEG_ALLOWED	'A'

註: 符號 ¬ 代表單一空白字元。

MQSEL_ * (特殊選取元值)

表 *357:* 常數值		
名稱	**小數值**	**十六進位值**
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_ * (選取器類型)

表 *358:* 常數值		
名稱	**小數值**	**十六進位值**
MQSELTYPE_NONE	0	X'00000000'
Mqseltype_standard	1	X'00000001'
MQSELTYPE_EXTENDED	2	X'00000002'

MQSID_ * (安全 ID)

表 *359:* 常數名稱和值	
名稱	**值**
MQSID_NONE	X'00...00' (40 個空值)
MQ Sid_NONE_ARRAY	'\0','\0',... (40 個空值)

MQSIDT_ * (安全 ID 類型)

表 *360:* 常數名稱和值	
名稱	**十六進位值**
無 MQSIDT_NONE	X'00'
MQSIDT_NT_SECURITY_ID	X'01'
MQSIDT_WAS_SECURITY_ID	X'02'

MQSMPO_ * (設定訊息內容選項和結構)

設定訊息內容選項結構

表 *361:* 常數的結構	
名稱	**結構**
MQSMPO_STRUC_ID	"SMPO"
MQSMPO_STRUC_ID_ARRAY	'S','M','P','O'

註: 符號 ¬ 代表單一空白字元。

表 *362:* 常數值		
名稱	**小數值**	**十六進位值**
MQSMPO_VERSION_1	1	X'00000001'
MQSMPO_CURRENT_VERSION	1	X'00000001'

設定訊息內容選項

表 *363:* 常數值		
名稱	**小數值**	**十六進位值**
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'

表 *363:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
MQSMPO_NONE	0	X'00000000'

MQSO_ * (訂閱選項)

表 *364:* 常數值		
名稱	**小數值**	**十六進位值**
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'
MQ 回復	4	X'00000004'
MQ 可延續	8	X'00000008'
MQ 群組 _ 訂閱	16	X'00000010'
受管理 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'
僅 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_ * (同步點可用性)

表 *365:* 常數值		
名稱	**小數值**	**十六進位值**
MQSP_available	1	X'00000001'
MQSP_NOT_AVAILABLE	0	X'00000000'

MQSQQM_ * (共用佇列佇列管理程式名稱)

表 *366:* 常數值		
名稱	**小數值**	**十六進位值**
MQSQQM_USE	0	X'00000000'
MQSQQM_IGNORE	1	X'00000001'

MQSR_ * (動作)

表 *367:* 常數值		
名稱	**小數值**	**十六進位值**
MQSR_ACTION_PUBLICATION	1	X'00000001'

MQSRO_ * (訂閱要求選項結構)

表 *368:* 常數的結構	
名稱	**結構**
MQSRO_STRUC_ID	"SRO¬"
MQSRO_STRUC_ID_ARRAY	'S','R','O','¬'

註: 符號 ¬ 代表單一空白字元。

表 *369:* 常數值		
名稱	**小數值**	**十六進位值**
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_ * (區段狀態)

表 *370:* 常數名稱和結構	
名稱	**結構**
MQSS_NOT_A_SEGMENT	'¬'
MQ SS_SEGMENT	'S'
MQ SS_LAST_SEGMENT	'L'

註: 符號 ¬ 代表單一空白字元。

MQSSL_ * (SSL FIPS 需求)

表 *371:* 常數值		
名稱	**小數值**	**十六進位值**
MQSSL_FIPS_NO	0	X'00000000'
MQSSL_FIPS_YES	1	X'00000001'

MQSTAT_ * (Stat 選項)

MQSTAT_TYPE_ASYNC_ERROR	0	X'00000000'
MQSTAT_TYPE_RECONNECTION	0	X'00000000'
MQSTAT_TYPE_RECONNECTION_ERROR	0	X'00000000'

MQSTS_ * (狀態報告結構)

表 *372:* 常數的結構	
名稱	**結構**
MQSTS_STRUC_ID	"STAT"
MQSTS_STRUC_ID_ARRAY	'S','T','A','T'

註: 符號 ¬ 代表單一空白字元。

表 *373:* 常數值		
名稱	**小數值**	**十六進位值**
MQSTS_VERSION_1	1	X'00000001'
MQSTS_CURRENT_VERSION	1	X'00000001'

MQSUB_ * (可延續訂閱)

可延續訂閱

表 *374:* 常數值		
名稱	**小數值**	**十六進位值**
MQSUB_DURABLE_AS_PARENT	0	X'00000000'
容許 MQSUB_DURABLE_ALLOWED	1	X'00000001'
MQSUB_DURABLE_INHIBITED	2	X'00000002'

可延續訂閱

表 *375:* 常數值		
名稱	**小數值**	**十六進位值**
MQSUB_DURABLE_ALL	-1	X'FFFFFFFF'
MQSUB_DURABLE_YES	1	X'00000001'
MQSUB_DURABLE_NO	2	X'00000002'

MQSUBTYPE_ * (指令格式「訂閱類型」)

表 *376:* 常數值		
名稱	**小數值**	**十六進位值**
MQSUBTYPE_API	1	X'00000001'
MQSUBTYPE_ADMIN	2	X'00000002'
MQ 子類型 _Proxy	3	X'00000003'
MQSUBTYPE_ALL	-1	X'FFFFFFFF'
MQSUBTYPE_USER	-2	X'FFFFFFFE'

MQSUS_ * (指令格式「暫停狀態」)

表 *377:* 常數值		
名稱	**小數值**	**十六進位值**
MQSUS_YES	1	X'00000001'
MQSUS_NO	0	X'00000000'

MQSVC_ * (服務)

服務類型

表 *378:* 常數值		
名稱	**小數值**	**十六進位值**
MQSVC_TYPE_COMMAND	0	X'00000000'
MQSVC_TYPE_SERVER	1	X'00000001'

服務控制

表 *379:* 常數值		
名稱	**小數值**	**十六進位值**
MQSVC_CONTROL_Q_MGR	0	X'00000000'
MQSVC_CONTROL_Q_MGR_START	1	X'00000001'
MQSVC_CONTROL_MANUAL	2	X'00000002'

服務狀態

表 *380:* 常數值		
名稱	**小數值**	**十六進位值**
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_ * (Pub/Sub 移轉的指令格式同步點值)

表 *381:* 常數值		
名稱	**小數值**	**十六進位值**
MQ 同步點 _ 是	0	X'00000000'
MQSYNCPOINT_IFPER	1	X'00000001'

MQSYSP_ * (指令格式「系統參數值」)

表 *382:* 常數值		
名稱	**小數值**	**十六進位值**
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'

表 *382:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
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'
MQSYSP_STATUS_COPYING_BSDS	35	X'00000023'
MQSYSP_STATUS_COPYING_LOG	36	X'00000024'

MQTA_ * (主題屬性)

萬用字元

表 *383:* 常數值		
名稱	**小數值**	**十六進位值**
MQTA_BLOCK	1	X'00000001'
MQTA_PASSTHRU	2	X'00000002'

容許訂閱

表 *384:* 常數值		
名稱	**小數值**	**十六進位值**
MQTA_SUB_AS_PARENT	0	X'00000000'
MQTA_SUB_INHIBITED	1	X'00000001'
容許 MQTA_SUB_ALLOWED	2	X'00000002'

Proxy 訂閱傳播

表 *385:* 常數值		
名稱	**小數值**	**十六進位值**
MQTA_PROXY_SUB_FORCE	1	X'00000001'
MQTA_PROXY_SUB_FIRSTUSE	2	X'00000002'

容許發佈

表 *386:* 常數值		
名稱	**小數值**	**十六進位值**
MQTA_PUB_AS_PARENT	0	X'00000000'
MQTA_PUB_INHIBITED	1	X'00000001'
容許 MQTA_PUB_ALLOWED	2	X'00000002'

MQTC_ * (觸發控制)

表 *387:* 常數值		
名稱	**小數值**	**十六進位值**
MQTC_OFF	0	X'00000000'
MQTC_ON	1	X'00000001'

MQTCPKEEP_ * (TCP Keepalive)

表 *388:* 常數值		
名稱	**小數值**	**十六進位值**
MQTCPKEEP_NO	0	X'00000000'
MQTCPKEEP_YES	1	X'00000001'

MQTCPSTACK_ * (TCP 堆疊類型)

表 *389:* 常數值		
名稱	**小數值**	**十六進位值**
MQTCPSTACK_SINGLE	0	X'00000000'
MQTCPSTACK_MULTIPLE	1	X'00000001'

MQTIME_ * (指令格式時間單位)

表 *390:* 常數值		
名稱	**小數值**	**十六進位值**
MQTIME_UNIT_MINS	0	X'00000000'
MQTIME_UNIT_SECS	1	X'00000001'

MQTM_ * (觸發訊息結構)

表 *391:* 常數的結構	
名稱	**結構**
MQTM_STRUC_ID	"TM¬¬"
MQTM_STRUC_ID_ARRAY	'T','M','¬','¬'

註: 符號 ¬ 代表單一空白字元。

表 *392:* 常數值		
名稱	**小數值**	**十六進位值**
MQTM_VERSION_1	1	X'00000001'
MQTM_CURRENT_VERSION	1	X'00000001'

MQTMC_ * (觸發訊息字元格式結構)

表 *393:* 常數的結構	
名稱	**結構**
MQTMC_STRUC_ID	"TMC¬"
MQTMC_STRUC_ID_ARRAY	'T','M','C','¬'
MQTMC_VERSION_1	"¬¬¬1"

表 *393:* 常數的結構 *(繼續)*	
名稱	**結構**
MQTMC_VERSION_2	"¬¬¬2"
MQTMC_CURRENT_VERSION	"¬¬¬2"
MQTMC_VERSION_1_ARRAY	'¬','¬','¬','1'
MQTMC_VERSION_2_ARRAY	'¬','¬','¬','2'
MQTMC_CURRENT_VERSION_ARRAY	'¬','¬','¬','2'

註: 符號 ¬ 代表單一空白字元。

MQTOPT_ * (主題類型)

表 *394:* 常數值		
名稱	**小數值**	**十六進位值**
本端 MQTOPT_LOCAL	0	X'00000000'
MQTOPT_CLUSTER	1	X'00000001'
MQTOPT_ALL	2	X'00000002'

MQTRAXSTR_ * (通道起始程式追蹤自動啟動)

表 *395:* 常數值		
名稱	**小數值**	**十六進位值**
MQTRAXSTR_NO	0	X'00000000'
MQTRAXSTR_YES	1	X'00000001'

MQTSCOPE_ * (訂閱範圍)

表 *396:* 常數值		
名稱	**小數值**	**十六進位值**
MQTSCOPE_QMGR	1	X'00000001'
MQTSCOPE_ALL	2	X'00000002'

MQTT_ * (觸發類型)

表 *397:* 常數值		
名稱	**小數值**	**十六進位值**
MQTT_NONE	0	X'00000000'
MQTT_FIRST	1	X'00000001'
MQTT_EVERY	2	X'00000002'
MQ tt_DEPTH	3	X'00000003'

MQTYPE_ * (內容資料類型)

表 *398:* 常數值		
名稱	**小數值**	**十六進位值**
MQTYPE_AS_SET	0	X'00000000'
MQTYPE_NULL	2	X'00000002'

表 *398:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
MQ 類型 _ 布林	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'
MQTYPE_STRING	1024	X'00000400'

MQUA_ * (發佈/訂閱使用者屬性選取器)

表 *399:* 常數值		
名稱	**小數值**	**十六進位值**
MQUA_FIRST	65536	X'00010000'
MQUA_LAST	999999999	X'3B9AC9FF'

MQUIDSUPP_ * (指令格式「使用者 ID 支援」)

表 *400:* 常數值		
名稱	**小數值**	**十六進位值**
MQUIDSUPP_NO	0	X'00000000'
MQUIDSUPP_YES	1	X'00000001'

MQUNDELIVERED_ * (Pub/Sub 移轉的指令格式「未遞送」值)

表 *401:* 常數值		
名稱	**小數值**	**十六進位值**
MQUNDELIVERED_NORMAL	0	X'00000000'
MQUNDELIVERED_SAFE	1	X'00000001'
MQUNDELIVERED_DISCARD	2	X'00000002'
MQUNDELIVERED_KEEP	3	X'00000003'

MQUOWST_ * (指令格式「UOW 狀態」)

表 *402:* 常數值		
名稱	**小數值**	**十六進位值**
MQUOWST_NONE	0	X'00000000'
MQUOWST_ACTIVE	1	X'00000001'
MQUOWST_PREPARED	2	X'00000002'
未解析 MQUOWST_UNRESOLVED	3	X'00000003'

MQUOWT_ * (指令格式 UOW 類型)

表 *403:* 常數值

名稱	小數值	十六進位值
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_ * (佇列使用情形)

表 *404:* 常數值

名稱	小數值	十六進位值
MQUS_NORMAL	0	X'00000000'
MQ 使用者 _ 傳輸	1	X'00000001'

MQUSAGE_ * (指令格式「頁集使用值」和「資料集使用值」)

指令格式「頁集用法值」

表 *405:* 常數值

名稱	小數值	十六進位值
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_EXPAND_USER	1	X'00000001'
MQUSAGE_EXPAND_SYSTEM	2	X'00000002'
MQUSAGE_EXPAND_NONE	3	X'00000003'

指令格式資料集用法值

表 *406:* 常數值

名稱	小數值	十六進位值
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_ * (值長度)

表 *407:* 常數值

名稱	小數值	十六進位值
MQVL_NULL_TERMINATED	-1	X'FFFFFFFF'
MQVL_EMPTY_STRING	0	X'00000000'

MQVU_ * (可變使用者 ID)

表 *408:* 常數值		
名稱	**小數值**	**十六進位值**
MQVU_FIXED_USER	1	X'00000001'
MQVU_ANY_USER	2	X'00000002'

MQWDR_ * (叢集工作量結束程式目的地記錄結構)

表 *409:* 常數的結構	
名稱	**結構**
MQWDR_STRUC_ID	"WDR¬"
MQWDR_STRUC_ID_ARRAY	'W','D','R','¬'

註: 符號 ¬ 代表單一空白字元。

表 *410:* 常數值		
名稱	**小數值**	**十六進位值**
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_ * (等待間隔)

表 *411:* 常數值		
名稱	**小數值**	**十六進位值**
MQWI_UNLIMITED	-1	X'FFFFFFFF'

MQWIH_ * (工作量資訊標頭結構及旗標)

工作量資訊標頭結構

表 *412:* 常數的結構	
名稱	**結構**
MQWIH_STRUC_ID	"WIH¬"
MQWIH_STRUC_ID_ARRAY	'W','I','H','¬'

註: 符號 ¬ 代表單一空白字元。

表 *413:* 常數值		
名稱	**小數值**	**十六進位值**
MQWIH_VERSION_1	1	X'00000001'
MQWIH_CURRENT_VERSION	1	X'00000001'
MQWIH_LENGTH_1	120	X'00000078'
MQWIH_CURRENT_LENGTH	120	X'00000078'

工作量資訊標頭旗標

表 *414:* 常數值		
名稱	**小數值**	**十六進位值**
MQWIH_NONE	0	X'00000000'

MQWQR_ * (叢集工作量結束程式佇列記錄結構)

表 *415:* 常數的結構	
名稱	**結構**
MQWQR_STRUC_ID	"WQR¬"
MQWQR_STRUC_ID_ARRAY	'W','Q','R','¬'

註: 符號 ¬ 代表單一空白字元。

表 *416:* 常數值		
名稱	**小數值**	**十六進位值**
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_ * (萬用字元綱目)

表 *417:* 常數值		
名稱	**小數值**	**十六進位值**
MQWS_DEFAULT	0	X'00000000'
MQWS_CHAR	1	X'00000001'
MQWS_TOPIC	2	X'00000002'

MQWXP_ * (叢集工作量結束程式參數結構)

MQWXP_ * (叢集工作量結束程式參數結構)

表 *418:* 常數的結構	
名稱	**結構**
MQWXP_STRUC_ID	"WXP¬"
MQWXP_STRUC_ID_ARRAY	'W','X','P','¬'

註: 符號 ¬ 代表單一空白字元。

表 *419:* 常數值		
名稱	**小數值**	**十六進位值**
MQWXP_VERSION_1	1	X'00000001'

表 *419:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
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_ * (叢集工作量旗標)

表 *420:* 常數值		
名稱	**小數值**	**十六進位值**
MQWXP_PUT_BY_CLUSTER_CHL	2	X'00000002'

相關參考

MQWXP-叢集工作量結束程式參數結構中的欄位

MQXACT_ * (API 呼叫程式類型)

表 *421:* 常數值		
名稱	**小數值**	**十六進位值**
MQXACT_EXTERNAL	1	X'00000001'
MQXACT_INTERNAL	2	X'00000002'

MQXC_ * (結束指令)

表 *422:* 常數值		
名稱	**小數值**	**十六進位值**
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_ * (結束程式回應)

表 *423:* 常數值		
名稱	**小數值**	**十六進位值**
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'

表 *423:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
MQXCC_SUPPRESS_EXIT	-5	X'FFFFFFFB'
MQXCC_CLOSE_CHANNEL	-6	X'FFFFFFFA'
MQXCC_REQUEST_ACK	-7	X'FFFFFFF9'
MQXCC_FAILED	-8	X'FFFFFFF8'

MQXDR_ * (結束回應)

表 *424:* 常數值		
名稱	**小數值**	**十六進位值**
MQXDR_OK	0	X'00000000'
MQXDR_CONVERsion_FAILED	1	X'00000001'

MQXE_ * (環境)

表 *425:* 常數值		
名稱	**小數值**	**十六進位值**
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_ * (登錄進入點選項結構及結束選項)

登錄進入點選項結構

表 *426:* 常數的結構	
名稱	**結構**
MQXEP_STRUC_ID	"XEPO"
MQXEPO_STRU_ID_ARRAY	'X','E','P','O'

註: 符號 ¬ 代表單一空白字元。

表 *427:* 常數值		
名稱	**小數值**	**十六進位值**
MQXEPO_VERSION_1	1	X'00000001'
MQXEPO_CURRENT_VERSION	1	X'00000001'

結束選項

表 *428:* 常數值		
名稱	**小數值**	**十六進位值**
MQXEPO_NONE	0	X'00000000'

MQXF_ * (API 函數 ID)

表 *429:* 常數值

名稱	小數值	十六進位值
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_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 交互結束程式參數結構)

表 *430:* 常數的結構	
名稱	**結構**
MQXP_STRUC_ID	"XP¬¬"
MQXP_STRUC_ID_ARRAY	'X','P','¬','¬'

註: 符號 ¬ 代表單一空白字元。

表 *431:* 常數值		
名稱	**小數值**	**十六進位值**
MQXP_VERSION_1	1	X'00000001'

MQXPDA_ * (問題判斷區域)

表 *432:* 常數名稱和值	
名稱	**值**
MQXPDA_NONE	X'00...00' (48 個空值)
MQXPDA_NONE_ARRAY	'\0','\0',... (48 個空值)

MQXPT_ * (傳輸類型)

表 *433:* 常數值		
名稱	**小數值**	**十六進位值**
MQXPT_ALL	-1	X'FFFFFFFF'
本端 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_ * (傳輸佇列標頭結構)

表 *434:* 常數的結構	
名稱	**結構**
MQXqh_STRUC_ID	"XQH¬"
MQXqh_STRU_ID_ARRAY	'X','Q','H','¬'

註: 符號 ¬ 代表單一空白字元。

表 *435:* 常數值		
名稱	**小數值**	**十六進位值**
MQXQH_VERSION_1	1	X'00000001'
MQXqh_CURRENT_VERSION	1	X'00000001'

MQXR_ * (結束原因)

表 *436:* 常數值

名稱	小數值	十六進位值
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'
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'
已接收 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_* (結束回應 2)

表 *437:* 常數值

名稱	小數值	十六進位值
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_ * (結束程式 ID)

表 *438:* 常數值		
名稱	**小數值**	**十六進位值**
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'
MQXT_CLUSTER_WORKLOAD_EXIT	20	X'00000014'
MQXT_PUBSUB_ROUTING_EXIT	21	X'00000015'

MQXUA_ * (結束使用者區域值)

表 *439:* 常數名稱和值	
名稱	**值**
MQXUA_NONE	X'00...00' (16 個空值)
MQXUA_NONE_ARRAY	'\0','\0',... (16 個空值)

MQXWD_ * (結束等待描述子結構)

表 *440:* 常數的結構	
名稱	**結構**
MQXWD_STRUC_ID	"XWD¬"
MQXWD_STRUC_ID_ARRAY	'X','W','D','¬'

註: 符號 ¬ 代表單一空白字元。

表 *441:* 常數值		
名稱	**小數值**	**十六進位值**
MQXWD_VERSION_1	1	X'00000001'

MQZAC_ * (應用程式環境定義結構)

表 *442:* 常數的結構	
名稱	**結構**
MQZAC_STRUC_ID	"ZAC¬"
MQZAC_STRUC_ID_ARRAY	'Z','A','C','¬'

註: 符號 ¬ 代表單一空白字元。

表 *443:* 常數值		
名稱	**小數值**	**十六進位值**
MQZAC_VERSION_1	1	X'00000001'
MQ zAC_CURRENT_VERSION	1	X'00000001'

MQZAD_ * (權限資料結構)

表 *444:* 常數的結構	
名稱	**結構**
MQZAD_STRUC_ID	"ZAD¬"
MQZAD_STRUC_ID_ARRAY	'Z','A','D','¬'

註: 符號 ¬ 代表單一空白字元。

表 *445:* 常數值		
名稱	**小數值**	**十六進位值**
MQZAD_VERSION_1	1	X'00000001'
MQZAD_VERSION_2	2	X'00000002'
MQZAD_CURRENT_VERSION	2	X'00000002'

MQZAET_ * (可安裝服務實體類型)

表 *446:* 常數值		
名稱	**小數值**	**十六進位值**
MQZAET_NONE	0	X'00000000'
MQZAET_PRINCIPAL	1	X'00000001'
MQZAET_GROUP	2	X'00000002'
MQZAET_UNKNOWN	3	X'00000003'

MQZAO_ * (可安裝服務授權)

表 *447:* 常數值		
名稱	**小數值**	**十六進位值**
MQZAO_CONNECT	1	X'00000001'
MQ 導覽 _ 瀏覽	2	X'00000002'
MQZAO_ 輸入	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_ 發佈	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'

表 *447:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
MQZAO_CHANGE	524288	X'00080000'
MQZAO_clear	1048576	X'00100000'
MQZAO_CONTROL	2097152	X'00200000'
已延伸 MQZAO_CONTROL_EXTENDED	4194304	X'00400000'
MQZAO_XX_ENCODE_CASE_ONE authorize	8388608	X'00800000'
MQZAO_ALL_ADMIN	16646144	X'00FE0000'
MQZAO_ALL	16662527	X'00FE3FFF'
MQZAO_REMOVE	16777216	X'01000000'
MQZAO_NONE	0	X'00000000'

MQZAS_ * (可安裝服務服務介面版本)

表 *448:* 常數值		
名稱	**小數值**	**十六進位值**
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_ * (鑑別類型)

表 *449:* 常數值		
名稱	**小數值**	**十六進位值**
MQZAT_INITIAL_CONTEXT	0	X'00000000'
MQZAT_CHANGE_CONTEXT	1	X'00000001'

MQZCI_ * (可安裝服務接續指示器)

表 *450:* 常數值		
名稱	**小數值**	**十六進位值**
MQZCI_DEFAULT	0	X'00000000'
MQ 配置項目 _ 繼續	0	X'00000000'
停止 MQZCI_STOP	1	X'00000001'

MQZED_ * (實體資料結構)

表 *451:* 常數的結構	
名稱	**結構**
MQZED_STRUC_ID	"ZED¬"
MQZED_STRUC_ID_ARRAY	'Z','E','D','¬'

註: 符號 ¬ 代表單一空白字元。

表 *452:* 常數值		
名稱	**小數值**	**十六進位值**
MQZED_VERSION_1	1	X'00000001'
MQZED_VERSION_2	2	X'00000002'
MQZED_CURRENT_VERSION	2	X'00000002'

MQZFP_ * (可用參數結構)

表 *453:* 常數的結構	
名稱	**結構**
MQZFP_STRUC_ID	"ZFP¬"
MQZFP_STRUC_ID_ARRAY	'Z','F','P','¬'

註: 符號 ¬ 代表單一空白字元。

表 *454:* 常數值		
名稱	**小數值**	**十六進位值**
MQZFP_VERSION_1	1	X'00000001'
MQZFP_CURRENT_VERSION	1	X'00000001'

MQZIC_ * (身分環境定義結構)

表 *455:* 常數的結構	
名稱	**結構**
MQZIC_STRUC_ID	"ZIC¬"
MQZIC_STRUC_ID_ARRAY	'Z','I','C','¬'

註: 符號 ¬ 代表單一空白字元。

表 *456:* 常數值		
名稱	**小數值**	**十六進位值**
MQZIC_VERSION_1	1	X'00000001'
MQZIC_CURRENT_VERSION	1	X'00000001'

MQZID_ * (服務的函數 ID)

所有服務共用的函數 ID

表 *457:* 常數值		
名稱	**小數值**	**十六進位值**
MQZID_INIT	0	X'00000000'
MQZID_TERMENT	1	X'00000001'

權限服務的功能 ID

表 *458:* 常數值		
名稱	**小數值**	**十六進位值**
MQZID_INIT_AUTHORITY	0	X'00000000'

表 *458:* 常數值 *(*繼續*)*		
名稱	**小數值**	**十六進位值**
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'
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_特許	13	X'0000000D'

名稱服務的函數 ID

表 *459:* 常數值		
名稱	**小數值**	**十六進位值**
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'

使用者 ID 服務的功能 ID

表 *460:* 常數值		
名稱	**小數值**	**十六進位值**
MQZID_INIT_USERID	0	X'00000000'
MQZID_TERM_USERID	1	X'00000001'
MQZID_Find_USERID	2	X'00000002'

MQZIO_ * (可安裝的服務起始設定選項)

表 *461:* 常數值		
名稱	**小數值**	**十六進位值**
MQZIO_PRIMARY	0	X'00000000'
次要 MQZIO_SECONDARY	1	X'00000001'

MQZNS_ * (名稱服務介面版本)

表 *462:* 常數值		
名稱	**小數值**	**十六進位值**
MQZNS_VERSION_1	1	X'00000001'

MQZSE_ * (可安裝服務啟動-列舉指示器)

表 *463:* 常數值		
名稱	**小數值**	**十六進位值**
MQZSE_START	1	X'00000001'
MQ ZSE_continue	0	X'00000000'

MQZSL_ * (可安裝服務選取器指示器)

表 *464:* 常數值		
名稱	**小數值**	**十六進位值**
MQZSL_NOT_RETURNED	0	X'00000000'
MQZSL_returned	1	X'00000001'

MQZTO_ * (可安裝服務終止選項)

表 *465:* 常數值		
名稱	**小數值**	**十六進位值**
MQZTO_PRIMARY	0	X'00000000'
次要 MQZTO_SECONDARY	1	X'00000001'

MQZUS_ * (使用者 ID 服務介面版本)

表 *466:* 常數值		
名稱	**小數值**	**十六進位值**
MQZUS_VERSION_1	1	X'00000001'

MQI 中使用的資料類型

可在 MQI 中使用之資料類型的相關資訊。 每一種資料類型之相關語言的說明、欄位及語言宣告。

在 MQI 中使用的資料類型簡介

本節介紹 MQI 中使用的資料類型，並提供在支援的程式設計語言中使用它們的一些指引。

基本資料類型

本節包含 MQI (或結束函數) 中所使用資料類型的相關資訊。 這些詳細說明，後面接著範例，顯示如何在下列主題中，以支援的程式設計語言來宣告基本資料類型。

MQI (或結束程式函數) 中使用的資料類型如下:

- 基本資料類型，或
- 基本資料類型 (陣列或結構) 的聚集

在 MQI (或結束函數中) 中使用下列基本資料類型:

基本資料類型名稱	資料類型	說明
MQBOOL	布林	MQBOOL 資料類型代表布林值。 值 0 代表 false。 任何其他值都代表 true。 MQBOOL 必須與 MQLONG 資料類型對齊。
MQBYTE	位元組	MQBYTE 資料類型代表資料的單位元組。 不會在位元組上放置任何特定解譯; 它會被視為位元字串，而不是二進位數字或字元。 不需要特殊對齊方式。 在使用不同字集或編碼的佇列管理程式之間傳送 MQBYTE 資料時，MQBYTE 資料 不會 以任何方式轉換。 MQMD 結構中的 *MsgId* 及 *CorrelId* 欄位如下所示。 MQBYTE 陣列有時用來代表佇列管理程式不知道的主儲存體區域。 例如，區域可能包含應用程式訊息資料或結構。 此區域的界限對齊必須與其中包含的資料本質相容。 在 C 程式設計語言中，任何資料類型都可以用於顯示為 MQBYTE 陣列的函數參數。 這是因為這類參數一律由位址傳遞，而在 C 中，函數參數宣告為指向 void 的指標。
MQBYTEn	*n* 個位元組的字串	每一個 MQBYTEn 資料類型都代表 *n* 個位元組的字串，其中 *n* 可以採用下列任何值: 8、16、24、32、40 或 128。 MQBYTE 資料類型會說明每一個位元組。 不需要特殊對齊方式。 如果位元組字串中的資料短於定義的字串長度，則必須以空值填補資料，以填入字串。 當佇列管理程式將位元組字串傳回應用程式 (例如，在 MQGET 呼叫上) 時，佇列管理程式會以空值填補字串的定義長度。 已命名常數可用來定義位元組字串欄位的長度。 這些在 第 49 頁的『常數』 中列出

基本資料類型名稱	資料類型	說明
MQCHAR	字元	MQCHAR 資料類型代表單位元組字元，或雙位元組或多位元組字元的一個位元組。 不需要特殊對齊方式。 在使用不同字集或編碼的佇列管理程式之間傳送 MQCHAR 資料時，MQCHAR 資料通常需要轉換才能正確解譯資料。 佇列管理程式會針對 MQMD 結構中的 MQCHAR 資料自動執行此動作。 應用程式訊息資料中 MQCHAR 資料的轉換是由 MQGET 呼叫上指定的 MQGMO_CONVERT 選項所控制; 如需進一步詳細資料，請參閱 第 323 頁的『MQGMO-取得訊息選項』 中此選項的說明。
MQCHARn	*n* 個字元的字串	每一個 MQCHARn 資料類型代表一個包含 *n* 個字元的字串，其中 *n* 可以採用下列任何值: 4、8、12、20、28、32、48、64、128 或 256。 每一個字元都由 MQCHAR 資料類型說明。 不需要特殊對齊方式。 如果字串中的資料短於定義的字串長度，則必須以空白填補資料以填入字串。 在某些情況下，可以使用空值字元來提早結束字串，而不是填補空白; 空值字元及其後面的字元會被視為空白，直到字串的定義長度為止。 在呼叫及資料類型說明中識別可以使用空值的位置。 當佇列管理程式將字串傳回應用程式 (例如，在 MQGET 呼叫中) 時，佇列管理程式一律會以空白填補字串的定義長度; 佇列管理程式不會使用空值字元來定界字串。 已命名常數可用來定義字串欄位的長度，並列在 第 49 頁的『常數』 中。
MQFLOAT32	32 位元浮點數字	MQFLOAT32 資料類型是使用標準 IEEE 浮點數格式所代表的 32 位元浮點數。 MQFLOAT32 必須在 4 位元組界限上對齊。 在 z/OS 上使用 C 中的 MQFLOAT32 需要使用 FLOAT (IEEE) 編譯器旗標。 在 COBOL 中使用 MQFLOAT32 僅限於支援 IEEE 格式浮點數的編譯器。 這可能需要使用 FLOAT (NATIVE) 編譯器旗標。

基本資料類型名稱	資料類型	說明
MQFLOAT64	64 位元浮點數字	MQFLOAT64 資料類型是使用標準 IEEE 浮點數格式所代表的 64 位元浮點數。 MQFLOAT64 必須在 8 位元組界限上對齊。 在 z/OS 上 C 中使用 MQFLOAT64 需要使用 FLOAT (IEEE) 編譯器旗標。 在 COBOL 中使用 MQFLOAT64 僅限於支援 IEEE 格式浮點數字的編譯器。 這可能需要使用 FLOAT (NATIVE) 編譯器旗標。
MQHCONFIG	配置控點	MQHCONFIG 資料類型代表配置控點，即針對特定可安裝服務所配置的元件。 配置控點必須在其自然界限上對齊。 應用程式不得依賴儲存在此控點內的資料格式。 如果有效，則其值預期可在進一步 MQI 呼叫中使用，但不預期具有除該目的以外的任何意義。
MQHCONN	連線控點	MQHCONN 資料類型代表連線控點，即與特定佇列管理程式的連線。 連線控點必須在 4 位元組界限上對齊。 應用程式不得依賴儲存在此控點內的資料格式。 如果有效，則其值預期可在進一步 MQI 呼叫中使用，但不預期具有除該目的以外的任何意義。
MQHMSG	訊息控點 (message handle)	MQHMSG 資料類型代表提供訊息存取權的訊息控點。 訊息控點必須在 8 位元組界限上對齊。 應用程式不得依賴儲存在此控點內的資料格式。 如果有效，則其值預期可在進一步 MQI 呼叫中使用，但不預期具有除該目的以外的任何意義。
MQHOBJ	物件控點	MQHOBJ 資料類型代表提供物件存取權的物件控點。 物件控點必須在 4 位元組界限上對齊。 應用程式不得依賴儲存在此控點內的資料格式。 如果有效，則其值預期可在進一步 MQI 呼叫中使用，但不預期具有除該目的以外的任何意義。

基本資料類型名稱	資料類型	說明
MQINT8	8 位元帶正負號整數	MQINT8 資料類型是 8 位元帶正負號的整數，除非環境定義另有限制，否則可以採用 -128 至 +127 範圍內的任何值。
MQINT16	16 位元帶正負號整數	MQINT16 資料類型是 16 位元帶正負號的整數，除非環境定義另有限制，否則可以採用 -32 768 至 +32 767 範圍內的任何值。 MQINT16 必須在 2 位元組界限上對齊。
MQINT32	32 位元帶正負號整數	MQINT32 資料類型是 32 位元帶正負號的二進位整數，除非環境定義另有限制，否則它可以採用 -2 147 483 648 至 + 2 147 483 647 範圍內的任何值。 請參閱 MQLONG 的定義。
MQINT64	64 位元有符號整數	MQINT64 資料類型是 64 位元帶正負號的整數，除非環境定義另有限制，否則可以採用 -9 223 372 036 854 775 808 至 + 9 223 372 036 854 775 807 範圍內的任何值。 若為 COBOL ，有效範圍限制為 -999 999 999 999 999 至 +999 999 999 999 999 999。 64 位元整數必須在 8 位元組界限上對齊。
MQLONG	32 位元帶正負號整數	除非環境定義另有限制，否則 MQLONG 資料類型是 32 位元帶正負號的二進位整數，可以採用 -2 147 483 648 至 + 2 147 483 647 範圍內的任何值。 對於 COBOL ，有效範圍限制為 -999 999 到 +999 999 999。 MQLONG 必須在 4 位元組界限上對齊。
MQPID	處理程序 ID	WebSphere MQ 處理程序 ID。 此 ID 與 MQ 追蹤及 FFST™ 傾出中使用的 ID 相同，但可能與作業系統處理程序 ID 不同。

基本資料類型名稱	資料類型	說明
MQPTR	指標	MQPTR 資料類型是任何類型資料的位址。 指標必須在其自然界限上對齊; 這是 IBM i 上的 16 位元組界限，以及其他平台上的 8 位元組界限。 部分程式設計語言支援鍵入的指標; MQI 也會在少數情況下使用這些指標 (例如，在 C 程式設計語言中使用 PMQCHAR 及 PMQLONG)。
MQTID	執行緒 ID	WebSphere MQ 執行緒 ID。 此 ID 與 MQ 追蹤及 FFST™ 傾出中使用的 ID 相同，但可能與作業系統執行緒 ID 不同。
MQUINT8	8 位元不帶正負號整數	MQUINT8 資料類型是 8 位元不帶正負號的整數，除非環境定義另有限制，否則可以採用 0 到 + 255 範圍內的任何值。
MQUINT16	16 位元無正負號整數	MQUINT16 資料類型是 16 位元不帶正負號的整數，除非環境定義另有限制，否則可以採用 0 到 + 65 535 範圍內的任何值。 MQUINT16 必須在 2 位元組界限上對齊。
MQUINT32	32 位元不帶正負號的整數	MQUINT32 資料類型是 32 位元無正負號二進位整數。 請參閱 MQULONG 的定義。
MQUINT64	64 位元不帶正負號的整數	MQINT64 資料類型是 64 位元不帶正負號的整數，除非環境定義另有限制，否則可以採用 0 至 +18 446 744 073 709 551 615 範圍內的任何值。 對於 COBOL，有效範圍限制為 0 到 +999 999 999 999 999。 64 位元整數必須在 8 位元組界限上對齊。
MQULONG	32 位元不帶正負號的整數	除非環境定義另有限制，否則 MQULONG 資料類型是 32 位元不帶正負號的二進位整數，可以採用 0 到 + 4 294 967 294 範圍內的任何值。 對於 COBOL，有效範圍限制為 0 到 +999 999 999。 MQULONG 必須在 4 位元組界限上對齊。
PMQACH	指標	MQACH 類型的資料結構指標
PMQAIR	指標	MQAIR 類型的資料結構指標

基本資料類型名稱	資料類型	說明
PMQAXC	指標	MQAXC 類型的資料結構指標
PMQAXP	指標	MQAXP 類型的資料結構指標
PMQBMHO	指標	MQBMHO 類型的資料結構指標
PMQBO	指標	MQBO 類型的資料結構指標
PMQBOOL	指標	MQBOOL 類型資料的指標
PMQBYTE	指標	MQBYTE 類型的資料指標
PMQBYTEn	指標	MQBYTEn 類型的資料指標，其中 n 可以是 8、16、24、32、40、128
PMQCBC	指標	MQCBC 類型的資料結構指標
PMQCBD	指標	MQCBD 類型的資料結構指標
PMQCHAR	指標	MQCHAR 類型的資料指標
PMQCHARN	指標	MQCHARN 資料類型的指標，其中 n 可以是 4、8、12、20、28、32、48、64、128、256、264
PMQCHARV	指標	MQCHARV 類型的資料結構指標
PMQCIH	指標	MQCIH 類型的資料結構指標
PMQCMHO	指標	MQCMHO 類型的資料結構指標
PMQCNO	指標	MQCNO 類型的資料結構指標
PMQCSP	指標	MQCSP 類型的資料結構指標
PMQCTLO	指標	MQCTLO 類型的資料結構指標
PMQDH	指標	MQDH 類型的資料結構指標
PMQDHO	指標	MQDHO 類型的資料結構指標
PMQDLH	指標	MQDLH 類型的資料結構指標
PMQDMHO	指標	MQDMHO 類型的資料結構指標
PMQDMPO	指標	MQDMPO 類型的資料結構指標
PMQEPH	指標	MQEPH 類型的資料結構指標
PMQFLOAT32	指標	指向 MQFLOAT32 類型之資料結構的指標
PMQFLOAT64	指標	指向 MQFLOAT64 類型之資料結構的指標
PMQFUNC	指標	函數的指標
PMQGMO	指標	MQGMO 類型的資料結構指標
PMQHCONFIG	指標	MQHCONFIG 類型資料的指標
PMQHCONN	指標	MQHCONN 類型資料的指標
PMQHMSG	指標	MQHMSG 類型資料的指標
PMQHOBJ	指標	MQHOBJ 類型資料的指標

基本資料類型名稱	資料類型	說明
PMQIIH	指標	MQIIH 類型的資料結構指標
PMQIMPO	指標	MQIMPO 類型的資料結構指標
PMQINT8	指標	指向類型 MQINT8 資料的指標
PMQINT16	指標	指向類型 MQINT16 之資料的指標
PMQINT32	指標	指向類型 MQINT32 資料的指標
PMQINT64	指標	指向類型 MQINT64 之資料的指標
PMQLONG	指標	MQLONG 類型資料的指標
PMQMD	指標	MQMD 類型結構的指標
PMQMDE	指標	MQMDE 類型的資料結構指標
PMQMD1	指標	指向類型 MQMD1 的資料結構的指標
PMQMD2	指標	指向 MQMD2 類型之資料結構的指標
PMQMHBO	指標	MQMHBO 類型的資料結構指標
PMQOD	指標	MQOD 類型的資料結構指標
PMQOR	指標	MQOR 類型的資料結構指標
PMQPD	指標	MQPD 類型的資料結構指標
PMQPID	指標	程序 ID 的指標
PMQMD	指標	MQMD 類型的資料結構指標
PMQPMO	指標	MQPMO 類型的資料結構指標
PMQPTR	指標	MQPTR 類型資料的指標
PMQRFH	指標	MQRFH 類型的資料結構指標
PMQRFH2	指標	指向 MQRFH2 類型之資料結構的指標
PMQRMH	指標	MQRMH 類型的資料結構指標
PMQRR	指標	MQRR 類型的資料結構指標
PMQSCO	指標	MQSCO 類型資料結構的指標
PMQSD	指標	MQSD 類型的資料結構指標
PMQSMPO	指標	MQSMPO 類型的資料結構指標
PMQSRO	指標	MQSRO 類型的資料結構指標
PMSSTS	指標	MQSTS 類型的資料結構指標
PMQTID	指標	執行緒 ID 的指標
PMQTM	指標	MQTM 類型的資料結構指標
PMQTMC2	指標	指向 MQTMC2 類型之資料結構的指標
PMQUINT8	指標	指向 MQUINT8 資料類型的指標
PMQUINT16	指標	指向 MQUINT16 資料類型的指標

基本資料類型名稱	資料類型	說明
PMQUINT32	指標	指向 MQUINT32 資料類型的指標
PMQUINT64	指標	指向 MQUINT64 資料類型的指標
PMQULONG	指標	MQULONG 資料類型的指標
PMQVOID	指標	
PMQWIH	指標	MQWIH 類型的資料結構指標
PMQXQH	指標	MQXQH 類型的資料結構指標

C 宣告

資料類型	呈現
MQBOOL	`typedef MQLONG MQBOOL;`
MQBYTE	`typedef unsigned char MQBYTE;`
MQBYTE8	`typedef MQBYTE MQBYTE8[8];`
MQBYTE16	`typedef MQBYTE MQBYTE16[16];`
MQBYTE24	`typedef MQBYTE MQBYTE24[24];`
MQBYTE32	`typedef MQBYTE MQBYTE32[32];`
MQBYTE40	`typedef MQBYTE MQBYTE40[40];`
MQCHAR	`typedef char MQCHAR;`
MQCHAR4	`typedef MQCHAR MQCHAR4[4];`
MQCHAR8	`typedef MQCHAR MQCHAR8[8];`
MQCHAR12	`typedef MQCHAR MQCHAR12[12];`
MQCHAR20	`typedef MQCHAR MQCHAR20[20];`
MQCHAR28	`typedef MQCHAR MQCHAR28[28];`
MQCHAR32	`typedef MQCHAR MQCHAR32[32];`
MQCHAR48	`typedef MQCHAR MQCHAR48[48];`
MQCHAR64	`typedef MQCHAR MQCHAR64[64];`

資料類型	呈現
MQCHAR128	`typedef MQCHAR MQCHAR128[128];`
MQCHAR256	`typedef MQCHAR MQCHAR256[256];`
MQFLOAT32	`typedef float MQFLOAT32;`
MQFLOAT64	`typedef double MQFLOAT64;`
MQHCONFIG	`typedef void MQPOINTER MQHCONFIG;`
MQHCONN	`typedef MQLONG MQHCONN;`
MQHOBJ	`typedef MQLONG MQHOBJ;`
MQINT8	`typedef signed char MQINT8;`
MQINT16	`typedef short MQINT16;`

MQINT64	在 64 位元 UNIX 系統上:
	`typedef long;`
	在 32 位元 AIX、Solaris 及 HP-UX 上:
	`typedef int64_t;`
	在 IBM i、 Linux 和 z/OS 上:
	`typedef long long;`
	在 Windows 上:
	`typedef _int64;`

MQLONG	在 IBM i 上:
	`typedef long MQLONG;`
	其他平台:
	`if defined(MQ_64_BIT)`
	`  typedef int MQLONG;`
	`else`
	`  typedef long MQLONG;`

MQPID	`typedef MQLONG MQPID;`
MQPTR	`typedef void MQPOINTER MQPTR;`
MQTID	`typedef MQLONG MQTID;`

資料類型	呈現
MQUINT8	`typedef unsigned char MQUINT8;`
MQUINT16	`typedef unsigned short MQUINT16;`

MQUINT64	在 64 位元 UNIX 系統上:
	`typedef unsigned long;`
	在 32 位元 AIX、Solaris 及 HP-UX 上:
	`typedef uint64_t;`
	在 IBM i、 Linux 和 z/OS 上:
	`typedef unsigned long long;`
	在 Windows 上:
	`typedef unsigned _int64;`

MQULONG	在 IBM i 上:
	`typedef unsigned long MQULONG;`
	其他平台:
	`if defined(MQ_64_BIT)`
	`  typedef unsigned int MQULONG;`
	`else`
	`  typedef unsigned long MQULONG;`

PMQBO	`typedef MQBO MQPOINTER PMQBO;`
PMQBOOL	`typedef MQBOOL MQPOINTER PMQBOOL;`
PMQBYTE	`typedef MQBYTE MQPOINTER PMQBYTE;`
PMQBYTE8	`typedef MQBYTE8[8] MQPOINTER PMQBYTE8[8];`
PMQBYTE16	`typedef MQBYTE16[16] MQPOINTER PMQBYTE16[16];`
PMQBYTE24	`typedef MQBYTE24[24] MQPOINTER PMQBYTE24[24];`
PMQBYTE32	`typedef MQBYTE32[32] MQPOINTER PMQBYTE32[32];`
PMQBYTE40	`typedef MQBYTE40[40] MQPOINTER PMQBYTE40[40];`
PMQBYTE128	`typedef MQBYTE128[128] MQPOINTER PMQBYTE128[128];`
PMQCHAR	`typedef MQCHAR MQPOINTER PMQCHAR;`

資料類型	呈現
PMQCHAR4	`typedef MQCHAR4[4] MQPOINTER PMQCHAR4[4];`
PMQCHAR8	`typedef MQCHAR8[8] MQPOINTER PMQCHAR8[8];`
PMQCHAR12	`typedef MQCHAR12[12] MQPOINTER PMQCHAR12[12];`
PMQCHAR20	`typedef MQCHAR20[20] MQPOINTER PMQCHAR20[20];`
PMQCHAR28	`typedef MQCHAR28[28] MQPOINTER PMQCHAR28[28];`
PMQCHAR32	`typedef MQCHAR32[32] MQPOINTER PMQCHAR32[32];`
PMQCHAR48	`typedef MQCHAR48[48] MQPOINTER PMQCHAR48[48];`
PMQCHAR64	`typedef MQCHAR64[64] MQPOINTER PMQCHAR64[64];`
PMQCHAR128	`typedef MQCHAR128[128] MQPOINTER PMQCHAR128[128];`
PMQCHAR256	`typedef MQCHAR256[256] MQPOINTER PMQCHAR256[256];`
PMQCHAR264	`typedef MQCHAR264[264] MQPOINTER PMQCHAR264[264];`
PMQCIH	`typedef MQCIH MQPOINTER PMQCIH;`
PMQCNO	`typedef MQCNO MQPOINTER PMQCNO;`
PMQDLH	`typedef MQDLH MQPOINTER PMQDLH;`
PMQFUNC	`typedef void MQPOINTER PMQFUNC;`
PMQFLOAT32	`typedef MQFLOAT32 MQPOINTER PMQFLOAT32;`
PMQFLOAT64	`typedef MQFLOAT64 MQPOINTER PMQFLOAT64;`
PMQGMO	`typedef MQGMO MQPOINTER PMQGMO;`
PMQHCONFIG	`typedef MQHCONFIG MQPOINTER PMQHCONFIG;`
PMQHCONN	`typedef MQHCONN MQPOINTER PMQHCONN;`
PMQHOBJ	`typedef MQHOBJ MQPOINTER PMQHOBJ;`

資料類型	呈現
PMQIIH	`typedef MQIIH MQPOINTER PMQIIH;`
PMQINT8	`typedef MQINT8 MQPOINTER PMQINT8;`
PMQINT16	`typedef MQINT16 MQPOINTER PMQINT16;`
PMQLONG	`typedef MQLONG MQPOINTER PMQLONG;`
PMQMD	`typedef MQMD MQPOINTER PMQMD;`
PMQMD1	`typedef MQMD1[1] MQPOINTER PMQMD1[1];`
PMQMDE	`typedef MQMDE MQPOINTER PMQMDE;`
PMQOD	`typedef MQOD MQPOINTER PMQOD;`
PMQPMO	`typedef MQPMO MQPOINTER PMQPMO;`
PMQPTR	`typedef MQPTR MQPOINTER PMQPTR;`
PMQRFH	`typedef MQRFH MQPOINTER PMQRFH;`
PMQRFH2	`typedef MQRFH2[2] MQPOINTER PMQRFH2[2];`
PMQRMH	`typedef MQRMH MQPOINTER PMQRMH;`
PMQTM	`typedef MQTM MQPOINTER PMQTM;`
PMQTMC2	`typedef MQTMC2[2] MQPOINTER PMQTMC2[2];`
PMQUINT8	`typedef MQUINT8 MQPOINTER PMQUINT8;`
PMQUINT16	`typedef MQUINT16 MQPOINTER PMQUINT16;`
PMQULONG	`typedef MQULONG MQPOINTER PMQULONG;`
PMQVOID	`typedef void MQPOINTER PMQVOID;`
PMQWIH	`typedef MQWIH MQPOINTER PMQWIH;`
PMQXQH	`typedef MQXQH MQPOINTER PMQXQH;`

資料類型	呈現
PPMQBO	`typedef PMQBO MQPOINTER PPMQBO;`
PPMQBYTE	`typedef PMQBYTE MQPOINTER PPMQBYTE;`
PPMQCHAR	`typedef PMQCHAR MQPOINTER PPMQCHAR;`
PPMQCNO	`typedef PMQCNO MQPOINTER PPMQCNO;`
PPMQGMO	`typedef PMQGMO MQPOINTER PPMQGMO;`
PPMQHCONN	`typedef PMQHCONN MQPOINTER PPMQHCONN;`
PPMQHOBJ	`typedef PMQHOBJ MQPOINTER PPMQHOBJ;`
PPMQLONG	`typedef PMQLONG MQPOINTER PPMQLONG;`
PPMQMD	`typedef PMQMD MQPOINTER PPMQMD;`
PPMQOD	`typedef PMQOD MQPOINTER PPMQOD;`
PPMQPMO	`typedef PMQPMO MQPOINTER PPMQPMO;`
PPMQULONG	`typedef PMQULONG MQPOINTER PPMQULONG;`
PPMQVOID	`typedef PMQVOID MQPOINTER PPMQVOID;`
其中 `defined(MQ_64_BIT)` 表示 64 位元平台。	

如需 MQPOINTER 巨集變數的說明，請參閱 第 235 頁的『資料類型』。

COBOL 宣告

資料類型	呈現
MQBOOL	`PIC S9(9) BINARY`
MQBYTE	`PIC X`
MQBYTE8	`PIC X(8)`
MQBYTE16	`PIC X(16)`
MQBYTE24	`PIC X(24)`

資料類型	呈現
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)`
MQCHAR128	`PIC X(128)`
MQCHAR256	`PIC X(256)`
MQFLOAT32	`USAGE COMP-1`
MQFLOAT64	`USAGE COMP-2`
MQHCONN	`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`

資料類型	呈現
MQPTR	`POINTER`
MQUINT8	`PIC 9(2) BINARY`
MQUINT16	`PIC 9(4) BINARY`
MQUINT64	`PIC 9(18) BINARY`
MQULONG	`PIC 9(9) BINARY`

PL/I 宣告
在 z/OS 上支援 PL/I。

資料類型	呈現
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)`
MQCHAR32	`char(32)`

資料類型	呈現
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)`
MQUINT64	`fixed bin(64)`
MQULONG	`fixed bin(32)`

System/390 組譯器宣告
System/390 組譯器僅在 z/OS 上受支援。

資料類型	呈現
MQBOOL	`DS   F`
MQBYTE	`DS   XL1`

資料類型	呈現
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`
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`

資料類型	呈現
MQINT16	`DS H`
MQINT64	`DS D`
MQLONG	`DS F`
MQPTR	`DS F`
MQUINT8	`DS XL1`
MQUINT16	`DS H`
MQUINT64	`DS D`
MQULONG	`DS F`

結構資料類型-簡介

本節介紹 MQI 中使用的結構資料類型。 後續各節會說明結構資料類型本身。

摘要

下表彙總 MQI 中使用的結構資料類型。

表 *467: MQI* 呼叫 *(*或結束函數*)* 所使用的結構資料類型*:*		
結構	**說明**	**使用的呼叫數**
MQACH	API 結束程式鏈標頭	
MQAIR	鑑別資訊記錄	MQCONNX
MQAXC	API 結束程式環境定義	
MQAXP	API 結束程式參數	
MQBMHO	緩衝區至訊息控點選項	MQBUFMH
MQBO	開始選項	MQBEGIN
MQCBD	回呼描述子	MQCB
MQCBO	建立工具袋選項	mqCreate 工具袋
MQCHARV	可變長度字串	MQINQMP
MQCNO	連線選項	MQCONNX
MQCSP	安全參數	MQCONNX
MQCTLO	回呼選項	MQCTL
MQDMPO	刪除訊息內容選項	MQDLTMP
MQGMO	取得訊息選項	MQGet
MQIMPO	查詢訊息內容選項	MQINQMP

表 *467: MQI* 呼叫 (或結束函數) 所使用的結構資料類型*: (*繼續*)*		
結構	**說明**	**使用的呼叫數**
MQMD	訊息描述子	MQBUFMH、 MQMHBUF、 MQCB、 MQGET、 MQPUT、 MQPUT1
MQMHBO	緩衝區選項的訊息控點	MQMHBUF
MQOD	物件描述子 (object descriptor)	MQOPEN， MQPUT1
MQOR	物件記錄	MQOPEN， MQPUT1
MQPD	內容描述子	MQSETMP
MQPMO	放置訊息選項	MQPUT、 MQPUT1
MQPMR	放置訊息記錄	MQPUT、 MQPUT1
MQRR	回應記錄	MQOPEN、 MQPUT、 MQPUT1
MQSCO	SSL 配置選項	MQCONNX
MQSD	訂閱描述子	MQSUB
MQSMPO	設定訊息內容選項	MQSETMP
MQSRO	訂閱要求選項	MQSUBRQ
MQSTS	狀態報告結構	MQSTAT

表 *468:* 訊息資料中使用的結構資料類型*:*	
結構	**說明**
MQCIH	CICS 資訊標頭
MQCFH	PCF 標頭
MQEPH	內嵌的 PCF 標頭
MQDH	配送標頭
MQDLH	無法傳送的郵件 (未遞送的訊息) 標頭
MQIIH	IMS 資訊標頭
MQMDE	訊息描述子延伸
MQRFH	規則及格式化標頭
MQRFH2	規則及格式化標頭 2
MQRMH	參照訊息標頭
MQTM	觸發訊息
MQTMC2	觸發訊息 (字元格式 2)
MQWIH	工作資訊標頭
MQXQH	傳輸佇列標頭

註: 第 791 頁的『資料轉換結束程式』中說明 MQDXP 結構 (資料轉換結束程式參數)， 以及相關聯的資料轉換呼叫。

結構資料類型的規則

程式設計語言的支援層次各不相同，而且會採用某些規則及慣例，以在每一種程式設計語言中一致地對映 MQI 結構:

1. 結構必須在它們的自然界限上對齊。
   - 大部分 MQI 結構都需要 4 位元組對齊。
   - 在 IBM i 上，包含指標的結構需要 16 位元組對齊; 這些是 :MQCNO、MQOD、MQPMO。
2. 結構中的每一個欄位必須在其自然界限上對齊。
   - 資料類型等於 MQLONG 的欄位必須在 4 位元組界限上對齊。
   - 在 IBM i 上，資料類型等於 MQPTR 的欄位必須在 16 位元組界限上對齊，在其他環境中則必須在 4 位元組界限上對齊。
   - 其他欄位在 1 位元組界限上對齊。
3. 結構的長度必須是其界限對齊的倍數。
   - 大部分 MQI 結構的長度都是 4 位元組的倍數。
   - 在 IBM i 上，包含指標的結構長度是 16 位元組的倍數。
4. 必要時，必須新增填補位元組或欄位，以確保符合上述規則。

說明中使用的慣例

每一種結構資料類型的說明包括:

- 結構的用途及使用概觀
- 結構中欄位的說明，其格式與程式設計語言無關
- 在每一種支援的程式設計語言中如何宣告結構的範例

每一種結構資料類型的說明包含下列區段:

結構名稱
結構名稱，後面接著結構中欄位的摘要。

概觀
結構用途及用法的簡要說明。

欄位
欄位的說明。 對於每一個欄位，欄位名稱後面接著括弧 () 中的基本資料類型。 在文字中，使用斜體字體來顯示欄位名稱; 例如 *Version*。

還有欄位用途的說明，以及欄位可以採用的任何值清單。 常數名稱以大寫顯示; 例如 MQGMO_STRUC_ID。 會使用 * 字元來顯示一組具有相同字首的常數，例如 :MQIA_ *。

在欄位的說明中，使用下列術語:

輸入(input)
您在打電話時在欄位中提供資訊。

輸出
當呼叫完成或失敗時，佇列管理程式會在欄位中傳回資訊。

輸入/輸出
當您進行呼叫時，您在欄位中提供資訊，當呼叫完成或失敗時，佇列管理程式會變更資訊。

起始值
顯示隨 MQI 提供之資料定義檔中每一個欄位的起始值的表格。

C 宣告
C 中結構的典型宣告。

COBOL 宣告
COBOL 中結構的一般宣告。

PL/I 宣告
典型的 PL/I 結構宣告

System/390 組譯器宣告
System/390 組譯語言中結構的一般宣告。

Visual Basic 宣告
Visual Basic 中結構的一般宣告。

C 程式設計

本節包含的資訊可協助您使用來自 C 程式設計語言的 MQI。

標頭檔

標頭檔可協助您撰寫使用 MQI 的 C 應用程式。

這些標頭檔在 第 235 頁的表 469 中彙總。

表 *469: C* 標頭檔

檔案	內容
CMQC	主要 MQI 的函數原型、資料類型及具名常數
CMQXC	資料轉換結束程式的函數原型、資料類型及具名常數
CMQEC	主要 MQI、資料轉換結束程式及介面進入點結構 (CMQEC 包括 CMQXC 及 CMQC) 的函數原型、資料類型及具名常數。

若要改進應用程式的可攜性，請在 `#include` 前置處理器指引上以小寫形式撰寫標頭檔的名稱:

```
#include "cmqec.h"
```

函數

您不需要在每次呼叫函數時指定位址所傳遞的所有參數。

- 依值傳遞 僅輸入 且類型為 MQHCONN、MQHOBJ 或 MQLONG 的參數。
- 依位址傳遞所有其他參數。

如果不需要特定參數，請使用空值指標作為函數呼叫上的參數，以取代參數資料的位址。 可以在呼叫說明中識別的參數。

未傳回任何參數作為函數的值; 在 C 術語中，這表示所有函數都會傳回 `void`。

函數的屬性由 MQENTRY 巨集變數定義; 此巨集變數的值視環境而定。

具有未定義資料類型的參數

MQGET、MQPUT 及 MQPUT1 函數上的 *Buffer* 參數具有未定義的資料類型。 此參數用來傳送及接收應用程式的訊息資料。

此排序的參數在 C 範例中顯示為 MQBYTE 的陣列。 您可以用這種方式來宣告參數，但通常更方便將它們宣告為說明訊息中資料佈置的特定結構。 將實際函數參數宣告為 void 的指標，並將任何資料類型的位址指定為函數呼叫上的參數。

資料類型

使用 C `typedef` 陳述式定義所有資料類型。 對於每一種資料類型，也請定義對應的指標資料類型。 指標資料類型的名稱是基本或結構資料類型的名稱，字首為字母 P 以表示指標。 使用 MQPOINTER 巨集變數定義指標的屬性; 此巨集變數的值視環境而定。 下列說明如何宣告指標資料類型:

```
#define MQPOINTER *                /* depends on environment */
...
typedef MQLONG MQPOINTER PMQLONG;  /* pointer to MQLONG */
typedef MQMD   MQPOINTER PMQMD;    /* pointer to MQMD */
```

操作二進位字串

將二進位資料的字串宣告為其中一個 MQBYTEn 資料類型。

每當您複製、比較或設定此類型的欄位時，請使用 C 函數 memcpy、 memcmp 或 memset; 例如:

```
#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));
```

請勿使用字串函數 strcpy、 strcmp、 strncpy 或 strncmp，因為對於以 MQBYTEn 資料類型宣告的資料，這些函數無法正確運作。

操作字串

當佇列管理程式將字元資料傳回應用程式時，佇列管理程式一律會以空白填補字元資料至欄位的定義長度; 佇列管理程式 不會 傳回以空值結尾的字串。

因此，在複製、比較或連結這類字串時，請使用字串函數 strncpy、 strncmp 或 strncat。

請勿使用需要以空值終止字串的字串函數 (strcpy、 strcmp、 strcat)。 此外，請勿使用函數 strlen 來決定字串的長度; 請改用 sizeof 函數來決定欄位的長度。

結構的起始值

標頭檔會定義各種巨集變數，當您宣告這些結構的實例時，可用來提供 MQ 結構的起始值。

這些巨集變數的名稱格式為 MQxxx_DEFAULT ，其中 MQxxx 代表結構的名稱。 它們以下列方式使用:

```
MQMD   MyMsgDesc = {MQMD_DEFAULT};
MQPMO  MyPutOpts = {MQPMO_DEFAULT};
```

對於部分字元欄位 (例如，在大部分結構中出現的 *StrucId* 欄位，或在 MQMD 中出現的 *Format* 欄位) ，MQI 定義有效的特定值。 針對每一個有效值，提供 兩個 巨集變數:

- 一個巨集變數將值定義為具有長度 (不包括隱含空值相符項) 的字串，完全符合欄位定義的長度。 例如，對於 MQMD 中的 *Format* 欄位，會提供下列巨集變數 (¬ 代表空白字元):

  ```
  #define MQFMT_STRING "MQSTR¬¬¬"
  ```

  將此表單與 memcpy 及 memcmp 函數搭配使用。
- 另一個巨集變數將值定義為字元陣列; 此巨集變數的名稱是字串形式的名稱，字尾為 _ARRAY。 例如:

  ```
  #define MQFMT_STRING_ARRAY 'M','Q','S','T','R','¬','¬','¬'
  ```

  當您使用不同於 MQMD_DEFAULT 巨集變數所提供的值來宣告結構實例時，請使用此表單來起始設定欄位。 (不一定需要這樣做; 在某些環境中，您可以在這兩種情況下使用值的字串形式。 不過，您可以將陣列表單用於宣告，因為這是與 C++ 程式設計語言相容的必要項目。)

動態結構的起始值

當需要可變數目的結構實例時，通常會在使用 calloc 或 malloc 函數動態取得的主儲存體中建立實例。 若要起始設定此類結構中的欄位，請考量下列技術:

1. 使用適當的 MQxxx_DEFAULT 巨集變數來宣告結構的實例，以起始設定結構。 此實例會變成其他實例的模型:

   ```
   MQMD Model = {MQMD_DEFAULT};  /* declare model instance */
   ```

   static 或 auto 關鍵字可以在宣告上編碼，以便視需要提供模型實例靜態或動態生命期限。

2. 使用 calloc 或 malloc 函數來取得結構動態實例的儲存體:

```
PMQMD Instance;
Instance = malloc(sizeof(MQMD));  /* get storage for dynamic instance */
```

3. 使用 memcpy 功能，將模型實例複製到動態實例:

```
memcpy(Instance,&Model,sizeof(MQMD));  /* initialize dynamic instance */
```

從 *C++* 使用

對於 C++ 程式設計語言，標頭檔包含下列其他陳述式，只有在您使用 C++ 編譯器時才會包含這些陳述式:

```
#ifdef __cplusplus
  extern "C" {
#endif

/* rest of header file */

#ifdef __cplusplus
  }
#endif
```

符號慣例

此資訊顯示如何呼叫函數及宣告參數。

在某些情況下，參數是大小不是固定的陣列。 對於這些，會使用小寫 n 來代表數值常數。 當您撰寫該參數的宣告時，請將 n 取代為所需的數值。

COBOL 程式設計

本節包含的資訊可協助您使用 COBOL 程式設計語言中的 MQI。

COPY 檔案

提供各種 COPY 檔案來協助您撰寫使用 MQI 的 COBOL 應用程式。 有兩個包含已命名常數的檔案，以及每一個結構的兩個檔案。

每一個結構都以兩種形式提供: 具有起始值的表單，以及不含下列項目的表單:

- 在 COBOL 程式的 WORKING-STORAGE SECTION 中使用具有起始值的結構; 它們包含在 COPY 檔案中，名稱字尾為字母 V (代表值)。
- 在 COBOL 程式的 LINKAGE SECTION 中使用不含起始值的結構; 它們包含在 COPY 檔案中，名稱以字母 L 為字尾 (適用於鏈結)。

COPY 檔案在 第 237 頁的表 470 中進行彙總。 並非所有列出的檔案都可在所有環境中使用。

表 *470: COBOL COPY* 檔案

檔案 (含起始值)	**檔案 (不含起始值)**	**內容**
CMQAIRV	CMQAIRL	鑑別資訊記錄
CMQBOV	CMQBOL	開始選項結構
CMQCIHV	CMQCIHL	CICS 資訊標頭結構
CMQCNOV	CMQCNOL	連接選項結構
CMQDHV	CMQDHL	配送標頭結構
CMQDLHV	CMQDLHL	無法傳送的郵件標頭結構
CMQDXPV	CMQDXPL	資料轉換結束程式參數結構
CMQGMOV	CMQGMOL	取得訊息選項結構

表 *470: COBOL COPY* 檔案 *(*繼續*)*		
檔案 (含起始值)	**檔案 (不含起始值)**	**內容**
CMQIIHV	CMQIIHL	IMS 資訊標頭結構
CMQMDV	CMQMDL	訊息描述子結構
CMQMDEV	CMQMDEL	訊息描述子延伸結構
CMQMD1V	CMQMD1L	訊息描述子結構第 1 版
CMQODV	CMQODL	物件描述子結構
CMQORV	CMQORL	物件記錄結構
CMQPMOV	CMQPMOL	放置訊息選項結構
CMQRFHV	CMQRFHL	規則和格式化標頭結構
CMQRFH2V	CMQRFH2L	規則和格式化標頭結構第 2 版
CMQRMHV	CMQRMHL	參照訊息標頭結構
CMQRRV	CMQRRL	回應記錄結構
CMQSCOV	CMQSCOL	SSL 配置選項
CMQTMV	CMQTML	觸發訊息結構
CMQTMCV	CMQTMCL	觸發訊息結構 (字元格式)
CMQTMC2V	CMQTMC2L	觸發訊息結構 (字元格式) 第 2 版
CMQWIHV	CMQWIHL	工作資訊標頭結構
CMQXQHV	CMQXQHL	傳輸佇列標頭結構
CMQV	-	主要 MQI 的已命名常數
CMQXV	-	資料轉換結束程式的具名常數
CMQMD2V	CMQMD2L	訊息描述子結構第 2 版

結構

在 COPY 檔案中，每一個結構宣告都以 level-10 項目開始; 這可讓您透過編碼 level-01 宣告，然後使用 COPY 陳述式在結構宣告的其餘部分中進行複製，來宣告結構的數個實例。 若要參照適當的實例，請使用 IN 關鍵字:

```
* 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.
```

在適當的界限上對齊結構。 如果您使用 COPY 陳述式，在不是 level-01 項目的項目之後併入結構，請確保結構從 level-01 項目開頭的適當偏移開始。 大部分 MQI 結構都需要 4 位元組對齊; 但 MQCNO、MQOD 及 MQPMO 例外，它們在 IBM i 上需要 16 位元組對齊。

在此區段中，會顯示結構中的欄位名稱，不含字首。 在 COBOL 中，欄位名稱以結構名稱為字首，後面接著連字號。 不過，如果結構名稱以數字結尾，指出結構是原始結構的第二個或更新版本，則字首會省略數字。 COBOL 中的欄位名稱以大寫顯示 (雖然必要時可以使用小寫或大小寫混合格式)。 例如， 第 363 頁的『MQMD-訊息描述子』 中說明的 *MsgType* 欄位在 COBOL 中變成 MQMD-MSGTYPE。

V 字尾結構會以所有欄位的起始值來宣告; 您只需要設定您想要的值不同於所提供起始值的那些欄位。

指標
部分結構需要處理可能與結構不連續的選用資料，例如 MQOD 結構所處理的 MQOR 及 MQRR 記錄。

為了處理此選用資料，結構包含以指標資料類型宣告的欄位。 不過， COBOL 並不支援所有環境中的指標資料類型。 因此，也可以使用包含從結構開始的資料偏移的欄位來處理選用資料。

如果您想要在環境之間連接應用程式，請確定指標資料類型是否在所有預期的環境中都可用。 如果不是，應用程式必須使用偏移欄位而非指標欄位來處理選用資料。

在不支援指標的環境中，請將指標欄位宣告為適當長度的位元組字串，起始值為全空值位元組字串。 如果您使用偏移欄位，請勿變更此起始值。

具名常數

在此區段中，會顯示常數的名稱，其中包含底線字元 (_) 作為名稱的一部分。 在 COBOL 中，使用連字號字元 (-) 來取代底線。

具有字串值的常數會使用單引號作為字串定界字元 (')。 在某些環境中，您可能必須指定適當的編譯器選項，讓編譯器接受單引號作為字串定界字元，以取代雙引號。

在 COPY 檔案中已命名的常數宣告為 level-10 項目。 如果要使用常數，請明確宣告 level-01 項目，然後使用 COPY 陳述式來複製常數的宣告:

```
* Declare a structure to hold the constants
 01  MY-MQ-CONSTANTS.
     COPY CMQV.
```

前述方法會導致常數佔用程式中的儲存體，即使未參照它們也一樣。 如果您在相同執行單元內的許多個別程式中併入常數，則會存在多個常數副本，不必要地耗用主儲存體。 請使用下列其中一項技術來避免此效果:

- 將 GLOBAL 子句新增至 level-01 宣告:

  ```
  * Declare a global structure to hold the constants
   01  MY-MQ-CONSTANTS GLOBAL.
       COPY CMQV.
  ```

  這會導致在執行單元內只配置一組常數的儲存體。 不過，執行單元內的任何程式都可以參照常數，而不只是包含 level-01 宣告的程式。

  註: 並非所有環境都支援 GLOBAL 子句。
- 請只將該程式所參照的常數手動複製到每一個程式中。 請勿使用 COPY 陳述式將所有常數複製到程式中。

符號慣例

本節中的後幾個主題顯示如何呼叫呼叫及宣告參數。 在某些情況下，參數是其大小不是固定的表格或字串。 對於這些，會使用小寫 n 來代表數值常數。 當您撰寫該參數的宣告時，請將 n 取代為所需的數值。

System/390 組譯器程式設計

本節包含的資訊可協助您使用來自 System/390 Assembler 程式設計語言的 MQI。

巨集
提供各種巨集來協助您撰寫使用 MQI 的組譯器應用程式。

已命名常數有兩個巨集，每個結構有一個巨集。 這些檔案在 第 239 頁的表 471 中彙總。

表 *471:* 組譯器巨集

檔案	內容
CMQA	主要 MQI 的具名常數 (相等)
CMQCIHA	CICS 資訊標頭結構
CMQCNOA	連接選項結構

表 *471:* 組譯器巨集 *(*繼續*)*	
檔案	**內容**
CMQDLHA	無法傳送的郵件標頭結構
CMQDXPA	資料轉換結束程式參數結構
CMQGMOA	取得訊息選項結構
CMQIIHA	IMS 資訊標頭結構
CMQMDA	訊息描述子結構
CMQMDEA	訊息描述子延伸結構
CMQODA	物件描述子結構
CMQPMOA	放置訊息選項結構
CMQRFHA	規則和格式化標頭結構
CMQRFH2A	規則和格式化標頭結構第 2 版
CMQRMHA	參照訊息標頭結構
CMQTMA	觸發訊息結構
CMQTMC2A	觸發訊息結構 (字元格式) 第 2 版
CMQVERA	結構版本控制
CMQWIHA	工作資訊標頭結構
CMQXA	資料轉換結束程式的具名常數
CMQXPA	API 交互結束程式參數結構
CMQXQHA	傳輸佇列標頭結構

結構

結構由巨集產生，巨集具有各種參數來控制巨集的動作。 下列各節說明這些參數。

不時引進新版本的 MQ 結構。 新版本中的其他欄位可能會導致先前小於 256 個位元組的結構變成大於 256 個位元組。 因此，請撰寫預期複製 MQ 結構或將 MQ 結構設為空值的組譯器指示，以正確使用可能大於 256 個位元組的結構。 或者，搭配使用 DCLVER 巨集參數或 CMQVERA 巨集與 VERSION 參數，以宣告特定版本的結構。

指定結構的名稱

為了宣告結構的多個實例，巨集會以使用者指定的字串和底線作為結構中每一個欄位名稱的字首。

使用的字串是在呼叫巨集時指定的標籤。 如果未指定標籤，則會使用結構名稱來建構字首:

```
* Declare two object descriptors
         CMQODA  ,        Prefix used="MQOD_" (the default)
MY_MQOD  CMQODA  ,        Prefix used="MY_MQOD_"
```

本節中顯示的結構宣告使用預設字首。

指定結構的形式

結構宣告可以由下列兩種形式之一的巨集產生，由 DSECT 參數控制:

DSECT = YES

組譯器 DSECT 指示用來啟動新的資料區段; 結構定義緊跟在 DSECT 陳述式後面。 巨集呼叫上的標籤會作為資料區段的名稱; 如果未指定標籤，則會使用結構的名稱。

DSECT = NO
組譯器 DC 指示用來定義常式中現行位置的結構。 欄位會以值來起始設定，您可以在巨集呼叫上撰寫相關參數來指定這些值。 在巨集呼叫上未指定任何值的欄位會以預設值起始設定。

指定的值必須是大寫。 如果未指定 DSECT 參數，則會採用 DSECT=NO 。

控制結構的版本
依預設，巨集一律會宣告每一個結構的最新版本。

雖然您可以使用 VERSION 巨集參數來指定結構中 *Version* 欄位的值，但該參數會定義 *Version* 欄位的起始值，且不會控制實際宣告的結構版本。 若要控制所宣告結構的版本，請使用 DCLVER 參數:

DCLVER=CURRENT
宣告的版本是現行 (最新) 版本。

DCLVER=SPECIFIED
宣告的版本是 VERSION 參數指定的版本。 如果您省略 VERSION 參數，則預設值為第 1 版。

如果您指定 VERSION 參數，則值必須是自行定義的數值常數，或所需版本的具名常數 (例如，MQCNO_VERSION_3)。 如果您指定某個其他值，則即使 VERSION 的值解析為有效值，也會將結構宣告為已指定 DCLVER=CURRENT 。

指定的值必須是大寫。 如果您省略 DCLVER 參數，則會從 MQDCLVER 廣域巨集變數取得所使用的值。 您可以使用 CMQVERA 巨集來設定此變數。

宣告一個結構內嵌在另一個結構內

若要將一個結構宣告為另一個結構的元件，請使用 NESTED 參數:

NESTED=YES
結構宣告在另一個宣告內形成巢狀。

NESTED=NO
結構宣告未巢狀在另一個宣告內。

指定的值必須是大寫。 如果省略 NESTED 參數，則會採用 NESTED=NO 。

指定欄位的起始值
指定要用來起始設定結構中欄位的值，方法是將該欄位的名稱 (不含字首) 編碼為巨集呼叫上的參數，並伴隨必要的值。

例如，若要宣告訊息描述子結構，且 *MsgType* 欄位以 MQMT_REQUEST 起始設定，而 *ReplyToQ* 欄位以字串 "MY_REPLY_TO_QUEUE" 起始設定，請使用下列指令:

```
MY_MQMD  CMQMDA  MSGTYPE=MQMT_REQUEST,                                X
               REPLYTOQ=MY_REPLY_TO_QUEUE
```

如果您在巨集呼叫中指定已命名常數 (等於) 作為值，請使用 CMQA 巨集來定義已命名常數。 請勿以單引號括住字串值。

控制清單

使用 LIST 參數來控制結構宣告在組譯器清單中的外觀:

LIST = YES
結構宣告會出現在組譯器清單中。

LIST = NO
結構宣告不會出現在組譯器清單中。

指定的值必須是大寫。 如果省略 LIST 參數，則會採用 LIST=NO 。

CMQVERA 巨集

此巨集可讓您設定要用於結構巨集上 DCLVER 參數的預設值。 只有在呼叫結構巨集時省略 DCLVER 參數時，結構巨集才會使用 CMQVERA 指定的值。 預設值是透過使用 DCLVER 參數編碼 CMQVERA 巨集來設定:

DCLVER=CURRENT
預設版本設為現行 (最新) 版本。

DCLVER=SPECIFIED
預設版本會設為 VERSION 參數所指定的版本。

您必須指定 DCLVER 參數，且值必須是大寫。 CMQVERA 設定的值會保留預設值，直到下次呼叫 CMQVERA 或組件結束為止。 如果您省略 CMQVERA ，預設值為 DCLVER=CURRENT。

符號慣例

後面各節顯示如何呼叫呼叫及宣告參數。 在某些情況下，參數是陣列或字串，其大小不是固定的，會使用小寫 n 來代表數值常數。 當您撰寫該參數的宣告時，請將 n 取代為所需的數值。

MQAIR-鑑別資訊記錄

MQAIR 結構代表鑑別資訊記錄。

下表彙總結構中的欄位。

表 *472: MQAIR* 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
版本	結構版本號碼	版本
AuthInfoType	鑑別資訊的類型	AuthInfoType
AuthInfoConnName	LDAP CRL 伺服器的連線名稱	AuthInfoConnName
LDAPUserNamePtr	LDAP 使用者名稱的位址	LDAPUserNamePtr
LDAPUserName 偏移	從 MQSCO 開始的 LDAP 使用者名稱的偏移	LDAPUserName 偏移
LDAPUserName 長度	LDAP 使用者名稱的長度	LDAPUserName 長度
LDAPPassword	存取 LDAP 伺服器的密碼	LDAPPassword
註: 如果 版小於 MQAIR_VERSION_2，則會忽略其餘欄位。		
OCSPResponderURL	可聯絡 OCSP 回應端的 URL	OCSPResponderURL

MQAIR 概觀

MQAIR 結構可讓作為 WebSphere MQ MQI 用戶端執行的應用程式，指定要用於用戶端連線之鑑別器的相關資訊。 此結構是 MQCONNX 呼叫的輸入參數。

可用性: AIX、 HP-UX、Solaris Linux 及 Windows 用戶端。

字集及編碼:MQAIR 中的資料必須是本端佇列管理程式的字集及編碼; 這些是由 **CodedCharSetId** 佇列管理程式屬性及 MQENC_NATIVE 所提供。

MQAIR 的欄位

MQAIR 結構包含下列欄位; 這些欄位按 **字母順序**進行說明:

AuthInfoConnName (MQCHAR264)
這是執行 LDAP 伺服器之主機的主機名稱或網址。 後面可以接著以括弧括住的選用埠號。 預設埠號為 389。

如果該值短於欄位長度，請以空字元終止該值，或以空白填補該值至欄位長度。 如果值無效，則呼叫會失敗，原因碼為 MQRC_AUTH_INFO_CONN_NAME_ERROR。

這是輸入欄位。 此欄位的長度由 MQ_AUTH_INFO_CONN_NAME_LENGTH 提供。 此欄位的起始值是 C 中的空字串，以及其他程式設計語言中的空白字元。

AuthInfo 類型 *(MQLONG)*
這是記錄中包含的鑑別資訊類型。

此值可以是下列兩個參數之一:

MQAIT_CRL_LDAP
使用 LDAP 伺服器進行憑證撤銷檢查。

MQAIT_OCSP
使用 OCSP 進行憑證撤銷檢查。

如果值無效，則呼叫會失敗，原因碼為 MQRC_AUTH_INFO_TYPE_ERROR。

這是輸入欄位。 此欄位的起始值是 MQAIT_CRL_LDAP。

LDAPPassword (MQCHAR32)
這是存取 LDAP CRL 伺服器所需的密碼。 如果該值短於欄位長度，請以空字元終止該值，或以空白填補該值至欄位長度。

如果 LDAP 伺服器不需要密碼，或您省略 LDAP 使用者名稱，則 *LDAPPassword* 必須是空值或空白。 如果您省略 LDAP 使用者名稱，且 *LDAPPassword* 不是空值或空白，則呼叫會失敗，原因碼為 MQRC_LDAP_PASSWORD_ERROR。

這是輸入欄位。 此欄位的長度由 MQ_LDAP_PASSWORD_LENGTH 提供。 此欄位的起始值是 C 中的空字串，以及其他程式設計語言中的空白字元。

LDAPUserName 長度 *(MQLONG)*

這是 *LDAPUserNamePtr* 或 *LDAPUserNameOffset* 欄位所定址 LDAP 使用者名稱的長度 (以位元組為單位)。 值必須在 0 到 MQ_DISTINGUISHED_NAME_LENGTH 的範圍內。 如果值無效，則呼叫會失敗，原因碼為 MQRC_LDAP_USER_NAME_LENGTH_ERR。

如果涉及的 LDAP 伺服器不需要使用者名稱，請將此欄位設為零。

這是輸入欄位。 此欄位的起始值為 0。

LDAPUserName 偏移 *(MQLONG)*
這是 LDAP 使用者名稱從 MQAIR 結構開始的偏移 (以位元組為單位)。

偏移可以是正數或負數。 如果 *LDAPUserNameLength* 為零，則會忽略該欄位。

您可以使用 *LDAPUserNamePtr* 或 *LDAPUserNameOffset* 來指定 LDAP 使用者名稱，但不能同時指定兩者; 如需詳細資料，請參閱 *LDAPUserNamePtr* 欄位的說明。

這是輸入欄位。 此欄位的起始值為 0。

LDAPUserNamePtr (PMQCHAR)
這是 LDAP 使用者名稱。

它包含嘗試存取 LDAP CRL 伺服器之使用者的「識別名稱」。 如果值短於 *LDAPUserNameLength* 指定的長度，請以空值字元來終止值，或以空白填補值，使其長度為 *LDAPUserNameLength*。 如果 *LDAPUserNameLength* 為零，則會忽略該欄位。

您可以使用下列兩種方式之一來提供 LDAP 使用者名稱:

- 使用指標欄位 *LDAPUserNamePtr*

  在此情況下，應用程式可以宣告與 MQAIR 結構不同的字串，並將 *LDAPUserNamePtr* 設為字串的位址。

  考量將 *LDAPUserNamePtr* 用於以可攜至不同環境 (例如， C 程式設計語言) 的方式支援指標資料類型的程式設計語言。

- 使用偏移欄位 *LDAPUserNameOffset*

  在此情況下，應用程式必須宣告包含 MQSCO 結構的複合結構，後面接著 MQAIR 記錄的陣列，後面接著 LDAP 使用者名稱字串，並將 *LDAPUserNameOffset* 設為從 MQAIR 結構開始算起適當名稱字串的偏移。 請確定此值是正確的，且具有可在 MQLONG 內容納的值 (最嚴格的程式設計語言是 COBOL ，其有效範圍是 -999 999 999 至 +999 999 999)。

考量將 *LDAPUserNameOffset* 用於不支援指標資料類型的程式設計語言，或以可能無法移植到不同環境 (例如 COBOL 程式設計語言) 的方式來實作指標資料類型的程式設計語言。

不論選擇何種技術，請只使用 *LDAPUserNamePtr* 和 *LDAPUserNameOffset* 之一; 如果兩者都不是零，則呼叫會失敗，原因碼為 MQRC_LDAP_USER_NAME_ERROR。

這是輸入欄位。 在那些支援指標的程式設計語言中，此欄位的起始值是空值指標，否則是全空值位元組字串。

註: 在程式設計語言不支援指標資料類型的平台上，此欄位宣告為適當長度的位元組字串。

OCSPResponderURL (MQCHAR256)
對於代表 OCSP 回應端連線詳細資料的 MQAIR 結構，此欄位包含可聯絡回應端的 URL。

此欄位的值是 HTTP URL。 此欄位優先於 AuthorityInfoAccess (AIA) 憑證延伸中的 URL。

除非下列陳述式皆為 true ，否則會忽略此值:

- MQAIR 結構是第 2 版或更新版本 (「版本」欄位設為 MQAIR_VERSION_2 或更新版本)。
- AuthInfo 類型欄位設為 MQAIT_OCSP。

如果欄位未包含正確格式的 HTTP URL (且未被忽略) ，則 MQCONNX 呼叫會失敗，原因碼為 MQRC_OCSP_URL_ERROR。

此欄位區分大小寫。 它必須以小寫字串 http:// 開頭。 視 OCSP 伺服器實作而定， URL 的其餘部分可能區分大小寫。

此欄位不會進行資料轉換。

StrucId (MQCHAR4)

值必須為:

MQAIR_STRUC_ID
鑑別資訊記錄的 ID。

對於 C 程式設計語言，也會定義常數 MQAIR_STRUC_ID_ARRAY; 此值與 MQAIR_STRUC_ID 相同，但卻是字元陣列而非字串。

這一律是輸入欄位。 此欄位的起始值為 MQAIR_STRUC_ID。

版本 *(MQLONG)*
MQAIR 結構的版本號碼。

此值必須是下列其中一個:

MQAIR_VERSION_1
Version-1 鑑別資訊記錄。

MQAIR_VERSION_2
Version-2 鑑別資訊記錄。

下列常數指定現行版本的版本號碼:

MQAIR_CURRENT_VERSION
鑑別資訊記錄的現行版本。

這一律是輸入欄位。 此欄位的起始值為 MQAIR_VERSION_1。

MQAIR 的起始值及語言宣告

表 *473: MQAIR* 中欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQAIR_STRUC_ID	'AIR¬'
版本	MQAIR_VERSION_1	1

表 *473: MQAIR* 中欄位的起始值 *(*繼續*)*

欄位名稱	常數名稱	常數值
AuthInfoType	MQAIT_CRL_LDAP	1
AuthInfoConnName	無	空字串或空白
LDAPUserNamePtr	無	空值指標或空值位元組
LDAPUserName 偏移	無	0
LDAPUserName 長度	無	0
LDAPPassword	無	空字串或空白
OCSPResponderURL	無	空字串或空白

附註:

1. 符號 ¬ 代表單一空白字元。
2. 在 C 程式設計語言中，巨集變數 MQAIR_DEFAULT 包含上面列出的值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQAIR MyAIR = {MQAIR_DEFAULT};
```

C 宣告

```
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 */

};
```

COBOL 宣告

```
**   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).
```

Visual Basic 宣告

```
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
```

MQBMHO-緩衝區至訊息控點選項

下表彙總結構中的欄位。 MQBMHO 結構-緩衝區至訊息處理選項

表 *474: MQBMHO* 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
Options	控制 MQBMHO 動作的選項	選項

MQBMHO 概觀

可用性: 全部。 緩衝區至訊息控點選項結構-概觀

目的:MQBMHO 結構可讓應用程式指定選項，以控制如何從緩衝區產生訊息控點。 此結構是 MQBUFMH 呼叫上的輸入參數。

字集及編碼:MQBMHO 中的資料必須在應用程式的字集及應用程式的編碼中 (MQENC_NATIVE)。

MQBMHO 的欄位

緩衝區至訊息控點選項結構-欄位

MQBMHO 結構包含下列欄位; 這些欄位按 **字母順序**說明:

選項 *(MQLONG)*
緩衝區至訊息控點結構-選項欄位

值可以為:

MQBHO_DELETE_PROPERTIES
新增至訊息控點的內容會從緩衝區中刪除。 如果呼叫失敗，則不會刪除任何內容。

預設選項: 如果您不需要說明的選項，請使用下列選項:

MQBMHO_NONE
未指定選項。

這一律是輸入欄位。 此欄位的起始值是 MQBHO_DELETE_PROPERTIES。

StrucId (MQCHAR4)
緩衝區至訊息控點結構- StrucId 欄位

這是結構 ID。 值必須為:

MQBMHO_STRUC_ID
緩衝區至訊息控點結構的 ID。

對於 C 程式設計語言，也會定義常數 MQBMHO_STRUC_ID_ARRAY; 其值與 MQBMHO_STRUC_ID 相同，但卻是字元陣列而非字串。

這一律是輸入欄位。 此欄位的起始值是 MQBMHO_STRUC_ID。

版本 *(MQLONG)*
緩衝區至訊息控點結構-版本欄位

這是結構版本號碼。 值必須為:

MQBMHO_VERSION_1
緩衝區至訊息控點結構的版本號碼。

下列常數指定現行版本的版本號碼:

MQBMHO_CURRENT_VERSION
緩衝區至訊息控點結構的現行版本。

這一律是輸入欄位。 此欄位的起始值為 MQBMHO_VERSION_1。

MQBMHO 的起始值和語言宣告
緩衝區至訊息控點結構-起始值

表 *475: MQBMHO* 中欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQBMHO_STRUC_ID	'BMHO'
Version	MQBMHO_VERSION_1	1
Options	MQBMHO_NONE	0

附註:

1. 在 C 程式設計語言中，巨集變數 MQBMHO_DEFAULT 包含上述值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQBMHO MyBMHO = {MQBMHO_DEFAULT};
```

C 宣告
緩衝區至訊息控點結構-C 語言宣告

```
typedef struct tagMQBMHO MQBMHO;
struct tagMQBMHO {
  MQCHAR4  StrucId;        /* Structure identifier */
  MQLONG   Version;        /* Structure version number */
  MQLONG   Options;        /* Options that control the action of
                              MQBUFMH */
};
```

COBOL 宣告
緩衝區至訊息控點結構-COBOL 語言宣告

```
**   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.
```

PL/I 宣告
緩衝區至訊息控點結構-PL/I 語言宣告

```
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 */
```

High Level Assembler 宣告
緩衝區至訊息控點結構-組譯器語言宣告

```
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)
```

MQBO-開始選項

下表彙總結構中的欄位。

表 *476: MQBO* 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
Options	控制 MQBEGIN 動作的選項	選項

MQBO 概觀

可用性: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows; 不適用於 WebSphere MQ MQI 用戶端。

目的:MQBO 結構可讓應用程式指定與建立工作單元相關的選項。 結構是 MQBEGIN 呼叫上的輸入/輸出參數。

字集及編碼:MQBO 中的資料必須採用 *CodedCharSetId* 佇列管理程式屬性所提供的字集，以及 MQENC_NATIVE 所提供本端佇列管理程式的編碼。 不過，如果應用程式以 MQ MQI 用戶端身分執行，則該結構必須採用用戶端的字集及編碼。

MQBO 的欄位

MQBO 結構包含下列欄位; 這些欄位按 **字母順序**說明:

選項 *(MQLONG)*
此欄位一律是輸入欄位。 其起始值為 MQBO_NONE。

值必須為:

MQBO_NONE
未指定選項。

StrucId (MQCHAR4)
此欄位一律是輸入欄位。 其起始值為 MQBO_STRUC_ID。

值必須為:

MQBO_STRUC_ID
begin-options 結構的 ID。

對於 C 程式設計語言，也會定義常數 MQBO_STRUC_ID_ARRAY; 此值與 MQBO_STRUC_ID 相同，但卻是字元陣列而非字串。

版本 *(MQLONG)*
此欄位一律是輸入欄位。 其起始值為 MQBO_VERSION_1。

值必須為:

MQBO_VERSION_1
begin-options 結構的版本號碼。

下列常數指定現行版本的版本號碼:

MQBO_CURRENT_VERSION
begin-options 結構的現行版本。

MQBO 的起始值和語言宣告

表 *477: MQBO* 中 *MQBO* 的欄位起始值

欄位名稱	常數名稱	常數值
StrucId	MQBO_STRUC_ID	'BO¬¬'
Version	MQBO_VERSION_1	1
Options	MQBO_NONE	0

附註:

1. 符號 ¬ 代表單一空白字元。
2. 在 C 程式設計語言中，巨集變數 MQBO_DEFAULT 包含上述值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQBO MyBO = {MQBO_DEFAULT};
```

C 宣告

```
typedef struct tagMQBO MQBO;
struct tagMQBO {
  MQCHAR4  StrucId;  /* Structure identifier */
  MQLONG   Version;  /* Structure version number */
  MQLONG   Options;  /* Options that control the action of MQBEGIN */
};
```

COBOL 宣告

```
**   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.
```

PL/I 宣告

```
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 */
```

Visual Basic 宣告

```
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
```

MQCBC-回呼環境定義

下表彙總結構中的欄位。 說明回呼常式的結構。

表 *478: MQCBC* 中的欄位		
欄位	**說明**	**主題**
StrucID	結構 ID	StrucID
Version	結構版本號碼	版本
CallType	呼叫函數的原因	CallType
Hobj	物件控點	HOBJ
CallbackArea	要使用的回呼函數欄位	CallbackArea
ConnectionArea	要使用的回呼函數欄位	ConnectionArea
CompCode	完成碼	CompCode
Reason	原因碼	原因
State	指示現行消費者的狀態	狀態
DataLength	訊息長度	DataLength
BufferLength	訊息緩衝區的長度 (以位元組為單位)	BufferLength
Flags	一般旗標	旗標
註: 如果「版本」小於 MQCBC_VERSION_2，則會忽略剩餘欄位		
ReconnectDelay	嘗試重新連線之前的毫秒數	ReconnectDelay

MQCBC 概觀

可用性: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows、 z/OS，以及連接至這些系統的 WebSphere MQ MQI 用戶端。

目的:MQCBC 結構用來指定傳遞給回呼函數的環境定義資訊。

此結構是呼叫訊息消費者常式時的輸入/輸出參數。

版本:MQCBC 的現行版本是 MQCBC_VERSION_2。

字集及編碼:MQCBC 中的資料必須採用 *CodedCharSetId* 佇列管理程式屬性所提供的字集，以及 MQENC_NATIVE 所提供本端佇列管理程式的編碼。 不過，如果應用程式以 MQ MQI 用戶端身分執行，則結構會在用戶端的字集及編碼中。

MQCBC 的欄位

MQCBC 結構的欄位英文字母清單。

MQCBC 結構包含下列欄位; 這些欄位按字母順序說明:

BufferLength (MQLONG)
此欄位是已傳遞給此函數的訊息緩衝區長度 (以位元組為單位)。

緩衝區可以大於針對消費者定義的 MaxMsg 長度值以及 MQGMO 中的 ReturnedLength 值。

實際訊息長度在 DataLength 欄位中提供。

在回呼函數期間，應用程式可以基於自己的目的使用整個緩衝區。

這是訊息消費者函數的輸入欄位; 它與異常狀況處理程式函數無關。

CallbackArea (MQPTR)
此欄位可供回呼函數使用。

佇列管理程式不會根據此欄位的內容來做出任何決策，且會從 MQCBD 結構中的 第 257 頁的『CallbackArea (MQPTR)』 欄位 (這是用來定義回呼函數的 MQCB 呼叫中的參數) 依原樣傳遞。

在呼叫 *HObj* 的回呼函數時，會保留對 *CallbackArea* 所做的變更。 此欄位不會與其他控點的回呼函數共用。

這是回呼函數的輸入/輸出欄位。 此欄位的起始值是空值指標或空值位元組。

CallType (MQLONG)
此欄位包含為何呼叫此函數的相關資訊; 定義如下。

訊息遞送呼叫類型: 這些呼叫類型包含訊息的相關資訊。 *DataLength* 和 *BufferLength* 參數適用於這些呼叫類型。

MQCBCT_MSG_REMOVED
已使用已從物件控點破壞性移除的訊息來呼叫訊息消費者函數。

如果 *CompCode* 的值為 MQCC_WARNING，則 *Reason* 欄位的值為 MQRC_TRUNCATED_MSG_ACCEPTED 或其中一個指出資料轉換問題的代碼。

MQCBCT_MSG_NOT_REMOVED
已使用尚未從物件控點破壞性移除的訊息來呼叫訊息消費者函數。 可以使用 *MsgToken* 從物件控點中破壞性地移除訊息。

訊息可能尚未移除，因為:

- MQGMO 選項要求瀏覽作業 MQGMO_BROWSE_ *
- 訊息大於可用的緩衝區，且 MQGMO 選項未指定 MQGMO_ACCEPT_TRUNCATED_MSG

如果 *CompCode* 的值為 MQCC_WARNING，則 *Reason* 欄位的值為 MQRC_TRUNCATED_MSG_FAILED 或其中一個指出資料轉換問題的代碼。

回呼控制項呼叫類型: 這些呼叫類型包含回呼控制項的相關資訊，但不包含訊息的詳細資料。 這些呼叫類型是使用 MQCBD 結構中的 選項 來要求。

DataLength 和 *BufferLength* 參數對這些呼叫類型無效。

MQCBCT_REGISTER_CALL
此呼叫類型的目的是容許回呼函數執行部分起始設定。

在登錄回呼之後，即使用 MQOP_REGISTER 的 *Operation* 欄位值從 MQCB 呼叫傳回時，會立即呼叫回呼函數。

此呼叫類型同時用於訊息消費者及事件處理程式。

如果要求，這是第一次呼叫回呼函數。

Reason 欄位的值是 MQRC_NONE。

MQCBCT_START_CALL
此呼叫類型的目的是容許回呼函數在啟動時執行部分設定，例如，恢復先前停止時已清除的資源。

當使用 MQOP_START 或 MQOP_START_WAIT 啟動連線時，會呼叫回呼函數。

如果回呼函數登錄在另一個回呼函數內，則會在回呼傳回時呼叫此呼叫類型。

此呼叫類型僅用於訊息消費者。

Reason 欄位的值是 MQRC_NONE。

MQCBCT_STOP_CALL
此呼叫類型的目的是容許回呼函數在停止一段時間時執行一些清理，例如清除在耗用訊息期間獲得的其他資源。

使用 MQOP_STOP 的 *Operation* 欄位值發出 MQCTL 呼叫時，會呼叫回呼函數。

此呼叫類型僅用於訊息消費者。

Reason 欄位的值已設定為指出停止的原因。

MQCBCT_DEREGISTER_CALL
此呼叫類型的目的是容許回呼函數在 consume 處理程序結束時執行最終清理。 當下列時，會呼叫回呼函數:

- 使用 MQOP_DEREGISTER 的 MQCB 呼叫來取消登錄回呼函數。
- 佇列已關閉，導致隱含取消登錄。 在此實例中，回呼函數會傳遞 MQHO_UNUSABLE_HOBJ 作為物件控點。
- MQDISC 呼叫完成-導致隱含關閉，因此取消登錄。 在此情況下，連線不會立即斷線，且尚未確定任何進行中的交易。

如果在回呼函數本身內採取其中任何動作，則會在回呼傳回之後呼叫該動作。

此呼叫類型同時用於訊息消費者及事件處理程式。

如果要求，這是回呼函數的最後一次呼叫。

Reason 欄位的值已設定為指出停止的原因。

MQCBCT_EVENT_CALL
事件處理程式函數

當佇列管理程式或連線停止或靜止時，已在沒有訊息的情況下呼叫事件處理程式函數。

此呼叫可用來對所有回呼函數採取適當的動作。

訊息消費者函數

當偵測到物件控點特有的錯誤 (*CompCode* = MQCC_FAILED) 時，已呼叫訊息消費者函數，但沒有訊息; 例如 *Reason* code = MQRC_GET_INHIBITED。

Reason 欄位的值設定為指出呼叫的原因。

MQCBCT_MC_EVENT_CALL

已針對多重播送事件呼叫事件處理程式函數; 事件處理程式會傳送 WebSphere MQ 多重播送事件，而非 'normal' WebSphere MQ 事件。

如需 MQCBCT_MC_EVENT_CALL 的相關資訊，請參閱 多重播送異常狀況報告 。

CompCode (MQLONG)
此欄位是完成碼。 它指出耗用訊息是否有任何問題。

此值是下列其中一個:

MQCC_OK
順利完成

MQCC_WARNING
警告 (局部完成)

MQCC_FAILED
通話失敗

這是輸入欄位。 此欄位的起始值是 MQCC_OK。

ConnectionArea (MQPTR)
此欄位可供回呼函數使用。

佇列管理程式不會根據此欄位的內容做出任何決策，而且會從 MQCTLO 結構中的 第 299 頁的『ConnectionArea (MQPTR)』 欄位傳遞它，該結構是用來控制回呼函數的 MQCTL 呼叫上的參數。

回呼函數對這個欄位所做的任何變更，都會在回呼函數的呼叫期間保留。 此區域可用來傳遞要由所有回呼函數共用的資訊。 與 *CallbackArea* 不同，此區域在連線控點的所有回呼之間是共用的。

這是輸入及輸出欄位。 此欄位的起始值是空值指標或空值位元組。

DataLength (MQLONG)
這是訊息中應用程式資料的長度 (以位元組為單位)。 如果值為零，則表示訊息不包含應用程式資料。

DataLength 欄位包含訊息的長度，但不一定是傳遞給消費者的訊息資料長度。 可能是訊息被截斷。 請使用 MQGMO 中的 ReturnedLength 欄位來判斷實際傳遞給消費者的資料量。

如果原因碼指出訊息已截斷，您可以使用 DataLength 欄位來判斷實際訊息的大小。 這可讓您判斷容納訊息資料所需的緩衝區大小，然後發出 MQCB 呼叫，以適當的值來更新 MaxMsg 長度 。

如果指定 MQGMO_CONVERT 選項，則轉換後的訊息可能大於 DataLength 所傳回的值。 在這類情況下，應用程式可能需要發出 MQCB 呼叫，將 MaxMsg 長度 更新為大於佇列管理程式針對 DataLength 傳回的值。

若要避免訊息截斷問題，請將 MaxMsg 長度指定為 MQCBD_FULL_MSG_LENGTH。 這會導致佇列管理程式在資料轉換之後配置完整訊息長度的緩衝區。 不過，請注意，即使指定此選項，仍可能無法使用足夠的儲存體來正確處理要求。 應用程式應該一律檢查傳回的原因碼。 例如，如果無法配置足夠的儲存體來轉換訊息，則會將訊息以未轉換的狀態傳回給應用程式。

這是訊息消費者函數的輸入欄位; 它與事件處理程式函數無關。

旗標 *(MQLONG)*
包含此消費者相關資訊的旗標。

下列是已定義的選項:

MQCBCF_READA_BUFFER_EMPTY
如果使用 MQCO_QUIESCE 選項的前一個 MQCLOSE 呼叫失敗，原因碼為 MQRC_READ_AHEAD_MSGS ，則會傳回此旗標。

此代碼指出正在傳回前次先讀訊息，且緩衝區現在是空的。 如果應用程式使用 MQCO_QUIESCE 選項發出另一個 MQCLOSE 呼叫，則它會成功。

請注意，應用程式不保證會收到已設定此旗標的訊息，因為先讀緩衝區中可能仍有不符合現行選取準則的訊息。 在此實例中，會以原因碼 MQRC_HOBJ_QUIESCED 來呼叫消費者函數。

如果先讀緩衝區完全空白，則會使用 MQCBCF_READA_BUFFER_EMPTY 旗標及原因碼 MQRC_HOBJ_QUIESCED_NO_MSGS 來呼叫消費者。

這是訊息消費者函數的輸入欄位; 它與事件處理程式函數無關。

Hobj (MQHOBJ)
這是訊息消費者呼叫的物件控點。

若為事件處理程式，此值為 MQHO_NONE

如果訊息尚未從佇列中移除，則應用程式可以使用此控點及「取得訊息選項」區塊中的訊息記號來取得訊息。

這一律是輸入欄位。 此欄位的起始值是 MQHO_UNUSABLE_HOBJ

原因 *(MQLONG)*
這是限定 *CompCode* 的原因碼。

這是輸入欄位。 此欄位的起始值為 MQRC_NONE。

狀態 *(MQLONG)*
指出現行消費者的狀態。 當非零原因碼傳遞至消費者函數時，此欄位對應用程式最有價值。

您可以使用此欄位來簡化應用程式設計，因為您不需要針對每一個原因碼撰寫行為的程式碼。

這是輸入欄位。 此欄位的起始值為 MQCS_NONE

狀態	佇列管理程式動作	常數值
MQCS_NONE 此原因碼代表沒有其他原因資訊的正常呼叫	無; 這是正常作業。	0
MQCS_SUSPENDED_TEMPORARY 這些原因碼代表暫時狀況。	會呼叫回呼常式來報告條件，然後暫停。 經過一段時間之後，系統可能會再次嘗試該作業，這可能會導致再次發生相同的狀況。	1
MQCS_SUSPENDED_USER_ACTION 這些原因碼代表回呼需要採取動作來解決條件的條件。	消費者已暫停，且會呼叫回呼常式來報告條件。 如果可能的話，回呼常式應該解決此狀況，並 RESUME 或關閉連線。	2
MQCS_SUSPENDED 這些原因碼代表阻止進一步訊息回呼的失敗。	佇列管理程式會自動暫停回呼函數。 如果回復回呼函數，則可能會再次收到相同的原因碼。	3
MQCS_STOPPED 這些原因碼代表訊息耗用的結束。	遞送至異常狀況處理程式及指定 MQCBDO_STOP_CALL 的回呼。 無法使用進一步的訊息。	4

StrucId (MQCHAR4)
此欄位中的值是結構 ID。

值必須為:

MQCBC_STRUC_ID
回呼環境定義結構的 ID。

對於 C 程式設計語言，也會定義常數 MQCBC_STRUC_ID_ARRAY; 此值與 MQCBC_STRUC_ID 相同，但卻是字元陣列而非字串。

這一律是輸入欄位。 此欄位的起始值是 MQCBC_STRUC_ID。

版本 *(MQLONG)*
此欄位中的值是結構版本號碼。

值必須為:

MQCBC_VERSION_1
Version-1 回呼環境定義結構。

下列常數指定現行版本的版本號碼:

MQCBC_CURRENT_VERSION
回呼環境定義結構的現行版本。

這一律是輸入欄位。 此欄位的起始值為 MQCBC_VERSION_1。

回呼函數一律會傳遞最新版本的結構。

ReconnectDelay (MQLONG)
ReconnectDelay 指出在嘗試重新連接之前佇列管理程式將等待的時間長度。 事件處理程式可以修改此欄位，以變更延遲或完全停止重新連線。

只有在回呼環境定義中 **原因** 欄位的值為 MQRC_RECONNECTING 時，才使用 ReconnectDelay 欄位。

在進入事件處理程式時， ReconnectDelay 的值是佇列管理程式在嘗試重新連線之前將等待的毫秒數。 第 255 頁的表 479 列出您可以設定的值，以從事件處理程式返回時修改佇列管理程式的行為。

表 *479: ReconnectDelay* 值		
姓名	**值**	**說明**
MQRD_NO_RECONNECT	-1	不再嘗試重新連線。 傳回錯誤給應用程式。
MQRD_NO_DELAY	0	請嘗試立即重新連接。
Milliseconds	>0	請等待此毫秒數，然後重試連線。

MQCBC 的起始值和語言宣告

回呼環境定義結構-起始值

MQCBC 結構沒有起始值。 結構會以參數形式傳遞至回呼常式。 佇列管理程式會起始設定結構; 應用程式絕不會起始設定它。

C 宣告

回呼環境定義結構-C 語言宣告

```
typedef struct tagMQCBC MQCBC;
struct tagMQCBC {
   MQCHAR4   StrucId;                /* Structure identifier */
   MQLONG    Version;                /* Structure version number */
   MQLONG    CallType;               /* Why Function was called */
   MQHOBJ    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 */
```

COBOL 宣告

```
** 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 **
```

PL/I 宣告

```
dcl
 1 MQCBC based,
  3 StrucId             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 */
```

High Level Assembler 宣告

```
    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)
```

MQCBD-回呼描述子

下表彙總結構中的欄位。 指定回呼函數的結構。

表 *480: MQCBD* 中的欄位

欄位	說明	主題
StrucID	結構 ID	StrucID
Version	結構版本號碼	版本
CallbackType	回呼函數的類型	CallbackType
Options	控制訊息耗用的選項	選項
Callback Area	要使用的回呼函數欄位	CallbackArea
CallbackFunction	是否以 API 呼叫方式呼叫函數	CallbackFunction
CallbackName	是否以動態鏈結程式來呼叫函數	CallbackName
MaxMsgLength	可讀取的最長訊息長度	MaxMsgLength

MQCBD 概觀

可用性: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows、 z/OS 及連接至這些系統的 WebSphere MQ MQI 用戶端。

目的:MQCBD 結構用來指定回呼函數，以及由佇列管理程式控制其使用的選項。

此結構是 MQCB 呼叫上的輸入參數。

版本:MQCBD 的現行版本是 MQCBD_VERSION_1。

字集及編碼:MQCBD 中的資料必須採用 *CodedCharSetId* 佇列管理程式屬性所提供的字集，以及 MQENC_NATIVE 所提供本端佇列管理程式的編碼。 不過，如果應用程式以 MQ MQI 用戶端身分執行，則該結構必須採用用戶端的字集及編碼。

MQCBD 的欄位

MQCBD 結構的欄位英文字母清單。

MQCBD 結構包含下列欄位; 這些欄位按字母順序說明:

CallbackArea (MQPTR)
回呼描述子結構- CallbackArea 欄位

這是可供回呼函數使用的欄位。

佇列管理程式不會根據這個欄位的內容來做出任何決策，且會從 MQCBC 結構中的 第 251 頁的『CallbackArea (MQPTR)』 欄位 (這是回呼函數宣告上的參數)，以不變的方式來傳遞它。

該值僅在具有值 MQOP_REGISTER 的 *Operation* 上使用，目前未定義回呼，它不會取代先前的定義。

這是回呼函數的輸入及輸出欄位。 此欄位的起始值是空值指標或空值位元組。

CallbackFunction (MQPTR)
回呼描述子結構- CallbackFunction 欄位

回呼函數會作為函數呼叫來呼叫。

請利用這個欄位來指定回呼函數的指標。

您 必須 指定 *CallbackFunction* 或 *CallbackName*。 如果您同時指定這兩者，則會傳回原因碼 MQRC_CALLBACK_ROUTINE_ERROR。

如果既未設定 *CallbackName* 也未設定 *CallbackFunction*，則呼叫會失敗，原因碼為 MQRC_CALLBACK_ROUTINE_ERROR。

下列環境不支援此選項: 不支援函數指標參照的程式設計語言及編譯器。 在這類狀況下，呼叫會失敗，原因碼為 MQRC_CALLBACK_ROUTINE_ERROR。

在 z/OS 上，函數必須預期使用 OS 鏈結慣例來呼叫。 例如，在 C 程式設計語言中，指定:

```
#pragma linkage(MQCB_FUNCTION,OS)
```

這是輸入欄位。 此欄位的起始值是空值指標或空值位元組。

註: 搭配使用 CICS 與 WebSphere MQ V7.0.1 時，如果發生下列情況，則支援非同步使用:

- Apar PK66866 已套用至 CICS TS 3.2
- Apar PK89844 適用於 CICS TS 4.1

CallbackName (MQCHAR128)
回呼描述子結構- CallbackName 欄位

呼叫回呼函數作為動態鏈結的程式。

您 必須 指定 *CallbackFunction* 或 *CallbackName*。 如果您同時指定這兩者，則會傳回原因碼 MQRC_CALLBACK_ROUTINE_ERROR。

如果既未設定 *CallbackName* 也未設定 *CallbackFunction*，則呼叫會失敗，原因碼為 MQRC_CALLBACK_ROUTINE_ERROR。

當登錄要使用的第一個回呼常式時，會載入模組，當要使用它的最後一個回呼常式取消登錄時，會卸載模組。

除非在下列文字中註明，否則名稱在欄位內是向左對齊的，不含內嵌空白; 名稱本身會以空白填補欄位長度。 在下列說明中，方括弧 ([]) 表示選用資訊:

IBM i
回呼名稱可以是下列其中一種格式:

- 程式庫 "/" 程式
- 程式庫 "/" ServiceProgram "("FunctionName")"

例如，`MyLibrary/MyProgram(MyFunction)`。

檔案庫名稱可以是 *LIBL。 檔案庫及程式名稱都限制為最多 10 個字元。

UNIX 系統
回呼名稱是可動態載入模組或程式庫的名稱，字尾是位於該程式庫中的函數名稱。 函數名稱必須以括弧括住。 程式庫名稱可以選擇性地以目錄路徑作為字首:

```
[path]library(function)
```

如果未指定路徑，則會使用系統搜尋路徑。

名稱限制為最多 128 個字元。

Windows
回呼名稱是動態鏈結程式庫的名稱，字尾是位於該程式庫中的函數名稱。 函數名稱必須以括弧括住。媒體庫名稱可以選擇性地以目錄路徑及磁碟機作為字首:

```
[d:][path]library(function)
```

如果未指定磁碟機和路徑，則會使用系統搜尋路徑。

名稱限制為最多 128 個字元。

z/OS
回呼名稱是載入模組的名稱，適用於 LINK 或 LOAD 巨集的 EP 參數上的規格。

名稱限制為最多 8 個字元。

z/OS CICS
回呼名稱是載入模組的名稱，適用於 EXEC CICS LINK 指令巨集的 PROGRAM 參數上的規格。

名稱限制為最多 8 個字元。

程式可以使用所安裝 PROGRAM 定義的 REMOTESYTEM 選項或由動態遞送程式定義為遠端。

如果程式要使用 WebSphere MQ API 呼叫，則遠端 CICS 區域必須連接至 WebSphere MQ 。 不過請注意， MQCBC 結構中的 第 253 頁的『Hobj (MQHOBJ)』 欄位在遠端系統中無效。

如果嘗試載入 *CallbackName* 時發生失敗，則會傳回下列其中一個錯誤碼給應用程式:

- 找不到 MQRC_MODULE_NOT_FOUND
- MQRC_MODULE_INVALID
- MQRC_MODULE_ENTRY_NOT_FOUND

也會將訊息寫入錯誤日誌中，其中包含嘗試載入的模組名稱，以及作業系統的失敗原因碼。

這是輸入欄位。 此欄位的起始值是空字串或空白。

CallbackType (MQLONG)
回呼描述子結構- CallbackType 欄位

這是回呼函數的類型。 值必須是下列其中一項:

MQCBT_MESSAGE_CONSUMER
將此回呼定義為訊息消費者函數。

當物件控點有符合指定選取準則的訊息可用且連線已啟動時，會呼叫訊息消費者回呼函數。

MQCBT_EVENT_HANDLER
將此回呼定義為非同步事件常式; 不會驅動它耗用控點的訊息。

在定義事件處理程式的 MQCB 呼叫上不需要 *Hobj* ，如果指定的話，則會忽略它。

針對會影響整個訊息消費者環境的條件呼叫事件處理程式。 當發生事件 (例如，佇列管理程式或連線停止或靜止) 時，會在沒有訊息的情況下呼叫消費者函數。 不會針對單一訊息消費者特定的條件來呼叫它，例如 MQRC_GET_INHIBITED。

不論連線是已啟動還是已停止，都會將事件遞送至應用程式，但在下列環境中除外:

- CICS on z/OS 環境
- 非執行緒應用程式

如果呼叫者未傳遞其中一個值，則呼叫會失敗，且 *Reason* 程式碼為 MQRC_CALLBACK_TYPE_ERROR

這一律是輸入欄位。 此欄位的起始值為 MQCBT_MESSAGE_CONSUMER。

MaxMsg 長度 *(MQLONG)*
這是可從控點讀取並提供給回呼常式的最長訊息長度 (以位元組為單位)。 回呼描述子結構- MaxMsg 長度欄位

如果訊息長度較長，回呼常式會收到 *MaxMsgLength* 個位元組的訊息，原因碼為:

- MQRC_TRUNCATED_MSG_FAILED 或
- 如果您指定 MQGMO_ACCEPT_TRUNCATED_MSG ，則 MQRC_TRUNCATED_MSG_ACCEPTED。

實際訊息長度在 MQCBC 結構的 第 253 頁的『DataLength (MQLONG)』 欄位中提供。

下列是已定義的特殊值:

MQCBD_FULL_MSG_LENGTH
系統會調整緩衝區長度，以傳回訊息而不截斷。

如果記憶體不足，無法配置緩衝區來接收訊息，系統會以 MQRC_STORAGE_NOT_AVAILABLE 原因碼來呼叫回呼函數。

例如，如果您要求資料轉換，且沒有足夠的記憶體可用來轉換訊息資料，則會將未轉換的訊息傳遞至回呼函數。

這是輸入欄位。 *MaxMsgLength* 欄位的起始值為 MQCBD_FULL_MSG_LENGTH。

選項 *(MQLONG)*
回呼描述子結構-選項欄位

可以指定下列任何一項或全部。 如果需要多個選項，則值可以是:

- 加在一起 (不要多次新增相同的常數) ，或
- 利用執行位元 OR 運算來組合（如果程式設計語言支援位元運算）。

MQCBDO_FAIL_IF_QUIESCING
如果佇列管理程式處於靜止狀態，則 MQCB 呼叫會失敗。

在 z/OS 上，如果連線 (若為 CICS 或 IMS 應用程式) 處於靜止狀態，則此選項也會強制 MQCB 呼叫失敗。

在 MQGMO 呼叫上傳遞的 MQGMO 選項中指定 MQGMO_FAIL_IF_QUIESCING ，以在訊息消費者靜止時向訊息消費者發出通知。

控制選項: 下列選項控制是否在消費者狀態變更時呼叫回呼函數，而不顯示訊息:

MQCBDO_REGISTER_CALL
使用呼叫類型 MQCBCT_REGISTER_CALL 來呼叫回呼函數。

MQCBDO_START_CALL
使用呼叫類型 MQCBCT_START_CALL 呼叫回呼函數。

MQCBDO_STOP_CALL
以呼叫類型 MQCBCT_STOP_CALL 來呼叫回呼函數。

MQCBDO_DEREGISTER_CALL
使用呼叫類型 MQCBCT_DEREGISTER_CALL 呼叫回呼函數。

MQCBDO_EVENT_CALL
使用呼叫類型 MQCBCT_EVENT_CALL 來呼叫回呼函數。

MQCBDO_MC_EVENT_CALL
使用呼叫類型 MQCBCT__MC_EVENT_CALL 來呼叫回呼函數。

如需這些通話類型的進一步詳細資料，請參閱 第 251 頁的『CallType (MQLONG)』 。

預設選項: 如果您不需要任何說明的選項，請使用下列選項:

MQCBDO_NONE
使用這個值來指出未指定其他選項；所有選項都採用其預設值。

MQCBDO_NONE 已定義為輔助程式文件; 此選項並非預期與任何其他選項搭配使用，但由於其值為零，因此無法偵測到這類使用。

這是輸入欄位。 *Options* 欄位的起始值是 MQCBDO_NONE。

StrucId (MQCHAR4)
回呼描述子結構- StrucId 欄位

這是結構 ID; 值必須是:

MQCBD_STRUC_ID
回呼描述子結構的 ID。

對於 C 程式設計語言，也會定義常數 MQCBD_STRUC_ARRAY; 此值與 MQCBD_STRUC_ID 相同，但卻是字元陣列而非字串。

這一律是輸入欄位。 此欄位的起始值是 MQCBD_STRUC_ID。

版本 *(MQLONG)*
回呼描述子結構-版本欄位

這是結構版本號碼; 值必須是:

MQCBD_VERSION_1
Version-1 回呼描述子結構。

下列常數指定現行版本的版本號碼:

MQCBD_CURRENT_VERSION
回呼描述子結構的現行版本。

這一律是輸入欄位。 此欄位的起始值為 MQCBD_VERSION_1。

MQCBD 的起始值和語言宣告
回呼描述子結構-起始值

表 *481: MQCBD* 中欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQCBD_STRUC_ID	'CBD¬'
Version	MQCBD_VERSION_1	1
CallBackType	MQCBT_MESSAGE_CONSUMER	1
Options	MQCBDO_NONE	0
CallbackArea	無	空值指標或空值空白
CallbackFunction	無	空值指標或空值空白

表 *481: MQCBD* 中欄位的起始值 *(*繼續*)*

欄位名稱	**常數名稱**	**常數值**
CallbackName	無	空字串或空白
MaxMsgLength	MQCBD_FULL_MSG_LENGTH	-1

附註:

1. 符號 ¬ 代表單一空白字元。
2. 空值字串或空白表示 C 程式設計語言中的空值字串，以及其他程式設計語言中的空白字元。
3. 在 C 程式設計語言中，巨集變數 MQCBD_DEFAULT 包含上面列出的值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQCBD MyCBD = {MQCBD_DEFAULT};
```

C 宣告
回呼描述子結構-C 語言宣告

```
typedef struct tagMQCBD MQCBD;
 struct tagMQCBD {
   MQCHAR4    StrucId;                /* 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 */
 };
```

COBOL 宣告

```
** 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.
```

PL/I 宣告

```
dcl
 1 MQCBD based,
   3 StrucId             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 */
```

MQCHARV-可變長度字串

下表彙總結構中的欄位。

欄位	說明	主題
VSPtr	可變長度字串的指標	VSPtr
VSOffset	從包含此 MQCHARV 結構的結構開始算起，可變長度字串的偏移 (以位元組為單位)	VSOffset
VSLength	VSPtr 或 VSOffset 欄位所定址之可變長度字串的長度 (以位元組為單位)。	VSLength
VSBufSize	VSPtr 或 VSOffset 欄位所定址的緩衝區大小 (以位元組為單位)。	VSBufSize
VSCCSID	VSPtr 或 VSOffset 欄位所定址之可變長度字串的字集 ID。	VSCCSID

MQCHARV 概觀

可用性: AIX、 HP-UX、Solaris、 Linux、 IBM i、 Windows，以及連接至這些系統的 WebSphere MQ MQI 用戶端。

目的: 使用 MQCHARV 結構來說明可變長度字串。

字集及編碼:MQCHARV 中的資料必須採用 MQENC_NATIVE 所提供本端佇列管理程式的編碼，以及結構內 VSCCSID 欄位的字集。 如果應用程式以 MQ 用戶端身分執行，則結構必須採用用戶端編碼。 部分字集具有視編碼而定的表示法。 如果 VSCCSID 是其中一個字集，則使用的編碼與 MQCHARV 中其他欄位的編碼相同。 VSCCSID 所識別的字集可以是雙位元組字集 (DBCS)。

用法:MQCHARV 結構會處理可能與包含它的結構不連續的資料。 若要處理此資料，可以使用以指標資料類型宣告的欄位。 請注意， COBOL 不支援所有環境中的指標資料類型。 因此，也可以使用欄位來處理資料，這些欄位包含從包含 MQCHARV 的結構開始算起的資料偏移。

COBOL 程式設計

如果您想要在環境之間連接應用程式，則必須確定指標資料類型是否在所有預期環境中都可用。 如果沒有，應用程式必須使用偏移欄位而非指標欄位來處理資料。

在不支援指標的環境中，您可以將指標欄位宣告為適當長度的位元組字串，起始值為全空值位元組字串。 如果您使用偏移欄位，請勿變更此起始值。 在不變更所提供的記錄冊的情況下執行此動作的一種方法是使用下列:

```
COPY CMQCHRVV REPLACING POINTER BY ==BINARY PIC S9(9)==.
```

其中 CMQCHRVV 可以交換要使用的記錄定義檔。

MQCHARV 的欄位

MQCHARV 結構包含下列欄位; 這些欄位按 **字母順序**說明:

VSBufSize (MQLONG)
這是 VSPtr 或 VSOffset 欄位所定址的緩衝區大小 (以位元組為單位)。

在函數呼叫中使用 MQCHARV 結構作為輸出欄位時，必須以提供的緩衝區長度來起始設定此欄位。 如果 VSLength 的值大於 VSBufSize ，則只會將 VSBufSize 個位元組資料傳回給緩衝區中的呼叫者。

此值必須是大於或等於零的值，或下列可辨識的特殊值:

MQVS_USE_VSLENGTH
指定時，會從 MQCHARV 結構中的 VSLength 欄位取得緩衝區的長度。 當使用結構作為輸出欄位並提供緩衝區時，請勿使用此值。

這是此欄位的起始值。

VSCCSID (MQLONG)
這是 VSPtr 或 VSOffset 欄位所定址之可變長度字串的字集 ID。

此欄位的起始值是 MQ 所定義的 MQCCSI_APPL，指出它應該變更為現行處理程序的真實字集 ID。 因此，MQCCSI_APPL 值絕不會與可變長度字串相關聯。 透過為編譯單元的常數 MQCCSI_APPL 定義適當的應用程式程式設計語言方法，可以變更此欄位的起始值。

VSLength (MQLONG)
VSPtr 或 VSOffset 欄位所定址之可變長度字串的長度 (以位元組為單位)。

此欄位的起始值為 0。 此值必須大於或等於零，或下列可辨識的特殊值:

MQVS_NULL_TERMINATED

如果未指定 MQVS_NULL_TERMINATED，則會併入 VSLength 位元組作為字串的一部分。 如果存在空值字元，則不會對字串進行定界。

如果指定 MQVS_NULL_TERMINATED，則會以字串中發現的第一個空值來區隔字串。 空值本身不會併入為該字串的一部分。

註: 如果指定 MQVS_NULL_TERMINATED，則用來終止字串的空值字元是 VSCCSID 所指定字碼集的空值。

例如，在 UTF-16 (UCS-2 CCSID 1200 及 13488) 中，這是兩個位元組的 Unicode 編碼，其中空值以全部為零的 16 位元數字表示。 在 UTF-16 中，通常會尋找設為全部零的單一位元組 (例如， 7 位元 ASCII 字元)，但只有在偶數位元組界限上找到兩個「零」位元組時，字串才會以空值終止。 如果它們是有效字元的每一部分，則可以在奇數界限上取得兩個「零」位元組。 例如， x '01' x '00' x '00' x '30' 代表兩個有效的 Unicode 字元，且字串結尾不是空值。

VSOffset (MQLONG)
偏移可以是正數或負數。 您可以使用 VSPtr 或 VSOffset 欄位來指定可變長度字串，但不能同時指定兩者。 可變長度字串從 MQCHARV 或包含它的結構開始的偏移 (以位元組為單位)。

當 MQCHARV 結構內嵌在另一個結構中時，此值是從包含此 MQCHARV 結構的結構開始算起可變長度字串的偏移 (以位元組為單位)。 當 MQCHARV 結構未內嵌在另一個結構中時，例如，如果將它指定為函數呼叫的參數，則偏移相對於 MQCHARV 結構的開頭。

此欄位的起始值為 0。

VSPtr (MQPTR)
這是可變長度字串的指標。

您可以使用 VSPtr 或 VSOffset 欄位來指定可變長度字串，但不能同時指定兩者。

此欄位的起始值是空值指標或空值位元組。

MQCHARV 的起始值及語言宣告

MQCHARV 中欄位的起始值

欄位名稱	常數名稱	常數值
VSPtr	無	空值指標或空值位元組。
VSOffset	無	0
VSBufSize	MQVS_USE_VSLENGTH	0
VSLength	無	0
VSCCSID	MQCCSI_APPL	-3

欄位名稱	常數名稱	常數值
註: 在 C 程式設計語言中，巨集變數 MQCHARV_DEFAULT 包含上述值。 您可以用下列方式來提供結構中欄位的起始值: `MQCHARV MyVarStr = {MQCHARV_DEFAULT};`		

C 宣告

```
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 */
};
```

MQCHARV 的 *COBOL* 宣告

```
** 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.
```

PL/I 宣告

```
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 */
```

High Level Assembler 宣告

```
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)
```

重新定義 *MQCCSI_APPL*

下列範例顯示如何在各種程式設計語言中置換 MQCCSI_APPL 的值。 您可以變更 MQCCSI_APPL 的值，而不需要個別設定每一個可變長度字串的 VSCCSID。

在這些範例中， CCSID 會設為 1208; 請將此值變更為您需要的值。 這會變成預設值，您可以在 MQCHARV 的任何特定實例中設定 VSCCSID 來置換。

C 用法

```
#define MQCCSI_APPL 1208
#include <cmqc.h>
```

COBOL 用法

```
COPY CMQXYZV REPLACING -3 BY 1208.
```

PL/I 用法

```
%MQCCSI_APPL = '1208';
%include syslib(cmqp);
```

System/390 組譯器用法

```
MQCCSI_APPL EQU 1208
         CMQA LIST=NO
```

MQCIH- CICS 橋接器標頭

下表彙總結構中的欄位。

表 *482: MQCIH* 中的欄位

欄位	說明	主題
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
StrucLength	MQCIH 結構的長度	StrucLength
Encoding	已保留	編碼
CodedCharSetId	已保留	CodedCharSetId
Format	MQCIH 之後的資料的 MQ 格式名稱	格式
Flags	旗標	旗標
ReturnCode	橋接器的回覆碼	ReturnCode
CompCode	MQ 完成碼或 CICS EIBRESP	CompCode
Reason	MQ 原因或回饋碼，或 CICS EIBRESP2	原因
UOWControl	工作單元控制項	UOWControl
GetWaitInterval	橋接器作業發出的 MQGET 呼叫的等待間隔	GetWait 間隔
LinkType	鏈結類型	LinkType
OutputDataLength	輸出 COMMAREA 資料長度	OutputData 長度
FacilityKeepTime	橋接器機能釋放時間	FacilityKeep 時間
ADSDescriptor	傳送/接收 ADS 描述子	ADSDcriptor
ConversationalTask	作業是否可為交談式	ConversationalTask
TaskEndStatus	作業結束時的狀態	TaskEnd 狀態
Facility	橋接器機能記號	機能

表 *482: MQCIH* 中的欄位 *(*繼續*)*		
欄位	**說明**	**主題**
Function	MQ 呼叫名稱或 CICS EIBFN 函數	函數
AbendCode	異常終止碼	AbendCode
Authenticator	密碼或 PassTicket	鑑別程式
Reserved1	已保留	Reserved1
ReplyToFormat	回覆訊息的 MQ 格式名稱	ReplyTo 格式
RemoteSysId	要使用的遠端 CICS 系統 ID	RemoteSysID
RemoteTransId	CICS 要使用的 RTRANSID	RemoteTransID
TransactionId	要附加的交易	TransactionId
FacilityLike	終端機模的擬屬性	FacilityLike
AttentionId	AID 金鑰	AttentionId
StartCode	交易起始碼	StartCode
CancelCode	異常終止交易碼	CancelCode
NextTransactionId	要連接的下一個交易	NextTransactionID
Reserved2	已保留	Reserved2
Reserved3	已保留	Reserved3
註: 如果 *Version* 小於 MQCIH_VERSION_2，則其餘欄位不存在。		
CursorPosition	游標位置	CursorPosition
ErrorOffset	訊息中的錯誤偏移	ErrorOffset
InputItem	已保留	InputItem
Reserved4	已保留	Reserved4

MQCIH 概觀

可用性: AIX、 HP-UX、 z/OS、Solaris、 Linux、 Windows，以及連接至這些系統的 WebSphere MQ MQI 用戶端。

目的:MQCIH 結構說明在透過 WebSphere MQ for z/OS 傳送至 CICS 橋接器的訊息開始時可呈現的資訊。

格式名稱:MQFMT_CICS。

版本:MQCIH 的現行版本是 MQCIH_VERSION_2。 僅存在於結構最新版本中的欄位在接下來的說明中如此識別。

為支援的程式設計語言提供的標頭、COPY 及 INCLUDE 檔案包含最新版本的 MQCIH，且 *Version* 欄位的起始值設為 MQCIH_VERSION_2。

字集及編碼: 特殊條件適用於 MQCIH 結構及應用程式訊息資料所使用的字集及編碼:

- 連接至擁有 CICS 橋接器佇列之佇列管理程式的應用程式，必須以佇列管理程式的字集及編碼方式提供 MQCIH 結構。 這是因為在此情況下不會執行 MQCIH 結構的資料轉換。
- 連接至其他佇列管理程式的應用程式可以提供採用任何受支援字集及編碼的 MQCIH 結構; 連接至擁有 CICS 橋接器佇列之佇列管理程式的接收訊息通道代理程式會轉換 MQCIH 結構。
- MQCIH 結構後面的應用程式訊息資料必須使用與 MQCIH 結構相同的字集及編碼。 您無法使用 MQCIH 結構中的 *CodedCharSetId* 及 *Encoding* 欄位來指定應用程式訊息資料的字集及編碼。

如果資料不是佇列管理程式支援的內建格式之一，您必須提供資料轉換結束程式來轉換應用程式訊息資料。

用法: 如果應用程式需要的值與 第 275 頁的表 484 中顯示的起始值相同，且橋接器是以 AUTH=LOCAL 或 AUTH=ENDY 執行，您可以從訊息中省略 MQCIH 結構。 在所有其他情況下，結構必須存在。

橋接器接受 version-1 或 version-2 MQCIH 結構，但對於 3270 交易，您必須使用 version-2 結構。

應用程式必須確保記載為要求欄位的欄位在傳送至橋接器的訊息中具有適當的值; 這些欄位是橋接器的輸入。

記載為回應欄位的欄位是由 CICS 橋接器在橋接器傳送至應用程式的回覆訊息中設定。 在 *ReturnCode*、*Function*、 *CompCode*、 *Reason* 及 *AbendCode* 欄位中傳回錯誤資訊，但並非所有這些欄位都已設定。 第 267 頁的表 483 顯示針對 *ReturnCode* 的不同值設定哪些欄位。

表 *483: MQCIH* 的 *MQCIH* 結構中錯誤資訊欄位的內容

ReturnCode	***Function***	***CompCode***	***Reason***	***AbendCode***
MQCRC_OK	-	-	-	-
MQCRC_BRIDGE_ERROR	-	-	MQFB_CICS_ *	-
MQCRC_MQ_API_ERROR MQCRC_BRIDGE_TIMEOUT	MQ 呼叫名稱	MQ *CompCode*	MQ *Reason*	-
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

MQCIH 的欄位

MQCIH 結構包含下列欄位; 這些欄位按 **字母順序**進行說明:

AbendCode (MQCHAR4)
AbendCode 是回應欄位。 此欄位的長度由 MQ_ABEND_CODE_LENGTH 提供。 此欄位的起始值為 4 個空白字元。

僅當 *ReturnCode* 欄位具有值 MQCRC_APPLICATION_ABEND 或 MQCRC_BRIDGE_ABEND 時，此欄位中傳回的值才有效。 如果有， *AbendCode* 會包含 CICS ABCODE 值。

ADSDescriptor (MQLONG)
此欄位是指定是否在 SEND 及 RECEIVE BMS 要求上傳送 ADS 描述子的指示器。

已定義下列值:

MQCADSD_NONE
不傳送或接收 ADS 描述子。

MQCADSD_SEND
傳送 ADS 描述子。

MQCADSD_RECV
接收 ADS 描述子。

MQCADSD_MSGFORMAT
對 ADS 描述子使用訊息格式。

這會使用 ADS 描述子的長格式來傳送或接收 ADS 描述子。 長格式具有在 4 位元組界限上對齊的欄位。

設定 *ADSDescriptor* 欄位，如下所示:

- 如果您不是使用 ADS 描述子，請將欄位設為 MQCADSD_NONE。

- 如果您在每一個環境中使用具有 相同 CCSID 的 ADS 描述子，請將欄位設為 MQCADSD_SEND 及 MQCADSD_RECV 的總和。
- 如果您在每一個環境中使用具有 不同 CCSID 的 ADS 描述子，請將欄位設為 MQCADSD_SEND、MQCADSD_RECV 及 MQCADSD_MSGFORMAT 的總和。

這是僅用於 3270 交易的要求欄位。 此欄位的起始值為 MQCADSD_NONE。

AttentionId (MQCHAR4)
此欄位中的值決定啟動交易時 AID 索引鍵的起始值。 它是 1 位元組值，靠左對齊。

AttentionId 是僅用於 3270 交易的要求欄位。 此欄位的長度由 MQ_ATTENTION_ID_LENGTH 提供。 此欄位的起始值為四個空白。

鑑別器 *(MQCHAR8)*
此欄位的值是密碼或 passticket。

如果 CICS 橋接器的使用者 ID 鑑別處於作用中狀態，則 *Authenticator* 會與 MQMD 身分環境定義中的使用者 ID 搭配使用，以鑑別訊息傳送端。

這是要求欄位。 此欄位的長度由 MQ_AUTHENTICATOR_LENGTH 提供。 此欄位的起始值為 8 個空白。

CancelCode (MQCHAR4)
此欄位中的值是用來終止交易的異常終止碼 (通常是要求更多資料的交談式交易)。 否則，此欄位會設為空白。

此欄位是僅用於 3270 交易的要求欄位。 此欄位的長度由 MQ_CANCEL_CODE_LENGTH 提供。 此欄位的起始值為四個空白。

CodedCharSetId (MQLONG)
CodedCharSetId 是保留欄位; 其值不顯著。 此欄位的起始值為 0。

遵循 MQCIH 結構之受支援結構的字集 ID 與 MQCIH 結構本身的字集 ID 相同，且取自任何之前的 WebSphere MQ 標頭。

CompCode (MQLONG)
此欄位是回應欄位。 其起始值為 MQCC_OK

此欄位中傳回的值取決於 *ReturnCode*; 請參閱 第 267 頁的表 483。

ConversationalTask (MQLONG)
此欄位是一個指示器，指定是否容許作業發出要求以取得相關資訊，或停止作業並發出異常終止訊息。

值必須是下列其中一個選項:

MQCCT_YES
作業是交談式。

MQCCT_NO
作業不是交談式。

此欄位是僅用於 3270 交易的要求欄位。 此欄位的起始值為 MQCCT_NO。

CursorPosition (MQLONG)
當啟動交易時，此欄位中的值會顯示起始游標位置。 若為交談式交易，游標位置是在 RECEIVE 向量中。

此欄位是僅用於 3270 交易的要求欄位。 此欄位的起始值為 0。 如果 *Version* 小於 MQCIH_VERSION_2，則此欄位不存在。

編碼 *(MQLONG)*
此欄位是保留欄位; 其值不顯著。 其起始值為 0。

遵循 MQCIH 結構之受支援結構的編碼與 MQCIH 結構本身的編碼相同，且取自任何先前的 WebSphere MQ 標頭。

ErrorOffset (MQLONG)
ErrorOffset 欄位會顯示橋接器結束程式所偵測到無效資料的位置。 此欄位提供從訊息開頭到無效資料位置的偏移。

ErrorOffset 是僅用於 3270 交易的回應欄位。 此欄位的起始值為 0。 如果 *Version* 小於 MQCIH_VERSION_2，則此欄位不存在。

機能 *(MQBYTE8)*
此欄位顯示 8 位元組橋接器機能記號。

橋接器機能記號可讓虛擬交談中的多個交易使用相同的橋接器機能 (虛擬 3270 終端機)。 在虛擬交談中的第一個 (或唯一) 訊息中，將值設為 MQCFAC_NONE。 此值告知 CICS 為此訊息配置新的橋接器機能。 在輸入訊息上指定非零 *FacilityKeepTime* 時，會在回應訊息中傳回橋接器機能記號。 然後，虛擬交談內的後續輸入訊息必須使用相同的橋接器機能記號。

下列是已定義的特殊值:

MQCFAC_NONE
未指定機能記號。

對於 C 程式設計語言，也會定義常數 MQCFAC_NONE_ARRAY，其值與 MQCFAC_NONE 相同，但它是字元陣列而非字串。

此欄位既是要求，也是僅用於 3270 交易的回應欄位。 此欄位的長度由 MQ_FACILITY_LENGTH 指定。 此欄位的起始值為 MQCFAC_NONE。

FacilityKeep 時間 *(MQLONG)*
FacilityKeep 時間是在使用者交易結束之後保留橋接器機能的時間長度 (以秒為單位)。

若為虛擬交談式交易，請指定對應於虛擬交談預期持續時間的值; 若為虛擬交談的最後一個交易，則指定零，若為其他交易類型，則指定零。

此欄位是僅用於 3270 交易的要求欄位。 此欄位的起始值為 0。

FacilityLike (MQCHAR4)
FacilityLike 是要用作橋接器機能模型的已安裝終端機名稱。

空白值表示從橋接器交易設定檔定義取得 *FacilityLike*，或使用預設值。

此欄位是僅用於 3270 交易的要求欄位。 此欄位的長度由 MQ_FACILITY_LIKE_LENGTH 指定。 此欄位的起始值為四個空白。

旗標 *(MQLONG)*
此欄位是要求欄位。 此欄位的起始值為 MQCIH_NONE。

值必須為:

MQCIH_NONE
沒有旗標。

MQCIH_PASS_EXPIRATION
回覆訊息包含:

- 與要求訊息相同的期限報告選項。
- 要求訊息中的剩餘到期時間，未對橋接器的處理時間進行調整。

如果您省略此值，則到期時間會設為 無限制。

MQCIH_REPLY_WITHOUT_NULLS
調整 CICS DPL 程式要求的回覆訊息長度，以排除 DPL 程式所傳回 COMMAREA 結尾的尾端空值 (X'00')。 如果未設定此值，則空值可能很重要，且會傳回完整 COMMAREA。

MQCIH_SYNC_ON_XX_ENCODE_CASE_ONE return
DPL 要求的 CICS 鏈結使用 SYNCONRETURN 選項，如果程式已出貨至另一個 CICS 區域，則會導致 CICS 在程式完成時採取同步點。 橋接器未指定要將要求傳送至哪個 CICS 區域; 這是由 CICS 程式定義或工作量平衡機能所控制。

格式 *(MQCHAR8)*
此欄位顯示遵循 MQCIH 結構之資料的 WebSphere MQ 格式名稱。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。 此欄位的編碼規則與 MQMD 中 *Format* 欄位的編碼規則相同。

如果 *ReplyToFormat* 欄位具有值 MQFMT_NONE，則此格式名稱也用於回覆訊息。

- 對於 DPL 要求， *Format* 必須是 COMMAREA 的格式名稱。
- 若為 3270 要求， *Format* 必須是 CSQCBDCI，橋接器會將「回覆」訊息的格式設為 CSQCBDC0 。

這些格式的資料轉換結束程式必須安裝在要執行它們的佇列管理程式上。

如果要求訊息產生錯誤回覆訊息，則錯誤回覆訊息的格式名稱為 MQFMT_STRING。

此欄位是要求欄位。 此欄位的長度由 MQ_FORMAT_LENGTH 提供。 此欄位的起始值為 MQFMT_NONE。

函數 *(MQCHAR4)*
此欄位是回應欄位。 此欄位的長度由 MQ_FUNCTION_LENGTH 提供。 此欄位的起始值為 MQCFUNC_NONE。

此欄位中傳回的值取決於 *ReturnCode*; 請參閱 第 267 頁的表 483。 當 *Function* 包含 WebSphere MQ 呼叫名稱時，可能有下列值:

MQCFUNC_MQCONN
MQCONN 呼叫。

MQCFUNC_MQGET
MQGET 呼叫。

MQCFUNC_MQINQ
MQINQ 呼叫。

MQCFUNC_MQOPEN
MQOPEN 呼叫。

MQCFUNC_MQPUT
MQPUT 呼叫。

MQCFUNC_MQPUT1
MQPUT1 呼叫。

MQCFUNC_NONE
沒有電話

在所有情況下，對於 C 程式設計語言，也會定義常數 MQCFUNC_ * _ARRAY; 這些常數與對應的 MQCFUNC_ * 常數具有相同的值，但它們是字元陣列而非字串。

GetWait 間隔 *(MQLONG)*
此欄位是要求欄位。 其起始值為 MQCGWI_DEFAULT。

只有在 *UOWControl* 具有 MQCUOWC_FIRST 值時，此欄位才適用。 它可讓傳送端應用程式指定橋接器發出的 MQGET 呼叫將等待此訊息所啟動工作單元的第二個及後續要求訊息的大約時間 (毫秒)。 此機能會置換橋接器使用的預設等待間隔。 您可以使用下列特殊值:

MQCGWI_DEFAULT
預設等待間隔。

此值會導致 CICS 橋接器等待啟動橋接器時指定的時間。

MQWI_UNLIMITED
無限制等待間隔。

InputItem (MQLONG)
此欄位是保留欄位。 值必須是 0。

如果 *Version* 小於 MQCIH_VERSION_2，則此欄位不存在。

LinkType (MQLONG)
此欄位是要求欄位。 其起始值為 MQCLT_PROGRAM。

此值指出橋接器嘗試鏈結的物件類型。 它必須是下列其中一個值:

MQCLT_PROGRAM
DPL 程式。

MQCLT_TRANSACTION
3270 交易。

NextTransactionID (MQCHAR4)
此值是使用者交易 (通常是 EXEC CICS RETURN TRANSID) 傳回的下一個交易名稱。 如果沒有下一個交易，則此欄位會設為空白。

此欄位是僅用於 3270 交易的回應欄位。 此欄位的長度由 MQ_TRANSACTION_ID_LENGTH 提供。 此欄位的起始值為四個空白。

OutputData 長度 *(MQLONG)*
此欄位是僅用於 DPL 程式的要求欄位。 其起始值為 MQCODL_AS_INPUT。

此值是要在回覆訊息中傳回用戶端的使用者資料長度。 此長度包括 8 位元組程式名稱。 傳遞給鏈結程式的 COMMAREA 長度是此欄位的最大值，以及要求訊息中使用者資料的長度減 8。

註: 訊息中使用者資料的長度是訊息的長度，不包括 MQCIH 結構。

如果要求訊息中使用者資料的長度小於 *OutputDataLength*，則會使用 LINK 指令的 DATALENGTH 選項，讓 LINK 能夠有效地運作傳送至另一個 CICS 區域。

您可以使用下列特殊值:

MQCODL_AS_INPUT
輸出長度與輸入長度相同。

即使未要求任何回覆，也可能需要此值，以確保傳遞至鏈結程式的 COMMAREA 大小足夠。

原因 *(MQLONG)*
此欄位是回應欄位。 其起始值為 MQRC_NONE。

此欄位中傳回的值取決於 *ReturnCode*; 請參閱 第 267 頁的表 483。

RemoteSysID (MQCHAR4)
此欄位顯示處理要求之 CICS 系統的 CICS 系統 ID。

如果此欄位空白，則會在與橋接器監視器相同的 CICS 系統上處理 CICS 系統要求。 使用的 SYSID 會在回覆訊息中傳回。

對於 3270 虛擬交談，交談中的所有後續訊息都必須指定在起始回覆中傳回的遠端 SYSID。 如果指定的話，SYSID 必須:

- 在作用中。
- 有權存取 WebSphere MQ 要求佇列。
- 可從橋接器監視器的 CICS 系統存取 CICS ISC 鏈結。

RemoteTransID (MQCHAR4)
此欄位是選用的「要求」欄位。 此欄位的長度由 MQ_TRANSACTION_ID_LENGTH 提供。

如果指定的話，則會使用該欄位作為 CICS START 的 RTRANSID 值。

ReplyTo 格式 *(MQCHAR8)*
此欄位的值是為了回應現行訊息而傳送之回覆訊息的 WebSphere MQ 格式名稱。

此欄位的編碼規則與 MQMD 中 *Format* 欄位的編碼規則相同。

此欄位是僅用於 DPL 程式的要求欄位。 此欄位的長度由 MQ_FORMAT_LENGTH 提供。 此欄位的起始值為 MQFMT_NONE。

Reserved1 (MQCHAR8)
此欄位是保留欄位。 值必須是 8 個空白。

Reserved2 (MQCHAR8)
此欄位是保留欄位。 值必須是 8 個空白。

Reserved3 (MQCHAR8)
此欄位是保留欄位。 值必須是 8 個空白。

Reserved4 (MQLONG)
此欄位是保留欄位。 值必須是 0。

如果 *Version* 小於 MQCIH_VERSION_2，則此欄位不存在。

ReturnCode (MQLONG)
此欄位的值是來自 CICS 橋接器的回覆碼，說明橋接器所執行處理的結果。 此欄位是回應欄位，起始值為 MQCRC_OK。

Function、 *CompCode*、 *Reason* 和 *AbendCode* 欄位可能包含其他資訊 (請參閱 第 267 頁的表 483)。
此值是下列其中一個：

MQCRC_APPLICATION_ABEND
(5 , X'005 ') 應用程式異常結束。

MQCRC_BRIDGE_ABEND
(4 , X'004 ') CICS 橋接器異常結束。

MQCRC_BRIDGE_ERROR
(3 , X'003 ') CICS 橋接器偵測到錯誤。

MQ CRC_BRIDGE_TIMEOUT
(8 , X'008 ') 現行工作單元內未在指定時間內收到第二個或更新的訊息。

MQCRC_CICS_EXEC_ERROR
(1 , X'001 ') EXEC CICS 陳述式偵測到錯誤。

MQCRC_MQ_API_ERROR
(2 , X'002 ') MQ 呼叫偵測到錯誤。

MQCRC_OK
(0 , X'000 ') 無錯誤。

MQCRC_PROGRAM_NOT_AVAILABLE
(7 , X'007 ') 程式無法使用。

MQCRC_SECURITY_ERROR
(6 , X'006 ') 發生安全錯誤。

MQCRC_TRANSID_NOT_AVAILABLE
(9 , X'009 ') 異動無法使用。

StartCode (MQCHAR4)
此欄位的值是一個指示器，指定橋接器是否模擬終端機交易或以 START 起始的交易。

此值必須是下列其中一個：

MQCSC_START
開始。

MQCSC_STARTDATA
啟動資料。

MQCSC_TERMINPUT
終端機輸入。

MQCSC_NONE
無。

無論如何，對於 C 程式設計語言，也會定義常數 MQCSC_ * _ARRAY; 這些常數與對應的 MQCSC_ * 常數具有相同的值，但它們是字元陣列而非字串。

在來自橋接器的回應中，此欄位設為 *NextTransactionId* 欄位中包含的下一個交易 ID 所適用的起始碼。回應中可能有下列起始碼:

- MQCSC_START
- MQCSC_STARTDATA
- MQCSC_TERMINPUT

若為 CICS Transaction Server 1.2，此欄位僅為要求欄位; 其在回應中的值未定義。

若為 CICS Transaction Server 1.3 版及後續版本，此欄位既是要求，也是回應欄位。

此欄位僅用於 3270 交易。 此欄位的長度由 MQ_START_CODE_LENGTH 提供。 此欄位的起始值為 MQCSC_NONE。

StrucId (MQCHAR4)
此欄位是要求欄位，其起始值為 MQCIH_STRUC_ID。

值必須為:

MQCIH_STRUC_ID
CICS 資訊標頭結構的 ID。

對於 C 程式設計語言，也會定義常數 MQCIH_STRUC_ID_ARRAY; 此值與 MQCIH_STRUC_ID 相同，但卻是字元陣列而非字串。

StrucLength (MQLONG)
此欄位是要求欄位，起始值為 MQCIH_LENGTH_2。

此值必須是下列其中一個:

MQCIH_LENGTH_1
version-1 CICS 資訊標頭結構的長度。

MQCIH_LENGTH_2
version-2 CICS 資訊標頭結構的長度。

下列常數指定現行版本的長度:

MQCIH_CURRENT_LENGTH
CICS 資訊標頭結構現行版本的長度。

TaskEnd 狀態 *(MQLONG)*
此欄位是回應欄位，顯示作業結束時使用者交易的狀態。 此欄位僅用於 3270 交易，且其起始值為 MQCTES_NOSYNC。

會傳回下列其中一個值:

MQCTES_NOSYNC
未同步。

使用者交易尚未完成且尚未同步。 在此情況下， MQMD 中的 *MsgType* 欄位是 MQMT_REQUEST。

MQCTES_COMMIT
確定工作單元。

使用者交易尚未完成，但已同步指出第一個工作單元。 在此情況下， MQMD 中的 *MsgType* 欄位是 MQMT_DATAGRAM。

MQCTES_BACKOUT
退出工作單元。

使用者交易尚未完成。 已取消現行工作單元。 在此情況下， MQMD 中的 *MsgType* 欄位是 MQMT_DATAGRAM。

MQCTES_ENDTASK
結束作業。

使用者交易已結束 (或異常終止)。 在此情況下， MQMD 中的 *MsgType* 欄位是 MQMT_REPLY。

TransactionId (MQCHAR4)
此欄位是要求欄位。 其長度由 MQ_TRANSACTION_ID_LENGTH 提供。 此欄位的起始值為四個空白。

如果 *LinkType* 具有值 MQCLT_TRANSACTION ，則 *TransactionId* 是要執行之使用者交易的交易 ID; 在此情況下，請指定非空白值。

如果 *LinkType* 具有值 MQCLT_PROGRAM ，則 *TransactionId* 是交易碼，在此交易碼下將執行工作單元內的所有程式。 如果您指定空白值，則會使用 CICS DPL 橋接器預設交易碼 (CKBP)。 如果值為非空白，則您必須已將它定義給 CICS ，以作為具有起始程式 CSQCBP00 的區域交易。 僅當 *UOWControl* 具有值 MQCUOWC_FIRST 或 MQCUOWC_ONLY 時，此欄位才適用。

UOWControl (MQLONG)
此欄位是要求欄位，用於控制 CICS 橋接器所執行的工作單元處理。 此欄位的起始值為 MQCUOWC_ONLY。

您可以要求橋接器執行單一交易，或在工作單元內執行一或多個程式。 此欄位指出 CICS 橋接器是否啟動工作單元，在現行工作單元內執行所要求的功能，或透過確定或取消工作單元來結束工作單元。 支援各種組合，以最佳化資料傳輸流程。

此值必須是下列其中一個:

僅 MQCUOWC_ONLY
啟動工作單元，執行功能，然後確定工作單元。

MQCUOWC_CONTINUE
現行工作單元的其他資料 (僅限 3270)。

MQCUOWC_FIRST
啟動工作單元並執行功能。

MQCUOWC_MIDDLE
在現行工作單元內執行功能

MQCUOWC_LAST
執行功能，然後確定工作單元。

MQCUOWC_COMMIT
確定工作單元 (僅限 DPL)。

MQCUOWC_BACKOUT
退出工作單元 (僅限 DPL)。

版本 *(MQLONG)*
此欄位是要求欄位。 其起始值為 MQCIH_VERSION_2。

此值必須是下列其中一個:

MQCIH_VERSION_1
Version-1 CICS 資訊標頭結構。

MQCIH_VERSION_2
Version-2 CICS 資訊標頭結構。

僅存在於結構最新版本中的欄位會在欄位說明中如此識別。 下列常數指定現行版本的版本號碼:

MQCIH_CURRENT_VERSION
CICS 資訊標頭結構的 現行 版本。

MQCIH 的起始值及語言宣告

表 *484: MQCIH* 中 *MQCIH* 欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQCIH_STRUC_ID	'CIH¬'
Version	MQCIH_VERSION_2	2
StrucLength	MQCIH_LENGTH_2	180
Encoding	無	0
CodedCharSetId	無	0
Format	MQFMT_NONE	空白
Flags	MQCIH_NONE	0
ReturnCode	MQCRC_OK	0
CompCode	MQCC_OK	0
Reason	MQRC_NONE	0
UOWControl	僅 MQCUOWC_ONLY	273
GetWaitInterval	MQCGWI_DEFAULT	-2
LinkType	MQCLT_PROGRAM	1
OutputDataLength	MQCODL_AS_INPUT	-1
FacilityKeepTime	無	0
ADSDescriptor	MQCADSD_NONE	0
ConversationalTask	MQCCT_NO	0
TaskEndStatus	MQCTES_NOSYNC	0
Facility	MQCFAC_NONE	空值
Function	MQCFUNC_NONE	空白
AbendCode	無	空白
Authenticator	無	空白
Reserved1	無	空白
ReplyToFormat	MQFMT_NONE	空白
RemoteSysId	無	空白
RemoteTransId	無	空白
TransactionId	無	空白
FacilityLike	無	空白
AttentionId	無	空白
StartCode	MQCSC_NONE	空白
CancelCode	無	空白
NextTransactionId	無	空白
Reserved2	無	空白

表 *484: MQCIH* 中 *MQCIH* 欄位的起始值 *(*繼續*)*

欄位名稱	常數名稱	常數值
Reserved3	無	空白
CursorPosition	無	0
ErrorOffset	無	0
InputItem	無	0
Reserved4	無	0

附註:

1. 符號 ¬ 代表單一空白字元。
2. 在 C 程式設計語言中，巨集變數 MQCIH_DEFAULT 包含上述值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQCIH MyCIH = {MQCIH_DEFAULT};
```

C 宣告

```
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 */
};
```

COBOL 宣告

```
**   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-STRUCLENGTH        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.
```

PL/I 宣告

```
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 */
```

High Level Assembler 宣告

```
MQCIH                   DSECT
MQCIH_STRUCID           DS   CL4  Structure identifier
MQCIH_VERSION           DS   F    Structure version number
MQCIH_STRUCLENGTH       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_ADSDESCRIPTOR     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)
```

Visual Basic 宣告

```
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
```

MQCMHO-建立訊息控點選項

下表彙總結構中的欄位。

表 *485: MQCMHO* 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
Options	選項	選項

MQCMHO 概觀

可用性: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows、 z/OS 及 WebSphere MQ 用戶端。

目的: **MQCMHO** 結構可讓應用程式指定控制如何建立訊息處理的選項。 結構是 **MQCRTMH** 呼叫上的輸入參數。

字集及編碼: **MQCMHO** 中的資料必須在應用程式的字集及應用程式的編碼中 (**MQENC_NATIVE**)。

MQCMHO 的欄位

MQCMHO 結構包含下列欄位; 這些欄位按 **字母順序**說明:

選項 *(MQLONG)*
此欄位一律是輸入欄位。 其起始值為 MQCMHO_DEFAULT_VALIDATION。

可以指定下列其中一個選項:

MQ 指令 _ 驗證

當呼叫 **MQSETMP** 來設定這個訊息控點中的內容時，會驗證內容名稱，以確定它:

- 不包含無效字元。
- 不會開始 JMS 或 usr.JMS ，但下列情況除外:
  - JMSCorrelationID
  - JMSReplyTo
  - JMSType
  - JMSXGroupID
  - JMSXGroupSeq

  這些名稱保留用於 JMS 內容。
- 不是下列其中一個關鍵字 (大小寫混合):
  - 和
  - BETWEEN
  - Esc 鍵
  - FALSE
  - IN
  - 是
  - 類似
  - 不是
  - NULL
  - 或
  - TRUE
- 不會開始「主體」。 或 Root。 (Root.MQMD 除外。)

如果內容是 MQ-defined (mq. *) 且名稱可辨識，內容描述子欄位會設為內容的正確值。 如果無法辨識內容，則內容描述子的 *Support* 欄位會設為 **MQPD_OPTIONAL**。

MQCMHO_DEFAULT_VALIDATION

此值指定發生內容名稱的預設驗證層次。

預設驗證層次相當於 **MQCMHO_VALIDATE** 指定的層次。

此值為預設值。

MQCMHO_NO_VALIDATION

不會驗證內容名稱。 請參閱 **MQCMHO_VALIDATE** 的說明。

預設選項: 如果不需要上述任何選項，則可以使用下列選項:

MQCMHO_NONE

所有選項都採用其預設值。 使用此值可指出尚未指定其他選項。 **MQCMHO_NONE** 輔助程式文件; 此選項並非預期與任何其他選項搭配使用，但由於其值為零，因此無法偵測到這類使用。

StrucId (MQCHAR4)
此欄位一律是輸入欄位。 其起始值為 MQCMHO_STRUC_ID。

這是結構 ID; 值必須是:

MQCMHO_STRUC_ID
建立訊息控點選項結構的 ID。

對於 C 程式設計語言，也會定義常數 **MQCMHO_STRUC_ID_ARRAY** ; 其值與 **MQCMHO_STRUC_ID** 相同，但卻是字元陣列而非字串。

版本 *(MQLONG)*
此欄位一律是輸入欄位。 其起始值為 MQCMHO_VERSION_1。

這是結構版本號碼; 值必須是:

MQCMHO_VERSION_1
Version-1 建立訊息控點選項結構。

下列常數指定現行版本的版本號碼:

MQCMHO_CURRENT_VERSION
建立訊息控點選項結構的現行版本。

MQCMHO 的起始值和語言宣告

表 *486: MQCMHO* 中欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQCMHO_STRUC_ID	'CMHO'
Version	MQCMHO_VERSION_1	1
Options	MQCMHO_DEFAULT_VALIDATION	0

附註:

1. 在 C 程式設計語言中，巨集變數 MQCMHO_DEFAULT 包含上述值。 您可以用下列方式來提供結構中欄位的起始值:

```
MQCMHO MyCMHO = {MQCMHO_DEFAULT};
```

C 宣告

```
struct tagMQCMHO {
  MQCHAR4   StrucId;         /* Structure identifier */
  MQLONG    Version;         /* Structure version number */
  MQLONG    Options;         /* Options that control the action of MQCRTMH */
};
```

COBOL 宣告

```
**   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.
```

PL/I 宣告

```
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 */
```

High Level Assembler 宣告

```
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)
```

MQCNO-連接選項

下表彙總結構中的欄位。

表 *487: MQCNO* 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
Options	控制 MQCONNX 動作的選項	選項
註: 如果 *Version* 小於 MQCNO_VERSION_2，則會忽略其餘欄位。		
ClientConnOffset	用戶端連線的 MQCD 結構偏移	ClientConn 偏移
ClientConnPtr	用戶端連線的 MQCD 結構位址	ClientConnPtr
註: 如果 *Version* 小於 MQCNO_VERSION_3，則會忽略其餘欄位。		
ConnTag	佇列管理程式連線標籤	ConnTag
註: 如果 *Version* 小於 MQCNO_VERSION_4，則會忽略其餘欄位。		
SSLConfigPtr	用戶端連線的 MQSCO 結構位址	SSLConfigPtr

表 *487: MQCNO* 中的欄位 *(繼續)*		
欄位	**說明**	**主題**
SSLConfigOffset	用戶端連線的 MQSCO 結構偏移	SSLConfigOffset
註: 如果 *Version* 小於 MQCNO_VERSION_5，則會忽略其餘欄位。		
ConnectionId	唯一連線 ID	ConnectionId
SecurityParmsOffset	安全參數	SecurityParms 偏移
SecurityParmsPtr	安全參數	SecurityParmsPtr

相關工作
使用 MQCONNX

MQCNO 概觀

可用性: 除了 MQCNO_VERSION_4: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows，以及連接至這些系統的 WebSphere MQ MQI 用戶端。

目的:MQCNO 結構可讓應用程式指定與本端佇列管理程式連線相關的選項。 此結構是 MQCONNX 呼叫的輸入/輸出參數。 如需使用共用控點及 MQCONNX 呼叫的相關資訊，請參閱 與 MQCONNX 共用 (執行緒無關) 連線。

版本: 為支援的程式設計語言提供的標頭、COPY 和 INCLUDE 檔案包含 MQCNO 的最新版本，但 *Version* 欄位的起始值設為 MQCNO_VERSION_1。 若要使用不在 version-1 結構中的欄位，應用程式必須將 *Version* 欄位設為所需版本的版本號碼。

字集及編碼:MQCNO 中的資料必須採用 *CodedCharSetId* 佇列管理程式屬性所提供的字集，以及 MQENC_NATIVE 所提供本端佇列管理程式的編碼。 不過，如果應用程式是以 WebSphere MQ MQI 用戶端身分執行，則該結構必須採用用戶端的字集及編碼。

MQCNO 的欄位

MQCNO 結構包含下列欄位; 這些欄位按 **字母順序**進行說明:

ClientConn 偏移 (MQLONG)
ClientConnOffset 是 MQCD 通道定義結構從 MQCNO 結構開始的偏移 (以位元組為單位)。 偏移可以是正數或負數。 此欄位是起始值為 0 的輸入欄位。

只有在發出 MQCONNX 呼叫的應用程式是以 WebSphere MQ MQI 用戶端身分執行時，才會使用 *ClientConnOffset* 。 如需如何使用此欄位的相關資訊，請參閱 *ClientConnPtr* 欄位的說明。

如果 *Version* 小於 MQCNO_VERSION_2，則會忽略此欄位。

ClientConnPtr (MQPTR)
ClientConnPtr 是輸入欄位。 它的起始值是那些支援指標的程式設計語言中的空值指標，否則是全空值位元組字串。

僅當發出 MQCONNX 呼叫的應用程式以 WebSphere MQ MQI 用戶端身分執行時，才會使用 *ClientConnOffset* 及 *ClientConnPtr* 。 透過指定其中一個或其他欄位，應用程式可以透過提供包含必要值的 MQCD 通道定義結構來控制用戶端連線通道的定義。

如果應用程式以 WebSphere MQ MQI 用戶端身分執行，但未提供 MQCD 結構，則會使用 MQSERVER 環境變數來選取通道定義。 如果未設定 MQSERVER ，則會使用用戶端通道表格。

如果應用程式不是以 WebSphere MQ MQI 用戶端身分執行，則會忽略 *ClientConnOffset* 及 *ClientConnPtr* 。

如果應用程式提供 MQCD 結構，請將列出的欄位設為必要值; 會忽略 MQCD 中的其他欄位。 您可以用空白來填補欄位長度的字串，或用空值字元來終止它們。 如需 MQCD 結構中欄位的相關資訊，請參閱 第 924 頁的『欄位』 。

MQCD 中的欄位	值
ChannelName	通道名稱。
Version	結構版本號碼。 不得小於 MQCD_VERSION_7。
TransportType	任何支援的傳輸類型。
ModeName	LU 6.2 模式名稱。
TpName	LU 6.2 交易程式名稱。
SecurityExit	通道安全結束程式的名稱。
SendExit	通道傳送結束程式的名稱。
ReceiveExit	通道接收結束程式的名稱。
MaxMsgLength	可透過用戶端連線通道傳送的訊息長度上限 (以位元組為單位)。
SecurityUserData	安全結束程式的使用者資料。
SendUserData	傳送結束程式的使用者資料。
ReceiveUserData	接收結束程式的使用者資料。
UserIdentifier	用來建立 LU 6.2 階段作業的使用者 ID。
Password	用來建立 LU 6.2 階段作業的密碼。
ConnectionName	連線名稱。
HeartbeatInterval	活動訊號流之間的時間 (以秒為單位)。
StrucLength	MQCD 結構的長度。
ExitNameLength	*SendExitPtr* 和 *ReceiveExitPtr* 所說明的結束程式名稱長度。 如果 *SendExitPtr* 或 *ReceiveExitPtr* 設為非空值指標的值，則必須大於零。
ExitDataLength	*SendUserDataPtr* 和 *ReceiveUserDataPtr* 所說明之結束程式資料的長度。 如果 *SendUserDataPtr* 或 *ReceiveUserDataPtr* 設為非空值指標的值，則必須大於零。
SendExitsDefined	*SendExitPtr* 所處理的傳送結束程式數目。 如果為零， *SendExit* 和 *SendUserData* 會提供結束程式名稱和資料。 如果大於零， *SendExitPtr* 和 *SendUserDataPtr* 會提供結束程式名稱和資料，且 *SendExit* 和 *SendUserData* 必須為空白。
ReceiveExitsDefined	*ReceiveExitPtr* 所處理的接收結束程式數目。 如果為零， *ReceiveExit* 和 *ReceiveUserData* 會提供結束程式名稱和資料。 如果大於零， *ReceiveExitPtr* 和 *ReceiveUserDataPtr* 會提供結束程式名稱和資料，且 *ReceiveExit* 和 *ReceiveUserData* 必須為空白。
SendExitPtr	第一個傳送結束程式的名稱位址。
SendUserDataPtr	第一個傳送結束程式的資料位址。
ReceiveExitPtr	第一個接收結束程式的名稱位址。
ReceiveUserDataPtr	第一個接收結束程式的資料位址。
LongRemoteUserIdLength	長遠端使用者 ID 的長度。
LongRemoteUserIdPtr	長遠端使用者 ID 的位址。
RemoteSecurityId	遠端安全 ID。

MQCD 中的欄位	值
SSLCipherSpec	SSL CipherSpec。
SSLPeerNamePtr	SSL 同層級名稱的位址。
SSLPeerNameLength	SSL 同層級名稱的長度。
KeepAliveInterval	針對通道的保持作用中計時，傳遞至通訊堆疊的值
LocalAddress	本端通訊位址，包括要使用之本端網路配接卡的 IP 位址，以及用於送出連線的埠範圍。

以下列兩種方式之一提供通道定義結構:

- 使用偏移欄位 *ClientConnOffset*

  在此情況下，應用程式必須宣告包含 MQCNO 的複合結構，後面接著通道定義結構 MQCD ， 並將 *ClientConnOffset* 設為通道定義結構從 MQCNO 開始的偏移。 請確定此偏移是正確的。 *ClientConnPtr* 必須設為空值指標或空值位元組。

  對於不支援指標資料類型的程式設計語言，或以不可攜至不同環境 (例如 COBOL 程式設計語言) 的方式來實作指標資料類型的程式設計語言，請使用 *ClientConnOffset* 。

  對於 Visual Basic 程式設計語言，這是稱為的複合結構 MQCNOCD 在標頭檔 CMQXB.BAS; 此結構包含 MQCNO 結構，後面接著 MQCD 結構。 呼叫 MQCNOCD_DEFAULTS 子常式來起始設定 MQCNOCD。 MQCNOCD 與 MQCONNX MQCONNX 呼叫的任何變式; 如需進一步詳細資料，請參閱 MQCONNX 呼叫的說明。
- 使用指標欄位 *ClientConnPtr*

  在此情況下，應用程式可以與 MQCNO 結構分開宣告通道定義結構，並將 *ClientConnPtr* 設為通道定義結構的位址。 將 *ClientConnOffset* 設為零。

  以可攜至不同環境 (例如， C 程式設計語言) 的方式，將 *ClientConnPtr* 用於支援指標資料類型的程式設計語言。

  在 C 程式設計語言中，您可以使用巨集變數 MQCD_CLIENT_CONN_DEFAULT ， 為比 MQCD_DEFAULT 所提供的起始值更適合在 MQCONNX 呼叫中使用的結構提供起始值。

不論您選擇何種技術，都只能使用 *ClientConnOffset* 和 *ClientConnPtr* 其中之一; 如果兩者都不是零，則呼叫會失敗，原因碼為 MQRC_CLIENT_CONN_ERROR。

當 MQCONNX 呼叫已完成時，不會再次參照 MQCD 結構。

如果 *Version* 小於 MQCNO_VERSION_2，則會忽略此欄位。

註: 在程式設計語言不支援指標資料類型的平台上，此欄位宣告為適當長度的位元組字串，起始值為 all-null 位元組字串。

ConnectionId (MQBYTE24)
ConnectionId 是唯一的 24 位元組 ID ， 可讓 WebSphere MQ 可靠地識別應用程式。 應用程式可以在 PUT 和 GET 呼叫中使用此 ID 來進行相關性。 在所有程式設計語言中，此輸出參數的起始值為 24 個空值位元組。

佇列管理程式會指派唯一 ID 給所有連線，但會建立它們。 如果 MQCONNX 建立與第 5 版 MQCNO 的連線，則應用程式可以從傳回的 MQCNO 判定 ConnectionId 。 在 WebSphere MQ 產生的所有其他 ID (例如 CorrelId、 MsgID 和 GroupId) 中，指派的 ID 保證是唯一的。

使用 PCF 指令「查詢連線」或 MQSC 指令 DISPLAY CONN ， 以使用 ConnectionId 來識別長時間執行的工作單元。 MQSC 指令 (CONN) 所使用的 ConnectionId 衍生自這裡傳回的 ConnectionId 。 PCF「查詢」及「停止連線」指令可以使用這裡傳回的 ConnectionId ， 而無需修改。

您可以使用 ConnectionId 來強制結束長時間執行的工作單元，方法是使用 PCF 指令「停止連線」或 MQSC 指令 STOP CONN 來指定 ConnectionId 。 如需使用這些指令的相關資訊，請參閱 停止連線 及 STOP CONN 。

如果版本小於 MQCNO_VERSION_5，則不會傳回此欄位。

此欄位的長度由 MQ_CONNECTION_ID_LENGTH 提供。

ConnTag (MQBYTE128)
ConnTag 是佇列管理程式在此連線期間與受應用程式影響的資源相關聯的標籤。 每一個應用程式或應用程式實例都必須對標籤使用不同的值，以便佇列管理程式可以正確地序列化受影響資源的存取權。 此欄位是輸入欄位，其起始值為 MQCT_NONE。

如需不同應用程式要使用之值的進一步詳細資料，請參閱 MQCNO_ * _CONN_TAG_ * 選項的說明。 當應用程式終止或發出 MQDISC 呼叫時，標籤不再有效。

註: ASCII 或 EBCDIC 中以大寫、小寫或混合大小寫的 MQ 開頭的連線標籤值保留供 IBM 產品使用。 請勿使用以這些字母開頭的連線標籤值。

如果您不需要任何標籤，請使用下列特殊值:

MQCT_NONE
欄位長度的值為二進位零。

對於 C 程式設計語言，也會定義常數 MQCT_NONE_ARRAY; 此常數具有與 MQCT_NONE 相同的值，但它是字元陣列而非字串。

當連接至 z/OS 佇列管理程式時，會使用這個欄位。 在其他環境中，請指定值 MQCT_NONE。

此欄位的長度由 MQ_CONN_TAG_LENGTH 提供。 如果 *Version* 小於 MQCNO_VERSION_3，則會忽略此欄位。

選項 *(MQLONG)*
控制 MQCONNX 動作的選項。

帳戶選項

如果 *AccountingConnOverride* 佇列管理程式屬性設為 MQMON_ENABLED，下列選項可控制帳戶類型:

MQCNO_ACCOUNTING_MQI_ENABLED
透過將 *MQIAccounting* 屬性設為 MQMON_OFF，在佇列管理程式定義中關閉監視資料收集時，設定此旗標會啟用 MQI 結算資料收集。

MQCNO_ACCOUNTING_MQI_DISABLED
透過將 *MQIAccounting* 屬性設為 MQMON_OFF，在佇列管理程式定義中關閉監視資料收集時，設定此旗標會停止 MQI 結算資料收集。

MQCNO_ACCOUNTING_Q_ENABLED
透過將 *MQIAccounting* 屬性設為 MQMON_OFF，在佇列管理程式定義中關閉佇列結算資料收集時，設定此旗標會針對在其佇列定義的 *MQIAccounting* 欄位中指定佇列管理程式的那些佇列啟用結算資料收集。

MQCNO_ACCOUNTING_Q_DISABLED
透過將 *MQIAccounting* 屬性設為 MQMON_OFF，在佇列管理程式定義中關閉佇列結算資料收集時，設定此旗標會關閉在其佇列定義的 *MQIAccounting* 欄位中指定佇列管理程式的那些佇列的結算資料收集。

如果未定義任何這些旗標，則連線的結算會如「佇列管理程式」屬性中所定義。

連結選項

下列選項控制要使用的 WebSphere MQ 連結類型。 僅指定下列其中一個選項:

MQCN_STANDARD_BINDING
應用程式及本端佇列管理程式代理程式 (管理佇列作業的元件) 以個別執行單元執行 (通常在個別處理程序中)。 此安排會維護佇列管理程式的完整性; 亦即，它會保護佇列管理程式免受錯誤程式的影響。

如果佇列管理程式支援多種連結類型，且您設定 MQCNO_STANDARD_BINDING，則佇列管理程式會使用 qm.ini 檔案 (或對等的 Windows 登錄項目) 中 *Connection* 段落中的 *DefaultBindType* 屬性，來選取實際的連結類型。 如果未定義此段落，或該值無法使用或不適用於應用程式，則佇列管理程式會選取適當的連結類型。 佇列管理程式會設定連接選項中使用的實際連結類型。

如果應用程式可能尚未完全測試，或可能不可靠或不可靠，請使用 MQCNO_STANDARD_BINDING。 MQCN_STANDARD_BINDING 是預設值。

在所有環境中都支援此選項。

如果您要鏈結至 mqm 檔案庫，則會先嘗試使用預設連結類型的標準伺服器連線。 如果無法載入基礎伺服器程式庫，則會改為嘗試用戶端連線。

- 如果指定 MQ_CONNECT_TYPE 環境變數，則可以提供下列其中一個選項，以在指定 MQCNO_STANDARD_BINDING 時變更 MQCONN 或 MQCONNX 的行為。 (如果 MQ_CONNECT_TYPE 設為 LOCAL 或 STANDARD 來指定 MQCNO_FASTPATH_BINDING，則例外狀況是容許管理者將捷徑連線降級，而不需要對應用程式進行相關變更:

值	意義
用戶端	只嘗試用戶端連線。
Fastpath	此值在舊版中受支援，但如果指定的話，現在會被忽略。
本端	只嘗試伺服器連線。 捷徑連線會降級為標準伺服器連線。
標準	支援與舊版相容。 此值現在被視為 LOCAL。

- 如果在呼叫 MQCONNX 時未設定 MQ_CONNECT_TYPE 環境變數，則會嘗試使用預設連結類型的標準伺服器連線。 如果伺服器媒體庫無法載入，則會嘗試用戶端連線。

MQCNO_FASTPATH_BINDING
應用程式和本端佇列管理程式代理程式是相同執行單元的一部分。 這與一般連結方法相反，應用程式和本端佇列管理程式代理程式在個別執行單元中執行。

如果佇列管理程式不支援此類型的連結，則會忽略 MQCNO_FASTPATH_BINDING; 處理程序會繼續進行，好像尚未指定選項一樣。

MQCNO_FASTPATH_BINDING 可以在多個處理程序耗用比應用程式所使用整體資源更多資源的情況下發揮優勢。使用捷徑連結的應用程式稱為 授信應用程式。

在決定是否使用捷徑連結時，請考量下列重要要點:

- 使用 MQCNO_FASTPATH_BINDING 選項不會防止應用程式變更或毀損屬於佇列管理程式的訊息及其他資料區。 只有在您已完全評估這些問題的情況下，才使用這個選項。
- 應用程式不得搭配使用非同步信號或計時器岔斷 (例如 sigkill) 與 MQCNO_FASTPATH_BINDING。 共用記憶體區段的使用也有一些限制。
- 應用程式必須使用 MQDISC 呼叫來中斷與佇列管理程式的連線。
- 在您使用 endmqm 指令結束佇列管理程式之前，必須先完成應用程式。
- 在 IBM i 上，工作必須在屬於 QMQMADM 群組的使用者設定檔下執行。 此外，程式不得異常停止，否則可能會發生無法預期的結果。
- 在 UNIX 系統上， mqm 使用者 ID 必須是有效使用者 ID， mqm 群組 ID 必須是有效群組 ID。 若要讓應用程式以這種方式執行，請配置程式，使其由 mqm 使用者 ID 及 mqm 群組 ID 所擁有，然後在程式上設定 setuid 及 setgid 許可權位元。

  WebSphere MQ 物件權限管理程式 (OAM) 仍會使用實際使用者 ID 進行權限檢查。
- 在 Windows 上，程式必須是 mqm 群組的成員。 64 位元應用程式不支援捷徑連結。

下列環境支援 MQCNO_FASTPATH_BINDING 選項: AIX、 HP-UX、 IBM i、Solaris、 Linux 及 Windows。 在 z/OS 上，接受但忽略此選項。

如需使用授信應用程式之含意的相關資訊，請參閱 授信應用程式的限制 。

MQCNO_SHARED_BINDING

使用 MQCNO_SHARED_BINDING，應用程式及本端佇列管理程式代理程式會共用部分資源。 如果佇列管理程式不支援此類型的連結，則會忽略 MQCNO_SHARED_BINDING。 繼續執行處理程序，好像尚未指定選項。

MQCNO_ISOLATED_BINDING

在此情況下，應用程式程序與本端佇列管理程式代理程式會彼此隔離，因為它們不會共用資源。 如果佇列管理程式不支援此類型的連結，則會忽略 MQCNO_ISOLATED_BINDING。 繼續執行處理程序，好像尚未指定選項。

MQCNO_CLIENT_BINDING

指定此選項，只讓應用程式嘗試用戶端連線。 此選項具有下列限制:

- 在 z/OS 上使用 MQRC_OPTIONS_ERROR 拒絕 MQCNO_CLIENT_BINDING。
- 如果使用 MQCNO_STANDARD_BINDING 以外的任何 MQCNO 連結選項指定 MQCNO_CLIENT_BINDING，則會以 MQRC_OPTIONS_ERROR 拒絕 MQCNO_CLIENT_BINDING。
- MQCNO_CLIENT_BINDING 無法用於 Java 或 .NET，因為它們有自己的機制來選擇連結類型。
- 如果在呼叫 MQCONNX 時未設定 MQ_CONNECT_TYPE 環境變數，則會嘗試使用預設連結類型的標準伺服器連線。 如果伺服器媒體庫無法載入，則會嘗試用戶端連線。

MQCNO_LOCAL_BINDING

指定此選項可讓應用程式嘗試伺服器連線。 如果同時指定 MQCNO_FASTPATH_BINDING、MQCNO_ISOLATED_BINDING 或 MQCNO_SHARED_BINDING，則連線會改為屬於該類型，並記載在本節中。 否則，會使用預設連結類型來嘗試標準伺服器連線。 MQCNO_LOCAL_BINDING 具有下列限制:

- 在 z/OS 上，會忽略 MQCNO_LOCAL_BINDING。
- 如果 MQCNO_RECONNECT_AS_DEF 以外的任何 MQCNO 重新連接選項指定 MQCNO_LOCAL_BINDING，則會以 MQRC_OPTIONS_ERROR 拒絕 MQCNO_LOCAL_BINDING。
- MQCNO_LOCAL_BINDING 無法用於 Java 或 .NET，因為它們有自己的機制來選擇連結類型。
- 如果在呼叫 MQCONNX 時未設定 MQ_CONNECT_TYPE 環境變數，則會嘗試使用預設連結類型的標準伺服器連線。 如果伺服器媒體庫無法載入，則會嘗試用戶端連線。

在 AIX、 HP-UX、Solaris、 Linux 及 Windows 上，您可以將環境變數 MQ_CONNECT_TYPE 與 *Options* 欄位指定的連結類型搭配使用，以控制使用的連結類型。 如果您指定此環境變數，則它必須具有值 FASTPATH 或 STANDARD ; 如果它具有不同的值，則會忽略它。 環境變數的值區分大小寫; 如需相關資訊，請參閱 MQCONNX 環境變數 。

環境變數與 *Options* 欄位互動如下:

- 如果您省略環境變數，或為其提供不受支援的值，則快速路徑連結的使用僅由 *Options* 欄位決定。
- 如果您提供受支援的值給環境變數，則只有在 同時 環境變數及 *Options* 欄位指定捷徑連結時，才會使用捷徑連結。

連線標籤選項

只有在連接至 z/OS 佇列管理程式時，才支援這些選項，它們會控制連線標籤 *ConnTag* 的使用。 您只能指定下列其中一個選項:

MQCNO_SERIALIZE_CONN_TAG_Q_MGR

此選項要求在本端佇列管理程式內專用連線標籤。 如果本端佇列管理程式中已使用連線標籤，則 MQCONNX 呼叫會失敗，原因碼為 MQRC_CONN_TAG_IN_USE。 使用本端佇列管理程式所屬佇列共用群組中其他位置的連線標籤，不會影響呼叫的結果。

MQCNO_SERIALIZE_CONN_TAG_QSG

此選項要求專用本端佇列管理程式所屬佇列共用群組內的連線標籤。 如果連線標籤已在佇列共用群組中使用，則 MQCONNX 呼叫會失敗，原因碼為 MQRC_CONN_TAG_IN_USE。

MQCNO_RESTRICT_CONN_TAG_Q_MGR

此選項要求在本端佇列管理程式內共用連線標籤。 如果連線標籤已在本端佇列管理程式中使用，則當發出要求的應用程式在與標籤現有使用者相同的處理範圍中執行時， MQCONNX 呼叫可能會成功。 如果未滿足此條件，則 MQCONNX 呼叫會失敗，原因碼為 MQRC_CONN_TAG_IN_USE。 使用本端佇列管理程式所屬佇列共用群組中其他位置的連線標籤，不會影響呼叫的結果。

- 應用程式必須在相同的 MVS 位址空間內執行，才能共用連線標籤。 如果使用連線標籤的應用程式是用戶端應用程式，則不容許 MQCNO_RESTRICT_CONN_TAG_Q_MGR。

MQCNO_RESTRICT_CONN_TAG_QSG

此選項要求在本端佇列管理程式所屬的佇列共用群組內共用連線標籤。 如果連線標籤已在佇列共用群組中使用，則 MQCONNX 呼叫可以成功，前提是發出要求的應用程式在相同的處理範圍中執行，並以標籤的現有使用者身分連接至相同的佇列管理程式。

如果未滿足這些條件，則 MQCONNX 呼叫會失敗，原因碼為 MQRC_CONN_TAG_IN_USE。

- 應用程式必須在相同的 MVS 位址空間內執行，才能共用連線標籤。 如果使用連線標籤的應用程式是用戶端應用程式，則不容許 MQCNO_RESTRICT_CONN_TAG_QSG。

如果未指定任何這些選項，則不會使用 *ConnTag* 。 如果 *Version* 小於 MQCNO_VERSION_3，則這些選項無效。

控點共用選項

在下列環境中支援這些選項: AIX、 HP-UX、 IBM i、Solaris、 Linux 及 Windows。 它們控制在相同處理程序內不同執行緒 (平行處理單元) 之間共用控點。 您只能指定下列其中一個選項:

MQCNO_HANDLE_SHARE_NONE

此選項指出連線及物件控點只能由導致配置控點的執行緒使用 (亦即，發出 MQCONN、MQCONNX 或 MQOPEN 呼叫的執行緒)。 控點無法由屬於相同處理程序的其他執行緒使用。

MQCNO_HANDLE_SHARE_BLOCK

此選項指出處理程序的一個執行緒所配置的連線及物件控點，可供屬於相同處理程序的其他執行緒使用。 不過，一次只能有一個執行緒可以使用任何特定的控點; 也就是說，只允許循序使用一個控點。 如果執行緒嘗試使用另一個執行緒已在使用的控點，則呼叫會封鎖 (等待) ，直到控點變成可用為止。

MQCNO_HANDLE_SHARE_NO_BLOCK

這與 MQCNO_HANDLE_SHARE_BLOCK 相同，但如果另一個執行緒正在使用控點，則呼叫會立即完成 MQCC_FAILED 和 MQRC_CALL_IN_PROGRESS ，而不是封鎖直到控點變成可用為止。

執行緒可以有零個或一個非共用控點:

- 每一個指定 MQCNO_HANDLE_SHARE_NONE 的 MQCONN 或 MQCONNX 呼叫都會在第一次呼叫中傳回新的非共用控點，而在第二次及後續呼叫中則會傳回相同的非共用控點 (假設沒有中間 MQDISC 呼叫)。 原因碼是第二個及更新版本呼叫的 MQRC_ALREADY_CONNECTED。
- 每一個指定 MQCNO_HANDLE_SHARE_BLOCK 或 MQCNO_HANDLE_SHARE_NO_BLOCK 的 MQCONNX 呼叫都會在每一個呼叫上傳回新的共用控點。

物件控點繼承與建立物件控點之 MQOPEN 呼叫上指定的連線控點相同的共用內容。 此外，工作單元繼承與用來啟動工作單元的連線控點相同的共用內容; 如果使用共用控點在一個執行緒中啟動工作單元，則可以使用相同的控點在另一個執行緒中更新工作單元。

如果您未指定控點共用選項，則預設值由環境決定:

- 在 Microsoft Transaction Server (MTS) 環境中，預設值與 MQCNO_HANDLE_SHARE_BLOCK 相同。

- 在其他環境中，預設值與 MQCNO_HANDLE_SHARE_NONE 相同。

重新連線選項

重新連線選項決定連線是否可重新連接。 只能重新連接用戶端連線。

MQCNO_RECONNECT_AS_DEF

重新連線選項會解析成其預設值。 如果未設定預設值，則此選項的值會解析為 DISABLED 。 選項的值會傳遞至伺服器，並可由 PCF 及 MQSC 查詢。

MQCNO_RECONNECT

應用程式可以重新連接至與 MQCONNX 的 QmgrName 參數值一致的任何佇列管理程式。 只有在用戶端應用程式與其起始建立連線的佇列管理程式之間沒有親緣性時，才使用 MQCNO_RECONNECT 選項。 選項的值會傳遞至伺服器，並可由 PCF 及 MQSC 查詢。

已停用 MQCNO_RECONNECT_DISABLED

無法重新連接應用程式。 選項的值不會傳遞至伺服器。

MQCNO_RECONNECT_Q_MGR

應用程式只能重新連接至它原先連接的佇列管理程式。 如果用戶端可以重新連接，但用戶端應用程式與它最初建立連線的佇列管理程式之間有親緣性，請使用此值。 如果您要用戶端自動重新連接至高可用性佇列管理程式的待命實例，請選擇此值。 選項的值會傳遞至伺服器，並可由 PCF 及 MQSC 查詢。

僅針對用戶端連線使用選項 MQCNO_RECONNECT、MQCNO_RECONNECT_DISABLED 及 MQCNO_RECONNECT_Q_MGR。 如果選項用於連結連線，則 MQCONNX 會失敗，並出現完成碼 MQCC_FAILED 和原因碼 MQRC_OPTIONS_ERROR。 適用於 Java 的 WebSphere MQ 類別不支援自動用戶端重新連接

交談共用選項

下列選項僅適用於 TCP/IP 用戶端連線。 對於 SNA、SPX 及 NetBios 通道，會忽略這些值，且通道會如同在舊版產品中一樣執行

MQCNO_NO_CONV_SHARING

此選項不允許交談共用。

您可以在大量載入交談的情況下使用 MQCNO_NO_CONV_SHARING ，因此在共用交談所在通道實例的伺服器連線端可能發生競用。 當連接至支援交談共用的通道時， MQCNO_NO_CNV_SHARING 的行為類似於 sharecnv (1) ，當連接至不支援交談共用的通道時，則為 sharecnv (0)。

MQCNO_ALL_CONVS_SHARE

此選項允許交談共用; 應用程式不會對通道實例上的連線數設定任何限制。 此選項是預設值。

如果應用程式指出通道實例可以共用，但通道伺服器連線端上的 *SharingConversations* (SHARECNV) 定義設為 1 ，則不會進行共用，且不會對應用程式發出警告。

同樣地，如果應用程式指出允許共用，但伺服器連線 *SharingConversations* 定義設為零，則不會提供警告，且應用程式在 7.0 版之前的產品版本中會展現與用戶端相同的行為; 系統會忽略與共用交談相關的應用程式設定。

MQCNO_NO_CONV_SHARING 與 MQCNO_ALL_CONVS_SHARE 互斥。 如果在特定連線上同時指定兩個選項，則會拒絕連線，原因碼為 MQRC_OPTIONS_ERROR。

通道定義選項

下列選項控制使用 MQCNO 中所傳遞的通道定義結構:

MQCNO_CD_FOR_OUTPUT_ONLY

此選項允許 MQCNO 中的通道定義結構僅用於傳回在成功 MQCONNX 呼叫中使用的通道名稱。

如果未提供有效的通道定義結構，則呼叫會失敗，原因碼為 MQRC_CD_ERROR。

如果應用程式不是以用戶端身分執行，則會忽略該選項。

可以使用 MQCNO_USE_CD_SELECTION 選項在後續 MQCONNX 呼叫中使用傳回的通道名稱，以使用相同的通道定義重新連接。 當用戶端通道表格中有多個適用的通道定義時，這會很有用。

MQCNO_USE_CD_SELECTION

此選項允許 MQCONNX 呼叫使用 MQCNO 所傳遞的通道定義結構中包含的通道名稱來連接。

如果設定 MQSERVER 環境變數，則會使用它所定義的通道定義。 如果未設定 MQSERVER ，則會使用用戶端通道表格。

如果找不到通道名稱與佇列管理程式名稱相符的通道定義，則呼叫會失敗，原因碼為 MQRC_Q_MGR_NAME_ERROR。

如果未提供有效的通道定義結構，則呼叫會失敗，原因碼為 MQRC_CD_ERROR。

如果應用程式不是以用戶端身分執行，則會忽略該選項。

預設選項

如果您不需要上述任何選項，則可以使用下列選項:

MQCNO_NONE

未指定任何選項。

請使用 MQCNO_NONE 來協助程式文件。 此選項並非預期與任何其他 MQCNO_ * 選項搭配使用，但由於其值為零，因此無法偵測到此類使用。

SecurityParms 偏移 (MQLONG)
SecurityParms 偏移是 MQCSP 結構從 MQCNO 結構開始算起的偏移 (以位元組為單位)。 偏移可以是正數或負數。 此欄位是起始值為 0 的輸入欄位。

如果 版小於 MQCNO_VERSION_5，則會忽略此欄位。

MQCSP 結構定義在 第 295 頁的『MQCSP-安全參數』中。

SecurityParmsPtr (PMQCSP)
SecurityParmsPtr 是 MQCSP 結構的位址，用來指定使用者 ID 和密碼，以供授權服務進行鑑別。 此欄位是輸入欄位，且其起始值是空值指標或空值位元組。

如果 版小於 MQCNO_VERSION_5，則會忽略此欄位。

MQCSP 結構定義在 第 295 頁的『MQCSP-安全參數』中。

SSLConfigOffset (MQLONG)
SSLConfigOffset 是 MQSCO 結構從 MQCNO 結構開始的偏移 (以位元組為單位)。 偏移可以是正數或負數。 此欄位是起始值為 0 的輸入欄位。

只有在發出 MQCONNX 呼叫的應用程式是以 WebSphere MQ MQI 用戶端身分執行時，才會使用 *SSLConfigOffset* 。 如需如何使用此欄位的相關資訊，請參閱 *SSLConfigPtr* 欄位的說明。

如果 *Version* 小於 MQCNO_VERSION_4，則會忽略此欄位。

SSLConfigPtr (PMQSCO)
SSLConfigPtr 是輸入欄位。 在那些支援指標的程式設計語言中， itsinitial 值是空值指標，否則是全空值位元組字串。

只有在發出 MQCONNX 呼叫的應用程式是以 WebSphere MQ MQI 用戶端身分執行，且通道通訊協定是 TCP/IP 時，才會使用 *SSLConfigPtr* 及 *SSLConfigOffset* 。 如果應用程式不是以 WebSphere MQ 用戶端執行，或通道通訊協定不是 TCP/IP ，則會忽略 *SSLConfigPtr* 及 *SSLConfigOffset* 。

透過指定 *SSLConfigPtr* 或 *SSLConfigOffset*，加上 *ClientConnPtr* 或 *ClientConnOffset*，應用程式可以控制使用 SSL 進行用戶端連線。 以這種方式指定 SSL 資訊時，會忽略環境變數 MQSSLKEYR 和 MQSSLCRYP; 也會忽略用戶端通道定義表 (CCDT) 中的任何 SSL 相關資訊。

只能在下列項目上指定 SSL 資訊:

- 用戶端程序的第一個 MQCONNX 呼叫，或
- 使用 MQDISC 結束所有先前對佇列管理程式的 SSL/TLS 連線時的後續 MQCONNX 呼叫。

這些是唯一可以起始設定整個處理程序 SSL 環境的狀態。 如果在 SSL 環境已存在時發出 MQCONNX 呼叫指定 SSL 資訊，則會忽略呼叫的 SSL 資訊，並使用現有 SSL 環境建立連線; 在此情況下，呼叫會傳回完成碼 MQCC_WARNING 及原因碼 MQRC_SSL_ALREADY_INITIALIZED。

您可以使用與 MQCD 結構相同的方式來提供 MQSCO 結構，方法是在 *SSLConfigPtr* 中指定位址，或在 *SSLConfigOffset* 中指定偏移; 如需如何執行此動作的詳細資料，請參閱 *ClientConnPtr* 的說明。 不過，您只能使用 *SSLConfigPtr* 和 *SSLConfigOffset* 中的一個; 呼叫會失敗，原因碼為 MQRC_SSL_CONFIG_ERROR。 如果兩者都不是零。

一旦 MQCONNX 呼叫完成，就不會再次參照 MQSCO 結構。

如果 *Version* 小於 MQCNO_VERSION_4，則會忽略此欄位。

註: 在程式設計語言不支援指標資料類型的平台上，此欄位宣告為適當長度的位元組字串。

StrucId (MQCHAR4)
StrucId 一律是輸入欄位。 其起始值為 MQCNO_STRUC_ID。

值必須為:

MQCNO_STRUC_ID
連接選項結構的 ID。

對於 C 程式設計語言，也會定義常數 MQCNO_STRUC_ID_ARRAY; 此常數具有與 MQCNO_STRUC_ID 相同的值，但它是字元陣列而非字串。

版本 *(MQLONG)*
版本一律是輸入欄位。 其起始值為 MQCNO_VERSION_1。

此值必須是下列其中一個:

MQCNO_VERSION_1
Version-1 連接選項結構。

MQCNO_VERSION_2
Version-2 連接選項結構。

MQCNO_VERSION_3
Version-3 連接選項結構。

MQCNO_VERSION_4
Version-4 連接選項結構。

MQCNO_VERSION_5
Version-5 連接選項結構。

此版本的 MQCNO 結構在 z/OS 上延伸 MQCNO_VERSION_3 ，在所有其他平台上延伸 MQCNO_VERSION_4 。

僅存在於結構最新版本中的欄位會在欄位說明中如此識別。 下列常數指定現行版本的版本號碼:

MQCNO_CURRENT_VERSION
連接選項結構的 現行 版本。

MQCNO 的起始值和語言宣告

表 *488: MQCNO* 中 *MQCNO* 的欄位起始值

欄位名稱	**常數名稱**	**常數值**
StrucId	MQCNO_STRUC_ID	'CNO¬'
Version	MQCNO_VERSION_1	1
Options	MQCNO_NONE	0
ClientConnOffset	無	0
ClientConnPtr	無	空值指標或空值位元組
ConnTag	MQCT_NONE	空值
SSLConfigPtr	無	空值指標或空值位元組
SSLConfigOffset	無	0
ConnectionId	無	空值指標或空值位元組
SecurityParmsOffset	無	空值指標或空值位元組
SecurityParmsPtr	無	空值指標或空值位元組

附註:

1. 符號 ¬ 代表單一空白字元。
2. 在 C 程式設計語言中，巨集變數 MQCNO_DEFAULT 包含上述值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQCNO MyCNO = {MQCNO_DEFAULT};
```

C 宣告

```
typedef struct tagMQCNO MQCNO;
struct tagMQCNO {
  MQCHAR4    StrucId;           /* Structure identifier */
  MQLONG     Version;           /* Structure version number */
  MQLONG     Options;           /* Options that control the action of
                                   MQCONNX */
  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 */
 };
```

COBOL 宣告

```
**   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 MQCONNX
   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-CONNTAG          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.
```

PL/I 宣告

```
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 MQCONNX */
  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 */
```

High Level Assembler 宣告

```
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
*                                   MQCONNX
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)
```

Visual Basic 宣告

```
Type MQCNO
  StrucId          As String*4  'Structure identifier'
  Version          As Long      'Structure version number'
  Options          As Long      'Options that control the action of'
                                'MQCONNX'
  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'
   End Type
```

MQCSP-安全參數

下表彙總結構中的欄位。

表 *489: MQCSP* 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
AuthenticationType	鑑別的類型	AuthenticationType
Reserved1	IBM i 上指標對齊的必要項目	Reserved1
CSPUserIdPtr	使用者 ID 的位址	CSPUserIdPtr
CSPUserIdOffset	使用者 ID 的偏移	CSPUserId 偏移
CSPUserIdLength	使用者 ID 的長度	CSPUserId 長度
Reserved2	IBM i 上指標對齊的必要項目	Reserved2
CSPPasswordPtr	密碼的位址	CSPPasswordPtr
CSPPasswordOffset	密碼的偏移	CSPPasswordOffset
CSPPasswordLength	密碼的長度	CSPPasswordLength

MQCSP 概觀

可用性: 所有 WebSphere MQ 產品。

目的:MQCSP 結構可讓授權服務鑑別使用者 ID 和密碼。 您可以在 MQCONNX 呼叫中指定 MQCSP 連線安全參數結構。

字集及編碼:MQCSP 中的資料必須採用本端佇列管理程式的字集及編碼; 這些分別由 *CodedCharSetId* 佇列管理程式屬性及 MQENC_NATIVE 提供。

MQCSP 的欄位

MQCSP 結構包含下列欄位; 這些欄位按 **字母順序**說明:

AuthenticationType (MQLONG)
AuthenticationType 是輸入欄位。 其起始值為 MQCSP_AUTH_NONE。

這是要執行的鑑別類型。 有效值為:

MQCSP_AUTH_NONE
請勿使用使用者 ID 和密碼欄位。

MQCSP_AUTH_USER_ID_AND_PWD
鑑別使用者 ID 和密碼欄位。

CSPPasswordLength (MQLONG)
此欄位是要在鑑別中使用的密碼長度。

密碼的長度上限取決於平台，請參閱 使用者 ID。 如果密碼長度大於允許的長度上限，則鑑別要求會失敗，並產生 MQRC_NOT_AUTHORIZED。

此欄位是輸入欄位。 此欄位的起始值為 0。

CSPPasswordOffset (MQLONG)
這是要在鑑別中使用的密碼偏移 (以位元組為單位)。 偏移可以是正數或負數。

這是輸入欄位。 此欄位的起始值為 0。

CSPPasswordPtr (MQPTR)
這是要在鑑別中使用的密碼位址 (以位元組為單位)。

這是輸入欄位。 在那些支援指標的程式設計語言中，此欄位的起始值是空值指標，否則是全空值位元組字串。 如果 *Version* 小於 MQCNO_VERSION_5，則會忽略此欄位。

CSPUserId 長度 (MQLONG)
此欄位是要在鑑別中使用的使用者 ID 長度。

使用者 ID 的長度上限取決於平台，請參閱 使用者 ID。 如果使用者 ID 的長度大於允許的長度上限，則鑑別要求會因 MQRC_NOT_AUTHORIZED 而失敗。

此欄位是輸入欄位。 此欄位的起始值為 0。

CSPUserId 偏移 (MQLONG)
這是要在鑑別中使用之使用者 ID 的偏移 (以位元組為單位)。 偏移可以是正數或負數。

這是輸入欄位。 此欄位的起始值為 0。

CSPUserIdPtr (MQPTR)
這是要在鑑別中使用之使用者 ID 的位址 (以位元組為單位)。

這是輸入欄位。 在那些支援指標的程式設計語言中，此欄位的起始值是空值指標，否則是全空值位元組字串。 如果 *Version* 小於 MQCNO_VERSION_5，則會忽略此欄位。

Reserved1 (MQBYTE4)
保留欄位， IBM i 上的指標對齊方式需要此欄位。

這是輸入欄位。 此欄位的起始值都是空值。

Reserved2 (MQBYTE8)
保留欄位， IBM i 上的指標對齊方式需要此欄位。

這是輸入欄位。 此欄位的起始值都是空值。

StrucId (MQCHAR4)
結構 ID。

值必須為:

MQCSP_STRUC_ID
安全參數結構的 ID。

若為 C 程式設計語言，也會定義常數 MQCSP_STRUC_ID_ARRAY; 此值與 MQCSP_STRUC_ID 相同，但卻是字元陣列而非字串。

這一律是輸入欄位。 此欄位的起始值是 MQCSPSTRUC_ID。

版本 *(MQLONG)*
結構版本號碼。

值必須為:

MQCSP_VERSION_1
Version-1 安全參數結構。

下列常數指定現行版本的版本號碼:

MQCSP_CURRENT_VERSION
安全參數結構的現行版本。

這一律是輸入欄位。 此欄位的起始值為 MQCSP_VERSION_1。

MQCSP 的起始值和語言宣告

表 *490: MQCSP for MQCSP* 中欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQCSP_STRUC_ID	'CSP¬'
Version	MQCSP_VERSION_1	1
AuthenticationType	無	MQCSP_AUTH_NONE
Reserved1	無	空字串或空白
CSPUserIdPtr	無	空值指標或空值位元組
CSPUserIdOffset	無	0
CSPUserIdLength	無	0
Reserved2	無	空字串或空白
CSPPasswordPtr	無	空值指標或空值位元組
CSPPasswordOffset	無	0
CSPPasswordLength	無	0

附註:

1. 符號 ¬ 代表單一空白字元。
2. 在 C 程式設計語言中，巨集變數 MQCSP_DEFAULT 包含上述值。 您可以用下列方式來提供結構中欄位的起始值:

```
MQCSP MyCSP = {MQCSP_DEFAULT};
```

C 宣告

```
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 */
                                 alignment */
  MQPTR      CSPPasswordPtr;     /* Address of password */
  MQLONG     CSPPasswordOffset;  /* Offset of password */
```

```
  MQLONG     CSPPasswordLength;  /* Length of password */
  };
```

COBOL 宣告

```
**   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.
```

PL/I 宣告

```
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 */
```

Visual Basic 宣告

```
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
```

MQCTLO-控制回呼選項結構

下表彙總結構中的欄位。 指定控制回呼函數的結構。

表 491: MQCTLO 中的欄位		
欄位	**說明**	**主題**
StrucID	結構 ID	StrucID
Version	結構版本號碼	版本
Options	選項	選項
Reserved	保留欄位	選項
ConnectionArea	要使用的回呼函數欄位	ConnectionArea

MQCTLO 概觀

可用性: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows、 z/OS 及連接至這些系統的 WebSphere MQ MQI 用戶端。 MQCTLO 結構概觀。

目的:MQCTLO 結構用來指定與控制回呼函數相關的選項。

此結構是 第 593 頁的『MQCTL-控制回呼』 呼叫上的輸入及輸出參數。

版本:MQCTLO 的現行版本是 MQCTLO_VERSION_1。

字集及編碼:MQCTLO 中的資料必須採用 *CodedCharSetId* 佇列管理程式屬性所提供的字集，以及 MQENC_NATIVE 所提供本端佇列管理程式的編碼。 不過，如果應用程式以 MQ MQI 用戶端身分執行，則該結構必須採用用戶端的字集及編碼。

MQCTLO 的欄位

MQCTLO 結構的欄位英文字母清單。

MQCTLO 結構包含下列欄位; 這些欄位按字母順序說明:

ConnectionArea (MQPTR)
控制選項結構- ConnectionArea 欄位

這是可供回呼函數使用的欄位。

佇列管理程式不會根據此欄位的內容來做出任何決策，它會依原樣傳遞至 MQCBC 結構中的 第 252 頁的『ConnectionArea (MQPTR)』 欄位，這是回呼的輸入參數。

對於 MQOP_START 及 MQOP_START_WAIT 以外的所有作業，將忽略此欄位。

這是回呼函數的輸入及輸出欄位。 此欄位的起始值是空值指標或空值位元組。

選項 *(MQLONG)*
控制選項結構-選項欄位

控制 MQCTL 動作的選項。

MQCTLO_FAIL_IF_QUIESCING
如果佇列管理程式或連線處於靜止狀態，則強制 MQCTL 呼叫失敗。

在 MQGMO 呼叫上傳遞的 MQGMO 選項中指定 MQGMO_FAIL_IF_QUIESCING ，以在訊息消費者靜止時向訊息消費者發出通知。

MQ CTLO_THREAD_AFFINITY
這個選項會通知系統，應用程式要求在相同執行緒上呼叫相同連線的所有訊息消費者。 此執行緒將用於消費者的所有呼叫，直到連線停止為止。

預設選項: 如果您不需要任何說明的選項，請使用下列選項:

MQCTLO_NONE
使用這個值來指出未指定其他選項；所有選項都採用其預設值。 MQCTLO_NONE 定義為輔助程式文件; 此選項不預期與任何其他選項搭配使用，但由於其值為零，因此無法偵測到這類使用。

這是輸入欄位。 *Options* 欄位的起始值是 MQCTLO_NONE。

保留 *(MQLONG)*

這是保留欄位。 值必須為零。

StrucId (MQCHAR4)
控制選項結構- StrucId 欄位

這是結構 ID; 值必須是:

MQCTLO_STRUC_ID
「控制選項」結構的 ID。

若為 C 程式設計語言，也會定義常數 MQCTLO_STRUC_ARRAY; 此值與 MQCTLO_STRUC_ID 相同，但卻是字元陣列而非字串。

這一律是輸入欄位。 此欄位的起始值是 MQCTLO_STRUC_ID。

版本 *(MQLONG)*
控制選項結構-版本欄位

這是結構版本號碼; 值必須是:

MQCTLO_VERSION_1
Version-1 控制選項結構。

下列常數指定現行版本的版本號碼:

MQ CTLO_CURRENT_VERSION
控制選項結構的現行版本。

這一律是輸入欄位。 此欄位的起始值為 MQCTLO_VERSION_1。

MQCTLO 的起始值及語言宣告

控制選項結構-起始值

表 *492: MQCTLO* 中欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQCTLO_STRUC_ID	'CTLO'
Version	MQCTLO_VERSION_1	1
Options	MQCTLO_NONE	空值
Reserved	保留欄位	
ConnectionArea	無	空值指標或空值位元組

附註:

1. 在 C 程式設計語言中，巨集變數 MQCTLO_DEFAULT 包含上述值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQCTLO MyCTLO = {MQCTLO_DEFAULT};
```

C 宣告
控制選項結構-C 語言宣告

```
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 */
 };
```

COBOL 宣告

```
** 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
```

PL/I 宣告

```
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 */
```

MQDH-Distribution 標頭

下表彙總結構中的欄位。

表 *493: MQDH* 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
StrucLength	MQDH 結構的長度加上之後的記錄	StrucLength
Encoding	MQPMR 記錄陣列之後的資料的數字編碼	編碼
CodedCharSetId	MQPMR 記錄陣列之後的資料的字集 ID	CodedCharSetId
Format	MQPMR 記錄陣列之後的資料的格式名稱	格式
Flags	一般旗標	旗標
PutMsgRecFields	指出哪些 MQPMR 欄位存在的旗標	PutMsgRecFields
RecsPresent	存在的物件記錄數	RecsPresent
ObjectRecOffset	從 MQDH 開始的第一個物件記錄的偏移	ObjectRec 偏移
PutMsgRecOffset	從 MQDH 開始的第一個放置訊息記錄的偏移	PutMsgRecOffset

MQDH 概觀

可用性: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows，以及連接至這些系統的 WebSphere MQ 用戶端。

目的:MQDH 結構說明當訊息是傳輸佇列上儲存的配送清單訊息時，訊息中所呈現的其他資料。 配送清單訊息是傳送至多個目的地佇列的訊息。 其他資料包含 MQDH 結構，後面接著 MQOR 記錄陣列及 MQPMR 記錄陣列。

此結構是由將訊息直接放置在傳輸佇列上，或從傳輸佇列中移除訊息 (例如: 訊息通道代理程式) 的特殊化應用程式所使用。

要將訊息放入配送清單的應用程式不得使用此結構。 相反地，它們必須使用 MQOD 結構來定義配送清單中的目的地，並使用 MQPMO 結構來指定訊息內容或接收傳送至個別目的地之訊息的相關資訊。

格式名稱:MQFMT_DIST_HEADER。

字集及編碼:MQDH 中的資料必須採用 *CodedCharSetId* 佇列管理程式屬性所提供的字集，以及 MQENC_NATIVE 所提供本端佇列管理程式的編碼。

在下列欄位中，將 MQDH 的字集及編碼設為 *CodedCharSetId* 及 *Encoding* 欄位:

- MQMD (如果 MQDH 結構是在訊息資料的開頭) ，或
- MQDH 結構之前的標頭結構 (所有其他觀察值)。

用法: 當應用程式將訊息放入配送清單，且部分或所有目的地位於遠端時，佇列管理程式會以 MQXQH 及 MQDH 結構作為應用程式訊息資料的字首，並將訊息放置在相關傳輸佇列上。 因此，當訊息位於傳輸佇列時，資料會以下列順序出現:

- MQXQH 結構
- MQDH 結構加上 MQOR 及 MQPMR 記錄的陣列
- 應用程式訊息資料

視目的地而定，佇列管理程式可以產生多個這類訊息，並將它放在不同的傳輸佇列上。 在此情況下，那些訊息中的 MQDH 結構會識別應用程式所開啟的配送清單所定義目的地的不同子集。

將配送清單訊息直接放置在傳輸佇列上的應用程式必須符合上述順序，且必須確定 MQDH 結構正確。 如果 MQDH 結構無效，佇列管理程式可能會使 MQPUT 或 MQPUT1 呼叫失敗，原因碼為 MQRC_DH_ERROR。

只有在您將佇列定義成能夠支援配送清單訊息時 (請參閱 第 731 頁的『佇列的屬性』中說明的 *DistLists* 佇列屬性) ，才能以配送清單形式將訊息儲存在佇列中。 如果應用程式將配送清單訊息直接放置在不支援配送清單的佇列上，則佇列管理程式會將配送清單訊息分割成個別訊息，並改為將配送清單訊息放置在佇列上。

MQDH 的欄位

MQDH 結構包含下列欄位; 這些欄位按 **字母順序**進行說明:

CodedCharSetId (MQLONG)

這是遵循 MQOR 及 MQPMR 記錄陣列的資料字集 ID; 它不適用於 MQDH 結構本身中的字元資料。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。 您可以使用下列特殊值:

MQCCSI_INHERIT
繼承此結構的字集 ID。

此結構 後面 的資料中的字元資料與此結構的字集相同。

佇列管理程式會將訊息中所傳送結構的這個值變更為結構的實際字集 ID。 如果未發生任何錯誤，則 MQGET 呼叫不會傳回值 MQCCSI_INHERIT。

如果 MQMD 中 *PutApplType* 欄位的值為 MQAT_BROKER ，則無法使用 MQCCSI_INHERIT。

此值在下列環境中受支援: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows，以及連接至這些系統的 WebSphere MQ 用戶端。

此欄位的起始值為 MQCCSI_UNDEFINED。

編碼 *(MQLONG)*

這是遵循 MQOR 及 MQPMR 記錄陣列之資料的數值編碼; 它不適用於 MQDH 結構本身中的數值資料。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。

此欄位的起始值為 0。

旗標 *(MQLONG)*

您可以指定下列旗標:

MQDHHF_NEW_MSG_IDS
為配送清單中的每一個目的地產生新的訊息 ID。 只有在沒有放置訊息記錄存在時，或當記錄存在但不包含 *MsgId* 欄位時，才設定此選項。

使用此旗標會延遲產生訊息 ID，直到配送清單訊息最終分割成個別訊息為止。 這會最小化必須與配送清單訊息一起傳送的控制資訊數量。

當應用程式將訊息放入配送清單時，佇列管理程式會在 MQDH 中設定 MQDH_NEW_MSG_IDS，當同時符合下列兩項時，它會產生這些 MQDH:

- 應用程式未提供任何放置訊息記錄，或提供的記錄未包含 *MsgId* 欄位。
- MQMD 中的 *MsgId* 欄位是 MQMI_NONE，或者 MQPMO 中的 *Options* 欄位包括 MQPMO_NEW_MSG_ID

如果不需要旗標，請指定下列:

MQDHHF_NONE
未指定任何旗標。 MQDHHF_NONE 定義為輔助程式說明文件。 此常數並非預期與任何其他常數一起使用，但由於其值為零，因此無法偵測此類使用。

此欄位的起始值為 MQDHHF_NONE。

格式 *(MQCHAR8)*

這是遵循 MQOD 及 MQPMR 記錄陣列的資料格式名稱 (以最後出現者為準)。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。 此欄位的編碼規則與 MQMD 中 *Format* 欄位的編碼規則相同。

此欄位的起始值為 MQFMT_NONE。

ObjectRec 偏移 *(MQLONG)*

這會提供包含目的地佇列名稱之 MQOR 物件記錄陣列中第一筆記錄的偏移 (以位元組為單位)。 此陣列中有 *RecsPresent* 筆記錄。 這些記錄 (加上第一個物件記錄與前一個欄位之間跳過的任何位元組) 包含在 *StrucLength* 欄位給定的長度中。

配送清單必須一律包含至少一個目的地，因此 *ObjectRecOffset* 必須一律大於零。

此欄位的起始值為 0。

PutMsgRecFields (MQLONG)

您可以指定下列旗標中的任何一個或多個:

MQPMRF_MSG_ID
訊息 ID 欄位存在。

MQPMRF_CORREL_ID
存在相關性 ID 欄位。

MQPMRF_GROUP_ID
存在群組 ID 欄位。

MQPMRF_FEEDBACK
意見回饋欄位存在。

MQPMRF_ACCOUNTING_TOKEN
存在 accounting-token 欄位。

如果沒有 MQPMR 欄位，請指定下列項目:

MQPMRF_NONE
沒有任何放置訊息記錄欄位。 MQPMRF_NONE 定義為輔助程式文件。 此常數並非預期與任何其他常數一起使用，但由於其值為零，因此無法偵測此類使用。

此欄位的起始值為 MQPMRF_NONE。

PutMsgRecOffset (MQLONG)

這會提供包含訊息內容之 MQPMR 放置訊息記錄陣列中第一筆記錄的偏移 (以位元組為單位)。 如果存在，則此陣列中有 *RecsPresent* 筆記錄。 這些記錄 (加上在第一個放置訊息記錄與前一個欄位之間跳過的任何位元組) 包含在 *StrucLength* 欄位給定的長度中。

放置訊息記錄是選用的; 如果未提供任何記錄，則 *PutMsgRecOffset* 是零，且 *PutMsgRecFields* 具有值 MQPMRF_NONE。

此欄位的起始值為 0。

RecsPresent (MQLONG)

這是目的地數目。 配送清單必須一律包含至少一個目的地，因此 *RecsPresent* 必須一律大於零。

此欄位的起始值為 0。

StrucId (MQCHAR4)

值必須為:

MQDH_STRUC_ID
配送標頭結構的 ID。

對於 C 程式設計語言，也會定義常數 MQDH_STRUC_ARRAY; 此值與 MQDH_STRUC_ID 相同，但卻是字元陣列而非字串。

此欄位的起始值是 MQDH_STRUC_ID。

StrucLength (MQLONG)

這是從 MQDH 結構開始到訊息資料開始，遵循 MQOR 及 MQPMR 記錄陣列的位元組數。 資料按下列順序出現:

- MQDH 結構
- MQOR 記錄的陣列
- MQPMR 記錄的陣列
- 訊息資料

MQOR 及 MQPMR 記錄的陣列由 MQDH 結構內包含的偏移來處理。 如果這些偏移導致一個以上 MQDH 結構、記錄陣列及訊息資料之間的未用位元組，則那些未用位元組必須包含在 *StrucLength* 的值中，但佇列管理程式不會保留那些位元組的內容。 它適用於 MQPMR 記錄陣列之前的 MQOR 記錄陣列。

此欄位的起始值為 0。

版本 *(MQLONG)*

值必須為:

MQDH_VERSION_1
配送標頭結構的版本號碼。

下列常數指定現行版本的版本號碼:

MQDH_CURRENT_VERSION
配送標頭結構的現行版本。

此欄位的起始值為 MQDH_VERSION_1。

MQDH 的起始值及語言宣告

表 *494: MQDH for MQDH* 中欄位的起始值

欄位名稱	**常數名稱**	**常數值**
StrucId	MQDH_STRUC_ID	'DH¬¬'
Version	MQDH_VERSION_1	1
StrucLength	無	0
Encoding	無	0
CodedCharSetId	未定義 MQCCSI_UNDEFINED	0
Format	MQFMT_NONE	空白
Flags	MQDHHF_NONE	0
PutMsgRecFields	MQPMRF_NONE	0
RecsPresent	無	0
ObjectRecOffset	無	0
PutMsgRecOffset	無	0

附註:

1. 符號 ¬ 代表單一空白字元。
2. 在 C 程式設計語言中，巨集變數 MQDH_DEFAULT 包含上述值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQDH MyDH = {MQDH_DEFAULT};
```

C 宣告

```
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 */
};
```

COBOL 宣告

```
**   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-STRUCLENGTH     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.
```

PL/I 宣告

```
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 */
```

Visual Basic 宣告

```
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
```

MQDLH-無法傳送郵件的標頭

下表彙總結構中的欄位。

表 *495: MQDLH* 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
Reason	訊息抵達無法傳送郵件佇列的原因	原因
DestQName	原始目的地佇列的名稱	DestQName
DestQMgrName	原始目的地佇列管理程式的名稱	DestQMgrName
Encoding	MQDLH 之後的資料的數字編碼	編碼
CodedCharSetId	MQDLH 之後的資料的字集 ID	CodedCharSetId
Format	MQDLH 之後的資料的格式名稱	格式
PutApplType	將訊息放置在無法傳送郵件佇列的應用程式類型	PutApplType
PutApplName	將訊息放置在無法傳送郵件佇列的應用程式名稱	PutApplName
PutDate	訊息放置在無法傳送郵件佇列的日期	PutDate
PutTime	訊息放置在無法傳送郵件佇列的時間	PutTime

MQDLH 概觀

可用性: 所有 WebSphere MQ 平台。

目的:MQDLH 結構說明在無法傳送郵件 (無法遞送的訊息) 佇列上，將訊息的應用程式訊息資料作為字首的資訊。 訊息可以到達無法傳送郵件的佇列，因為佇列管理程式或訊息通道代理程式已將它重新導向至佇列，或因為應用程式已將訊息直接放置在佇列上。

格式名稱:MQFMT_DEAD_LETTER_HEADER。

字集和編碼:MQDLH 結構中的欄位採用 *CodedCharSetId* 和 *Encoding* 欄位所提供的字集和編碼。 這些指定在 MQDLH 之前的標頭結構中，或在 MQMD 結構中 (如果 MQDLH 位於應用程式訊息資料的開頭)。

對於佇列名稱中有效的字元，字集必須是具有單位元組字元的字集。

如果您使用適用於 Java/JMS 的 WMQ 類別，且 Java 虛擬機器不支援 MQMD 中定義的字碼頁，則會以 UTF-8 字集撰寫 MQDLH。

用法: 將訊息直接放置在無法傳送郵件的佇列上的應用程式必須以 MQDLH 結構作為訊息資料的字首，並以適當的值起始設定欄位。 不過，佇列管理程式不需要 MQDLH 結構存在，或已指定欄位的有效值。

如果訊息太長而無法放入無法傳送郵件的佇列，應用程式必須執行下列其中一項:

- 截斷訊息資料以符合無法傳送郵件的佇列。
- 記錄輔助儲存體上的訊息，並將異常狀況報告訊息放置在指出此狀況的無法傳送郵件佇列上。
- 捨棄訊息並將錯誤傳回給其發送端。 如果訊息是 (或可能是) 重要訊息，只有在已知發送端仍有訊息副本時，才執行此動作; 例如，訊息通道代理程式從通訊通道接收到訊息。

上述哪一項是適當的 (如果有的話) ，取決於應用程式的設計。

當訊息是前端具有 MQDLH 結構的區段時，佇列管理程式會執行特殊處理; 如需進一步詳細資料，請參閱 MQMDE 結構的說明。

將訊息放置在無法傳送郵件的佇列上: 當將訊息放置在無法傳送郵件的佇列上時，用於 MQPUT 或 MQPUT1 呼叫的 MQMD 結構必須與與訊息相關聯的 MQMD (通常是 MQGET 呼叫所傳回的 MQMD) 相同，但下列情況除外:

- 將 *CodedCharSetId* 和 *Encoding* 欄位設為 MQDLH 結構中用於欄位的任何字集和編碼。

- 將 *Format* 欄位設為 MQFMT_DEAD_LETTER_HEADER，以指出資料以 MQDLH 結構開始。
- 使用適用於下列情況的環境定義選項來設定環境定義欄位 (*AccountingToken*、*ApplIdentityData*、*ApplOriginData*、*PutApplName*、*PutApplType*、*PutDate*、*PutTime*、*UserIdentifier*):
  - 將與任何先前訊息無關的訊息放入無法傳送郵件的佇列上的應用程式必須使用 MQPMO_DEFAULT_CONTEXT 選項; 這會導致佇列管理程式將訊息描述子中的所有環境定義欄位設為其預設值。
  - 將剛收到的訊息放入無法傳送郵件的佇列中的伺服器應用程式必須使用 MQPMO_PASS_ALL_CONTEXT 選項來保留原始環境定義資訊。
  - 將 回覆 放置在無法傳送郵件的佇列上的伺服器應用程式必須使用 MQPMO_PASS_IDENTITY_CONTEXT 選項; 這會保留身分資訊，但會將原始資訊設為伺服器應用程式的原始資訊。
  - 訊息通道代理程式必須使用 MQPMO_SET_ALL_CONTEXT 選項，將從其通訊通道接收到的訊息放入無法傳送郵件的佇列中，以保留原始環境定義資訊。

在 MQDLH 結構本身中，設定欄位如下:

- 將 *CodedCharSetId*、*Encoding* 和 *Format* 欄位設為值，以說明遵循 MQDLH 結構的資料，通常是來自原始訊息描述子的值。
- 將環境定義欄位 *PutApplType*、*PutApplName*、*PutDate* 及 *PutTime* 設為適用於將訊息置於無法傳送郵件之佇列的應用程式的值; 這些值與原始訊息無關。
- 視需要設定其他欄位。

請確定所有欄位都有有效值，且在 MQDLH 結構中不會將空值及後續字元轉換成空白，因此請不要過早使用空值字元來結束字元資料。

從無法傳送郵件的佇列取得訊息: 從無法傳送郵件的佇列取得訊息的應用程式必須驗證訊息開頭是否為 MQDLH 結構。 應用程式可以透過檢查訊息描述子 MQMD 中的 *Format* 欄位，來判定 MQDLH 結構是否存在; 如果欄位具有值 MQFMT_DEAD_LETTER_HEADER，則訊息資料會以 MQDLH 結構開頭。 另請注意，如果應用程式從無法傳送郵件的佇列取得的訊息原本對佇列而言太長，則可能會被截斷。

MQDLH 的欄位

MQDLH 結構包含下列欄位; 這些欄位按 **字母順序**進行說明:

CodedCharSetId (MQLONG)
CodedCharSetId 是流經 MQDLH 結構 (通常是來自原始訊息的資料) 之資料的字集 ID; 它不適用於 MQDLH 結構本身中的字元資料。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。 可以使用下列特殊值:

MQCCSI_INHERIT
此結構之後的資料中的字元資料與此結構的字集相同。

佇列管理程式會將訊息中所傳送結構的這個值變更為結構的實際字集 ID。 如果沒有發生錯誤， MQGET 呼叫不會傳回值 MQCCSI_INHERIT。

如果 MQMD 中 *PutApplType* 欄位的值為 MQAT_BROKER，則無法使用 MQCCSI_INHERIT。

此值在下列環境中受支援: AIX、 HP-UX、 z/OS、 IBM i、Solaris、 Linux、 Windows，以及連接至這些系統的 WebSphere MQ MQI 用戶端。

此欄位的起始值為 MQCCSI_UNDEFINED。

DestQMgr 名稱 *(MQCHAR48)*
DestQMgr 名稱是作為訊息原始目的地的佇列管理程式名稱。

此欄位的長度由 MQ_Q_MGR_NAME_LENGTH 提供。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

DestQName (MQCHAR48)
DestQName 是作為訊息原始目的地的訊息佇列名稱。

此欄位的長度由 MQ_Q_NAME_LENGTH 指定。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

編碼 *(MQLONG)*
編碼是遵循 MQDLH 結構之資料的數值編碼 (通常是來自原始訊息的資料); 它不適用於 MQDLH 結構本身中的數值資料。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。

此欄位的起始值為 0。

格式 *(MQCHAR8)*
格式是遵循 MQDLH 結構之資料的格式名稱 (通常是來自原始訊息的資料)。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。 此欄位的編碼規則與 MQMD 中 *Format* 欄位的編碼規則相同。

此欄位的長度由 MQ_FORMAT_LENGTH 提供。 此欄位的起始值為 MQFMT_NONE。

PutAppl 名稱 *(MQCHAR28)*
PutAppl 名稱是將訊息放置在無法傳送的郵件 (無法遞送的訊息) 佇列上的應用程式名稱。

名稱的格式視 *PutApplType* 欄位而定。 此格式可以隨版次而改變。 請參閱 第 363 頁的『MQMD-訊息描述子』中 *PutApplName* 欄位的說明。

如果佇列管理程式將訊息重新導向至無法傳送郵件的佇列，則 *PutApplName* 會包含佇列管理程式名稱的前 28 個字元，必要的話會以空白填補。

此欄位的長度由 MQ_PUT_APPL_NAME_LENGTH 提供。 這個欄位的起始值在 C 中是空字串，在其他程式設計語言中是 28 個空白字元。

PutAppl 類型 *(MQLONG)*
PutAppl 類型是將訊息放置在無法傳送的郵件 (無法遞送的訊息) 佇列上的應用程式類型。

此欄位與訊息描述子 MQMD 中的 *PutApplType* 欄位具有相同的意義 (如需詳細資料，請參閱 第 363 頁的『MQMD-訊息描述子』)。

如果佇列管理程式將訊息重新導向至無法傳送郵件的佇列，則 *PutApplType* 具有值 MQAT_QMGR。

此欄位的起始值為 0。

PutDate (MQCHAR8)
PutDate 是將訊息放入無法傳送郵件 (無法遞送的訊息) 佇列的日期。

當佇列管理程式產生此欄位時，用於日期的格式為:

- YYYYMMDD

其中字元代表:

YYYY
年 (四個數字)

MM
月份 (01 到 12)

DD
日 (01 至 31)

「格林威治標準時間 (GMT)」用於 *PutDate* 及 *PutTime* 欄位，受精確設為 GMT 的系統時鐘所限制。

此欄位的長度由 MQ_PUT_DATE_LENGTH 提供。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 8 個空白字元。

PutTime (MQCHAR8)
PutTime 是將訊息放置在無法傳送郵件 (無法遞送的訊息) 佇列上的時間。

當佇列管理程式產生此欄位時，所使用的時間格式為:

- HHMMSSTH

其中字元代表:

HH
小時 (00 到 23)

MM
分鐘 (00 到 59)

不銹鋼
秒 (00 至 59; 請參閱附註)

T
十分之一秒 (0 到 9)

H
百分之一秒 (0 到 9)

註: 如果系統時鐘已同步至非常精確的時間標準，則在極少數情況下可能會在 *PutTime* 中傳回 60 或 61 秒。 將閏秒插入廣域時間標準時會發生這種情況。

「格林威治標準時間 (GMT)」用於 *PutDate* 及 *PutTime* 欄位，受精確設為 GMT 的系統時鐘所限制。

此欄位的長度由 MQ_PUT_TIME_LENGTH 提供。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 8 個空白字元。

原因 *(MQLONG)*
「原因」欄位識別訊息放置在無法傳送郵件的佇列上而非原始目的地佇列上的原因。

這可識別為何將訊息放置在無法傳送郵件的佇列而非原始目的地佇列上的原因。 它應該是其中一個 MQFB_ * 或 MQRC_ * 值 (例如 MQRC_Q_FULL)。 如需可能發生的一般 MQFB_ * 值的詳細資料，請參閱 第 363 頁的『MQMD-訊息描述子』 中 *Feedback* 欄位的說明。

如果值在 MQFB_IMS_FIRST 到 MQFB_IMS_LAST 的範圍內，則可以從 *Reason* 欄位的值減去 MQFB_IMS_ERROR 來判定實際的 IMS 錯誤碼。

部分 MQFB_ * 值僅出現在此欄位中。 它們與已傳送至無法傳送郵件之佇列的儲存庫訊息、觸發訊息或傳輸佇列訊息相關。 它們是:

MQFB_APPL_CANNOT_BE_STARTED (X'00000109')
處理觸發訊息的應用程式無法啟動觸發訊息的 *ApplId* 欄位中指定的應用程式 (請參閱 第 521 頁的『MQTM-觸發訊息』)。

在 z/OS 上， CKTI CICS 交易是處理觸發訊息的應用程式範例。

MQFB_APPL_TYPE_ERROR (X'0000010B')
處理觸發訊息的應用程式無法啟動應用程式，因為觸發訊息的 *ApplType* 欄位無效 (請參閱 第 521 頁的『MQTM-觸發訊息』)。

在 z/OS 上， CKTI CICS 交易是處理觸發訊息的應用程式範例。

MQFB_BIND_OPEN_CLUSRCVR_DEL (X'00000119')
訊息是在 SYSTEM.CLUSTER.TRANSMIT.QUEUE 預期用於使用 MQOO_BIND_ON_OPEN 選項開啟的叢集佇列，但在可以傳送訊息之前，已刪除要用來將訊息傳輸至目的地佇列的遠端叢集接收端通道。 因為已指定 MQOO_BIND_ON_OPEN ，所以只能使用開啟佇列時所選取的通道來傳輸訊息。 因為此通道不再可用，所以會將訊息放置在無法傳送郵件的佇列上。

MQFB_NOT_A_REPOSITORY_MSG (X'00000118')
訊息不是儲存庫訊息。

MQFB_STOPPED_BY_CHAD_EXIT (X'00000115')
通道自動定義結束程式已停止訊息。

MQFB_STOPPED_BY_MSG_EXIT (X'0000010D')
通道訊息結束程式已停止訊息。

MQFB_TM_ERROR (X'0000010A')
MQMD 中的 *Format* 欄位指定 MQFMT_TRIGGER，但訊息不是以有效的 MQTM 結構開頭。 例如，*StrucId* 助記鍵 eye-catcher 可能無效、 *Version* 可能無法辨識，或觸發訊息的長度可能不足以包含 MQTM 結構。

在 z/OS 上， CKTI CICS 交易是處理觸發訊息並可產生此回饋碼的應用程式範例。

MQFB_XMIT_Q_MSG_ERROR (X'0000010F')
訊息通道代理程式發現傳輸佇列上的訊息格式不正確。 訊息通道代理程式會使用此回饋碼將訊息放置在無法傳送郵件的佇列上。

此欄位的起始值為 MQRC_NONE。

StrucId (MQCHAR4)
StrucId 是結構 ID。

值必須為:

MQDLH_STRUC_ID
無法傳送郵件之標頭結構的 ID。

對於 C 程式設計語言，也會定義常數 MQDLH_STRUC_ID_ARRAY; 此值與 MQDLH_STRUC_ID 相同，但卻是字元陣列而非字串。

此欄位的起始值是 MQDLH_STRUC_ID。

版本 *(MQLONG)*
版本是結構版本號碼。

值必須為:

MQDLH_VERSION_1
無法傳送郵件的標頭結構的版本號碼。

下列常數指定現行版本的版本號碼:

MQDLH_CURRENT_VERSION
無法傳送郵件的標頭結構現行版本。

此欄位的起始值為 MQDLH_VERSION_1。

MQDLH 的起始值及語言宣告

表 *496: MQDLH* 中 *MQDLH* 欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQDLH_STRUC_ID	'DLH¬'
Version	MQDLH_VERSION_1	1
Reason	MQRC_NONE	0
DestQName	無	空字串或空白
DestQMgrName	無	空字串或空白
Encoding	無	0
CodedCharSetId	未定義 MQCCSI_UNDEFINED	0
Format	MQFMT_NONE	空白
PutApplType	無	0

表 *496: MQDLH* 中 *MQDLH* 欄位的起始值 *(*繼續*)*		
欄位名稱	**常數名稱**	**常數值**
PutApplName	無	空字串或空白
PutDate	無	空字串或空白
PutTime	無	空字串或空白

附註:

1. 符號 ¬ 代表單一空白字元。
2. 空值字串或空白值表示 C 中的空值字串，以及其他程式設計語言中的空白字元。
3. 在 C 程式設計語言中，巨集變數 MQDLH_DEFAULT 包含上述值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQDLH MyDLH = {MQDLH_DEFAULT};
```

C 宣告

```
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 */
};
```

COBOL 宣告

```
**   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).
```

PL/I 宣告

```
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 */
```

High Level Assembler 宣告

```
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)
```

Visual Basic 宣告

```
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
```

MQDMHO-刪除訊息控點選項

下表彙總結構中的欄位。

表 *497: MQDMHO* 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
Options	選項	選項

MQDMHO 概觀

可用性: 所有 WebSphere MQ 系統及 WebSphere MQ 用戶端。

目的: **MQDMHO** 結構可讓應用程式指定選項，以控制如何刪除訊息處理。 結構是 **MQDLTMH** 呼叫上的輸入參數。

字集及編碼: **MQDMHO** 中的資料必須在應用程式的字集及應用程式的編碼中 (**MQENC_NATIVE**)。

MQDMHO 的欄位

MQDMHO 結構包含下列欄位; 這些欄位按 **字母順序**說明:

選項 *(MQLONG)*

值必須為:

MQDMHO_NONE

未指定選項。

這一律是輸入欄位。 此欄位的起始值為 **MQDMHO_NONE**。

StrucId (MQCHAR4)

這是結構 ID; 值必須是:

MQDMHO_STRUC_ID

刪除訊息控點選項結構的 ID。

對於 C 程式設計語言，也會定義常數 **MQDMHO_STRUC_ID_ARRAY**；其值與 **MQDMHO_STRUC_ID** 相同，但卻是字元陣列而非字串。

這一律是輸入欄位。 此欄位的起始值為 **MQDMHO_STRUC_ID**。

版本 *(MQLONG)*

這是結構版本號碼; 值必須是:

MQDMHO_VERSION_1
Version-1 刪除訊息控點選項結構。

下列常數指定現行版本的版本號碼:

MQDMHO_CURRENT_VERSION
刪除訊息控點選項結構的現行版本。

這一律是輸入欄位。 此欄位的起始值為 **MQDMHO_VERSION_1**。

MQDMHO 的起始值和語言宣告

表 *498: MQDMHO* 中欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQDMHO_STRUC_ID	'DMHO'
Version	MQDMHO_VERSION_1	1
Options	MQDMHO_NONE	0

附註:

1. 在 C 程式設計語言中，巨集變數 MQDMHO_DEFAULT 包含上述值。 您可以用下列方式來提供結構中欄位的起始值:

```
MQDMHO MyDMHO = {MQDMHO_DEFAULT};
```

C 宣告

```
typedef struct tagMQDMHO;
struct tagMQDMHO {
  MQCHAR4   StrucId;         /* Structure identifier */
  MQLONG    Version;         /* Structure version number */
  MQLONG    Options;         /* Options that control the action of MQDLTMH */
};
```

COBOL 宣告

```
**   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.
```

PL/I 宣告

```
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 */
```

High Level Assembler 宣告

```
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)
```

MQDMPO-刪除訊息內容選項

下表彙總結構中的欄位。 MQDMPO 結構-刪除訊息內容選項

表 *499: MQDMPO* 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
Options	控制 MQDMPO 動作的選項	選項

MQDMPO 概觀

可用性: 所有 WebSphere MQ 系統及 WebSphere MQ 用戶端。

目的:MQDMPO 結構可讓應用程式指定選項來控制如何刪除訊息內容。 結構是 MQDLTMP 呼叫上的輸入參數。

字集及編碼:MQDMPO 中的資料必須位於應用程式的字集及應用程式的編碼 (MQENC_NATIVE)。

MQDMPO 的欄位

刪除訊息內容選項結構-欄位

MQDMPO 結構包含下列欄位; 這些欄位按 **字母順序**進行說明:

選項 *(MQLONG)*
刪除訊息內容選項結構-選項欄位

位置選項: 相較於內容游標，下列選項與內容的相對位置相關。

MQDMPO_DEL_FIRST
刪除第一個符合指定名稱的內容。

MQDMPO_DEL_PROP_UNDER_CURSOR
刪除內容游標所指向的內容; 即前次使用 MQIMPO_INQ_FIRST 或 MQIMPO_INQ_NEXT 選項來查詢的內容。

當重複使用訊息控點時，會重設內容游標。 當在 MQGET 呼叫的 MQGMO 結構的 *MsgHandle* 欄位中指定訊息控點時，或在 MQPUT 呼叫的 MQPMO 結構時，也會重設訊息控點。

如果在尚未建立內容游標時使用此選項，則呼叫會失敗，完成碼為 MQCC_FAILED 且原因為 MQRC_PROPERTY_NOT_AVAILABLE。 如果已刪除內容游標所指向的內容，則呼叫也會失敗，完成碼為 MQCC_FAILED 且原因為 MQRC_PROPERTY_NOT_AVAILABLE。

如果兩個選項都不需要，則可以使用下列選項:

MQDMPO_NONE
未指定選項。

此欄位一律是輸入欄位。 此欄位的起始值是 MQDMPO_DEL_FIRST。

StrucId (MQCHAR4)
刪除訊息內容選項結構- StrucId 欄位

這是結構 ID。 值必須為:

MQDMPO_STRUC_ID
刪除訊息內容選項結構的 ID。

對於 C 程式設計語言，也會定義常數 MQDMPO_STRUC_ID_ARRAY; 此值與 MQDMPO_STRUC_ID 相同，但卻是字元陣列而非字串。

這一律是輸入欄位。 此欄位的起始值是 MQDMPO_STRUC_ID。

版本 *(MQLONG)*
刪除訊息內容選項結構-版本欄位

這是結構版本號碼。 值必須為:

MQDMPO_VERSION_1
刪除訊息內容選項結構的版本號碼。

下列常數指定現行版本的版本號碼:

MQDMPO_CURRENT_VERSION
刪除訊息內容選項結構的現行版本。

這一律是輸入欄位。 此欄位的起始值為 MQDMPO_VERSION_1。

MQDMPO 的起始值及語言宣告
刪除訊息內容選項結構-起始值

表 *500: MQDPMO* 中欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQDMPO_STRUC_ID	'DMPO'
Version	MQDMPO_VERSION_1	1
Options	控制 MQDLTMP 動作的選項	MQDMPO_NONE

附註:

1. 在 C 程式設計語言中，巨集變數 MQDMPO_DEFAULT 包含上述值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQDMPO MyDMPO = {MQDMPO_DEFAULT};
```

C 宣告
刪除訊息內容選項結構-C 語言宣告

```
typedef struct tagMQDMPO MQDMPO;
struct tagMQDMPO {
  MQCHAR4  StrucId;        /* Structure identifier */
  MQLONG   Version;        /* Structure version number */
  MQLONG   Options;        /* Options that control the action of
                              MQDLTMP */
};
```

COBOL 宣告
刪除訊息內容選項結構-COBOL 語言宣告

```
**   MQDMPO structure
  10 MQDMPO.
**    Structure identifier
   15 MQDMPO-STRUCID               PIC X(4).
**    Structure version number
   15 MQDMPO-VERSION               PIC S9(9) BINARY.
**    Options that control the action of MQDLTMP
   15 MQDMPO-OPTIONS               PIC S9(9) BINARY.
```

PL/I 宣告
刪除訊息內容選項結構-PL/I 語言宣告

```
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 */
```

High Level Assembler 宣告
刪除訊息內容選項結構-組譯器語言宣告

```
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)
```

MQEPH-內嵌 PCF 標頭

下表彙總結構中的欄位。

表 *501: MQEPH* 中的欄位

欄位	說明	主題
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
StrucLength	MQEPH 結構的長度加上 MQCFH 及其後的參數結構	StrucLength
Encoding	在最後 PCF 參數結構之後的資料的數字編碼	編碼
CodedCharSetId	在最後 PCF 參數結構之後的資料的字集 ID	CodedCharSetId
Format	在最後 PCF 參數結構之後的資料的格式名稱	格式
Flags	旗標	旗標
PCFHeader	可程式指令格式 (PCF) 標頭	PCFHeader

MQEPH 概觀

可用性: 所有 WebSphere MQ 平台。

目的:MQEPH 結構說明當訊息是可程式化指令格式 (PCF) 訊息時，訊息中所呈現的其他資料。 *PCFHeader* 欄位定義遵循此結構的 PCF 參數，這可讓您遵循具有其他標頭的 PCF 訊息資料。

格式名稱:MQFMT_EMBEDDED_PCF

字集及編碼:MQEPH 中的資料必須採用 *CodedCharSetId* 佇列管理程式屬性所提供的字集，以及 MQENC_NATIVE 所提供本端佇列管理程式的編碼。

在下列欄位中，將 MQEPH 的字集及編碼設為 *CodedCharSetId* 及 *Encoding* 欄位:

- MQMD (如果 MQEPH 結構是在訊息資料的開頭) ，或
- MQEPH 結構之前的標頭結構 (所有其他情況)。

用法: 您無法使用 MQEPH 結構，將指令傳送至指令伺服器或任何其他佇列管理程式 PCF 接受伺服器。

同樣地，指令伺服器或任何其他佇列管理程式 PCF 接受伺服器不會產生包含 MQEPH 結構的回應或事件。

MQEPH 的欄位

MQEPH 結構包含下列欄位; 這些欄位按 **字母順序**進行說明:

CodedCharSetId (MQLONG)

這是遵循 MQEPH 結構及相關聯 PCF 參數的資料字集 ID; 它不適用於 MQEPH 結構本身中的字元資料。

此欄位的起始值為 MQCCSI_UNDEFINED。

編碼 *(MQLONG)*

這是遵循 MQEPH 結構及相關聯 PCF 參數的資料數值編碼; 它不適用於 MQEPH 結構本身中的字元資料。

此欄位的起始值為 0。

旗標 *(MQLONG)*

下列是可用的值:

MQEPH_NONE
未指定任何旗標。 MQEPH_NONE 定義為輔助程式文件。 此常數並非預期與任何其他常數一起使用，但由於其值為零，因此無法偵測此類使用。

MQEPH_CCSID_EMBEDDED
包含字元資料之參數的字集是在每一個結構中的 CodedCharSetId 欄位內個別指定。 StrucId 及格式欄位的字集是由 MQEPH 結構之前的標頭結構中的 CodedCharSetId 欄位所定義，或由 MQMD 中的 CodedCharSetId 欄位所定義 (如果 MQEPH 位於訊息開頭)。

此欄位的起始值為 MQEPH_NONE。

格式 *(MQCHAR8)*

這是遵循 MQEPH 結構及相關聯 PCF 參數的資料格式名稱。

此欄位的起始值為 MQFMT_NONE。

PCFHeader (MQCFH)

這是可程式指令格式 (PCF) 標頭，定義遵循 MQEPH 結構的 PCF 參數。 這可讓您使用其他標頭來追蹤 PCF 訊息資料。

一開始會使用下列值來定義 PCF 標頭:

表 *502: MQCFH* 中欄位的起始值

欄位名稱	**常數名稱**	**常數值**
Type	MQCFT_NONE	0
StrucLength	MQCFH_STRUC_LENGH	36
Version	MQCFH_VERSION_3	3
StrucLength	無	0
Command	MQCMD_NONE	0
MsgSeqNumber	無	1
Control	MQCFC_LAST	1

表 *502: MQCFH* 中欄位的起始值 *(繼續)*		
欄位名稱	**常數名稱**	**常數值**
CompCode	MQCC_OK	0
Reason	MQRC_NONE	0
ParameterCount	無	0

應用程式必須將 Type 從 MQCFT_NONE 變更為有效的結構類型，才能使用內嵌 PCF 標頭。

StrucId (MQCHAR4)

值必須為:

MQEPH_STRUC_ID
配送標頭結構的 ID。

對於 C 程式設計語言，也會定義常數 MQEPH_STRUC_ID_ARRAY; 此值與 MQDH_STRUC_ID 相同，但是字元陣列而非字串。

此欄位的起始值為 MQEPH_STRUC_ID。

StrucLength (MQLONG)

這是下一個標頭結構之前的資料量。 它包括:

- MQEPH 標頭的長度
- 標頭後面所有 PCF 參數的長度
- 在那些參數之後的任何空白填補

StrucLength 必須是 4 的倍數。

結構的固定長度部分由 MQEPH_STRUC_LENGTH_FIXED 定義。

此欄位的起始值為 68。

版本 *(MQLONG)*

值必須為:

MQEPH_VERSION_1
內嵌 PCF 標頭結構的版本號碼。

下列常數指定現行版本的版本號碼:

MQCFH_VERSION_3
內嵌 PCF 標頭結構的現行版本。

此欄位的起始值為 MQEPH_VERSION_1。

MQEPH 的起始值和語言宣告

表 *503: MQEPH* 中 *MQEPH* 的欄位起始值		
欄位名稱	**常數名稱**	**常數值**
StrucId	MQEPH_STRUC_ID	'EPH¬'
Version	MQEPH_VERSION_1	1
StrucLength	MQEPH_STRUC_LENGTH_FIXED	68
Encoding	無	0
CodedCharSetId	未定義 MQCCSI_UNDEFINED	0
Format	MQFMT_NONE	空白

表 503: MQEPH 中 MQEPH 的欄位起始值 (繼續)

欄位名稱	常數名稱	常數值
Flags	MQEPH_NONE	0
PCFHeader	第 319 頁的表 502 中定義的名稱和值	0

附註:

1. 符號 ¬ 代表單一空白字元。
2. 在 C 程式設計語言中，巨集變數 MQEPH_DEFAULT 包含上面列出的值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQEPH MyEPH = {MQEPH_DEFAULT};
```

C 宣告

```
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 */
 };
```

COBOL 宣告

```
**   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.
```

PL/I 宣告

```
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 */
```

High Level Assembler 宣告

```
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)
```

Visual Basic 宣告

```
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
```

MQGMO-取得訊息選項

下表彙總結構中的欄位。

表 *504: MQGMO* 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
Options	控制 MQGET 動作的選項	MQGMO-選項欄位
WaitInterval	等待間隔	WaitInterval
Signal1	信號(F)	Signal1
Signal2	信號 ID	Signal2
ResolvedQName	解析的目的地佇列名稱	ResolvedQName
註: 如果 *Version* 小於 MQGMO_VERSION_2，則會忽略其餘欄位。		
MatchOptions	控制 MQGET 使用的選取準則的選項	MatchOptions
GroupStatus	指出擷取的訊息是否位於群組中的旗標	GroupStatus
SegmentStatus	指出擷取的訊息是否為邏輯訊息區段的旗標	SegmentStatus
Segmentation	指出是否容許擷取的訊息進一步分段的旗標	分段
Reserved1	已保留	Reserved1
註: 如果 *Version* 小於 MQGMO_VERSION_3，則會忽略其餘欄位。		
MsgToken	訊息記號	MsgToken
ReturnedLength	傳回的訊息資料長度（位元組數）	ReturnedLength
註: 如果 *Version* 小於 MQGMO_VERSION_4，則會忽略其餘欄位。		
Reserved2	已保留	Reserved2
MsgHandle	要移入從佇列擷取之訊息內容的訊息控點。	MsgHandle

MQGMO 概觀

可用性: 所有 WebSphere MQ 平台。

目的:MQGMO 結構可讓應用程式控制如何從佇列中移除訊息。 此結構是 MQGET 呼叫上的輸入/輸出參數。

版本:MQGMO 的現行版本是 MQGMO_VERSION_4。 某些欄位只能在某些 MQGMO 版本中使用。 如果您需要在數個環境之間連接應用程式，則必須確保 MQGMO 的版本在所有環境之間一致。 僅存在於特定結構版本中的欄位在 第 323 頁的『MQGMO-取得訊息選項』 及欄位說明中被識別為如此。

為支援的程式設計語言提供的標頭、COPY 及 INCLUDE 檔案包含環境支援的 MQGMO 最新版本，但 *Version* 欄位的起始值設為 MQGMO_VERSION_1。 若要使用 version-1 結構中不存在的欄位，請將 *Version* 欄位設為所需版本的版本號碼。

字集及編碼:MQGMO 中的資料必須採用 *CodedCharSetId* 佇列管理程式屬性所提供的字集，以及 MQENC_NATIVE 所提供本端佇列管理程式的編碼。 不過，如果應用程式以 MQ MQI 用戶端身分執行，則該結構必須採用用戶端的字集及編碼。

MQGMO 的欄位

MQGMO 結構包含下列欄位; 這些欄位按 **字母順序**說明:

GroupStatus (MQCHAR)
此旗標指出擷取的訊息是否位於群組中。

它具有下列其中一個值:

MQGS_NOT_IN_GROUP
訊息不在群組中。

MQGS_MSG_IN_GROUP
訊息位於群組中，但不是群組中的最後一個。

MQGS_LAST_MSG_IN_GROUP
訊息是群組中的最後一個。

如果群組只包含一則訊息，則也會傳回此值。

這是輸出欄位。 此欄位的起始值是 MQGS_NOT_IN_GROUP。 如果 *Version* 小於 MQGMO_VERSION_2，則會忽略此欄位。

MatchOptions (MQLONG)

這些選項可讓應用程式選擇 *MsgDesc* 參數中的哪些欄位，以用來選取 MQGET 呼叫所傳回的訊息。 應用程式會在此欄位中設定必要選項，然後將 *MsgDesc* 參數中的對應欄位設為那些欄位所需的值。 只有在訊息的 MQMD 中具有這些值的訊息才是在 MQGET 呼叫中使用該 *MsgDesc* 參數進行擷取的候選項。 當選取要傳回的訊息時， 未 指定對應相符選項的欄位會被忽略。 如果您在 MQGET 呼叫中未指定選取準則 (亦即， 任何訊息都可接受) ， 請將 *MatchOptions* 設為 MQMO_NONE。

- 在 z/OS 上，可以使用的選取準則可能受限於用於佇列的索引類型。 如需進一步詳細資料，請參閱 *IndexType* 佇列屬性。

如果您指定 MQGMO_LOGICAL_ORDER ，則只有特定訊息可由下一個 MQGET 呼叫傳回:

- 如果沒有現行群組或邏輯訊息，則只有 *MsgSeqNumber* 等於 1 且 *Offset* 等於 0 的訊息才有資格傳回。 在此狀況下，您可以使用下列一或多個比對選項來選取要傳回哪些合格訊息:
  - MQMO_MATCH_MSG_ID
  - MQMO_MATCH_CORREL_ID
  - MQMO_MATCH_GROUP_ID
- 如果 有 現行群組或邏輯訊息，則只有群組中的下一個訊息或邏輯訊息中的下一個區段適合傳回，且無法透過指定 MQMO_ * 選項來變更。

在上述兩種情況下，您都可以指定不適用的比對選項，但 *MsgDesc* 參數中相關欄位的值必須符合要傳回的訊息中對應欄位的值; 呼叫失敗，原因碼為 MQRC_MATCH_OPTIONS_ERROR ，表示不滿足此條件。

如果您指定 MQGMO_MSG_UNDER_CURSOR 或 MQGMO_BROWSE_MSG_UNDER_CURSOR ，則會忽略 *MatchOptions* 。

根據訊息內容取得訊息不是使用比對選項來完成; 如需相關資訊，請參閱 第 421 頁的『SelectionString (MQCHARV)』 。

您可以指定下列一或多個符合選項:

MQMO_MATCH_MSG_ID
要擷取的訊息必須具有符合 MQGET 呼叫 *MsgDesc* 參數中 *MsgId* 欄位值的訊息 ID。 除了可能適用的任何其他相符項 (例如，相關性 ID) 之外，還會有此相符項。

如果省略此選項，則會忽略 *MsgDesc* 參數中的 *MsgId* 欄位，且任何訊息 ID 都將符合。

註: 訊息 ID MQMI_NONE 是特殊值，符合訊息 MQMD 中的 *any* 訊息 ID。 因此，使用 MQMI_NONE 指定 MQMO_MATCH_MSG_ID 與 不 指定 MQMO_MATCH_MSG_ID 相同。

MQMO_MATCH_CORREL_ID
要擷取的訊息必須具有相關性 ID ，其符合 MQGET 呼叫 *MsgDesc* 參數中 *CorrelId* 欄位的值。 除了可能適用的任何其他相符項 (例如，訊息 ID) 之外，還會有此相符項。

如果省略此選項，則會忽略 *MsgDesc* 參數中的 *CorrelId* 欄位，且任何相關性 ID 都將符合。

註: 相關性 ID MQCI_NONE 是特殊值，符合訊息 MQMD 中的 任何 相關性 ID。 因此，使用 MQCI_NONE 指定 MQMO_MATCH_CORREL_ID 與 不 指定 MQMO_MATCH_CORREL_ID 相同。

MQMO_MATCH_GROUP_ID
要擷取的訊息必須具有符合 MQGET 呼叫 *MsgDesc* 參數中 *GroupId* 欄位值的群組 ID。 除了可能適用的任何其他相符項 (例如，相關性 ID) 之外，還會有此相符項。

如果省略此選項，則會忽略 *MsgDesc* 參數中的 *GroupId* 欄位，且任何群組 ID 都將符合。

註: 群組 ID MQGI_NONE 是特殊值，符合訊息 MQMD 中的 任何 群組 ID。 因此，指定 MQMO_MATCH_GROUP_ID 與 MQGI_NONE 與 不 指定 MQMO_MATCH_GROUP_ID 相同。

MQMO_MATCH_MSG_SEQ_NUMBER
要擷取的訊息必須具有符合 MQGET 呼叫 *MsgDesc* 參數中 *MsgSeqNumber* 欄位值的訊息序號。 此相符項是任何其他可能適用的相符項 (例如，群組 ID) 之外的其他相符項。

如果您省略此選項，則會忽略 *MsgDesc* 參數中的 *MsgSeqNumber* 欄位，且任何訊息序號都將符合。

MQMO_MATCH_OFFSET
要擷取的訊息必須具有符合 MQGET 呼叫的 *MsgDesc* 參數中 *Offset* 欄位值的偏移。 此相符項是任何其他可能適用的相符項 (例如，訊息序號) 之外的其他相符項。

如果您省略此選項，則會忽略 *MsgDesc* 參數中的 *Offset* 欄位，且任何偏移都將符合。

- z/OS 不支援此選項。

MQMO_MATCH_MSG_TOKEN
要擷取的訊息必須具有符合 MQGET 呼叫所指定 MQGMO 結構中 *MsgToken* 欄位值的訊息記號。

您可以針對所有本端佇列指定此選項。 如果您針對具有 *IndexType* MQIT_MSG_TOKEN (WLM 受管理佇列) 的佇列指定它，則無法使用 MQMO_MATCH_MSG_TOKEN 指定其他相符選項。

您不能使用 MQGMO_WAIT 或 MQGMO_SET_SET_信號來指定 MQMO_MATCH_MSG_TOKEN。 如果應用程式想要等待訊息到達具有 *IndexType* MQIT_MSG_TOKEN 的佇列，請指定 MQMO_NONE。

如果您省略此選項，則會忽略 MQGMO 中的 *MsgToken* 欄位，且任何訊息記號都將符合。

如果您未指定任何說明的選項，則可以使用下列選項:

MQMO_NONE
在選取要傳回的訊息時不使用任何相符項; 佇列上的所有訊息都可以擷取 (但受制於 MQGMO_ALL_MSGS_AVAILABLE、MQGMO_ALL_SEGMENTS_AVAILABLE 及 MQGMO_COMPLETE_MSG 選項的控制)。

MQMO_NONE 會輔助程式說明文件。 此選項並非預期與任何其他 MQMO_ * 選項搭配使用，但由於其值為零，因此無法偵測到此類使用。

這是輸入欄位。 此欄位的起始值是 MQMO_MATCH_MSG_ID 與 MQMO_MATCH_CORREL_ID。 如果 *Version* 小於 MQGMO_VERSION_2，則會忽略此欄位。

註: *MatchOptions* 欄位的起始值是為了與較早的 MQSeries 佇列管理程式相容而定義。 不過，從佇列讀取一系列訊息而不使用選取準則時，此起始值需要應用程式在每一個 MQGET 呼叫之前，將 *MsgId* 及

CorrelId 欄位重設為 MQMI_NONE 及 MQCI_NONE。 將 *Version* 設為 MQGMO_VERSION_2，並將 *MatchOptions* 設為 MQMO_NONE ， 以避免需要重設 *MsgId* 和 *CorrelId* 。

MsgHandle (MQHMSG)

如果指定 MQGMO_PROPERTIES_AS_Q_DEF 選項，且 PropertyControl 佇列屬性未設為 MQPROP_FORCE_MQRFH2 ，則這是將移入從佇列擷取之訊息內容的訊息控點。 控點由 MQCRTMH 呼叫建立。 在擷取訊息之前，將會清除已與控點相關聯的任何內容。

也可以指定下列值:
MQHM_NONE
未提供訊息控點。

如果在輸出中提供並使用有效的訊息控點來包含訊息內容，則 MQGET 呼叫不需要任何訊息描述子，輸入欄位會使用與訊息控點相關聯的訊息描述子。

如果在 MQGET 呼叫上指定訊息描述子，它一律優先於與訊息控點相關聯的訊息描述子。

如果指定了 MQGMO_PROPERTIES_FORCE_MQRFH2 ，或指定了 MQGMO_PROPERTIES_AS_Q_DEF ，且 PropertyControl 佇列屬性為 MQPROP_FORCE_MQRFH2 ，則當未指定訊息描述子參數時，呼叫會失敗，原因碼為 MQRC_MD_ERROR。

從 MQGET 呼叫返回時，會更新與此訊息控點相關聯的內容及訊息描述子，以反映所擷取訊息的狀態 (以及訊息描述子 (如果已在 MQGET 呼叫上提供的話))。 然後可以使用 MQINQMP 呼叫來查詢訊息的內容。

除了訊息描述子延伸之外，如果有訊息描述子延伸，則訊息資料中不包含可使用 MQINQMP 呼叫來查詢的內容; 如果佇列上的訊息包含訊息資料中的內容，則會在將資料傳回應用程式之前從訊息資料中移除這些內容。

如果未提供任何訊息控點，或版本小於 MQGMO_VERSION_4 ，則您必須在 MQGET 呼叫上提供有效的訊息描述子。 任何訊息內容 (訊息描述子中包含的除外) 都會在訊息資料中傳回，這些訊息資料受限於 MQGMO 結構及 PropertyControl 佇列屬性中的內容選項值。

這是一律輸入欄位。 此欄位的起始值為 MQHM_NONE。 如果 Version 小於 MQGMO_VERSION_4，則會忽略此欄位。

MsgToken (MQBYTE16)
MsgToken 欄位-MQGMO 結構。 佇列管理程式會使用此欄位來唯一識別訊息。

這是佇列管理程式所產生的位元組字串，用來識別佇列上唯一的訊息。 訊息記號是在訊息第一次放置在佇列管理程式上時產生，除非佇列管理程式重新啟動，否則訊息會保留在訊息中，直到從佇列管理程式永久移除訊息為止。

從佇列中移除訊息時，識別該訊息實例的 *MsgToken* 不再有效，且永不重複使用。 如果重新啟動佇列管理程式，則在重新啟動之前識別佇列上訊息的 *MsgToken* 可能在重新啟動之後無效。 不過，絕不會重複使用 *MsgToken* 來識別不同的訊息實例。 「 *MsgToken* 」是由佇列管理程式所產生，且對於任何外部應用程式都不可見。

當 MQGET 呼叫提供第 3 版或更新版本 MQGMO 時傳回訊息，佇列管理程式會在 MQGMO 中傳回 *MsgToken* 來識別佇列上的訊息。 這有一個例外: 當從同步點以外的佇列移除訊息時，佇列管理程式可能不會傳回 *MsgToken* ，因為在後續 MQGET 呼叫中識別傳回的訊息並不有用。 應用程式應該只使用 *MsgToken* 來參照後續 MQGET 呼叫的訊息。

如果提供了 *MsgToken* 並且指定了 *MatchOption* MQMO_MATCH_MSG_TOKEN ，並且既未指定 MQGMO_MSG_UNDER_CURSOR 也未指定 MQGMO_BROWSE_MSG_UNDER_CURSOR ，則只能傳回該 *MsgToken* 所識別的訊息。 不論 INDXTYPE 為何，此選項在所有本端佇列上都有效，在 z/OS 上，您只能在「工作量管理程式 (WLM)」佇列上使用 INDXTYPE (MSGXX_ENCODE_CASE_ONE token)。

會檢查指定的任何其他 *MatchOptions* ，如果它們不相符，則會傳回 MQRC_NO_MSG_AVAILABLE。 如果 MQGMO_BROWSE_NEXT 編碼為 MQMO_MATCH_MSG_TOKEN ，則只有在 *MsgToken* 所識別的訊息超出呼叫控點的瀏覽游標時，才會傳回該訊息。

如果指定 MQGMO_MSG_UNDER_CURSOR 或 MQGMO_BROWSE_MSG_UNDER_CURSOR ，則會忽略 MQMO_MATCH_MSG_TOKEN。

具有下列取得訊息選項的 MQMO_MATCH_MSG_TOKEN 無效:

- MQGMO_WAIT
- MQGMO_SET_SIGNAL

對於指定 MQMO_MATCH_MSG_TOKEN 的 MQGET 呼叫，必須將第 3 版或更新版本的 MQGMO 提供給呼叫，否則會傳回 MQRC_WRONG_GMO_VERSION。

如果此時 *MsgToken* 無效，除非發生另一個錯誤，否則會傳回具有 MQRC_NO_MSG_AVAILABLE 的 MQCC_FAILED。

選項 *(MQLONG)*

MQGMO 選項控制 MQGET 的動作。 您可以指定零個以上選項。 如果您需要多個選用值:

- 新增值 (不要多次新增相同的常數) ，或
- 使用位元 OR 運算來結合值 (如果程式設計語言支援位元運算)。

會記下無效的選項組合; 所有其他組合都是有效的。

等待選項: 下列選項與等待訊息抵達佇列相關:

MQGMO_WAIT
應用程式會等待適當的訊息到達。 應用程式等待的時間上限指定在 *WaitInterval* 中。

重要: 如果立即有適當的訊息可用，則沒有等待或延遲。

如果禁止 MQGET 要求，或在等待時禁止 MQGET 要求，則會取消等待。 不論佇列上是否有適當的訊息，呼叫都會完成，並傳回 MQCC_FAILED 及原因碼 MQRC_GET_INHIBITED。

您可以將 MQGMO_WAIT 與 MQGMO_BROWSE_FIRST 或 MQGMO_BROWSE_NEXT 選項搭配使用。

如果數個應用程式在相同的共用佇列上等待，則下列規則會選取當適當的訊息到達時啟動哪個應用程式:

表 *505:* 在共用佇列上啟動 *MQGET* 呼叫的規則。

等待啟動的 MQGET 呼叫數		**結果**
使用 BROWSE 選項	**沒有 BROWSE 選項** [1]	
無	1 或更多	啟動一個沒有 BROWSE 選項的 MQGET 呼叫。
1 或更多	無	會啟動具有 BROWSE 選項的所有 MQGET 呼叫。
1 或更多	1 或更多	啟動一個沒有 BROWSE 選項的 MQGET 呼叫。 無法預期使用 BROWSE 選項啟動的 MQGET 呼叫數。

如果多個沒有 BROWSE 選項的 MQGET 呼叫正在相同佇列上等待，則只會啟動一個。 佇列管理程式會依下列順序嘗試提供等待呼叫的優先順序:

1. 只有特定訊息 (例如，具有特定 *MsgId* 或 *CorrelId* (或兩者) 的訊息) 才能滿足的特定 get-wait 要求。
2. 可由任何訊息滿足的一般 get-wait 要求。

註:

- 在第一個種類內，不會對更具體的 get-wait 要求提供額外的優先順序。 例如，同時指定 *MsgId* 和 *CorrelId* 的要求。
- 在任一種類內，無法預測選取哪個應用程式。 特別是，等待時間最長的應用程式不一定是選取的應用程式。
- 路徑長度及作業系統的優先順序排程考量，可能表示作業系統優先順序低於預期的等待中應用程式會擷取訊息。
- 也可能發生未等待的應用程式優先於的應用程式擷取訊息。

[1] 指定 MQGMO_LOCK 選項的 MQGET 呼叫會被視為非瀏覽呼叫。

在 z/OS 上，適用下列要點:

- 如果您想要應用程式在等待訊息送達時繼續進行其他工作，請考慮改用信號選項 (MQGMO_SET_SIGNAL)。 不過，信號選項是環境特定選項; 您在不同環境之間連接的應用程式不得使用它。
- 如果有多個 MQGET 呼叫正在等待相同的訊息，並混合等待和信號選項，則會平均考量每一個等待中的呼叫。 將 MQGMO_SET_SIGNAL 與 MQGMO_WAIT 一起指定是錯誤的。 將此選項與信號未處理的佇列控點一起指定也是錯誤。
- 如果您針對 *IndexType* 為 MQIT_MSG_TOKEN 的佇列指定 MQGMO_WAIT 或 MQGMO_SET_SIGNAL，則不允許任何選取準則。 這表示:
  - 如果您使用 version-1 MQGMO，請在 MQGET 呼叫 MQMI_NONE 及 MQCI_NONE 上指定的 MQMD 中設定 *MsgId* 及 *CorrelId* 欄位。
  - 如果您使用 version-2 或更新版本 MQGMO，請將 *MatchOptions* 欄位設為 MQMO_NONE。
- 對於共用佇列上的 MQGET 呼叫，且該呼叫是群組訊息的瀏覽要求或破壞性取得，而且 *MsgId* 或 *CorrelId* 都不符合， MQGET 呼叫會每 200 毫秒重新發出一次，直到適當的訊息到達佇列或等待間隔到期為止。

  此方法會導致非預期的處理額外負擔，而且在不常新增訊息時，不是有效的訊息擷取方法。 為了避免瀏覽案例的額外負擔，請在 MQGET 呼叫上指定相符的 *MsgId* (如果未檢索或由 *MsgId* 檢索) 或 *CorrelId* (如果由 *CorrelId* 檢索)。

如果指定 MQGMO_BROWSE_MSG_UNDER_CURSOR 或 MQGMO_MSG_UNDER_CURSOR，則會忽略 MQGMO_WAIT ; 不會引發任何錯誤。

MQGMO_NO_WAIT
如果沒有可用的適當訊息，應用程式不會等待。 MQGMO_NO_WAIT 與 MQGMO_WAIT 相反。 MQGMO_NO_WAIT 定義為輔助程式文件。 如果都未指定，則它是預設值。

MQGMO_SET_SIGNAL
將此選項與 *Signal1* 和 *Signal2* 欄位搭配使用。 它可讓應用程式在等待訊息到達時繼續進行其他工作。 它也容許 (如果有適當的作業系統機能可用的話) 應用程式等待訊息到達多個佇列。

註: MQGMO_SET_SIGNAL 選項是環境特有的; 請勿將它用於您要連接的應用程式。

在兩種情況下，呼叫完成的方式與未指定此選項相同:

1. 如果目前可用的訊息滿足訊息描述子中指定的準則。
2. 如果偵測到參數錯誤或其他同步錯誤。

如果目前沒有符合訊息描述子中所指定準則的訊息可用，則控制會回到應用程式，而不等待訊息送達。 *CompCode* 和 *Reason* 參數設為 MQCC_WARNING 和 MQRC_SIGNAL_REQUEST_ACCEPTED。 未設定訊息描述子中的其他輸出欄位以及 MQGET 呼叫的輸出參數。 當適當的訊息稍後到達時，會透過張貼歐洲央行來傳遞信號。

然後，呼叫者必須重新發出 MQGET 呼叫來擷取訊息。 應用程式可以使用作業系統提供的功能來等待此信號。

如果作業系統提供多重等待機制，您可以使用它來等待訊息到達數個佇列中的任何一個佇列。

如果指定非零 *WaitInterval*，則會在等待間隔到期之後遞送信號。 佇列管理程式也可以取消等待，在此情況下會遞送信號。

多個 MQGET 呼叫可以為相同的訊息設定信號。 應用程式的啟動順序與 MQGMO_WAIT 的說明相同。

如果有多個 MQGET 呼叫正在等待相同的訊息，則每一個等待中的呼叫都會被視為相等。 呼叫可以包含等待和信號選項的混合。

在特定條件下， MQGET 呼叫可以擷取訊息，並且可以遞送相同訊息到達所產生的信號。 當信號遞送時，應用程式必須準備好沒有可用的訊息。

佇列控點最多只能有一個信號要求未完成。

此選項不適用於下列任何選項:

- MQGMO_UNLOCK

- MQGMO_WAIT

對於共用佇列上的 MQGET 呼叫，且該呼叫是瀏覽要求或群組訊息的破壞性取得，且 *MsgId* 或 *CorrelId* 都不符合，則會在 200 毫秒之後 MQEC_MSG_ARRIVED 公佈使用者的信號 ECB。

即使適當的訊息可能尚未到達佇列，在等待間隔到期之前，當佇列隨 MQEC_WAIT_INTERVAL_EXPIRED 一起公佈時，仍會發生這種情況。 當 MQEC_MSG_ARRIVED 公佈時，您必須重新發出第二個 MQGET 呼叫來擷取訊息 (如果有的話)。

此技術是用來確保及時通知您訊息到達，但在與非共用佇列上的類似呼叫序列相比較時，可能會顯示為非預期的處理額外負擔。

當不常新增訊息時，這不是有效的訊息擷取方法。 為了避免瀏覽案例的額外負擔，請在 MQGET 呼叫上指定相符的 *MsgId* (如果未檢索或由 *MsgId* 檢索) 或 *CorrelId* (如果由 *CorrelId* 檢索)。

此選項僅在 z/OS 上受支援。

MQGMO_FAIL_IF_QUIESCING
如果佇列管理程式處於靜止狀態，則強制 MQGET 呼叫失敗。

在 z/OS 上，如果連線 (適用於 CICS 或 IMS 應用程式) 處於靜止狀態，此選項也會強制 MQGET 呼叫失敗。

如果此選項與 MQGMO_WAIT 或 MQGMO_SET_SIGNAL 一起指定，且在佇列管理程式進入靜止狀態時等待或信號未完成:

- 已取消等待，且呼叫會傳回完成碼 MQCC_FAILED ，原因碼為 MQRC_Q_MGR_QUIESCING 或 MQRC_CONNECTION_QUIESCING。
- 使用環境特定的信號完成碼來取消信號。

  在 z/OS 上，信號完成，事件完成碼為 MQEC_Q_MGR_QUIESCING 或 MQEC_CONNECTION_QUIESCING。

如果未指定 MQGMO_FAIL_IF_QUIESCING ，且佇列管理程式或連線進入靜止狀態，則不會取消等待或信號。

同步點選項: 下列選項與工作單元內 MQGET 呼叫的參與相關:

MQGMO_SYNCPOINT
要求是在正常工作單元通訊協定內運作。 訊息會標示為其他應用程式無法使用，但只有在確定工作單元時，才會從佇列中刪除它。 如果工作單元已取消，則訊息會重新變成可用。

您可以維持未設定 MQGMO_SYNCPOINT 和 MQGMO_NO_SYNCPOINT 。 在此情況下，工作單元通訊協定中取得要求的併入由執行佇列管理程式的環境決定。 它不是由執行應用程式的環境所決定。 在 z/OS 上，取得要求是在工作單元內。 在所有其他環境中，取得要求不在工作單元內。

由於這些差異，您想要埠的應用程式不得容許此選項預設; 請明確指定 MQGMO_SYNCPOINT 或 MQGMO_NO_SYNCPOINT 。

此選項不適用於下列任何選項:

- 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
要求是在正常工作單元通訊協定內運作，但 只有 在擷取的訊息持續存在時。 持續訊息在 MQMD 的 *Persistence* 欄位中具有值 MQPER_PERSISTENT 。

- 如果訊息持續存在，佇列管理程式會處理呼叫，如同應用程式已指定 MQGMO_SYNCPOINT 一樣。

- 如果訊息不是持續的，佇列管理程式會如同應用程式已指定 MQGMO_NO_SYNCPOINT 一樣處理呼叫。

此選項不適用於下列任何選項:

- 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

此選項在下列環境中受支援: AIX、 HP-UX、 z/OS、 IBM i、Solaris 及 Linux，以及連接至這些系統的 WebSphere MQ MQI 用戶端。

MQGMO_NO_SYNCPOINT
要求是在正常工作單元通訊協定之外運作。 如果您收到沒有瀏覽選項的訊息，則會立即從佇列中刪除該訊息。 藉由取消工作單元，無法再次使訊息可供使用。

如果您指定 MQGMO_BROWSE_FIRST 或 MQGMO_BROWSE_NEXT，則會假設此選項。

您可以維持未設定 MQGMO_SYNCPOINT 和 MQGMO_NO_SYNCPOINT 。 在此情況下，工作單元通訊協定中取得要求的併入由執行佇列管理程式的環境決定。 它不是由執行應用程式的環境所決定。 在 z/OS 上，取得要求是在工作單元內。 在所有其他環境中，取得要求不在工作單元內。

由於這些差異，您想要埠的應用程式不得容許此選項預設; 請明確指定 MQGMO_SYNCPOINT 或 MQGMO_NO_SYNCPOINT 。

此選項不適用於下列任何選項:

- MQGMO_MARK_SKIP_BACKOUT
- MQGMO_SYNCPOINT
- MQGMO_SYNCPOINT_IF_PERSISTENT

MQGMO_MARK_SKIP_BACKOUT
在佇列上還原以這個選項標示的訊息，而不回復工作單元。

此選項僅在 z/OS 上受支援。

如果指定此選項，則也必須指定 MQGMO_SYNCPOINT 。 MQGMO_MARK_SKIP_BACKOUT 不適用於下列任何選項:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_BROWSE_NEXT
- MQGMO_LOCK
- MQGMO_NO_SYNCPOINT
- MQGMO_SYNCPOINT_IF_PERSISTENT
- MQGMO_UNLOCK

註: 在 IMS 和 CICS 上，在取消包含標示為 MQGMO_MARK_SKIP_BACKOUT 之訊息的工作單元之後，您可能需要發出額外的 WebSphere MQ 呼叫。 在確定包含所標示訊息的新工作單元之前，您必須先發出 WebSphere MQ 呼叫。 此呼叫可以是您喜歡的任何 WebSphere MQ 呼叫。

1. 在 IMS 上，如果您尚未套用 IMS APAR PN60855 ，且正在執行 IMS MPP 或 BMP 應用程式。
2. 在 CICS 上，如果您正在執行任何應用程式。

在這兩種情況下，在確定包含已取消訊息的新工作單元之前，請先發出任何 WebSphere MQ 呼叫。

註: 在工作單元內，只能有一個 get 要求標示為 skipping backout ，以及一個或數個未標示的 get 要求。

如果應用程式退出工作單元，則使用 MQGMO_MARK_SKIP_BACKOUT 擷取的訊息不會還原至其先前狀態。 其他資源更新項目已取消。 訊息會被視為已在取消要求所啟動的新工作單元中擷取。 在沒有 MQGMO_MARK_SKIP_BACKOUT 選項的情況下擷取訊息。

如果在變更部分資源之後，工作單元顯然無法順利完成，則 MQGMO_MARK_SKIP_BACKOUT 非常有用。 如果您省略此選項，則取消工作單元會恢復佇列上的訊息。 下次擷取訊息時，會再次發生相同的事件順序。

不過，如果您在原始 MQGET 呼叫上指定 MQGMO_MARK_SKIP_BACKOUT，則取消工作單元會取消其他資源的更新。 訊息會被視為已在新的工作單元下擷取。 應用程式可以執行適當的錯誤處理。 它可以將報告訊息傳送給原始訊息的傳送者，或將原始訊息放在無法傳送郵件的佇列上。 然後，它可以確定新的工作單元。 確定新的工作單元會從原始佇列永久移除訊息。

MQGMO_MARK_SKIP_BACKOUT 會標示單一實體訊息。 如果訊息屬於訊息群組，則不會標示群組中的其他訊息。 同樣地，如果標示的訊息是邏輯訊息的區段，則不會標示邏輯訊息中的其他區段。

可以標示群組中的任何訊息，但如果使用 MQGMO_LOGICAL_ORDER 擷取訊息，則標示群組中的第一個訊息會非常有利。 如果取消工作單元，則會將第一個 (標示的) 訊息移至新的工作單元。 會在佇列上恢復群組中的第二個及更新版本訊息。 另一個應用程式無法使用 MQGMO_LOGICAL_ORDER 來擷取留在佇列上的訊息。 群組中的第一個訊息不再位於佇列上。 不過，支援工作單元的應用程式可以使用 MQGMO_LOGICAL_ORDER 選項，將第二個及稍後的訊息擷取至新的工作單元。 已擷取第一個訊息。

有時您可能需要退出新的工作單元。 例如，因為無法傳送郵件的佇列已滿，且不得捨棄訊息。 取消新工作單元會恢復原始佇列上的訊息，以防止訊息遺失。 不過，在此狀況下，無法繼續處理。 在取消新的工作單元之後，應用程式必須通知操作員或管理者發生無法復原的錯誤，然後完成。

只有在包含 get 要求的工作單元因應用程式取消而岔斷時， MQGMO_MARK_SKIP_BACKOUT 才會運作。 如果包含 get 要求的工作單元因交易或系統失敗而取消，則會忽略 MQGMO_MARK_SKIP_BACKOUT 。 使用此選項擷取的任何訊息在佇列上恢復的方式與未使用此選項擷取的訊息恢復的方式相同。

瀏覽選項: 下列選項與瀏覽佇列上的訊息相關:

MQGMO_BROWSE_FIRST
使用 MQOO_BROWSE 選項開啟佇列時， 會建立瀏覽游標，並以邏輯方式放置在佇列上第一個訊息之前。 然後，您可以使用指定 MQGMO_BROWSE_FIRST、 MQGMO_BROWSE_NEXT 或 MQGMO_BROWSE_MSG_UNDER_CURSOR 選項的 MQGET 呼叫，以非破壞性方式從佇列中擷取訊息。 瀏覽游標會標示佇列上的訊息內的位置，下一個 MQGET 呼叫 MQGMO_BROWSE_NEXT 會從該位置搜尋適當的訊息。

MQGMO_BROWSE_FIRST 不適用於下列任何選項:

- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_BROWSE_NEXT
- MQGMO_MARK_SKIP_BACKOUT
- MQGMO_MSG_UNDER_CURSOR
- MQGMO_SYNCPOINT
- MQGMO_SYNCPOINT_IF_PERSISTENT
- MQGMO_UNLOCK

如果未開啟佇列進行瀏覽，也會發生錯誤。

含有 MQGMO_BROWSE_FIRST 的 MQGET 呼叫會忽略瀏覽游標的前一個位置。 會擷取佇列上滿足訊息描述子中所指定條件的第一個訊息。 訊息會保留在佇列上，而瀏覽游標會定位在此訊息上。

在此呼叫之後，瀏覽游標會定位在已傳回的訊息上。 在發出具有 MQGMO_BROWSE_NEXT 的下一個 MQGET 呼叫之前，可能會從佇列中移除該訊息。 在此情況下，瀏覽游標會保留在佇列中訊息所佔用的位置，即使該位置現在是空的。

搭配使用 MQGMO_MSG_UNDER_CURSOR 選項與非瀏覽 MQGET 呼叫，以從佇列中移除訊息。

即使使用相同的 *Hobj* 控點，非瀏覽 MQGET 呼叫也不會移動瀏覽游標。 也不會由傳回完成碼 MQCC_FAILED 或原因碼 MQRC_TRUNCATED_MSG_FAILED 的瀏覽 MQGET 呼叫所移動。

指定 MQGMO_LOCK 選項與此選項，以鎖定所瀏覽的訊息。

您可以使用 MQGMO_* 和 MQMO_* 選項的任何有效組合來指定 MQGMO_BROWSE_FIRST，這些選項可控制邏輯訊息群組和區段中的訊息處理。

如果您指定 MQGMO_LOGICAL_ORDER，則會以邏輯順序瀏覽訊息。 如果您省略該選項，則會以實體順序瀏覽訊息。 如果您指定 MQGMO_BROWSE_FIRST，則可以在邏輯順序與實體順序之間切換。 使用 MQGMO_BROWSE_NEXT 的後續 MQGET 呼叫瀏覽佇列的順序，與為佇列控點指定 MQGMO_BROWSE_FIRST 的最新呼叫相同。

佇列管理程式會為 MQGET 呼叫保留兩組群組及區段資訊。 瀏覽呼叫的群組和區段資訊會與從佇列中移除訊息之呼叫的資訊分開保留。 如果您指定 MQGMO_BROWSE_FIRST，佇列管理程式會忽略用於瀏覽的群組和區段資訊。 它會掃描佇列，就像沒有現行群組和現行邏輯訊息一樣。 如果 MQGET 呼叫成功，完成碼 MQCC_OK 或 MQCC_WARNING，則用於瀏覽的群組和區段資訊會設為所傳回訊息的群組和區段資訊。 如果呼叫失敗，群組和區段資訊會維持與呼叫之前的相同。

MQGMO_BROWSE_NEXT
將瀏覽游標前進至佇列上滿足 MQGET 呼叫上指定的選取準則的下一個訊息。 訊息會傳回給應用程式，但會保留在佇列上。

MQGMO_BROWSE_NEXT 不適用於下列任何選項:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_MARK_SKIP_BACKOUT
- MQGMO_MSG_UNDER_CURSOR
- MQGMO_SYNCPOINT
- MQGMO_SYNCPOINT_IF_PERSISTENT
- MQGMO_UNLOCK

如果未開啟佇列進行瀏覽，也會發生錯誤。

如果 MQGMO_BROWSE_NEXT 是在開啟佇列進行瀏覽之後第一個瀏覽佇列的呼叫，則其行為方式與 MQGMO_BROWSE_FIRST 相同。

在發出具有 MQGMO_BROWSE_NEXT 的下一個 MQGET 呼叫之前，可能會從佇列中移除游標下的訊息。 即使該位置現在是空的，瀏覽游標在邏輯上仍會保留在佇列中訊息所佔用的位置。

訊息以兩種方式之一儲存在佇列上:

- 優先順序 (MQMDS_PRIORITY) 內的 FIFO，或
- FIFO 不論 優先順序 (MQMDS_FIFO)

MsgDeliverySequence 佇列屬性指出套用的方法 (如需詳細資料，請參閱 第 731 頁的『佇列的屬性』)。

佇列的 *MsgDeliverySequence* 可能是 MQMDS_PRIORITY。 訊息到達佇列的優先順序高於瀏覽游標目前所指向的佇列。 在此情況下，在使用 MQGMO_BROWSE_NEXT 對佇列進行現行清理期間，找不到較高優先順序的訊息。 只有在使用 MQGMO_BROWSE_FIRST 重設瀏覽游標或重新開啟佇列之後，才能找到它。

必要的話， MQGMO_MSG_UNDER_CURSOR 選項可以與非瀏覽 MQGET 呼叫搭配使用，以從佇列中移除訊息。

使用相同 *Hobj* 控點的非瀏覽 MQGET 呼叫不會移動瀏覽游標。

在此選項中指定 MQGMO_LOCK 選項，以鎖定所瀏覽的訊息。

您可以使用 MQGMO_* 和 MQMO_* 選項的任何有效組合來指定 MQGMO_BROWSE_NEXT，這些選項可控制邏輯訊息群組和區段中的訊息處理。

如果您指定 MQGMO_LOGICAL_ORDER，則會以邏輯順序瀏覽訊息。 如果您省略該選項，則會以實體順序瀏覽訊息。 如果您指定 MQGMO_BROWSE_FIRST，則可以在邏輯順序與實體順序之間切換。 使用 MQGMO_BROWSE_NEXT 的後續 MQGET 呼叫瀏覽佇列的順序，與為佇列控點指定 MQGMO_BROWSE_FIRST 的最新呼叫相同。 如果未滿足此條件，則呼叫會失敗，原因碼為 MQRC_INCONSISTENT_BROWSE。

註: 如果未指定 MQGMO_LOGICAL_ORDER，當使用 MQGET 呼叫來瀏覽訊息群組尾端之後，請特別小心。 例如，假設群組中的最後一則訊息在佇列上群組中的第一個訊息之前。 使用 MQGMO_BROWSE_NEXT 來瀏覽群組結尾之後，指定 MQMO_MATCH_MSG_SEQ_NUMBER 並將 *MsgSeqNumber* 設為 1 會傳回群組中已瀏覽過的第一個訊息。 此結果可能會立即發生，如果有中間群組，則稍後可能會有一些 MQGET 呼叫。 相同的考量適用於不在群組中的邏輯訊息。

瀏覽呼叫的群組和區段資訊會與從佇列中移除訊息之呼叫的資訊分開保留。

MQGMO_BROWSE_MSG_UNDER_CURSOR
擷取由瀏覽游標以非破壞性方式指向的訊息，不論 MQGMO 中 *MatchOptions* 欄位中指定的 MQMO_* 選項為何。

MQGMO_BROWSE_MSG_UNDER_CURSOR 不適用於下列任何選項:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_NEXT
- MQGMO_MARK_SKIP_BACKOUT
- MQGMO_MSG_UNDER_CURSOR
- MQGMO_SYNCPOINT
- MQGMO_SYNCPOINT_IF_PERSISTENT
- MQGMO_UNLOCK

如果未開啟佇列進行瀏覽，也會發生錯誤。

瀏覽游標所指向的訊息是前次使用 MQGMO_BROWSE_FIRST 或 MQGMO_BROWSE_NEXT 選項擷取的訊息。 如果在此佇列開啟之後未對此佇列發出任何這些呼叫，則呼叫會失敗。 如果已破壞性地擷取瀏覽游標下的訊息，則呼叫也會失敗。

此呼叫不會變更瀏覽游標的位置。

MQGMO_MSG_UNDER_CURSOR 選項可以與非瀏覽 MQGET 呼叫搭配使用，以從佇列中移除訊息。

即使使用相同的 *Hobj* 控點，非瀏覽 MQGET 呼叫也不會移動瀏覽游標。 也不會由傳回完成碼 MQCC_FAILED 或原因碼 MQRC_TRUNCATED_MSG_FAILED 的瀏覽 MQGET 呼叫所移動。

如果 MQGMO_BROWSE_MSG_UNDER_CURSOR 與 MQGMO_LOCK 一起指定:

- 如果已鎖定訊息，則它必須是游標下的訊息，因此會傳回該訊息，而不會重新解除鎖定及鎖定。 訊息會保持鎖定狀態。
- 如果沒有已鎖定的訊息，且瀏覽游標下有訊息，則會鎖定並傳回給應用程式。 如果瀏覽游標下沒有訊息，則呼叫會失敗。

如果在未指定 MQGMO_LOCK 的情況下指定 MQGMO_BROWSE_MSG_UNDER_CURSOR :

- 如果已鎖定訊息，則必須是游標下的訊息。 訊息會傳回至應用程式，然後解除鎖定。 因為訊息現在已解除鎖定，所以不保證可以再次瀏覽它，或由相同應用程式以破壞性方式擷取它。 從佇列取得訊息的另一個應用程式可能已破壞性地擷取它。
- 如果沒有已鎖定的訊息，且瀏覽游標之下有訊息，則會將它傳回給應用程式。 如果瀏覽游標下沒有訊息，則呼叫會失敗。

如果 MQGMO_COMPLETE_MSG 與 MQGMO_BROWSE_MSG_UNDER_CURSOR 一起指定，則瀏覽游標必須識別其 MQMD 中的 *Offset* 欄位為零的訊息。 如果未滿足此條件，則呼叫會失敗，原因碼為 MQRC_INVALID_MSG_UNDER_CURSOR。

瀏覽呼叫的群組和區段資訊會與從佇列中移除訊息之呼叫的資訊分開保留。

MQGMO_MSG_UNDER_CURSOR
擷取瀏覽游標所指向的訊息，不論 MQGMO 中 *MatchOptions* 欄位中指定的 MQMO_* 選項為何。 從佇列中移除訊息。

瀏覽游標所指向的訊息是前次使用 MQGMO_BROWSE_FIRST 或 MQGMO_BROWSE_NEXT 選項擷取的訊息。

如果 MQGMO_COMPLETE_MSG 與 MQGMO_MSG_UNDER_CURSOR 一起指定，則瀏覽游標必須識別其 MQMD 中的 *Offset* 欄位為零的訊息。 如果未滿足此條件，則呼叫會失敗，原因碼為 MQRC_INVALID_MSG_UNDER_CURSOR。

此選項不適用於下列任何選項:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_BROWSE_NEXT
- MQGMO_UNLOCK

如果未開啟佇列進行瀏覽及輸入，也會發生錯誤。 如果瀏覽游標目前未指向可擷取的訊息， MQGET 呼叫會傳回錯誤。

MQGMO_MARK_BROWSE_HANDLE
即會標示成功 MQGET 所傳回的訊息，或所傳回 *MsgToken* 所識別的訊息。 標記特定於呼叫中使用的物件控點。

訊息未從佇列中移除。

只有在同時指定下列其中一個選項時， MQGMO_MARK_BROWSE_HANDLE 才有效:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_BROWSE_NEXT

MQGMO_MARK_BROWSE_HANDLE 不適用於下列任何選項:

- MQGMO_ALL_MSGS_AVAILABLE
- MQGMO_ALL_SEGMENTS_AVAILABLE
- MQGMO_COMPLETE_MSG
- MQGMO_LOCK
- MQGMO_LOGICAL_ORDER
- MQGMO_UNLOCK

訊息會保持此狀態，直到發生下列其中一個事件為止:

- 所涉及的物件控點已正常關閉或以其他方式關閉。
- 透過呼叫 MQGET ，並使用選項 MQGMO_UNMARK_BROWSE_HANDLE，將此控點的訊息取消標示。
- 此訊息從破壞性 MQGET 的呼叫中傳回，該呼叫以 MQCC_OK 或 MQCC_WARNING 完成。 即使稍後回復 MQGET ，訊息狀態仍會維持變更。
- 訊息到期。

MQGMO_MARK_BROWSE_CO_OP
由成功 MQGET 傳回的訊息，或由傳回的 *MsgToken* 識別的訊息，會針對協同作業集中的所有控點標示。

合作層次標記是除了任何可能已設定的控點層次標記之外的其他標記。

訊息未從佇列中移除。

僅當對指定 MQOO_CO_OP 之 MQOPEN 的呼叫傳回所使用的物件控點時， MQGMO_MARK_BROWSE_CO_OP 才有效。 您也必須指定下列其中一個 MQGMO 選項:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_BROWSE_NEXT

此選項不適用於下列任何選項:

- MQGMO_ALL_MSGS_AVAILABLE

- MQGMO_ALL_SEGMENTS_AVAILABLE
- MQGMO_COMPLETE_MSG
- MQGMO_LOCK
- MQGMO_LOGICAL_ORDER
- MQGMO_UNLOCK

如果已標示訊息，且未指定 MQGMO_UNMARKED_BROWSE_MSG 選項，則呼叫會失敗，並傳回 MQCC_FAILED 及原因碼 MQRC_MSG_MARKED_BROWSE_CO_OP。

訊息會保持此狀態，直到發生下列其中一個事件為止:

- 會關閉協同作業集中的所有物件控點。
- 透過呼叫 MQGET 並使用選項 MQGMO_UNMARK_BROWSE_CO_OP，取消標示此訊息以用於協同作業的瀏覽器。
- 佇列管理程式會自動取消標示訊息。
- 從對非瀏覽 MQGET 的呼叫傳回此訊息。 即使稍後回復 MQGET，訊息狀態仍會維持變更。
- 訊息到期。

MQGMO_UNMARKED_BROWSE_MSG
指定 MQGMO_UNMARKED_BROWSE_MSG 的 MQGET 呼叫會傳回視為未標示其控點的訊息。 如果訊息已標示為其控點，則不會傳回訊息。 如果佇列是由使用選項 MQOO_CO_OP 對 MQOPEN 的呼叫所開啟，且訊息已由協同作業集的成員標示，則它也不會傳回訊息。

此選項不適用於下列任何選項:

- MQGMO_ALL_MSGS_AVAILABLE
- MQGMO_ALL_SEGMENTS_AVAILABLE
- MQGMO_COMPLETE_MSG
- MQGMO_LOCK
- MQGMO_LOGICAL_ORDER
- MQGMO_UNLOCK

MQGMO_UNMARK_BROWSE_CO_OP
在呼叫指定此選項的 MQGET 之後，該組協同作業控點中的任何開啟控點都不再考量該訊息，以標示該協同作業控點。 如果在此呼叫之前已在控點層次標示訊息，則仍會將該訊息視為在控點層次標示。

使用 MQGMO_UNMARK_BROWSE_CO_OP 只有在使用選項 MQOO_CO_OP 成功呼叫 MQOPEN 所傳回的控點時才有效。 即使訊息未被視為由一組協同作業的控點所標示， MQGET 仍會成功。

MQGMO_UNMARK_BROWSE_CO_OP 在非瀏覽 MQGET 呼叫中無效，或具有下列任何選項:

- 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
呼叫指定此選項的 MQGET 之後，不再將找到的訊息視為已被此控點標示。

即使未標示此控點的訊息，呼叫也會成功。

此選項在非瀏覽 MQGET 呼叫上無效，或具有下列任何選項:

- 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

鎖定選項: 下列選項與鎖定佇列上的訊息相關:

MQGMO_LOCK
鎖定已瀏覽的訊息，讓佇列開啟的任何其他控點都無法看見該訊息。 只有在同時指定下列其中一個選項時，才能指定選項:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_NEXT
- MQGMO_BROWSE_MSG_UNDER_CURSOR

每一個佇列控點只能鎖定一個訊息。 訊息可以是邏輯訊息或實體訊息:

- 如果您指定 MQGMO_COMPLETE_MSG，則構成邏輯訊息的所有訊息區段都會鎖定至佇列控點。 訊息必須全部出現在佇列上，且可供擷取。
- 如果您省略 MQGMO_COMPLETE_MSG，則只會將單一實體訊息鎖定至佇列控點。 如果此訊息碰巧是邏輯訊息的區段，則已鎖定區段會阻止其他應用程式使用 MQGMO_COMPLETE_MSG 來擷取或瀏覽邏輯訊息。

鎖定訊息一律是瀏覽游標下的訊息。 稍後指定 MQGMO_MSG_UNDER_CURSOR 選項的 MQGET 呼叫可以從佇列中移除訊息。 其他使用佇列控點的 MQGET 呼叫也可以移除訊息 (例如，指定已鎖定訊息之訊息 ID 的呼叫)。

如果呼叫傳回完成碼 MQCC_FAILED 或 MQCC_WARNING，原因碼為 MQRC_TRUNCATED_MSG_FAILED，則不會鎖定任何訊息。

如果應用程式未從佇列中移除訊息，則下列其中一個動作會釋放鎖定:

- 對這個控點發出另一個 MQGET 呼叫，並指定 MQGMO_BROWSE_FIRST 或 MQGMO_BROWSE_NEXT。 如果呼叫以 MQCC_OK 或 MQCC_WARNING 完成，則會釋放鎖定。 如果呼叫完成並出現 MQCC_FAILED，則訊息會保持鎖定。 不過，下列異常狀況適用:
  - 如果 MQCC_WARNING 隨 MQRC_TRUNCATED_MSG_FAILED 一起傳回，則不會解除鎖定訊息。
  - 如果隨 MQRC_NO_MSG_AVAILABLE 一起傳回 MQCC_FAILED，則會解除鎖定訊息。

  如果您也指定 MQGMO_LOCK，則會鎖定傳回的訊息。 如果您省略 MQGMO_LOCK，則在呼叫之後不會有鎖定訊息。

  如果您指定 MQGMO_WAIT，且沒有立即可用的訊息，則在開始等待之前會解除鎖定原始訊息。
- 使用 MQGMO_BROWSE_MSG_UNDER_CURSOR 針對此控點發出另一個 MQGET 呼叫，但不使用 MQGMO_LOCK。 如果呼叫以 MQCC_OK 或 MQCC_WARNING 完成，則會釋放鎖定。 如果呼叫完成並出現 MQCC_FAILED，則訊息會保持鎖定。 不過，下列異常狀況適用:
  - 如果 MQCC_WARNING 隨 MQRC_TRUNCATED_MSG_FAILED 一起傳回，則不會解除鎖定訊息。
- 使用 MQGMO_UNLOCK 針對此控點發出另一個 MQGET 呼叫。
- 使用控點發出 MQCLOSE 呼叫。 MQCLOSE 可能是隱含的，由應用程式結束所造成。

除了指定隨附的瀏覽選項所需的 MQOO_BROWSE 之外，不需要任何特殊 MQOPEN 選項來指定 MQGMO_LOCK。

MQGMO_LOCK 不適用於下列任何選項:

- MQGMO_MARK_SKIP_BACKOUT
- MQGMO_SYNCPOINT

- MQGMO_SYNCPOINT_IF_PERSISTENT
- MQGMO_UNLOCK

當您在 HP Integrity NonStop Server 上使用 IBM WebSphere MQ 用戶端時，如果由 TMF 協調，則無法將 MQGMO_LOCK 用於 z/OS 佇列管理程式。

MQGMO_UNLOCK
要解除鎖定的訊息先前必須已由具有 MQGMO_LOCK 選項的 MQGET 呼叫鎖定。 如果此控點未鎖定任何訊息，則呼叫會以 MQCC_WARNING 和 MQRC_NO_MSG_LOCKED 完成。

如果您指定 MQGMO_UNLOCK，則不會檢查或變更 *MsgDesc*、 *BufferLength*、 *Buffer* 及 *DataLength* 參數。 *Buffer* 中未傳回任何訊息。

指定 MQGMO_UNLOCK 不需要特殊開啟選項 (雖然首先需要 MQOO_BROWSE 才能發出鎖定要求)。

此選項不適用於下列選項以外的任何選項:

- MQGMO_NO_WAIT
- MQGMO_NO_SYNCPOINT

不論是否指定，都會假設這兩個選項。

訊息資料選項: 下列選項與從佇列讀取訊息時處理訊息資料相關:

MQGMO_ACCEPT_TRUNCATED_MSG
如果訊息緩衝區太小而無法保留完整訊息，請容許 MQGET 呼叫填入緩衝區。 MQGET 會盡可能在緩衝區中填入訊息。 它會發出警告完成碼，並完成其處理。 這表示:

- 瀏覽訊息時，瀏覽游標會進階至傳回的訊息。
- 移除訊息時，會從佇列中移除傳回的訊息。
- 如果未發生其他錯誤，則會傳回原因碼 MQRC_TRUNCATED_MSG_ACCEPTED。

如果沒有這個選項，緩衝區仍會盡可能填滿訊息。 已發出警告完成碼，但未完成處理。 這表示:

- 瀏覽訊息時，瀏覽游標不是進階的。
- 移除訊息時，不會從佇列中移除訊息。
- 如果未發生其他錯誤，則會傳回原因碼 MQRC_TRUNCATED_MSG_FAILED 。

MQGMO_CONVERT
此選項會轉換訊息中的應用程式資料，以符合 MQGET 呼叫上 *MsgDesc* 參數中指定的 *CodedCharSetId* 及 *Encoding* 值。 在將資料複製到 *Buffer* 參數之前，會先轉換資料。

轉換處理程序會假設放置訊息時指定的 *Format* 欄位，以識別訊息中資料的本質。 訊息資料由佇列管理程式轉換為內建格式，並由使用者撰寫的結束程式轉換為其他格式。 如需資料轉換結束程式的詳細資料，請參閱 第 791 頁的『資料轉換結束程式』 。

- 如果轉換成功，則從 MQGET 呼叫返回時， *MsgDesc* 參數中指定的 *CodedCharSetId* 和 *Encoding* 欄位保持不變。
- 如果僅轉換失敗，則會傳回未轉換的訊息資料。 *MsgDesc* 中的 *CodedCharSetId* 及 *Encoding* 欄位會設為未轉換訊息的值。 在此情況下，完成碼為 MQCC_WARNING 。

在任一情況下，這些欄位都會說明 *Buffer* 參數中所傳回訊息資料的字集 ID 及編碼。

如需佇列管理程式執行轉換的格式名稱清單，請參閱 第 363 頁的『MQMD-訊息描述子』 中說明的 *Format* 欄位。

群組和區段選項: 下列選項與處理邏輯訊息群組和區段中的訊息相關。 在選項說明之前，以下是一些重要術語的定義:

實體訊息
實體訊息是可以放置在佇列上或從佇列中移除的最小資訊單元。 它通常對應於在單一 MQPUT、 MQPUT1 或 MQGET 呼叫上指定或擷取的資訊。 每個實體訊息都有自己的訊息描述子 MQMD。 通常，實體訊息是透過訊息 ID (MQMD 中的 *MsgId* 欄位) 的不同值來識別。 佇列管理程式不會施行不同的值。

邏輯訊息

邏輯訊息是應用程式資訊的單一單元。 在沒有系統限制的情況下，邏輯訊息與實體訊息相同。 如果邏輯訊息很大，系統限制可能會建議或需要將邏輯訊息分割成兩個以上實體訊息 (稱為區段)。

已分段的邏輯訊息包含兩個以上具有相同非空值群組 ID (MQMD 中的 *GroupId* 欄位) 的實體訊息。 它們在 MQMD 中具有相同的訊息序號 *MsgSeqNumber* 欄位。 透過 MQMD 中區段偏移 *Offset* 欄位的不同值來識別區段。 區段偏移是實體訊息中資料從邏輯訊息中資料開始的偏移。 因為每一個區段都是實體訊息，所以邏輯訊息中的區段通常具有不同的訊息 ID。

尚未分段但傳送應用程式已允許分段的邏輯訊息也具有非空值群組 ID。 在此情況下，如果邏輯訊息不屬於訊息群組，則只有一個實體訊息具有該群組 ID。 除非邏輯訊息屬於訊息群組，否則傳送應用程式已禁止分段的邏輯訊息具有空值群組 ID MQGI_NONE。

訊息群組

訊息群組是一組具有相同非空值群組 ID 的一或多個邏輯訊息。 群組中的邏輯訊息由訊息序號的不同值來識別。 序號是 1 到 n 範圍內的整數，其中 n 是群組中邏輯訊息的數目。 如果已分段一個以上邏輯訊息，則群組中有 n 則以上的實體訊息。

MQGMO_LOGICAL_ORDER

MQGMO_LOGICAL_ORDER 控制佇列控點的連續 MQGET 呼叫所傳回訊息的順序。 必須在每一個呼叫上指定選項。

如果針對相同佇列控點的連續 MQGET 呼叫指定 MQGMO_LOGICAL_ORDER，則會依訊息序號的順序傳回群組中的訊息。 邏輯訊息的區段會依其區段偏移所給定的順序傳回。 此順序可能與那些訊息及區段在佇列上出現的順序不同。

註: 指定 MQGMO_LOGICAL_ORDER 不會對不屬於群組且不是區段的訊息產生負面影響。 實際上，會將這類訊息視為每一個都屬於只包含一個訊息的訊息群組。 從包含群組中混合訊息、訊息區段及未分段訊息 (不在群組中) 的佇列中擷取訊息時，可以放心地指定 MQGMO_LOGICAL_ORDER。

為了以所需順序傳回訊息，佇列管理程式會在連續 MQGET 呼叫之間保留群組和區段資訊。 群組及區段資訊可識別佇列控點的現行訊息群組及現行邏輯訊息。 它也會識別群組及邏輯訊息中的現行位置，以及是否在工作單元內擷取訊息。 由於佇列管理程式會保留此資訊，因此在每次 MQGET 呼叫之前，應用程式都不需要設定群組和區段資訊。 具體而言，它表示應用程式不需要在 MQMD 中設定 *GroupId*、*MsgSeqNumber* 及 *Offset* 欄位。 不過，應用程式必須在每次呼叫時正確設定 MQGMO_SYNCPOINT 或 MQGMO_NO_SYNCPOINT 選項。

開啟佇列時，沒有現行訊息群組及現行邏輯訊息。 當 MQGET 呼叫傳回具有 MQMF_MSG_IN_GROUP 旗標的訊息時，訊息群組會變成現行訊息群組。 在連續呼叫上指定 MQGMO_LOGICAL_ORDER 之後，該群組會保留現行群組，直到傳回具有下列項目的訊息為止:

- MQMF_LAST_MSG_IN_GROUP without MQMF_SEGMENT (亦即，群組中的最後一則邏輯訊息未分段)，或
- MQMF_LAST_MSG_IN_GROUP with MQMF_LAST_SEGMENT (亦即，傳回的訊息是群組中最後一個邏輯訊息的最後一個區段)。

當傳回這類訊息時，訊息群組會終止，當順利完成 MQGET 呼叫時，不再有現行群組。 同樣地，當 MQGET 呼叫傳回具有 MQMF_SEGMENT 旗標的訊息時，邏輯訊息會變成現行邏輯訊息。 當傳回具有 MQMF_LAST_SEGMENT 旗標的訊息時，邏輯訊息會終止。

如果未指定選取準則，則連續的 MQGET 呼叫會以正確的順序傳回佇列上第一個訊息群組的訊息。 然後，它們會傳回第二個訊息群組的訊息，依此類推，直到沒有其他可用的訊息為止。 在 *MatchOptions* 欄位中指定下列一或多個選項，即可選取傳回的特定訊息群組:

- MQMO_MATCH_MSG_ID
- MQMO_MATCH_CORREL_ID
- MQMO_MATCH_GROUP_ID

不過，只有在沒有現行訊息群組或邏輯訊息時，這些選項才有效。 如需進一步詳細資料，請參閱 第 323 頁的『MQGMO-取得訊息選項』 中說明的 *MatchOptions* 欄位。

第 339 頁的表 506 顯示當嘗試尋找要在 MQGET 呼叫中傳回的訊息時，佇列管理程式所尋找的 *MsgId*、*CorrelId*、*GroupId*、*MsgSeqNumber* 及 *Offset* 欄位值。 這些規則適用於從佇列中移除訊息，以及瀏覽佇列上的訊息。 在表格中，表示「是」或「否」:

LOG ORD
指出是否在通話中指定 MQGMO_LOGICAL_ORDER 選項。

Cur grp
指出在呼叫之前是否存在現行訊息群組。

Cur log msg
指出在呼叫之前是否存在現行邏輯訊息。

其他直欄
顯示佇列管理程式所尋找的值。「前一個」表示針對佇列控點前一個訊息中的欄位所傳回的值。

表 *506:* 與邏輯訊息群組及區段中的訊息相關的 *MQGET* 選項

您指定的選項	**呼叫之前的群組和日誌訊息狀態**		**佇列管理程式尋找的值**				
LOG ORD	**Cur grp**	**Cur log msg**	*MsgId*	*CorrelId*	*GroupId*	*MsgSeqNumber*	*Offset*
是	否	否	控制者 *MatchOptions*	控制者 *MatchOptions*	控制者 *MatchOptions*	1	0
是	否	是	任何訊息 ID	任何相關性 ID	前一個群組 ID	1	前一個偏移 + 前一個區段長度
是	是	否	任何訊息 ID	任何相關性 ID	前一個群組 ID	前一個序號 + 1	0
是	是	是	任何訊息 ID	任何相關性 ID	前一個群組 ID	前一個序號	前一個偏移 + 前一個區段長度
否	兩者擇一	兩者擇一	控制者 *MatchOptions*	控制者 *MatchOptions*	控制者 *MatchOptions*	控制者 *MatchOptions*	控制者 *MatchOptions*

如果多個訊息群組出現在佇列上且適合傳回，則會依每一個群組中第一個邏輯訊息的第一個區段在佇列上的位置所決定的順序來傳回群組。 亦即，訊息序號為 1 且偏移為 0 的實體訊息會決定傳回合格群組的順序。

MQGMO_LOGICAL_ORDER 選項會影響工作單元，如下所示:

- 如果在工作單元內擷取群組中的第一個邏輯訊息或區段，則如果使用相同的佇列控點，則必須在工作單元內擷取群組中的所有其他邏輯訊息及區段。 不過，它們不需要在相同的工作單元內擷取。 這可讓由許多實體訊息組成的訊息群組，在佇列控點的兩個以上連續工作單元之間分割。
- 如果 未 在工作單元內擷取群組中的第一個邏輯訊息或區段，且使用相同的佇列控點，則無法在工作單元內擷取群組中的任何其他邏輯訊息和區段。

如果未滿足這些條件，則 MQGET 呼叫會失敗，原因碼為 MQRC_INCONSISTENT_UOW。

指定 MQGMO_LOGICAL_ORDER 時， MQGET 呼叫上提供的 MQGMO 不得小於 MQGMO_VERSION_2，且 MQMD 不得小於 MQMD_VERSION_2。 如果未滿足此條件，則呼叫會在適當的情況下失敗，原因碼為 MQRC_WRONG_GMO_VERSION 或 MQRC_WRONG_MD_VERSION。

如果針對佇列控點的連續 MQGET 呼叫 不 指定 MQGMO_LOGICAL_ORDER ，則會傳回訊息，而不管它們是否屬於訊息群組，或它們是否為邏輯訊息的區段。 這表示來自特定群組或邏輯訊息的訊息或區段可能不正常傳回，或與來自其他群組或邏輯訊息的訊息或區段混合，或與不在群組中且不是區段的訊息混合。 在此狀況下，連續 MQGET 呼叫所傳回的特定訊息是由在那些呼叫上指定的 MQMO_* 選項所控制 (如需這些選項的詳細資料，請參閱 第 323 頁的『MQGMO-取得訊息選項』 中說明的 *MatchOptions* 欄位)。

這是在系統失效之後，可用來重新啟動中間的訊息群組或邏輯訊息的技術。 當系統重新啟動時，應用程式可以將 *GroupId*、 *MsgSeqNumber*、 *Offset* 及 *MatchOptions* 欄位設為適當的值，然後發出 MQGET 呼叫並設定 MQGMO_SYNCPOINT 或 MQGMO_NO_SYNCPOINT，但 不 指定 MQGMO_LOGICAL_ORDER。 如果此呼叫成功，佇列管理程式會保留群組和區段資訊，後續使用該佇列控點的 MQGET 呼叫可以正常指定 MQGMO_LOGICAL_ORDER。

佇列管理程式針對 MQGET 呼叫所保留的群組及區段資訊，與它針對 MQPUT 呼叫所保留的群組及區段資訊不同。 此外，佇列管理程式還會保留下列項目的個別資訊:

- 從佇列中移除訊息的 MQGET 呼叫。
- MQGET 呼叫可瀏覽佇列上的訊息。

對於任何給定的佇列控點，應用程式可以將指定 MQGMO_LOGICAL_ORDER 的 MQGET 呼叫與未指定的 MQGET 呼叫混合。 不過，請注意下列要點:

- 如果您省略 MQGMO_LOGICAL_ORDER，則每一個成功的 MQGET 呼叫都會導致佇列管理程式將已儲存的群組及區段資訊設為與所傳回訊息相對應的值; 這會取代佇列管理程式保留給佇列控點的現有群組及區段資訊。 只會修改適用於呼叫動作 (瀏覽或移除) 的資訊。
- 如果您省略 MQGMO_LOGICAL_ORDER，則會在有現行訊息群組或邏輯訊息時，呼叫不會失敗; 呼叫可能會成功，並傳回 MQCC_WARNING 完成碼。 第 340 頁的表 507 顯示可能產生的各種觀察值。 在這些情況下，如果完成碼不是 MQCC_OK，則原因碼是下列其中一項 (適當的話):
  - MQRC_INCOMPLETE_GROUP
  - MQRC_INCOMPLETE_MSG
  - MQRC_INCONSISTENT_UOW

  註: 當瀏覽佇列時，或當關閉已開啟以進行瀏覽但未輸入的佇列時，佇列管理程式不會檢查群組和區段資訊; 在這些情況下，完成碼一律為 MQCC_OK (假設沒有其他錯誤)。

表 *507: MQGET* 或 *MQCLOSE* 呼叫與群組和區段資訊不一致時的結果

現行呼叫為	前一個呼叫是 MQGET，使用 MQGMO_LOGICAL_ORDER	前一個通話是 MQGET 沒有 MQGMO_LOGICAL_ORDER
MQGET (MQGMO_LOGICAL_ORDER)	MQCC_FAILED	MQCC_FAILED
MQGET 沒有 MQGMO_LOGICAL_ORDER	MQCC_WARNING	MQCC_OK
MQCLOSE 具有未終止的群組或邏輯訊息	MQCC_WARNING	MQCC_OK

建議要以邏輯順序擷取訊息及區段的應用程式指定 MQGMO_LOGICAL_ORDER，因為這是最簡單的選項。 此選項可讓應用程式解除管理群組和區段資訊的需求，因為佇列管理程式會管理該資訊。 不過，特殊化應用程式可能需要比 MQGMO_LOGICAL_ORDER 選項所提供的更多控制，而不指定該選項即可達成此目的。 然後，應用程式必須確保在每一個 MQGET 呼叫之前正確設定 MQMD 中的 *MsgId*、 *CorrelId*、 *GroupId*、 *MsgSeqNumber* 及 *Offset* 欄位，以及 MQGMO 中 *MatchOptions* 的 MQMO_* 選項。

例如，想要 轉遞 所接收實體訊息的應用程式，不論這些訊息是在邏輯訊息的群組或區段中，都 不得 指定 MQGMO_LOGICAL_ORDER。 在傳送與接收佇列管理程式之間有多條路徑的複雜網路中，實體訊息可能不正常送達。 透過在 MQPUT 呼叫上既未指定 MQGMO_LOGICAL_ORDER，也未指定對應的 MQPMO_LOGICAL_ORDER，轉遞應用程式可以在每一個實體訊息到達時立即擷取並轉遞它，而不需要等待邏輯順序中的下一個實體訊息到達。

在適當的情況下，您可以使用任何其他 MQGMO_* 選項以及各種 MQMO_* 選項來指定 MQGMO_LOGICAL_ORDER (請參閱上文)。

- 在 z/OS 上，專用及共用佇列支援此選項，但佇列必須具有索引類型 MQIT_GROUP_ID。 對於共用佇列，佇列對映的 CFSTRUCT 物件必須是 CFLEVEL (3) 或 CFLEVEL (4)。
- 在 AIX、 HP-UX、 IBM i、Solaris、 Linux、Windows，以及連接至這些系統的 WebSphere MQ MQI 用戶端上，所有本端佇列都支援此選項。

MQGMO_COMPLETE_MSG

MQGET 呼叫只能傳回完整邏輯訊息。 如果邏輯訊息已分段，則佇列管理程式會重新組合區段，並將完整的邏輯訊息傳回至應用程式; 擷取邏輯訊息的應用程式不會明顯看到邏輯訊息已分段的事實。

註: 這是唯一會導致佇列管理程式重新組合訊息區段的選項。 如果未指定，則區段在佇列上存在時 (且滿足 MQGET 呼叫上指定的其他選取準則) 會個別傳回至應用程式。 不想要接收個別區段的應用程式必須一律指定 MQGMO_COMPLETE_MSG。

如果要使用這個選項，應用程式必須提供足夠大的緩衝區來容納完整訊息，或指定 MQGMO_ACCEPT_TRUNCATED_MSG 選項。

如果佇列包含部分區段遺漏的分段訊息 (可能是因為它們在網路中已延遲且尚未到達) ，則指定 MQGMO_COMPLETE_MSG 會阻止擷取屬於不完整邏輯訊息的區段。 不過，這些訊息區段仍會構成 *CurrentQDepth* 佇列屬性的值; 這表示即使 *CurrentQDepth* 大於零，也可能沒有可擷取的邏輯訊息。

對於 持續 訊息，佇列管理程式只能在工作單元內重新組合區段:

- 如果 MQGET 呼叫在使用者定義的工作單元內運作，則會使用該工作單元。 如果在重組處理程序期間呼叫失敗，則佇列管理程式會在佇列上恢復在重組期間移除的任何區段。 不過，失敗並不會阻止順利確定工作單元。
- 如果呼叫是在使用者定義工作單元之外運作，且不存在使用者定義工作單元，則佇列管理程式會在呼叫期間建立工作單元。 如果呼叫成功，佇列管理程式會自動確定工作單元 (應用程式不需要這麼做)。 如果呼叫失敗，佇列管理程式會取消工作單元。
- 如果呼叫是在使用者定義工作單元之外運作，但存在使用者定義工作單元，則佇列管理程式無法重新組合。 如果訊息不需要重組，呼叫仍會成功。 但是如果訊息需要重組，則呼叫會失敗，原因碼為 MQRC_UOW_NOT_AVAILABLE。

對於 非持續性 訊息，佇列管理程式不需要工作單元即可執行重新組合。

每一個屬於區段的實體訊息都有自己的訊息描述子。 對於構成單一邏輯訊息的區段，訊息描述子中的大部分欄位對於邏輯訊息中的所有區段都是相同的; 通常只有 *MsgId*、 *Offset* 和 *MsgFlags* 欄位在邏輯訊息中的區段之間會有所不同。 不過，如果區段放置在中間佇列管理程式的無法傳送郵件的佇列上，則 DLQ 處理程式會擷取指定 MQGMO_CONVERT 選項的訊息，這可能會導致變更區段的字集或編碼。 如果 DLQ 處理程式順利傳送區段，則當區段到達目的地佇列管理程式時，區段可能具有與邏輯訊息中其他區段不同的字集或編碼。

由 *CodedCharSetId* 和 *Encoding* 欄位不同的區段所組成的邏輯訊息，無法由佇列管理程式重新組合成單一邏輯訊息。 相反地，佇列管理程式會重新組合並傳回邏輯訊息開頭的前幾個連續區段，這些區段具有相同的字集 ID 及編碼，且 MQGET 呼叫會適當地完成，並具有完成碼 MQCC_WARNING 及原因碼 MQRC_INCONSISTENT_CCSIDS 或 MQRC_INCONSISTENT_ENCODINGS。 不論是否指定 MQGMO_CONVERT，都會發生這種情況。 若要擷取其餘區段，應用程式必須重新發出不含 MQGMO_COMPLETE_MSG 選項的 MQGET 呼叫，逐一擷取區段。 MQGMO_LOGICAL_ORDER 可用來依序擷取其餘區段。

放置區段的應用程式也可以將訊息描述子中的其他欄位設為區段之間不同的值。 不過，如果接收端應用程式使用 MQGMO_COMPLETE_MSG 來擷取邏輯訊息，則這樣做沒有任何好處。 當佇列管理程式重新組合邏輯訊息時，它會在訊息描述子中傳回來自 第一個 區段之訊息描述子的值; 唯一例外是 *MsgFlags* 欄位，佇列管理程式會設定該欄位以指出重新組合的訊息是唯一區段。

如果對報告訊息指定 MQGMO_COMPLETE_MSG，則佇列管理程式會執行特殊處理。 佇列管理程式會檢查佇列，以查看該報告類型與邏輯訊息中不同區段相關的所有報告訊息是否都存在於佇列上。 如果是，則可以透過指定 MQGMO_COMPLETE_MSG，將它們擷取為單一訊息。 若要這樣做，必須由支援分段的佇列管理程式或 MCA 產生報告訊息，或原始應用程式必須要求至少 100 個位元組的訊息資料 (亦即，必須指定適當的 MQRO_*_WITH_DATA 或 MQRO_*_WITH_FULL_DATA 選項)。 如果區段存在的應用程式資料量小於完整數量，則會在傳回的報告訊息中以空值取代遺漏的位元組。

如果使用 MQGMO_MSG_UNDER_CURSOR 或 MQGMO_BROWSE_MSG_UNDER_CURSOR 指定 MQGMO_COMPLETE_MSG，則瀏覽游標必須位於其 MQMD 中 *Offset* 欄位值為 0 的訊息上。 如果未滿足此條件，則呼叫會失敗，原因碼為 MQRC_INVALID_MSG_UNDER_CURSOR。

MQGMO_COMPLETE_MSG 默示 MQGMO_ALL_SEGMENTS_AVAILABLE，因此不需要指定。

MQGMO_COMPLETE_MSG 可以與 MQGMO_SYNCPOINT_IF_PERSISTENT 以外的任何其他 MQGMO_* 選項一起指定，也可以與 MQMO_MATCH_OFFSET 以外的任何 MQMO_* 選項一起指定。

- 在 z/OS 上，專用及共用佇列支援此選項，但佇列必須具有索引類型 MQIT_GROUP_ID。 對於共用佇列，佇列對映的 CFSTRUCT 物件必須位於 CFLEVEL (3) 或 CFLEVEL (4)。
- 在 AIX、 HP-UX、 IBM i、Solaris、 Linux、Windows ， 以及連接至這些系統的 WebSphere MQ MQI 用戶端上，所有本端佇列都支援此選項。

MQGMO_ALL_MSGS_AVAILABLE
只有在群組中的 所有 訊息都可用時，群組中的訊息才會變成可供擷取。 如果佇列包含遺漏部分訊息的訊息群組 (可能是因為它們在網路中已延遲且尚未到達) ，則指定 MQGMO_ALL_MSGS_AVAILABLE 會阻止擷取屬於不完整群組的訊息。 不過，那些訊息仍會構成 *CurrentQDepth* 佇列屬性的值; 這表示即使 *CurrentQDepth* 大於零，也可能沒有可擷取的訊息群組。 如果沒有其他可擷取的訊息，則在指定的等待間隔 (如果有的話) 過期之後會傳回原因碼 MQRC_NO_MSG_AVAILABLE 。

MQGMO_ALL_MSGS_AVAILABLE 的處理取決於是否同時指定 MQGMO_LOGICAL_ORDER :

- 如果同時指定這兩個選項，則 只有 在沒有現行群組或邏輯訊息時， MQGMO_ALL_MSGS_AVAILABLE 才會生效。 如果 有 現行群組或邏輯訊息，則會忽略 MQGMO_ALL_MSGS_AVAILABLE 。 這表示在以邏輯順序處理訊息時， MQGMO_ALL_MSGS_AVAILABLE 可以保持開啟狀態。
- 如果在沒有 MQGMO_LOGICAL_ORDER 的情況下指定 MQGMO_ALL_MSGS_AVAILABLE ，則 MQGMO_ALL_MSGS_AVAILABLE 一律 會有效果。 這表示在從佇列中移除群組中的第一個訊息之後，必須關閉此選項，才能移除群組中的其餘訊息。

順利完成指定 MQGMO_ALL_MSGS_AVAILABLE 的 MQGET 呼叫，表示在發出 MQGET 呼叫時，群組中的所有訊息都在佇列上。 不過，請注意，其他應用程式仍然可以從群組中移除訊息 (該群組未鎖定至擷取群組中第一個訊息的應用程式)。

如果省略此選項，則即使群組不完整，也可以擷取屬於群組的訊息。

MQGMO_ALL_MSGS_AVAILABLE 默示 MQGMO_ALL_SEGMENTS_AVAILABLE，因此不需要指定。

MQGMO_ALL_MSGS_AVAILABLE 可以與任何其他 MQGMO_* 選項一起指定，也可以與任何 MQMO_* 選項一起指定。

- 在 z/OS 上，專用及共用佇列支援此選項，但佇列必須具有索引類型 MQIT_GROUP_ID。 對於共用佇列，佇列對映的 CFSTRUCT 物件必須位於 CFLEVEL (3) 或 CFLEVEL (4)。
- 在 AIX、 HP-UX、 IBM i、Solaris、 Linux、Windows ， 以及連接至這些系統的 WebSphere MQ MQI 用戶端上，所有本端佇列都支援此選項。

MQGMO_ALL_SEGMENTS_AVAILABLE
只有當邏輯訊息中的 所有 區段都可用時，邏輯訊息中的區段才會變成可供擷取。 如果佇列包含部分區段遺漏的分段訊息 (可能是因為它們在網路中已延遲且尚未到達) ，則指定 MQGMO_ALL_SEGMENTS_AVAILABLE 會阻止擷取屬於不完整邏輯訊息的區段。 不過，那些區段仍會構成 *CurrentQDepth* 佇列屬性的值; 這表示即使 *CurrentQDepth* 大於零，也可能沒有可擷取的邏輯訊息。 如果沒有其他可擷取的訊息，則在指定的等待間隔 (如果有的話) 過期之後會傳回原因碼 MQRC_NO_MSG_AVAILABLE 。

MQGMO_ALL_SEGMENTS_AVAILABLE 的處理取決於是否同時指定 MQGMO_LOGICAL_ORDER :

- 如果同時指定這兩個選項，則 只有 在沒有現行邏輯訊息時， MQGMO_ALL_SEGMENTS_AVAILABLE 才會生效。 如果 有 現行邏輯訊息，則會忽略 MQGMO_ALL_SEGMENTS_AVAILABLE 。 這表示在以邏輯順序處理訊息時， MQGMO_ALL_SEGMENTS_AVAILABLE 可以保持開啟狀態。
- 如果在沒有 MQGMO_LOGICAL_ORDER 的情況下指定 MQGMO_ALL_SEGMENTS_AVAILABLE ，則 MQGMO_ALL_SEGMENTS_AVAILABLE 一律 會有效果。 這表示在從佇列中移除邏輯訊息中的第一個區段之後，必須關閉選項，才能移除邏輯訊息中的其餘區段。

如果未指定此選項，則即使邏輯訊息不完整，也可以擷取訊息區段。

雖然 MQGMO_COMPLETE_MSG 和 MQGMO_ALL_SEGMENTS_AVAILABLE 都需要所有區段都可用，才能擷取其中任何區段，但前者會傳回完整訊息，而後者則容許逐一擷取區段。

如果針對報告訊息指定 MQGMO_ALL_SEGMENTS_AVAILABLE，則佇列管理程式會檢查佇列，以查看組成完整邏輯訊息的每一個區段是否至少有一個報告訊息。 如果有，則滿足 MQGMO_ALL_SEGMENTS_AVAILABLE 條件。 不過，佇列管理程式不會檢查報告訊息的 類型，因此在與邏輯訊息區段相關的報告訊息中可能混合了報告類型。 因此， MQGMO_ALL_SEGMENTS_AVAILABLE 的成功並不表示 MQGMO_COMPLETE_MSG 會成功。 如果針對特定邏輯訊息的區段存在 混合 報告類型，則必須逐一擷取這些報告訊息。

您可以將 MQGMO_ALL_SEGMENTS_AVAILABLE 與任何其他 MQGMO_* 選項以及任何 MQMO_* 選項一起指定。

- 在 z/OS 上，專用及共用佇列支援此選項，但佇列必須具有索引類型 MQIT_GROUP_ID。 對於共用佇列，佇列對映的 CFSTRUCT 物件必須位於 CFLEVEL (3) 或 CFLEVEL (4)。
- 在 AIX、 HP-UX、 IBM i、Solaris、 Linux、Windows，以及連接至這些系統的 WebSphere MQ MQI 用戶端上，所有本端佇列都支援此選項。

內容選項：下列選項與訊息的內容相關：

MQGMO_PROPERTIES_AS_Q_DEF

訊息的內容 (訊息描述子或延伸中的除外) 應該如 *PropertyControl* 佇列屬性所定義來表示。 如果提供 *MsgHandle*，則會忽略此選項，且訊息的內容可透過 *MsgHandle* 取得，除非 *PropertyControl* 佇列屬性的值為 MQPROP_FORCE_MQRFH2。

如果未指定內容選項，這是預設動作。

MQGMO_PROPERTIES_IN_HANDLE

訊息的內容應該透過 *MsgHandle* 提供。 如果未提供訊息控點，則呼叫會失敗，原因為 MQRC_HMSG_ERROR。

註: 如果稍後未建立訊息控點的應用程式讀取訊息，則佇列管理程式會將任何訊息內容放入 MQRFH2 結構中。 您可能會發現存在非預期的 MQRFH2 標頭會破壞現有應用程式的行為。

MQGMO_NO_PROPERTIES

將不會擷取訊息的內容，但訊息描述子 (或延伸) 中包含的內容除外。 如果提供 *MsgHandle*，則會忽略它。

MQGMO_PROPERTIES_FORCE_MQRFH2

訊息的內容 (訊息描述子或延伸中的除外) 應該使用 MQRFH2 標頭來表示。 對於預期擷取內容但無法變更為使用訊息控點的應用程式，這可提供與舊版的相容性。 如果已提供 *MsgHandle*，這個選項會被忽略。

MQGMO_PROPERTIES_COMPATIBILITY
如果訊息包含字首為 **"mcd."**、 **"jms."**、 **"usr."**或 **"mqext."**的內容，則所有訊息內容都會遞送至 MQRFH2 標頭中的應用程式。 否則訊息的所有內容（訊息描述子或延伸中的除外）都會被捨棄，而不再能供應用程式存取。

預設選項: 如果不需要任何說明的選項，則可以使用下列選項:

MQGMO_NONE
使用這個值來指出未指定其他選項；所有選項都採用其預設值。 MQGMO_NONE 輔助工具程式文件; 此選項並非預期與任何其他選項搭配使用，但由於其值為零，因此無法偵測到此類使用。

Options 欄位的起始值為 MQGMO_NO_WAIT 加號 MQGMO_PROPERTIES_AS_Q_DEF。

Reserved1 (MQCHAR)

這是保留欄位。 此欄位的起始值是空白字元。 如果 *Version* 小於 MQGMO_VERSION_2，則會忽略此欄位。

Reserved2 (MQLONG)

這是保留欄位。 此欄位的起始值是空白字元。 如果 *Version* 小於 **MQGMO_VERSION_4**，則會忽略此欄位。

ResolvedQName (MQCHAR48)

這是佇列管理程式設定為從中擷取訊息之佇列本端名稱的輸出欄位，定義為本端佇列管理程式。 在下列情況下，這不同於用來開啟佇列的名稱:

- 已開啟別名佇列 (在此情況下，會傳回別名所解析的本端佇列名稱) ，或
- 已開啟模型佇列 (在此情況下，會傳回動態本端佇列的名稱)。

此欄位的長度由 MQ_Q_NAME_LENGTH 指定。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

ReturnedLength (MQLONG)

這是佇列管理程式設定為 *Buffer* 參數中 MQGET 呼叫所傳回訊息資料長度 (以位元組為單位) 的輸出欄位。 如果佇列管理程式不支援此功能，則 *ReturnedLength* 會設為值 MQRL_UNDEFINED。

在編碼或字集之間轉換訊息時，訊息資料有時會變更大小。 從 MQGET 呼叫傳回時:

- 如果 *ReturnedLength* 是 不 MQRL_UNDEFINED ，則 *ReturnedLength* 會提供傳回的訊息資料位元組數。
- 如果 *ReturnedLength* 具有值 MQRL_UNDEFINED ，則所傳回訊息資料的位元組數通常由較小的 *BufferLength* 及 *DataLength* 提供，但如果 MQGET 呼叫完成，且原因碼為 MQRC_TRUNCATED_MSG_ACCEPTED ，則可以 小於 此值。 如果發生這種情況， *Buffer* 參數中的不顯著位元組會設為空值。

下列是已定義的特殊值:

MQRL_UNDEFINED
未定義傳回資料的長度。

在 z/OS 上，針對 *ReturnedLength* 欄位傳回的值一律為 MQRL_UNDEFINED。

此欄位的起始值是 MQRL_UNDEFINED。 如果 *Version* 小於 MQGMO_VERSION_3，則會忽略此欄位。

分段 *(MQCHAR)*

這是一個旗標，指出擷取的訊息是否容許進一步分段。 它具有下列其中一個值:

MQSEG_INHIBITED
不容許分段。

容許 MQSEG_ALLOWED
容許分段。

在 z/OS 上，佇列管理程式一律會將此欄位設為 MQSEG_INHIBITED。

這是輸出欄位。 此欄位的起始值為 MQSEG_INHIBITED。 如果 *Version* 小於 MQGMO_VERSION_2，則會忽略此欄位。

SegmentStatus (MQCHAR)

這是一個旗標，指出擷取的訊息是否為邏輯訊息的區段。 它具有下列其中一個值:

MQSS_NOT_A_SEGMENT
訊息不是區段。

MQ SS_SEGMENT
訊息是區段，但不是邏輯訊息的最後一個區段。

MQ SS_LAST_SEGMENT
訊息是邏輯訊息的最後一個區段。

如果邏輯訊息只包含一個區段，則也會傳回此值。

在 z/OS 上，佇列管理程式一律會將此欄位設為 MQSS_NOT_A_SEGMENT。

這是輸出欄位。 此欄位的起始值為 MQSS_NOT_A_SEGMENT。 如果 *Version* 小於 MQGMO_VERSION_2，則會忽略此欄位。

Signal1 (MQLONG)

這是僅與 MQGMO_SET_SIGNAL 選項一起使用的輸入欄位; 它識別要在訊息可用時遞送的信號。

註: 此欄位的資料類型及用法由環境決定; 因此，您想要在不同環境之間連接的應用程式不得使用信號。

- 在 z/OS 上，此欄位必須包含「事件控制區塊 (ECB)」的位址。 在發出 MQGET 呼叫之前，應用程式必須先清除 ECB。 在佇列關閉之前，不得釋放包含 ECB 的儲存體。 佇列管理程式會公佈 ECB，並提供其中一個所述的信號完成碼。 這些完成碼以 ECB 的位元 2 到 31 來設定，該區域在 z/OS 對映巨集 IHAECB 中定義為用於使用者完成碼。
- 在所有其他環境中，這是保留欄位; 其值並不重要。

信號完成碼如下:

MQEC_MSG_REALED
佇列上已到達適當的訊息。 此訊息未保留給呼叫程式; 必須發出第二個 MQGET 要求，但另一個應用程式可能在發出第二個要求之前擷取訊息。

MQ ec_WAIT_INTERVAL_EXPIRED
指定的 *WaitInterval* 已過期，沒有適當的訊息送達。

已取消 MQEC_WAIT_CANCELED
由於不確定的原因 (例如佇列管理程式終止或停用佇列)，已取消等待。 如果您想要進一步診斷，請重新發出要求。

MQEC_Q_MGR_QUIESCING
已取消等待，因為佇列管理程式已進入靜止狀態 (MQGMO_FAIL_IF_QUIESCING 已在 MQGET 呼叫上指定)。

MQEC_CONNECTION_QUIESCING
等待已取消，因為連線已進入靜止狀態 (MQGMO_FAIL_IF_QUIESCING 已在 MQGET 呼叫上指定)。

此欄位的起始值由環境決定:

- 在 z/OS 上，起始值是空值指標。
- 在所有其他環境中，起始值為 0。

Signal2 (MQLONG)

這是僅與 MQGMO_SET_SIGNAL 選項一起使用的輸入欄位。 它是保留欄位; 其值不顯著。

此欄位的起始值為 0。

StrucId (MQCHAR4)

這是結構 ID。 值必須為:

MQGMO_STRUC_ID
get-message 選項結構的 ID。

對於 C 程式設計語言，也會定義常數 MQGMO_STRUC_ARRAY; 此值與 MQGMO_STRUC_ID 相同，但卻是字元陣列而非字串。

這一律是輸入欄位。 此欄位的起始值是 MQGMO_STRUC_ID。

版本 *(MQLONG)*
版本是結構版本號碼。

此值必須是下列其中一個:

MQGMO_VERSION_1
Version-1 取得訊息選項結構。

所有環境都支援此版本。

MQGMO_VERSION_2
Version-2 get-message 選項結構。

所有環境都支援此版本。

MQGMO_VERSION_3
Version-3 取得訊息選項結構。

所有環境都支援此版本。

MQGMO_VERSION_4
Version-4 取得訊息選項結構。

所有環境都支援此版本。

僅存在於結構最新版本中的欄位會在欄位說明中如此識別。 下列常數指定現行版本的版本號碼:

MQ GMO_CURRENT_VERSION
取得訊息選項結構的現行版本。

這一律是輸入欄位。 此欄位的起始值為 MQGMO_VERSION_1。

WaitInterval (MQLONG)

這是 MQGET 呼叫等待適當訊息到達的大約時間 (毫秒) (亦即，滿足 MQGET 呼叫 *MsgDesc* 參數中指定的選取準則的訊息)。

.

重要: 如果立即有適當的訊息可用，則沒有等待或延遲。

如需詳細資料，請參閱 第 363 頁的『MQMD-訊息描述子』 中說明的 *MsgId* 欄位。 如果在經歷此時間之後沒有合適的訊息到達，則呼叫會完成，且 MQCC_FAILED 及原因碼 MQRC_NO_MSG_AVAILABLE。

在 z/OS 上， MQGET 呼叫實際等待的時段受到系統載入及工作排程考量的影響，且在指定給 *WaitInterval* 的值與大約大於 *WaitInterval* 的 250 毫秒之間可能有所不同。

WaitInterval 與 MQGMO_WAIT 或 MQGMO_SET_SIGNAL 選項一起使用。 如果都未指定，則會忽略它。 如果指定其中一項，則 *WaitInterval* 必須大於或等於零，或下列特殊值:

MQWI_UNLIMITED
無限制等待間隔。

此欄位的起始值為 0。

MQGMO 的起始值及語言宣告

表 *508: MQGMO* 之 *MQGMO* 中欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQGMO_STRUC_ID	'GMO¬'
Version	MQGMO_VERSION_1	1
Options	MQGMO_NO_WAIT	0
WaitInterval	無	0
Signal1	無	z/OS 上的空值指標; 否則 0
Signal2	無	0
ResolvedQName	無	空字串或空白
MatchOptions	MQMO_MATCH_MSG_ID + MQMO_MATCH_CORREL_ID	3
GroupStatus	MQGS_NOT_IN_GROUP	'¬'
SegmentStatus	MQSS_NOT_A_SEGMENT	'¬'
Segmentation	MQSEG_INHIBITED	'¬'
Reserved1	無	'¬'

表 *508: MQGMO* 之 *MQGMO* 中欄位的起始值 *(*繼續*)*		
欄位名稱	**常數名稱**	**常數值**
MsgToken	MQMTOK_NONE	空值
ReturnedLength	MQRL_UNDEFINED	-1
Reserved2	無	'¬'
MsgHandle	MQHM_NONE	0

附註:

1. 符號 ¬ 代表單一空白字元。
2. 空值字串或空白值表示 C 中的空值字串，以及其他程式設計語言中的空白字元。
3. 在 C 程式設計語言中，巨集變數 MQGMO_DEFAULT 包含上述值。 您可以用下列方式來提供結構中欄位的起始值:

```
MQGMO MyGMO = {MQGMO_DEFAULT};
```

C 宣告

```
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 */
                             retrieved is in a group */
  MQCHAR    SegmentStatus;   /* Flag indicating whether message */
                             retrieved is a segment of a logical */
                             message */
  MQCHAR    Segmentation;    /* Flag indicating whether further */
                             segmentation is allowed for the */
                             message retrieved */
  MQCHAR    Reserved1;       /* Reserved */
  /* Ver:2 */
  MQBYTE16  MsgToken;        /* Message token */
  MQLONG    ReturnedLength;  /* Length of message data returned */
                             (bytes) */
  /* Ver:3 */
  MQLONG    Reserved2;       /* Reserved */
  MQHMSG    MsgHandle;       /* Message handle */
  /* Ver:4 */
};
```

- 在 z/OS 上， *Signal1* 欄位宣告為 PMQLONG。

COBOL 宣告

```
**   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.
```

- 在 z/OS 上， *Signal1* 欄位宣告為 POINTER。

PL/I 宣告

```
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
                                     retrieved is in a group */
  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 */
```

- 在 z/OS 上， *Signal1* 欄位宣告為 pointer。

High Level Assembler 宣告

```
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)
```

Visual Basic 宣告

```
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 MQBYTE16  'Message token'
  ReturnedLength As Long      'Length of message data returned (bytes)'
End Type
```

MQIIH- IMS 資訊標頭

下表彙總結構中的欄位。

表 *509: MQIIH* 中的欄位

欄位	說明	主題
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
StrucLength	MQIIH 結構的長度	StrucLength
Encoding	已保留	編碼
CodedCharSetId	已保留	CodedCharSetId
Format	MQIIH 之後的資料的 MQ 格式名稱	格式
Flags	旗標	旗標
LTermOverride	邏輯終端機置換	LTermOverride
MFSMapName	訊息格式服務對映圖名稱	MFSMapName
ReplyToFormat	回覆訊息的 MQ 格式名稱	ReplyTo 格式
Authenticator	RACF™ 密碼或通行通行證	鑑別程式
TranInstanceId	交易實例 ID	TranInstanceID

表 509: MQIIH 中的欄位 (繼續)		
欄位	**說明**	**主題**
TranState	交易狀態	TranState
CommitMode	確定模式	CommitMode
SecurityScope	安全範圍	SecurityScope
Reserved	已保留	已保留

MQIIH 概觀

可用性: 所有 WebSphere MQ 系統及 WebSphere MQ 用戶端。

目的:MQIIH 結構說明在透過 WebSphere MQ for z/OS 傳送至 IMS 橋接器的訊息開始時必須呈現的資訊。

格式名稱:MQFMT_IMS。

字集及編碼: 特殊條件適用於 MQIIH 結構及應用程式訊息資料所使用的字集及編碼:

- 連接至擁有 IMS 橋接器佇列之佇列管理程式的應用程式，必須提供採用佇列管理程式字集及編碼的 MQIIH 結構。 這是因為在此情況下不會執行 MQIIH 結構的資料轉換。
- 連接至其他佇列管理程式的應用程式可以提供採用任何受支援字集及編碼的 MQIIH 結構; 連接至擁有 IMS 橋接器佇列之佇列管理程式的接收訊息通道代理程式會轉換 MQIIH。
- MQIIH 結構後面的應用程式訊息資料必須使用與 MQIIH 結構相同的字集及編碼。 請勿使用 MQIIH 結構中的 *CodedCharSetId* 及 *Encoding* 欄位來指定應用程式訊息資料的字集及編碼。

  如果資料不是佇列管理程式支援的內建格式之一，您必須提供資料轉換結束程式來轉換應用程式訊息資料。

MQIIH 的欄位

MQIIH 結構包含下列欄位; 這些欄位按 **字母順序**說明:

鑑別器 *(MQCHAR8)*

這是 RACF 密碼或 PassTicket。 它是選用項目; 如果指定的話，它會與 MQMD 安全環境定義中的使用者 ID 搭配使用，以建置 U 記號，並傳送至 IMS 以提供安全環境定義。 如果未指定，則會使用使用者 ID 而不進行驗證。 這取決於 RACF 交換器的設定，這可能需要鑑別器存在。

如果第一個位元組是空白或空值，則會忽略此情況。 可以使用下列特殊值:

無 MQIAUT_NONE
無鑑別。

對於 C 程式設計語言，也會定義常數 MQIAUT_NONE_ARRAY; 此值與 MQIAUT_NONE 值相同，但它是字元陣列而非字串。

此欄位的長度由 MQ_AUTHENTICATOR_LENGTH 提供。 此欄位的起始值為 MQIAUT_NONE。

CodedCharSetId (MQLONG)

這是保留欄位; 其值不顯著。 此欄位的起始值為 0。

遵循 MQIIH 結構之受支援結構的字集 ID 與 MQIIH 結構本身相同，且取自任何之前的 MQ 標頭。

CommitMode (MQCHAR)

這是 IMS 確定模式。 如需 IMS 確定模式的相關資訊，請參閱 *OTMA* 參考手冊 。 此值必須是下列其中一個:

MQICM_COMMIT_THEN_SEND
確定然後傳送。

此模式意味著輸出的雙重佇列作業，但區域佔用時間較短。 捷徑和交談式交易無法以此模式執行。

MQICM_SEND_THEN_COMMIT
傳送然後確定。

由於 MQICM_SEND_THEN_COMMIT 的確定 mpde 而起始的任何 IMS 交易都會以 RESPONSE 模式執行，而不論交易在 IMS 系統定義中的定義方式 (TRANSACT 巨集中的 MSGTYPE 參數)。 這也適用於透過交易切換所起始的交易。

此欄位的起始值為 MQICM_COMMIT_THEN_SEND。

編碼 *(MQLONG)*

這是保留欄位; 其值不顯著。 此欄位的起始值為 0。

遵循 MQIIH 結構之受支援結構的編碼與 MQIIH 結構本身的編碼相同，並取自任何之前的 MQ 標頭。

旗標 *(MQLONG)*

旗標值必須為:

MQIIH_NONE
沒有旗標。

MQIH_PASS_EXPIRATION
回覆訊息包含:

- 與要求訊息相同的期限報告選項
- 要求訊息中未對橋接器處理時間進行調整的剩餘到期時間

如果未設定此值，則到期時間會設為 *unlimited*。

MQIIH_REPLY_FORMAT_NONE
設定 MQIIH.Format 欄位。

MQIIH_IGNORE_PURG
在 OTMA 字首中設定 TMAMIPRG 指示器，以要求 OTMA 忽略 TP PCB 上 CM0 交易的 PURG 呼叫。

MQIIH_CM0_REQUEST_RESPONSE
若為確定模式 0 (CM0) 交易，此旗標會在 OTMA 字首中設定 TMAMHRSP 指示器。 設定此指示器會要求在原始 IMS 應用程式未回覆 IOPCB 或訊息切換至另一個交易時， OTMA/IMS 產生 DFS2082 RESPONSE MODE TRANSACTION TERMINATED WITHOUT REPLY 訊息。

此欄位的起始值為 MQIIH_NONE。

格式 *(MQCHAR8)*

這會指定遵循 MQIIH 結構之資料的 MQ 格式名稱。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。

此欄位的長度由 MQ_FORMAT_LENGTH 提供。 此欄位的起始值為 MQFMT_NONE。

LTermOverride (MQCHAR8)

置於 IO PCB 欄位中的邏輯終端機置換。 它是選用的; 如果未指定，則會使用 TPIPE 名稱。 如果第一個位元組是空白或空值，則會忽略它。

此欄位的長度由 MQ_LTERM_OVERRIDE_LENGTH 指定。 此欄位的起始值為 8 個空白字元。

MFSMapName (MQCHAR8)

訊息格式服務對映名稱，放置在 IO PCB 欄位中。 它是選用項目。 在輸入上，它代表 MID ，在輸出上，它代表 MOD。 如果第一個位元組是空白或空值，則會忽略它。

此欄位的長度由 MQ_MFS_MAP_NAME_LENGTH 提供。 此欄位的起始值為 8 個空白字元。

ReplyTo 格式 *(MQCHAR8)*

這是為了回應現行訊息而傳送之回覆訊息的 MQ 格式名稱。 此欄位的長度由 MQ_FORMAT_LENGTH 提供。 此欄位的起始值為 MQFMT_NONE。

若要使用 MQGMO_CONVERT 轉換回覆訊息中的資料，請指定 MQIIH.replyToFormat= MQFMT_STRING 或 MQIIH.replyToFormat= MQFMT_IMS_VAR_STRING。 如需這些欄位的用法說明，請參閱 第 376 頁的『格式 (MQCHAR8)』。

如果在要求訊息上使用預設值 (MQIIH.replyToFormat= MQFMT_NONE) ，且使用 MQGMO_CONVERT 擷取回覆訊息，則不會執行資料轉換。

保留 (MQCHAR)

這是保留欄位; 它必須是空白。

SecurityScope (MQCHAR)

這指出需要 IMS 安全處理。 已定義下列值:

MQISS_CHECK
檢查安全範圍 :ACEE 建置在控制區域中，但不在相依區域中。

MQISS_FULL
完整安全範圍: 快取的 ACEE 建置在控制區域中，非快取的 ACEE 建置在相依區域中。 如果您使用 MQISS_FULL ，請確保為其建置 ACEE 的使用者 ID 有權存取相依區域中使用的資源。

如果此欄位既未指定 MQISS_CHECK 也未指定 MQISS_FULL ，則會採用 MQISS_CHECK。

此欄位的起始值是 MQISS_CHECK。

StrucId (MQCHAR4)

這是結構 ID。 值必須為:

MQIIH_STRUC_ID
IMS 資訊標頭結構的 ID。

對於 C 程式設計語言，也會定義常數 MQIH_STRUC_ID_ARRAY; 此值與 MQIIH_STRUC_ID 相同，但卻是字元陣列而非字串。

此欄位的起始值是 MQIIH_STRUC_ID。

StrucLength (MQLONG)

這是 MQIIH 結構的長度。 值必須為:

MQIIH_LENGTH_1
IMS 資訊標頭結構的長度。

此欄位的起始值為 MQIIH_LENGTH_1。

TranInstanceID (MQBYTE16)

這是交易實例 ID。 IMS 的輸出訊息會使用此欄位，因此在第一次輸入時會忽略此欄位。 如果您將 *TranState* 設為 MQITS_IN_CONVERSATION ，則必須在下一個輸入及所有後續輸入中提供此項目，才能讓 IMS 將訊息與正確交談產生關聯。 您可以使用下列特殊值:

MQITII_NONE
沒有交易實例 ID。

對於 C 程式設計語言，也會定義常數 MQITII_NONE_ARRAY; 此值與 MQITII_NONE 值相同，但它是字元陣列而非字串。

此欄位的長度由 MQ_TRAN_INSTANCE_ID_LENGTH 提供。 此欄位的起始值為 MQITII_NONE。

TranState (MQCHAR)

這指出 IMS 交談狀態。 因為不存在交談，所以在第一次輸入時忽略此情況。 在後續輸入上，它會指出交談是否為作用中。 輸出時由 IMS 設定。 此值必須是下列其中一個:

MQITS_IN_CONVERSATION
在交談中。

MQITS_NOT_IN_CONVERSATION
不在交談中。

MQITS_ARCHITECTED
以架構形式傳回交易狀態資料。

此值僅與 IMS /DISPLAY TRAN 指令搭配使用。 它會以 IMS 架構格式而非字元格式傳回交易狀態資料。

此欄位的起始值為 MQITS_NOT_IN_CONVERSATION。

版本 *(MQLONG)*

這是結構版本號碼。 值必須為:

MQIIH_VERSION_1
IMS 資訊標頭結構的版本號碼。

下列常數指定現行版本的版本號碼:

MQIIH_CURRENT_VERSION
IMS 資訊標頭結構的現行版本。

此欄位的起始值為 MQIIH_VERSION_1。

MQIIH 的起始值及語言宣告

表 *510: MQIIH* 中 *MQIIH* 的欄位起始值

欄位名稱	常數名稱	常數值
StrucId	MQIIH_STRUC_ID	'IIH?'
Version	MQIIH_VERSION_1	1
StrucLength	MQIIH_LENGTH_1	84
Encoding	無	0
CodedCharSetId	無	0
Format	MQFMT_NONE	空白
Flags	MQIIH_NONE	0
LTermOverride	無	空白
MFSMapName	無	空白
ReplyToFormat	MQFMT_NONE	空白
Authenticator	無 MQIAUT_NONE	空白
TranInstanceId	MQITII_NONE	空值
TranState	MQITS_NOT_IN_CONVERSATION	'?'
CommitMode	MQICM_COMMIT_THEN_SEND	'0'
SecurityScope	MQISS_CHECK	'C'
Reserved	無	'?'

表 *510: MQIIH* 中 *MQIIH* 的欄位起始值 *(*繼續*)*

欄位名稱	常數名稱	常數值

附註:

1. 那個符號**?** 代表單一空白字元。
2. 在 C 程式設計語言中，巨集變數 MQIIH_DEFAULT 包含上述值。 您可以用下列方式來提供結構中欄位的起始值:

```
MQIIH MyIIH = {MQIIH_DEFAULT};
```

C 宣告

```
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 */
};
```

COBOL 宣告

```
**   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.
```

PL/I 宣告

```
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 */
```

High Level Assembler 宣告

```
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_LTERMOVERRIDE   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)
```

Visual Basic 宣告

```
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
```

MQIMPO-查詢訊息內容選項

下表彙總結構中的欄位。 MQIMPO 結構-查詢訊息內容選項

表 *511: MQIMPO* 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
Options	控制 MQINQMP 動作的選項	選項
RequestedEncoding	要將所查詢內容轉換成的編碼	RequestedEncoding
RequestedCCSID	所查詢內容的字集	RequestedCCSID
ReturnedEncoding	回覆值的編碼	ReturnedEncoding
ReturnedCCSID	回覆值的字集	ReturnedCCSID
Reserved1	保留欄位	ReturnedCCSID
ReturnedName	所查詢內容的名稱	ReturnedName
TypeString	內容資料類型的字串表示法	TypeString

MQIMPO 概觀

查詢訊息內容選項結構。

可用性: 所有 WebSphere MQ 系統及 WebSphere MQ 用戶端。

目的:MQIMPO 結構可讓應用程式指定選項來控制如何查詢訊息內容。 結構是 MQINQMP 呼叫上的輸入參數。

字集及編碼:MQIMPO 中的資料必須是應用程式的字集及應用程式的編碼 (MQENC_NATIVE)。

MQIMPO 的欄位

查詢訊息內容選項結構-欄位

MQIMPO 結構包含下列欄位; 這些欄位按 **字母順序**說明:

選項 *(MQLONG)*
查詢訊息內容選項結構-選項欄位

下列選項控制 MQINQMP 的動作。 您可以指定下列一或多個選項，如果您需要多個選項，則值可以是:

- 加在一起 (不要多次新增相同的常數) ，或
- 利用執行位元 OR 運算來組合（如果程式設計語言支援位元運算）。

會記下無效的選項組合; 所有其他組合都是有效的。

值資料選項: 從訊息擷取內容時，下列選項與值資料的處理相關。

MQIMPO_CONVERT_VALUE
此選項要求轉換內容的值，以符合 MQINQMP 呼叫在 *Value* 區域中傳回內容值之前指定的 *RequestedCCSID* 及 *RequestedEncoding* 值。

- 如果轉換成功，則在從 MQINQMP 呼叫傳回時， *ReturnedCCSID* 和 *ReturnedEncoding* 欄位會設為與 *RequestedCCSID* 和 *RequestedEncoding* 相同。
- 如果轉換失敗，但 MQINQMP 呼叫已完成且沒有錯誤，則會傳回未轉換的內容值。

  如果內容是字串，則 *ReturnedCCSID* 及 *ReturnedEncoding* 欄位會設為未轉換字串的字集及編碼。

在此情況下，完成碼為 MQCC_WARNING，原因碼為 MQRC_PROP_VALUE_NOT_CONVERTED。 內容游標已進階至傳回的內容。

如果內容值在轉換期間展開，且超出 *Value* 參數的大小，則會傳回未轉換的值，完成碼為 MQCC_FAILED; 原因碼設為 MQRC_PROPERTY_VALUE_TOO_BIG。

MQINQMP 呼叫的 *DataLength* 參數會傳回內容值已轉換成的長度，以便讓應用程式決定容納已轉換內容值所需的緩衝區大小。 內容游標未變更。

這個選項也會要求:

- 如果內容名稱包含萬用字元，且
- *ReturnedName* 欄位會以所傳回名稱的位址或偏移來起始設定，

則會轉換傳回的名稱，以符合 *RequestedCCSID* 及 *RequestedEncoding* 值。

- 如果轉換成功，則 *ReturnedName* 的 *VSCCSID* 欄位及所傳回名稱的編碼會設為輸入值 *RequestedCCSID* 及 *RequestedEncoding* 。
- 如果轉換失敗，但 MQINQMP 呼叫完成而沒有錯誤或警告，則會取消轉換傳回的名稱。 在此情況下，完成碼為 MQCC_WARNING，原因碼為 MQRC_PROP_NAME_NOT_CONVERTED。

  內容游標已進階至傳回的內容。 如果未同時轉換值和名稱，則會傳回 MQRC_PROP_VALUE_NOT_CONVERTED。

如果傳回的名稱在轉換期間展開，且超出 *RequestedName* 的 *VSBufsize* 欄位大小，則會保留未轉換的傳回字串，完成碼為 MQCC_FAILED，且原因碼設為 MQRC_PROPERTY_NAME_TOO_BIG。

MQCHARV 結構的 *VSLength* 欄位會傳回內容值已轉換成的長度，以便讓應用程式決定容納已轉換內容值所需的緩衝區大小。 內容游標未變更。

MQIMPO_CONVERT_TYPE
此選項要求將內容的值從其現行資料類型轉換為 MQINQMP 呼叫的 *Type* 參數上指定的資料類型。

- 如果轉換成功，則在傳回 MQINQMP 呼叫時， *Type* 參數保持不變。
- 如果轉換失敗，但 MQINQMP 呼叫完成而未發生錯誤，則呼叫會失敗，原因為 MQRC_PROP_CONV_NOT_SUPPORTED。 內容游標未變更。

如果資料類型的轉換導致在轉換期間擴充值，且已轉換的值超出 *Value* 參數的大小，則會傳回未轉換的值，完成碼為 MQCC_FAILED 且原因碼設為 MQRC_PROPERTY_VALUE_TOO_BIG。

MQINQMP 呼叫的 *DataLength* 參數會傳回內容值已轉換成的長度，以便讓應用程式決定容納已轉換內容值所需的緩衝區大小。 內容游標未變更。

如果 MQINQMP 呼叫的 *Type* 參數值無效，則呼叫會失敗，原因為 MQRC_PROPERTY_TYPE_ERROR。

如果不支援所要求的資料類型轉換，則呼叫會失敗，原因為 MQRC_PROP_CONV_NOT_SUPPORTED。 支援下列資料類型轉換:

內容資料類型	支援的目標資料類型
MQ 類型 _ 布林	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

內容資料類型	支援的目標資料類型
MQTYPE_STRING	MQTYPE_Boolean、 MQTYPE_INT8、 MQTYPE_INT16、 MQTYPE_INT32、 MQTYPE_INT64、 MQTYPE_FLOAT32、 MQTYPE_FLOAT64
MQTYPE_NULL	無

控管支援轉換的一般規則如下:

- 數值內容值可以從一種資料類型轉換為另一種資料類型，前提是在轉換期間不會遺失任何資料。

  例如，資料類型為 MQTYPE_INT32 的內容值可以轉換為資料類型為 MQTYPE_INT64 的值，但無法轉換為資料類型為 MQTYPE_INT16 的值。
- 任何資料類型的內容值都可以轉換成字串。
- 字串內容值可以轉換為任何其他資料類型，前提是字串已正確格式化以進行轉換。 如果應用程式嘗試轉換未正確格式化的字串內容值， WebSphere MQ 會傳回原因碼 MQRC_PROP_NUMBER_FORMAT_ERROR。
- 如果應用程式嘗試不支援的轉換， WebSphere MQ 會傳回原因碼 MQRC_PROP_CONV_NOT_SUPPORTED。

將內容值從一個資料類型轉換成另一個資料類型的特定規則如下:

- 將 MQTYPE_BOOLEAN 內容值轉換為字串時，會將值 TRUE 轉換為字串 "TRUE" ，並將值 false 轉換為字串 "FALSE"。
- 將 MQTYPE_BOOLEAN 內容值轉換為數值資料類型時，值 TRUE 會轉換為 1 ，而值 FALSE 會轉換為零。
- 將字串內容值轉換為 MQTYPE_BOOLEAN 值時，會將字串 "TRUE" 或 "1" 轉換為 TRUE ，並將字串 "FALSE" 或 "0" 轉換為 FALSE。

  請注意，術語 "TRUE" 和 "FALSE" 不區分大小寫。

  無法轉換任何其他字串; WebSphere MQ 會傳回原因碼 MQRC_PROP_NUMBER_FORMAT_ERROR。
- 將字串內容值轉換為資料類型為 MQTYPE_INT8、 MQTYPE_INT16、 MQTYPE_INT32 或 MQTYPE_INT64 的值時，字串必須具有下列格式:

  ```
  [blanks][sign]digits
  ```

  字串元件的意義如下:

  blanks
  選用的前導空白字元

  sign
  選用加號 (+) 或減號 (-) 字元。

  digits
  一連串連續的數字字元 (0-9)。 至少必須呈現一個數字字元。

  在一連串數字字元之後，字串可以包含非數字字元的其他字元，但一旦達到這些字元的第一個字元，轉換就會停止。 假設字串代表十進位整數。

  WebSphere MQ 如果字串未正確格式化，則會傳回原因碼 MQRC_PROP_NUMBER_FORMAT_ERROR。
- 將字串內容值轉換為資料類型為 MQTYPE_FLOAT32 或 MQTYPE_FLOAT64 的值時，字串必須具有下列格式:

  ```
      [blanks][sign]digits[.digits][e_char[e_sign]e_digits]
  ```

  字串元件的意義如下:

  blanks
  選用的前導空白字元

sign
選用加號 (+) 或減號 (-) 字元。

digits
一連串連續的數字字元 (0-9)。 至少必須呈現一個數字字元。

e_char
指數字元，可以是 "E" 或 "e"。

e_sign
指數的選用加號 (+) 或減號 (-) 字元。

e_digits
指數的連續數字字元 (0-9)。 如果字串包含指數字元，則至少必須存在一個數字字元。

在一連串數字字元或代表指數的選用字元之後，字串可以包含不是數字字元的其他字元，但一旦達到這些字元的第一個字元，就會停止轉換。 假設字串代表具有 10 次方的指數的十進位浮點數字。

WebSphere MQ 如果字串未正確格式化，則會傳回原因碼 MQRC_PROP_NUMBER_FORMAT_ERROR。

- 將數值內容值轉換為字串時，該值會轉換為以十進位數表示的值字串，而不是包含該值的 ASCII 字元的字串。 例如，整數 65 會轉換為字串 "65"，而不是字串 "A"。
- 將位元組字串內容值轉換為字串時，每個位元組都會轉換為代表該位元組的兩個十六進位字元。 例如，位元組陣列 {0xF1, 0x12, 0x00, 0xFF} 會轉換為字串 "F11200FF"。

MQIMPO_QUERY_LENGTH
查詢內容值的類型和長度。 在 MQINQMP 呼叫的 *DataLength* 參數中傳回長度。 未傳回內容值。

如果指定 *ReturnedName* 緩衝區，則 MQCHARV 結構的 *VSLength* 欄位會填入內容名稱的長度。 未傳回內容名稱。

反覆運算選項: 下列選項與反覆運算內容相關，使用具有萬用字元的名稱

MQIMPO_INQ_FIRST
查詢第一個符合指定名稱的內容。 在此呼叫之後，會在傳回的內容上建立游標。

這是預設值。

MQIMPO_INQ_PROP_UNDER_CURSOR 選項隨後可以與 MQINQMP 呼叫搭配使用，以重新查詢相同的內容 (必要的話)。

請注意，只有一個內容游標; 因此，如果 MQINQMP 呼叫中指定的內容名稱變更游標。

此選項不適用於下列任一選項:

MQIMPO_INQ_NEXT
MQIMPO_INQ_PROP_UNDER_CURSOR

MQIMPO_INQ_NEXT
查詢下一個符合指定名稱的內容，繼續從內容游標搜尋。 游標會進階至所傳回的內容。

如果這是指定名稱的第一個 MQINQMP 呼叫，則會傳回第一個符合指定名稱的內容。

MQIMPO_INQ_PROP_UNDER_CURSOR 選項隨後可以與 MQINQMP 呼叫搭配使用，以重新查詢相同的內容。

如果已刪除游標下的內容， MQINQMP 會傳回已刪除內容之後的下一個相符內容。

如果新增的內容符合萬用字元，當疊代正在進行時，在疊代完成期間不一定會傳回該內容。 一旦使用 MQIMPO_INQ_FIRST 重新啟動反覆運算，即會傳回此內容。

在進行疊代時，如果內容符合已刪除的萬用字元，則在刪除之後不會傳回該內容。

此選項不適用於下列任一選項:

MQIMPO_INQ_FIRST
MQIMPO_INQ_PROP_UNDER_CURSOR

MQIMPO_INQ_PROP_UNDER_CURSOR
擷取內容游標所指向的內容值。 內容 cursor 所指向的內容是前次使用 MQIMPO_INQ_FIRST 或 MQIMPO_INQ_NEXT 選項所查詢的內容。

當重複使用訊息控點、在 MQGET 呼叫上 MQGMO 的 *MsgHandle* 欄位中指定訊息控點，或在 MQPUT 呼叫上 MQPMO 結構的 *OriginalMsgHandle* 或 *NewMsgHandle* 欄位中指定訊息控點時，會重設內容游標。

如果在尚未建立內容游標時使用此選項，或已刪除內容游標所指向的內容，則呼叫會失敗，完成碼為 MQCC_FAILED 且原因為 MQRC_PROPERTY_NOT_AVAILABLE。

此選項不適用於下列任一選項:

MQIMPO_INQ_FIRST
MQIMPO_INQ_NEXT

如果不需要上述任何選項，則可以使用下列選項:

MQIMPO_NONE
使用這個值來指出未指定其他選項；所有選項都採用其預設值。

MQIMPO_NONE 會輔助程式說明文件; 此選項不會與任何其他選項一起使用，但由於其值為零，因此無法偵測到這類使用。

這一律是輸入欄位。 此欄位的起始值為 MQIMPO_INQ_FIRST。

RequestedCCSID (MQLONG)
查詢訊息內容選項結構- RequestedCCSID 欄位

如果所查詢的內容值是字串，則要將其轉換成的字集。 這也是在指定 MQIMPO_CONVERT_VALUE 或 MQIMPO_CONVERT_TYPE 時，要轉換 *ReturnedName* 的字集。

此欄位的起始值為 MQCCSI_APPL。

RequestedEncoding (MQLONG)
查詢訊息內容選項結構- RequestedEncoding 欄位

這是在指定 MQIMPO_CONVERT_VALUE 或 MQIMPO_CONVERT_TYPE 時，所查詢的內容值要轉換成的編碼。

此欄位的起始值為 MQENC_NATIVE。

Reserved1 (MQCHAR)

這是保留欄位。 此欄位的起始值是空白字元 (4 位元組欄位)。

ReturnedCCSID (MQLONG)
查詢訊息內容選項結構- ReturnedCCSID 欄位

在輸出上，如果 MQINQMP 呼叫的 *Type* 參數是 MQTYPE_STRING ，這是所傳回值的字集。

如果指定 MQIMPO_CONVERT_VALUE 選項且轉換成功，則傳回時， *ReturnedCCSID* 欄位與傳入的值相同。

此欄位的起始值為零。

ReturnedEncoding (MQLONG)
查詢訊息內容選項結構- ReturnedEncoding 欄位

在輸出上，這是所傳回值的編碼。

如果指定 MQIMPO_CONVERT_VALUE 選項且轉換成功，則傳回時， *ReturnedEncoding* 欄位與傳入的值相同。

此欄位的起始值為 MQENC_NATIVE。

ReturnedName (MQCHARV)
查詢訊息內容選項結構- ReturnedName 欄位

所查詢內容的實際名稱。

在輸入時，可以使用 MQCHARV 結構的 *VSPtr* 或 *VSOffset* 欄位來傳入字串緩衝區。 字串緩衝區的長度是使用 MQCHARV 結構的 *VSBufsize* 欄位來指定。

從 MQINQMP 呼叫返回時，字串緩衝區會以所查詢內容的名稱來完成，但前提是字串緩衝區的長度足以完整包含名稱。 MQCHARV 結構的 *VSLength* 欄位會填入內容名稱的長度。 會填寫 MQCHARV 結構的 *VSCCSID* 欄位，以指出所傳回名稱的字集，是否轉換名稱失敗。

這是輸入/輸出欄位。 此欄位的起始值為 MQCHARV_DEFAULT。

StrucId (MQCHAR4)
查詢訊息內容選項結構- StrucId 欄位

這是結構 ID。 值必須為:

MQIMPO_STRUC_ID
查詢訊息內容選項結構的 ID。

對於 C 程式設計語言，也會定義常數 MQIMPO_STRUC_ID_ARRAY; 此值與 MQIMPO_STRUC_ID 相同，但是字元陣列而非字串。

這一律是輸入欄位。 此欄位的起始值是 MQIMPO_STRUC_ID。

TypeString (MQCHAR8)
查詢訊息內容選項結構- TypeString 欄位

內容資料類型的字串表示法。

如果在 MQRFH2 標頭中指定了內容，且無法辨識 MQRFH2 dt 屬性，則可以使用此欄位來決定內容的資料類型。 *TypeString* 以編碼字集 1208 (UTF-8) 傳回，並且是無法辨識內容之 dt 屬性值的前八個位元組

這一律是輸出欄位。 此欄位的起始值是 C 程式設計語言中的空字串，以及其他程式設計語言中的 8 個空白字元。

版本 *(MQLONG)*
查詢訊息內容選項結構-版本欄位

這是結構版本號碼。 值必須為:

MQIMPO_VERSION_1
查詢訊息內容選項結構的版本號碼。

下列常數指定現行版本的版本號碼:

MQ IMPO_CURRENT_VERSION
查詢訊息內容選項結構的現行版本。

這一律是輸入欄位。 此欄位的起始值為 MQIMPO_VERSION_1。

MQIMPO 的起始值及語言宣告
查詢訊息內容選項結構-起始值

表 *512: MQIPMO* 中欄位的起始值		
欄位名稱	**常數名稱**	**常數值**
StrucId	MQIMPO_STRUC_ID	'IMPO'
Version	MQIMPO_VERSION_1	1
Options	MQIMPO_INQ_FIRST	
RequestedEncoding	MQENC_NATIVE	
RequestedCCSID	MQCCSI_APPL	
ReturnedEncoding	MQENC_NATIVE	
ReturnedCCSID	0	
Reserved1	0	
ReturnedName	MQCHARV_DEFAULT	
TypeString	空字串或空白	

附註:

1. 空值字串或空白值表示 C 中的空值字串，以及其他程式設計語言中的空白字元。
2. 在 C 程式設計語言中，巨集變數 MQIMPO_DEFAULT 包含上述值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQIMPO MyIMPO = {MQIMPO_DEFAULT};
```

C 宣告
查詢訊息內容選項結構-C 語言宣告

```
typedef struct tagMQIMPO MQIMPO;
struct tagMQIMPO {
  MQCHAR4  StrucId;            /* 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 */
};
```

COBOL 宣告
查詢訊息內容選項結構-COBOL 語言宣告

```
**   MQIMPO structure
  10 MQIMPO.
**    Structure identifier
   15 MQIMPO-STRUCID               PIC X(4).
**    Structure version number
   15 MQIMPO-VERSION               PIC S9(9) BINARY.
**    Options that control the action of MQINQMP
   15 MQIMPO-OPTIONS               PIC S9(9) BINARY.
**    Requested encoding of VALUE
   15 MQIMPO-REQUESTEDENCODING     PIC S9(9) BINARY.
**    Requested character set identifier of VALUE
   15 MQIMPO-REQUESTEDCCSID        PIC S9(9) BINARY.
**    Returned encoding of VALUE
   15 MQIMPO-RETURNEDENCODING      PIC S9(9) BINARY.
```

```
**    Returned character set identifier of VALUE
   15 MQIMPO-RETURNEDCCSID          PIC S9(9) BINARY.
**    Reserved field
   15 MQIMPO-RESERVED1
**    Returned property name
   15 MQIMPO-RETURNEDNAME.
**     Address of variable length string
    20 MQIMPO-RETURNEDNAME-VSPTR    POINTER.
**     Offset of variable length string
    20 MQIMPO-RETURNEDNAME-VSOFFSET PIC S9(9) BINARY.
**     CCSID of variable length string
    20 MQIMPO-RETURNEDNAME-VSCCSID  PIC S9(9) BINARY.
**    Property data type as string
   15 MQIMPO-TYPESTRING             PIC S9(9) BINARY.
```

PL/I 宣告
查詢訊息內容選項結構-PL/I 語言宣告

```
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 */
```

High Level Assembler 宣告
查詢訊息內容選項結構-組譯器語言宣告

```
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)
```

MQMD-訊息描述子

下表彙總結構中的欄位。

表 513: MQMD 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
Report	報告訊息的選項	報告
MsgType	訊息類型	MsgType
Expiry	訊息期限	MQMD-期限欄位
Feedback	回饋碼或原因碼	MQMD-意見欄位
Encoding	訊息資料的數字編碼	編碼
CodedCharSetId	訊息資料的字集 ID	CodedCharSetId
Format	訊息資料的格式名稱	格式
Priority	訊息優先順序	優先順序
Persistence	訊息持續性	持續性
MsgId	訊息 ID	MQMD- MsgId 欄位
CorrelId	相關性 ID	CorrelId
BackoutCount	取消計數器	BackoutCount
ReplyToQ	回覆佇列的名稱	ReplyToQ
ReplyToQMgr	回覆佇列管理程式的名稱	回覆目的地佇列管理程式
UserIdentifier	使用者 ID	UserIdentifier
AccountingToken	帳戶記號	AccountingToken
ApplIdentityData	與身分相關的應用程式資料	ApplIdentityData
PutApplType	放置訊息的應用程式類型	PutApplType
PutApplName	放置訊息的應用程式名稱	PutApplName
PutDate	放置訊息的日期	PutDate
PutTime	放置訊息的時間	PutTime
ApplOriginData	與出處相關的應用程式資料	ApplOriginData
註: 如果 *Version* 小於 MQMD_VERSION_2，則會忽略其餘欄位。		
GroupId	群組 ID	GroupId
MsgSeqNumber	群組內的邏輯訊息的序號	MsgSeqNumber
Offset	從邏輯訊息開始的實體訊息中的資料偏移	偏移
MsgFlags	訊息旗標	MQMD- MsgFlags 欄位
OriginalLength	原始訊息的長度	OriginalLength

MQMD 概觀

可用性: 所有 WebSphere MQ 系統，以及連接至這些系統的 WebSphere MQ MQI 用戶端。

目的:MQMD 結構包含在傳送與接收應用程式之間傳送訊息時，應用程式資料所隨附的控制資訊。 此結構是 MQGET、MQPUT 及 MQPUT1 呼叫的輸入/輸出參數。

版本:MQMD 的現行版本是 MQMD_VERSION_2。 預期在數個環境之間可攜的應用程式必須確保所有相關環境都支援所需版本的 MQMD。 僅存在於結構最新版本中的欄位在接下來的說明中如此識別。

為支援的程式設計語言提供的標頭、COPY 和 INCLUDE 檔案包含環境支援的 MQMD 最新版本，但 *Version* 欄位的起始值設為 MQMD_VERSION_1。 若要使用不在 version-1 結構中的欄位，應用程式必須將 *Version* 欄位設為所需版本的版本號碼。

名稱為 MQMD1 的 version-1 結構宣告可用。

字集及編碼:MQMD 中的資料必須是本端佇列管理程式的字集及編碼; 這些是由 *CodedCharSetId* 佇列管理程式屬性及 MQENC_NATIVE 所提供。 不過，如果應用程式以 WebSphere MQ MQI 用戶端身分執行，則結構必須是在用戶端的字集及編碼中。

如果傳送及接收佇列管理程式使用不同的字集或編碼，則會自動轉換 MQMD 中的資料。 應用程式不需要轉換 MQMD。

使用不同版本的 MQMD: version-2 MQMD 等同於使用 version-1 MQMD ，並以 MQMDE 結構作為訊息資料的字首。 不過，如果 MQMDE 結構中的所有欄位都有其預設值，則可以省略 MQMDE。 使用 version-1 MQMD 加上 MQMDE ，如下所述:

- 在 MQPUT 及 MQPUT1 呼叫上，如果應用程式提供 version-1 MQMD ，則應用程式可以選擇性地以 MQMDE 作為訊息資料的字首，並將 MQMD 中的 *Format* 欄位設為 MQFMT_MD_EXTENSION ，以指出 MQMDE 存在。 如果應用程式未提供 MQMDE ，則佇列管理程式會採用 MQMDE 中欄位的預設值。

  註: 存在於 version-2 MQMD 中但不存在於 version-1 MQMD 中的數個欄位是 MQPUT 及 MQPUT1 呼叫上的輸入/輸出欄位。 不過，在 MQPUT 及 MQPUT1 呼叫的輸出上，佇列管理程式 不會 在 MQMDE 的對等欄位中傳回任何值; 如果應用程式需要這些輸出值，則必須使用 version-2 MQMD。
- 在 MQGET 呼叫上，如果應用程式提供 version-1 MQMD ，則佇列管理程式會以 MQMDE 作為傳回訊息的字首，但只有在 MQMDE 中有一或多個欄位具有非預設值時。 MQMD 中的 *Format* 欄位將具有值 MQFMT_MD_EXTENSION ，以指出 MQMDE 存在。

佇列管理程式用於 MQMDE 中欄位的預設值與那些欄位的起始值相同，如 第 409 頁的表 518 中所示。

當訊息位於傳輸佇列時， MQMD 中的部分欄位會設為特定值; 如需詳細資料，請參閱 第 539 頁的『MQXQH-傳輸佇列標頭』 。

訊息環境定義:MQMD 中的某些欄位包含訊息環境定義。 訊息環境定義有兩種類型: 身分環境定義 和 原始環境定義。 通常:

- 身分環境定義與 原始 放置訊息的應用程式相關
- 原始環境定義與 最近 放置訊息的應用程式相關。

這兩個應用程式可以是相同的應用程式，但也可以是不同的應用程式 (例如，當訊息從一個應用程式轉遞至另一個應用程式時)。

雖然身分及原始環境定義通常具有說明的意義，但 MQMD 中兩種環境定義欄位類型的內容取決於放置訊息時指定的 MQPMO_ * _CONTEXT 選項。 因此，身分環境定義不一定與最初放置訊息的應用程式相關，而原始環境定義不一定與最近放置訊息的應用程式相關; 它取決於應用程式套組的設計。

訊息通道代理程式 (MCA) 永不變更訊息環境定義。 從遠端佇列管理程式接收訊息的 MCA 在 MQPUT 或 MQPUT1 呼叫上使用環境定義選項 MQPMO_SET_ALL_CONTEXT。 這可讓接收 MCA 確切保留與來自傳送 MCA 的訊息一起移動的訊息環境定義。 不過，結果是原始環境定義與傳送及接收訊息的 MCA 都不相關。 原始環境定義參照放置訊息的較早應用程式。 如果所有中間應用程式都已傳遞訊息環境定義，則原始環境定義會參照原始應用程式本身。

在說明中，環境定義欄位的說明如同先前所說明的一樣。 如需訊息環境定義的相關資訊，請參閱 訊息環境定義。

MQMD 的欄位

MQMD 結構包含下列欄位; 這些欄位按 **字母順序**進行說明:

表 514: MQMD 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
Report	報告訊息的選項	報告
MsgType	訊息類型	MsgType
Expiry	訊息期限	MQMD-期限欄位
Feedback	回饋碼或原因碼	MQMD-意見欄位
Encoding	訊息資料的數字編碼	編碼
CodedCharSetId	訊息資料的字集 ID	CodedCharSetId
Format	訊息資料的格式名稱	格式
Priority	訊息優先順序	優先順序
Persistence	訊息持續性	持續性
MsgId	訊息 ID	MQMD- MsgId 欄位
CorrelId	相關性 ID	CorrelId
BackoutCount	取消計數器	BackoutCount
ReplyToQ	回覆佇列的名稱	ReplyToQ
ReplyToQMgr	回覆佇列管理程式的名稱	回覆目的地佇列管理程式
UserIdentifier	使用者 ID	UserIdentifier
AccountingToken	帳戶記號	AccountingToken
ApplIdentityData	與身分相關的應用程式資料	ApplIdentityData
PutApplType	放置訊息的應用程式類型	PutApplType
PutApplName	放置訊息的應用程式名稱	PutApplName
PutDate	放置訊息的日期	PutDate
PutTime	放置訊息的時間	PutTime
ApplOriginData	與出處相關的應用程式資料	ApplOriginData
註: 如果 *Version* 小於 MQMD_VERSION_2，則會忽略其餘欄位。		
GroupId	群組 ID	GroupId
MsgSeqNumber	群組內的邏輯訊息的序號	MsgSeqNumber
Offset	從邏輯訊息開始的實體訊息中的資料偏移	偏移
MsgFlags	訊息旗標	MQMD- MsgFlags 欄位
OriginalLength	原始訊息的長度	OriginalLength

AccountingToken (MQBYTE32)

這是帳戶記號，這是訊息 **身分環境定義** 的一部分。 如需訊息環境定義的相關資訊，請參閱 第 364 頁的『MQMD 概觀』；另請參閱 訊息環境定義。

AccountingToken 可讓應用程式對因訊息而完成的工作適當地收費。 佇列管理程式會將此資訊視為位元字串，且不會檢查其內容。

佇列管理程式會產生此資訊，如下所示:

- 欄位的第一個位元組設為後面位元組中呈現的帳戶資訊長度; 此長度介於 0 到 30 之間，並以二進位整數儲存在第一個位元組中。
- 第二個及後續的位元組 (由長度欄位指定) 會設為適用於環境的帳戶資訊。
  - 在 z/OS 上，帳戶資訊設為:
    - 對於 z/OS 批次，來自 JES JOB 卡或 EXEC 卡中 JES ACCT 陳述式的帳戶資訊 (逗點分隔字元會變更為 X'FF ')。 必要的話，此資訊會截斷為 31 個位元組。
    - 若為 TSO ，使用者的帳號。
    - 若為 CICS，則為 LU 6.2 工作單元 ID (UEPUOWDS) (26 位元組)。
    - 對於 IMS，這是與 16 個字元 IMS 回復記號連結的 8 個字元 PSB 名稱。
  - 在 IBM i 上，帳戶資訊會設為工作的帳戶碼。
  - 在 UNIX 系統上，帳戶資訊設為 ASCII 字元的數值使用者 ID。
  - 在 Windows 上，帳戶資訊會以壓縮格式設為 Windows 安全 ID (SID)。 SID 可唯一識別儲存在 *UserIdentifier* 欄位中的使用者 ID。 當 SID 儲存在 *AccountingToken* 欄位時，會省略 6 個位元組的「ID 權限」(位於 SID 的第三個及後續的位元組)。 例如，如果 Windows SID 長度為 28 個位元組， *AccountingToken* 欄位中會儲存 22 個位元組的 SID 資訊。
- 統計欄位的最後一個位元組 (位元組 32) 設為統計記號類型 (在此情況下為 MQACTT_NT_SECURITY_ID , x '0b'):

  MQACTT_CICS_LUOW_ID
  CICS LUOW ID。

  MQACTT_NT_SECURITY_ID
  Windows 安全 ID。

  MQACTT_OS400_ACCOUNT_TOKEN
  IBM i 帳戶記號。

  MQACTT_UNIX_NUMERIC_ID
  UNIX 系統數值 ID。

  MQACTT_USER
  使用者定義帳戶記號。

  MQACTT_UNKNOWN
  不明 accounting-token 類型。

  帳戶記號類型僅在下列環境中設為明確值: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows，以及連接至這些系統的 WebSphere MQ MQI 用戶端。 在其他環境中， accounting-token 類型會設為值 MQACTT_UNKNOWN。 在這些環境中，請使用 *PutApplType* 欄位來推斷所接收帳戶記號的類型。
- 所有其他位元組都設為二進位零。

對於 MQPUT 及 MQPUT1 呼叫，如果在 *PutMsgOpts* 參數中指定 MQPMO_SET_IDENTITY_CONTEXT 或 MQPMO_SET_ALL_CONTEXT ，則這是輸入/輸出欄位。 如果未指定 MQPMO_SET_IDENTITY_CONTEXT 或 MQPMO_SET_ALL_CONTEXT ，則輸入時會忽略此欄位，且此欄位是僅限輸出欄位。 如需訊息環境定義的相關資訊，請參閱 訊息環境定義 。

順利完成 MQPUT 或 MQPUT1 呼叫之後，此欄位包含在將訊息放入佇列時隨訊息一起傳輸的 *AccountingToken* 。 這將是隨訊息保留的 *AccountingToken* 值 (如需保留的發佈資訊的詳細資料，請參閱第 437 頁的『MQPMO 選項 (MQLONG)』 中 MQPMO_RETAIN 的說明) ，但當訊息以發佈資訊形式傳送給訂閱者時，不會用作 *AccountingToken* ，因為它們提供值來置換傳送給訂閱者的所有發佈資訊中的 *AccountingToken* 。 如果訊息沒有環境定義，則欄位完全是二進位零。

這是 MQGET 呼叫的輸出欄位。

此欄位不會根據佇列管理程式的字集進行任何轉換; 該欄位會被視為位元字串，而不是字元字串。

佇列管理程式不會對這個欄位中的資訊執行任何動作。 如果應用程式想要將資訊用於會計用途，則必須解譯資訊。

您可以對 *AccountingToken* 欄位使用下列特殊值:

MQACT_NONE
未指定帳戶記號。

欄位長度的值為二進位零。

對於 C 程式設計語言，也會定義常數 MQACT_NONE_ARRAY; 此值與 MQACT_NONE 相同，但它是字元陣列而非字串。

此欄位的長度由 MQ_ACCOUNTING_TOKEN_LENGTH 提供。 此欄位的起始值是 MQACT_NONE。

ApplIdentity 資料 *(MQCHAR32)*

這是訊息 **身分環境定義** 的一部分。 如需訊息環境定義的相關資訊，請參閱 第 364 頁的『MQMD 概觀』 和 訊息環境定義 。

ApplIdentityData 是應用程式套組所定義的資訊，可用來提供訊息或其發送端的其他相關資訊。 佇列管理程式會將此資訊視為字元資料，但不會定義其格式。 當佇列管理程式產生此資訊時，它會完全空白。

對於 MQPUT 及 MQPUT1 呼叫，如果在 *PutMsgOpts* 參數中指定 MQPMO_SET_IDENTITY_CONTEXT 或 MQPMO_SET_ALL_CONTEXT，則這是輸入/輸出欄位。 如果存在空值字元，佇列管理程式會將空值及任何後續字元轉換為空白。 如果未指定 MQPMO_SET_IDENTITY_CONTEXT 或 MQPMO_SET_ALL_CONTEXT，則輸入時會忽略此欄位，且此欄位是僅限輸出欄位。 如需訊息環境定義的相關資訊，請參閱 訊息環境定義。

順利完成 MQPUT 或 MQPUT1 呼叫之後，此欄位包含在將訊息放入佇列時隨訊息一起傳輸的 *ApplIdentityData* 。 這將是隨訊息保留的 *ApplIdentityData* 值 (如需保留的發佈資訊的詳細資料，請參閱 MQPMO_RETAIN 的說明)，但在將訊息當作發佈資訊傳送給訂閱者時不會用作 *ApplIdentityData* ，因為它們提供值來置換傳送給訂閱者的所有發佈資訊中的 *ApplIdentityData* 。 如果訊息沒有環境定義，則欄位會完全空白。

這是 MQGET 呼叫的輸出欄位。 此欄位的長度由 MQ_APPL_IDENTITY_DATA_LENGTH 提供。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 32 個空白字元。

ApplOrigin 資料 *(MQCHAR4)*

這是訊息 **原始環境定義** 的一部分。 如需訊息環境定義的相關資訊，請參閱 第 364 頁的『MQMD 概觀』 和 訊息環境定義 。

ApplOriginData 是應用程式套組所定義的資訊，可用來提供訊息來源的其他相關資訊。 例如，它可以由以適當使用者權限執行的應用程式設定，以指出身分資料是否受信任。

佇列管理程式會將此資訊視為字元資料，但不會定義其格式。 當佇列管理程式產生此資訊時，它會完全空白。

對於 MQPUT 及 MQPUT1 呼叫，如果在 *PutMsgOpts* 參數中指定 MQPMO_SET_ALL_CONTEXT，則這是輸入/輸出欄位。 會捨棄欄位中空值字元之後的任何資訊。 佇列管理程式會將空值字元及任何後續字元轉換為空白。 如果未指定 MQPMO_SET_ALL_CONTEXT，則輸入時會忽略此欄位，且此欄位是僅限輸出欄位。

這是 MQGET 呼叫的輸出欄位。 此欄位的長度由 MQ_APPL_ORIGIN_DATA_LENGTH 提供。 這個欄位的起始值在 C 中是空字串，在其他程式設計語言中是 4 個空白字元。

當發佈訊息時，雖然已設定 *ApplOrigin* 資料，但它在所接收的訂閱中是空白的。

BackoutCount (MQLONG)

這是 MQGET 呼叫先前作為工作單元的一部分傳回訊息並隨後取消的次數。 它可協助應用程式偵測基於訊息內容的處理錯誤。 此計數會排除指定任何 MQGMO_BROWSE_ * 選項的 MQGET 呼叫。

HardenGetBackout 佇列屬性會影響此計數的精確度; 請參閱 第 731 頁的『佇列的屬性』。

在 z/OS 上，值 255 表示訊息已取消 255 次以上; 傳回的值永不會大於 255。

這是 MQGET 呼叫的輸出欄位。 MQPUT 和 MQPUT1 呼叫會忽略它。 此欄位的起始值為 0。

CodedCharSetId (MQLONG)
此欄位指定訊息內文中字元資料的字集 ID。

註: MQMD 及其他 MQ 資料結構中的字元資料 (即呼叫上的參數) 必須在佇列管理程式的字集中。 這是由佇列管理程式的 *CodedCharSetId* 屬性所定義; 如需此屬性的詳細資料，請參閱 第 701 頁的『佇列管理程式的屬性』 。

在選項中使用 MQGMO_CONVERT 呼叫 MQGET 時，如果此欄位設為 MQCCSI_Q_MGR ，則用戶端與伺服器應用程式之間的行為會不同。 對於伺服器應用程式，用於字元轉換的字碼頁是佇列管理程式的 *CodedCharSetId* ; 對於用戶端應用程式，用於字元轉換的字碼頁是現行語言環境字碼頁。

對於用戶端應用程式，會根據用戶端的語言環境而非佇列管理程式上的語言環境來填寫 MQCCSI_Q_MGR。 該規則的例外是當您將訊息放入 IMS Bridge 佇列時; 在 MQMD 的 *CodedCharSetId* 欄位中，傳回的是佇列管理程式的 CCSID。

您不得使用下列特殊值:

MQCCSI_APPL
當使用 MQGET 呼叫搭配 MQGMO_CONVERT 選項來接收訊息時，這會導致 MQMD 的 *CodedCharSetId* 欄位值不正確，並導致回覆碼 MQRC_SOURCE_CCSID_ERROR (若為 z/OS，則為 MQRC_FORMAT_ERROR)。

您可以使用下列特殊值:

MQCCSI_Q_MGR
訊息中的字元資料是在佇列管理程式的字集中。

在 MQPUT 及 MQPUT1 呼叫上，佇列管理程式會將隨訊息一起傳送的 MQMD 中的此值變更為佇列管理程式的真實字集 ID。 因此， MQGET 呼叫永不會傳回 MQCCSI_Q_MGR 值。

MQCCSI_DEFAULT
String 欄位中資料的 *CodedCharSetId* 由 MQCFH 結構之前的標頭結構中的 *CodedCharSetId* 欄位定義，如果 MQCFH 位於訊息開頭，則由 MQMD 中的 *CodedCharSetId* 欄位定義。

MQCCSI_INHERIT
訊息中字元資料的字集與此結構相同; 這是佇列管理程式的字集。 (僅適用於 MQMD , MQCCSI_INHERIT 與 MQCCSI_Q_MGR 具有相同的意義)。

在與訊息一起傳送的 MQMD 中，佇列管理程式會將此值變更為 MQMD 的實際字集 ID。 如果沒有發生錯誤， MQGET 呼叫不會傳回值 MQCCSI_INHERIT。

如果 MQMD 中 *PutApplType* 欄位的值是 MQAT_BROKER ，請勿使用 MQCCSI_INHERIT。

MQCCSI_Embedded
訊息中的字元資料是具有訊息資料本身所含 ID 的字集。 可以在訊息資料內內嵌任意數目的字集 ID ，以套用至資料的不同部分。 此值必須用於包含混合字集資料的 PCF 訊息 (格式為 MQFMT_ADMIN、MQFMT_EVENT 或 MQFMT_PCF)。 PCF 訊息中包含的每一個 MQCFST、MQCFSL 和 MQCFSF 結構必須指定明確字集 ID ，而不是 MQCCSI_DEFAULT。

如果 MQFMT_EMBEDDED_PCF 格式的訊息要包含混合字集的資料，請不要使用 MQCCSI_EMBEDDED。 請改為在 MQEPH 結構的旗標欄位中設定 MQEPH_CCSID_EMBEDDED。 這相當於在前述結構中設定 MQCCSI_EMBEDDED。 然後， PCF 訊息中包含的每一個 MQCFST、MQCFSL 及 MQCFSF 結構必須指定明確字集 ID ，而不是 MQCCSI_DEFAULT。 如需 MQEPH 結構的相關資訊，請參閱 第 318 頁的『MQEPH-內嵌 PCF 標頭』。

僅在 MQPUT 及 MQPUT1 呼叫上指定此值。 如果在 MQGET 呼叫中指定它，則會阻止轉換訊息。

在 MQPUT 及 MQPUT1 呼叫上，佇列管理程式會變更與上述訊息一起傳送的 MQMD 中的 MQCCSI_Q_MGR 及 MQCCSI_INHERIT 值，但不會變更 MQPUT 或 MQPUT1 呼叫上指定的 MQMD。 不會對指定的值執行其他檢查。

擷取訊息的應用程式必須比較這個欄位與應用程式預期的值; 如果值不同，應用程式可能需要轉換訊息中的字元資料。

如果您在 MQGET 呼叫上指定 MQGMO_CONVERT 選項，則此欄位是輸入/輸出欄位。 應用程式指定的值是必要時要轉換訊息資料的編碼字集 ID。 如果轉換成功或不需要，則該值保持不變 (但 MQCCSI_Q_MGR 或 MQCCSI_INHERIT 值會轉換為實際值)。 如果轉換不成功，則 MQGET 呼叫之後的值代表傳回應用程式之未轉換訊息的編碼字集 ID。

否則，這是 MQGET 呼叫的輸出欄位，以及 MQPUT 和 MQPUT1 呼叫的輸入欄位。 此欄位的起始值為 MQCCSI_Q_MGR。

CorrelId (MQBYTE24)
CorrelId 欄位是訊息標頭中的內容，可用來識別特定訊息或訊息群組。

這是位元組字串，應用程式可以用來將訊息關聯至另一個訊息，或將訊息關聯至應用程式正在執行的其他工作。 相關性 ID 是訊息的永久內容，在重新啟動佇列管理程式之後會持續保存。 因為相關性 ID 是位元組字串而非字串，所以當訊息從一個佇列管理程式流向另一個佇列管理程式時，在字集之間 不會 轉換相關性 ID。

對於 MQPUT 和 MQPUT1 呼叫，應用程式可以指定任何值。 佇列管理程式會隨訊息一起傳輸此值，並將它遞送至發出訊息取得要求的應用程式。

如果應用程式指定 MQPMO_NEW_CORREL_ID，則佇列管理程式會產生與訊息一起傳送的唯一相關性 ID，並在從 MQPUT 或 MQPUT1 呼叫輸出時傳回給傳送應用程式。

佇列管理程式所產生的相關性 ID 包含 3 個位元組的產品 ID (ASCII 或 EBCDIC 中的 AMQ 或 CSQ)，後面接著一個保留位元組，以及唯一字串的產品特定實作。 在 WebSphere MQ 中，這個產品特定的實作字串包含佇列管理程式名稱的前 12 個字元，以及從系統時鐘衍生的值。 因此，所有可交互通訊的佇列管理程式都必須具有前 12 個字元不同的名稱，以確保訊息 ID 是唯一的。 產生唯一字串的能力也取決於未向後變更的系統時鐘。 若要消除佇列管理程式所產生的訊息 ID 複製應用程式所產生的訊息 ID 的可能性，應用程式必須避免產生其起始字元在 ASCII 或 EBCDIC (X'41 '至 X'49' 及 X'C1'至 X'C9') 中介於 A 到 I 範圍內的 ID。 不過，不會阻止應用程式產生起始字元在這些範圍內的 ID。

此產生的相關性 ID 會與訊息一起保留 (如果已保留)，並在將訊息作為發佈資訊傳送給訂閱者時用作相關性 ID，這些訂閱者在 MQSUB 呼叫所傳遞的 MQSD 中的 SubCorrelID 欄位中指定 MQCI_NONE。 如需保留發佈資訊的詳細資料，請參閱 MQPMO 選項 。

當佇列管理程式或訊息通道代理程式產生報告訊息時，它會以原始訊息的 *Report* 欄位 (MQRO_COPY_MSG_ID_TO_CORREL_ID 或 MQRO_PASS_CORREL_ID) 指定的方式來設定 *CorrelId* 欄位。 產生報告訊息的應用程式也必須執行此動作。

對於 MQGET 呼叫， *CorrelId* 是五個欄位之一，可用來選取要從佇列擷取的特定訊息。 如需如何指定此欄位值的詳細資料，請參閱 *MsgId* 欄位的說明。

指定 MQCI_NONE 作為相關性 ID 的效果與 不 指定 MQMO_MATCH_CORREL_ID 的效果相同，亦即， 任何相關性 ID 都將符合。

如果在 MQGET 呼叫的 *GetMsgOpts* 參數中指定 MQGMO_MSG_UNDER_CURSOR 選項，則會忽略這個欄位。

從 MQGET 呼叫傳回時， *CorrelId* 欄位會設為所傳回訊息的相關性 ID (如果有的話)。

可以使用下列特殊值:

MQCI_NONE
未指定相關性 ID。

欄位長度的值為二進位零。

對於 C 程式設計語言，也會定義常數 MQCI_NONE_ARRAY; 此值與 MQCI_NONE 相同，但卻是字元陣列而非字串。

MQCI_NEW_SESSION
訊息是新階段作業的開始。

CICS 橋接器會將此值辨識為指出新階段作業的開始，即新訊息序列的開始。

對於 C 程式設計語言，也會定義常數 MQCI_NEW_SESSION_ARRAY; 此值與 MQCI_NEW_SESSION 相同，但卻是字元陣列而非字串。

對於 MQGET 呼叫，這是輸入/輸出欄位。 對於 MQPUT 及 MQPUT1 呼叫，如果 未 指定 MQPMO_NEW_CORREL_ID，則這是輸入欄位; 如果指定 MQPMO_NEW_CORREL_ID 為，則是輸出欄位。 此欄位的長度由 MQ_CORREL_ID_LENGTH 提供。 此欄位的起始值為 MQCI_NONE。

註:

您無法在階層中傳遞發佈的相關性 ID。 佇列管理程式會使用此欄位。

編碼 *(MQLONG)*

這會指定訊息中數值資料的數值編碼; 它不適用於 MQMD 結構本身中的數值資料。 數值編碼定義用於二進位整數、聚集十進位整數及浮點數字的表示法。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。 佇列管理程式不會檢查欄位是否有效。 下列是已定義的特殊值:

MQENC_NATIVE
編碼是應用程式執行所在的程式設計語言和機器的預設值。

註: 這個常數的值取決於程式設計語言和環境。 因此，必須使用適用於應用程式執行所在環境的標頭、巨集、COPY 或 INCLUDE 檔案來編譯應用程式。

放置訊息的應用程式通常會指定 MQENC_NATIVE。 擷取訊息的應用程式必須將這個欄位與 MQENC_NATIVE 值相互比較; 如果值不同，應用程式可能需要轉換訊息中的數值資料。 使用 MQGMO_CONVERT 選項可要求佇列管理程式在處理 MQGET 呼叫時轉換訊息。 如需如何建構 *Encoding* 欄位的詳細資料，請參閱 第 785 頁的『機器編碼』 。

如果您在 MQGET 呼叫上指定 MQGMO_CONVERT 選項，則此欄位是輸入/輸出欄位。 必要的話，應用程式指定的值是要將訊息資料轉換成的編碼。 如果轉換成功或不需要，則該值保持不變。 如果轉換不成功，MQGET 呼叫之後的值代表傳回給應用程式之未轉換訊息的編碼。

在其他情況下，這是 MQGET 呼叫的輸出欄位，以及 MQPUT 和 MQPUT1 呼叫的輸入欄位。 此欄位的起始值為 MQENC_NATIVE。

期限 *(MQLONG)*

這是由放置訊息的應用程式所設定的一段時間 (以十分之一秒為單位)。 如果訊息在這段時間之前未從目的地佇列中移除，則該訊息變成適合捨棄。

此值會減少，以反映訊息在目的地佇列上花費的時間，以及在任何中間傳輸佇列上花費的時間 (如果放置到遠端佇列的話)。 它也可以由訊息通道代理程式減少，以反映傳輸時間 (如果這些時間很重要的話)。 同樣地，將此訊息轉遞至另一個佇列的應用程式可能會在必要時減少此值 (如果它已保留訊息很長時間的話)。 不過，有效期限會被視為近似，而且值不需要減少以反映小的時間間隔。

當應用程式使用 MQGET 呼叫來擷取訊息時， *Expiry* 欄位代表仍保留的原始到期時間量。

在經歷訊息的到期時間之後，它會變成適合由佇列管理程式捨棄。 當發生瀏覽或非瀏覽 MQGET 呼叫時，如果訊息尚未過期，則會捨棄該訊息。 例如， MQGMO 中的 *MatchOptions* 欄位設為 MQMO_NONE 從 FIFO 排序佇列讀取的非瀏覽 MQGET 呼叫會捨棄所有過期訊息，直到第一個未過期訊息為止。 使用優先順序排序的佇列，相同的呼叫會捨棄優先順序較高的過期訊息，以及在第一個未過期訊息之前到達佇列的優先順序相等訊息。

已過期的訊息永不會傳回至應用程式 (透過瀏覽或非瀏覽 MQGET 呼叫) ，因此在成功 MQGET 呼叫之後，訊息描述子的 *Expiry* 欄位中的值會大於零，或特殊值 MQEI_UNLIMITED。

如果將訊息放置在遠端佇列上，則在訊息到達目的地佇列之前，訊息可能會在中間傳輸佇列上到期 (並捨棄)。

如果訊息指定其中一個 MQRO_EXPIRATION_ * 報告選項，則會在捨棄過期訊息時產生報告。 如果未指定任何這些選項，則不會產生此類報告; 假設訊息在此時段之後不再相關 (可能是因為後面的訊息已取代它)。

對於同步點內放置的訊息，期限間隔從放置訊息的時間開始，而不是確定同步點的時間。 在確定同步點之前，期限間隔可能已過。 在此情況下，在確定作業之後會捨棄訊息，且不會將訊息傳回應用程式以回應 MQGET 作業。

根據到期時間捨棄訊息的任何其他程式也必須傳送適當的報告訊息 (如果要求的話)。

註:

1. 如果放置的訊息具有 *Expiry* 時間零或大於 999 999 999 的數字， MQPUT 或 MQPUT1 呼叫會失敗，原因碼為 MQRC_EXPIRY_ERROR; 在此情況下不會產生報告訊息。
2. 因為具有已經歷到期時間的訊息可能要等到稍後才會捨棄，所以佇列上可能有訊息已超過到期時間，因此不適合擷取。 不過，這些訊息會計入佇列上所有用途的訊息數，包括深度觸發。

3. 如果要求，則會在捨棄訊息時產生有效期限報告，而不是在訊息變成適合捨棄時產生。
4. 捨棄過期訊息，並在要求時產生到期報告，絕不是應用程式工作單元的一部分，即使訊息已排定捨棄，因為 MQGET 呼叫在工作單元內運作。
5. 如果工作單元內的 MQGET 呼叫擷取了接近過期的訊息，且隨後取消該工作單元，則該訊息可能變成適合在重新擷取之前予以捨棄。
6. 如果具有 MQGMO_LOCK 的 MQGET 呼叫已鎖定接近過期的訊息，則在具有 MQGMO_MSG_UNDER_CURSOR 的 MQGET 呼叫可以擷取訊息之前，該訊息可能變成適合捨棄; 如果發生這種情況，則會在此後續 MQGET 呼叫上傳回原因碼 MQRC_NO_MSG_UNDER_CURSOR。
7. 當擷取到期時間大於零的要求訊息時，應用程式可以在傳送回覆訊息時採取下列其中一個動作:
   - 將要求訊息中剩餘的到期時間複製到回覆訊息。
   - 請將回覆訊息中的到期時間設為大於零的明確值。
   - 將回覆訊息中的到期時間設為 MQEI_UNLIMITED。

   要採取的動作取決於應用程式的設計。 不過，將訊息放入無法傳送的郵件 (無法遞送的訊息) 佇列的預設動作必須是保留訊息的剩餘到期時間，並繼續減少它。
8. 觸發訊息一律使用 MQEI_UNLIMITED 產生。
9. *Format* 名稱為 MQFMT_XMIT_Q_HEADER 的訊息 (通常位於傳輸佇列上) 在 MQXQH 內具有第二個訊息描述子。 因此，它有兩個相關聯的 *Expiry* 欄位。 在此情況下，應注意下列其他要點:
   - 當應用程式將訊息放入遠端佇列時，佇列管理程式會起始將訊息放在本端傳輸佇列上，並以 MQXQH 結構作為應用程式訊息資料的字首。 佇列管理程式會將兩個 *Expiry* 欄位的值設定為與應用程式指定的值相同。

     如果應用程式將訊息直接放置在本端傳輸佇列上，則訊息資料必須已以 MQXQH 結構開頭，且格式名稱必須是 MQFMT_XMIT_Q_HEADER。 在此情況下，應用程式不需要將這兩個 *Expiry* 欄位的值設為相同。 (佇列管理程式會檢查 MQXQH 內的 *Expiry* 欄位是否包含有效值，以及訊息資料的長度是否足以包含它)。 對於可以直接寫入傳輸佇列的應用程式，應用程式必須使用內嵌的訊息描述子來建立傳輸佇列標頭。 不過，如果寫入傳輸佇列的訊息描述子中的期限值與內嵌訊息描述子中的值不一致，則會發生期限錯誤拒絕。
   - 從佇列擷取 *Format* 名稱為 MQFMT_XMIT_Q_HEADER 的訊息時 (不論這是正常或傳輸佇列) ，佇列管理程式會減少 兩者 這些 *Expiry* 欄位，並具有在佇列上等待所花費的時間。 如果訊息資料不夠長，無法在 MQXQH 中包含 *Expiry* 欄位，則不會發生任何錯誤。
   - 佇列管理程式會使用個別訊息描述子中的 *Expiry* 欄位 (亦即，不是 MQXQH 結構內內嵌的訊息描述子中的訊息描述子) 來測試訊息是否適合捨棄。
   - 如果兩個 *Expiry* 欄位的起始值不同，則擷取訊息時個別訊息描述子中的 *Expiry* 時間可能大於零 (因此訊息不適合捨棄) ，而 MQXQH 中根據 *Expiry* 欄位的時間已過。 在此情況下， MQXQH 中的 *Expiry* 欄位會設為零。
10. 除非 MQIIH 的「旗標」欄位中設定 MQQIH_PASS_EXPIRATION ，否則從 IMS 橋接器傳回的回覆訊息的到期時間無限制。 如需相關資訊，請參閱 旗標 。

可辨識下列特殊值:

MQEI_UNLIMITED
訊息有效期限無限制。

這是 MQGET 呼叫的輸出欄位，以及 MQPUT 和 MQPUT1 呼叫的輸入欄位。 此欄位的起始值為 MQEI_UNLIMITED。

回饋 *(MQLONG)*
「意見回饋」欄位與 MQMT_REPORT 類型的訊息一起使用，以指出報告的本質，且僅對該類型的訊息有意義。

欄位可以包含其中一個 MQFB_ * 值，或其中一個 MQRC_ * 值。 回饋碼分組如下:

MQFB_NONE
未提供任何意見。

MQFB_SYSTEM_FIRST
系統產生回饋的最低值。

MQFB_SYSTEM_LAST
系統產生回饋的最高值。

系統產生的回饋碼 MQFB_SYSTEM_FIRST 到 MQFB_SYSTEM_LAST 的範圍包括本主題中列出的一般回饋碼 (MQFB_ *) ，以及無法將訊息放入目的地佇列時可能發生的原因碼 (MQRC_ *)。

MQFB_APPL_FIRST
應用程式所產生意見回饋的最低值。

MQFB_APPL_LAST
應用程式所產生意見回饋的最高值。

產生報告訊息的應用程式不得使用系統範圍 (非 MQFB_QUIT) 中的回饋碼，除非它們想要模擬佇列管理程式或訊息通道代理程式所產生的報告訊息。

在 MQPUT 或 MQPUT1 呼叫上，指定的值必須是 MQFB_NONE ，或在系統範圍或應用程式範圍內。 不論 *MsgType* 的值為何，都會勾選此選項。

一般回饋碼:

MQFB_COA
確認到達目的地佇列 (請參閱 MQRO_COA)。

MQFB_COD
確認遞送至接收端應用程式 (請參閱 MQRO_COD)。

MQFB_EXPIRATION
已捨棄訊息，因為在到期時間已過之前未從目的地佇列中移除該訊息。

MQFB_PAN
正面動作通知 (請參閱 MQRO_PAN)。

MQFB_NAN
負面動作通知 (請參閱 MQRO_NAN)。

MQFB_QUIT
結束應用程式。

工作量排程程式可以使用它來控制執行中應用程式的實例數。 將具有此回饋碼的 MQMT_REPORT 訊息傳送至應用程式實例，會向該實例指出它應該停止處理。 不過，遵循此慣例是應用程式的問題; 它不是由佇列管理程式所強制執行。

通道回饋碼:

MQFB_CHANNEL_COMPLETED
通道正常結束。

MQFB_CHANNEL_FAIL
通道異常結束並進入 STOPPED 狀態。

MQFB_CHANNEL_FAIL_RETRY
通道異常結束並進入 RETRY 狀態。

IMS-橋接器回饋碼

當收到非預期的 IMS-OTMA 感應碼時，會使用這些代碼。 感應碼或當感應碼為 0x1A 時，與該感應碼相關聯的原因碼會在 *Feedback* 中指出。

1. 對於 MQFB_IMS_FIRST (300) 到 MQFB_IMS_LAST (399) 範圍內的 *Feedback* 代碼，收到 0x1A 以外的感應碼。 感應碼 由表示式提供 (*Feedback* - MQFB_IMS_FIRST+1)
2. 對於範圍 MQFB_IMS_NACK_1A_REASON_FIRST (600) 到 MQFB_IMS_NACK_1A_REASON_LAST (855) 之間的 意見 代碼，收到感應碼 0x1A 。 與感應碼相關聯的 原因碼 由表示式提供 (*Feedback* - MQFB_IMS_NACK_1A_REASON_FIRST)

Open Transaction Manager Access Guide and Reference 中說明 IMS-OTMA 感應碼及對應原因碼的意義。

IMS 橋接器可以產生下列回饋碼:

MQFB_DATA_LENGTH_ZERO
在訊息的應用程式資料中，區段長度為零。

MQFB_DATA_LENGTH_NEGATIVE
在訊息的應用程式資料中，區段長度是負數。

MQ fb_data_length_too_big
在訊息的應用程式資料中，區段長度太大。

MQ fb_BUFFER_溢位
其中一個長度欄位的值會導致資料溢位訊息緩衝區。

MQFB_LENGTH_OFF_BY_ONE
其中一個長度欄位的值太短 1 個位元組。

MQFB_IIH_ERROR
MQMD 中的 *Format* 欄位指定 MQFMT_IMS，但訊息的開頭不是有效的 MQIIH 結構。

MQFB_NOT_AUTHORIZED_FOR_IMS
訊息描述子 MQMD 中包含的使用者 ID，或 MQIIH 結構中 *Authenticator* 欄位中包含的密碼，未通過 IMS 橋接器所執行的驗證。 因此，訊息未傳遞至 IMS。

MQFB_IMS_ERROR
IMS 傳回非預期的錯誤。 如需錯誤的相關資訊，請參閱 IMS 橋接器所在系統上的 WebSphere MQ 錯誤日誌。

MQFB_IMS_FIRST
當 IMS-OTMA 感應碼不是 0x1A 時， IMS 產生的回饋碼位於 MQFB_IMS_FIRST (300) 到 MQFB_IMS_LAST (399) 的範圍內。 IMS-OTMA 感應碼本身是 *Feedback* 減去 MQFB_IMS_ERROR。

MQFB_IMS_LAST
當感應碼不是 0x1A 時， IMS 產生的意見回饋的最高值。

MQFB_IMS_NACK_1A_REASON_FIRST
當感應碼為 0x1A 時， IMS 產生的回饋碼位於 MQFB_IMS_NACK_1A_REASON_FIRST (600) 至 MQFB_IMS_NACK_1A_REASON_LAST (855) 範圍內。

MQFB_IMS_NACK_1A_REASON_LAST
當感應碼為 0x1A 時， IMS 產生的意見回饋的最高值

CICS 橋接器回饋碼: CICS 橋接器可以產生下列回饋碼:

MQFB_CICS_APPL_ABENDED
訊息中指定的應用程式異常結束。 此回饋碼僅出現在 MQDLH 結構的 *Reason* 欄位中。

MQFB_CICS_APPL_NOT_STARTED
訊息中所指定應用程式的 EXEC CICS LINK 失敗。 此回饋碼僅出現在 MQDLH 結構的 *Reason* 欄位中。

MQFB_CICS_BRIDGE_FAILURE
CICS 橋接器已異常終止，但未完成正常錯誤處理。

MQFB_CICS_CCSID_ERROR
字集 ID 無效。

MQFB_CICS_CIH_ERROR
CICS 資訊標頭結構遺漏或無效。

MQFB_CICS_XX_ENCODE_CASE_ONE commarea_error
CICS COMMAREA 的長度無效。

MQFB_CICS_CORREL_ID_ERROR
相關性 ID 無效。

MQFB_CICS_DLQ_ERROR
CICS 橋接器作業無法將此要求的回覆複製到無法傳送郵件的佇列。 已取消要求。

MQFB_CICS_ENCODING_ERROR
編碼無效。

MQFB_CICS_INTERNAL_ERROR
CICS 橋接器發生非預期的錯誤。

此回饋碼僅出現在 MQDLH 結構的 *Reason* 欄位中。

MQFB_CICS_NOT_AUTHORIZED
使用者 ID 未獲授權或密碼無效。

此回饋碼僅出現在 MQDLH 結構的 *Reason* 欄位中。

MQFB_CICS_UOW_BACKED_OUT
由於下列其中一個原因，已取消工作單元:

- 處理相同工作單元內的另一個要求時偵測到失敗。
- 工作單元進行時發生 CICS 異常終止。

MQFB_CICS_UOW_ERROR
工作單元控制欄位 *UOWControl* 無效。

追蹤路徑訊息回饋碼:

MQFB_ACTIVITY
與 MQFMT_EMBEDDED_PCF 格式搭配使用，以容許使用者資料在活動報告之後的選項。

MQFB_MAX_ACTIVITIES
因訊息所涉及的活動數目超出活動上限而捨棄追蹤路徑訊息時傳回。

MQFB_NOT_FORWARDED
因為追蹤路徑訊息即將傳送至不支援追蹤路徑訊息的遠端佇列管理程式，所以捨棄該訊息時傳回。

MQ fb_not_delivered
因為追蹤路徑訊息即將放入本端佇列而捨棄時傳回。

MQFB_UNSUPPORTED_FORWARDING
因為 forwarding 參數中的值無法辨識，且在拒絕的位元遮罩中，而捨棄 trace-route 訊息時傳回。

MQ fb_UNSUPPORTED_Delivery
因為遞送參數中的值無法辨識，且在拒絕的位元遮罩中，而捨棄追蹤路徑訊息時傳回。

WebSphere MQ 原因碼: 對於異常狀況報告訊息， *Feedback* 包含 WebSphere MQ 原因碼。 可能的原因碼如下:

MQRC_PUT_INHIBITED
(2051 ， X'803 ') 佇列禁止放置呼叫。

MQRC_Q_FULL
(2053 ， X'805 ') 佇列已包含訊息數目上限。

MQRC_NOT_AUTHORIZED
(2035 ， X'7F3') 未獲授權存取。

無法使用 MQRC_Q_SPACE_NOT_AVAILABLE
(2056 ， X'808 ') 磁碟上沒有可供佇列使用的空間。

MQRC_PERSISTENT_NOT_ALLOWED
(2048 ， X'800 ') 佇列不支援持續訊息。

MQRC_MSG_TOO_BIG_FOR_Q_MGR
(2031 ， X'7EF') 訊息長度大於佇列管理程式的上限。

MQRC_MSG_TOO_BIG_FOR_Q
(2030 ， X'7EE') 訊息長度大於佇列的上限。

如需原因碼的完整清單，請參閱:

- 若為 WebSphere MQ for z/OS，請參閱 API 原因碼。
- 對於所有其他平台，請參閱 API 完成及原因碼。

.

這是 MQGET 呼叫的輸出欄位，以及 MQPUT 和 MQPUT1 呼叫的輸入欄位。 此欄位的起始值為 MQFB_NONE。

格式 *(MQCHAR8)*

這是訊息傳送端用來向接收端指出訊息中資料本質的名稱。 可以針對名稱指定佇列管理程式字集中的任何字元，但您必須將名稱限制為下列:

- 大寫 A 到 Z
- 數字 0 到 9

如果使用其他字元，可能無法在傳送端和接收端佇列管理程式的字集之間轉換名稱。

以空白填補名稱至欄位長度，或使用空值字元來終止欄位結尾之前的名稱; 空值及任何後續字元會被視為空白。 請勿指定含有前導或內含空白的名稱。 對於 MQGET 呼叫，佇列管理程式會傳回以空白填補欄位長度的名稱。

佇列管理程式不會檢查名稱是否符合上述建議。

以大寫、小寫及混合大小寫的 MQ 開頭的名稱具有佇列管理程式所定義的意義; 對於您自己的格式，請勿使用以這些字母開頭的名稱。佇列管理程式內建格式如下:

MQFMT_NONE
未定義資料的本質: 使用 MQGMO_CONVERT 選項從佇列擷取訊息時，無法轉換資料。

如果您在 MQGET 呼叫中指定 MQGMO_CONVERT，且訊息中資料的字集或編碼與 *MsgDesc* 參數中指定的不同，則會傳回訊息，並具有下列完成碼及原因碼 (假設沒有其他錯誤):

- 如果 MQFMT_NONE 資料位於訊息開頭，則完成碼 MQCC_WARNING 及原因碼 MQRC_FORMAT_ERROR。
- 如果 MQFMT_NONE 資料位於訊息結尾 (亦即，前面有一個以上 MQ 標頭結構)，則完成碼 MQCC_OK 及原因碼 MQRC_NONE。 在此情況下， MQ 標頭結構會轉換為所要求的字集及編碼。

對於 C 程式設計語言，也會定義常數 MQFMT_NONE_ARRAY; 此值與 MQFMT_NONE 相同，但卻是字元陣列而非字串。

MQFMT_ADMIN
訊息是可程式化指令格式 (PCF) 的指令伺服器要求或回覆訊息。 如果在 MQGET 呼叫上指定 MQGMO_CONVERT 選項，則可以轉換此格式的訊息。 如需使用可程式化指令格式訊息的相關資訊，請參閱 使用可程式化指令格式 。

對於 C 程式設計語言，也會定義常數 MQFMT_ADMIN_ARRAY; 此值與 MQFMT_ADMIN 相同，但卻是字元陣列而非字串。

MQFMT_CICS
訊息資料以 CICS 資訊標頭 MQCIH 開頭，後面接著應用程式資料。 應用程式資料的格式名稱由 MQCIH 結構中的 *Format* 欄位提供。

在 z/OS 上，請在 MQGET 呼叫上指定 MQGMO_CONVERT 選項，以轉換 MQFMT_CICS 格式的訊息。

對於 C 程式設計語言，也會定義常數 MQFMT_CICS_ARRAY; 此值與 MQFMT_CICS 相同，但卻是字元陣列而非字串。

MQFMT_COMMAND_1
訊息是包含物件計數、完成碼及原因碼的 MQSC 指令-伺服器回覆訊息。 如果在 MQGET 呼叫上指定 MQGMO_CONVERT 選項，則可以轉換此格式的訊息。

對於 C 程式設計語言，也會定義常數 MQFMT_COMMAND_1_ARRAY ; 此值與 MQFMT_COMMAND_1 相同，但卻是字元陣列而非字串。

MQFMT_COMMAND_2
此訊息是 MQSC 指令-伺服器回覆訊息，包含所要求物件的相關資訊。 如果在 MQGET 呼叫上指定 MQGMO_CONVERT 選項，則可以轉換此格式的訊息。

對於 C 程式設計語言，也會定義常數 MQFMT_COMMAND_2_ARRAY ; 此值與 MQFMT_COMMAND_2 相同，但卻是字元陣列而非字串。

MQFMT_DEAD_LETTER_HEADER
訊息資料以無法傳送郵件的標頭 MQDLH 開頭。 來自原始訊息的資料緊跟在 MQDLH 結構後面。 原始訊息資料的格式名稱由 MQDLH 結構中的 *Format* 欄位提供; 如需此結構的詳細資料，請參閱 第 306 頁的『MQDLH-無法傳送郵件的標頭』。 如果在 MQGET 呼叫上指定 MQGMO_CONVERT 選項，則可以轉換此格式的訊息。

對於 *Format* 為 MQFMT_DEAD_LETTER_HEADER 的訊息，不會產生 COA 及 COD 報告。

對於 C 程式設計語言，也會定義常數 MQFMT_DEAD_LETTER_HEADER_ARRAY; 此值與 MQFMT_DEAD_LETTER_HEADER 值相同，但卻是字元陣列而非字串。

MQFMT_DIST_HEADER
訊息資料以配送清單標頭 MQDH 開頭; 這包括 MQOR 及 MQPMR 記錄的陣列。 distribution-list 標頭後面可以接著其他資料。 其他資料的格式 (如果有的話) 由 MQDH 結構中的 *Format* 欄位提供; 如需此結構的詳細資料，請參閱 第 301 頁的『MQDH-Distribution 標頭』。 如果在 MQGET 呼叫上指定 MQGMO_CONVERT 選項，則可以轉換格式為 MQFMT_DIST_HEADER 的訊息。

在下列環境中支援此格式: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows，以及連接至這些系統的 WebSphere MQ MQI 用戶端。

對於 C 程式設計語言，也會定義常數 MQFMT_DIST_HEDER_ARRAY; 此值與 MQFMT_DIST_HEADER 相同，但卻是字元陣列而非字串。

MQFMT_EMBEDDED_PCF
追蹤路徑訊息的格式，前提是 PCF 指令值設為 MQCMD_TRACE_ROUTE。 使用此格式可讓使用者資料與追蹤路徑訊息一起傳送，但前提是其應用程式可以應付之前的 PCF 參數。

PCF 標頭 **必須** 是第一個標頭，否則不會將訊息視為追蹤路徑訊息。 這表示訊息不能在群組中，且追蹤路徑訊息無法分段。 如果在群組中傳送追蹤路徑訊息，則會拒絕訊息，原因碼為 MQRC_MSG_NOT_ALLOWED_IN_GROUP。

請注意， MQFMT_ADMIN 也可以用於追蹤路徑訊息的格式，但在此情況下，無法隨追蹤路徑訊息一起傳送任何使用者資料。

MQFMT_EVENT
此訊息是 MQ 事件訊息，報告發生的事件。 事件訊息具有與可程式化指令相同的結構; 如需此結構的相關資訊，請參閱 PCF 指令訊息 ，以及 事件監視 ，以取得事件的相關資訊。

如果在 MQGET 呼叫上指定 MQGMO_CONVERT 選項，則可在所有環境中轉換 Version-1 事件訊息。 Version-2 事件訊息只能在 z/OS 上轉換。

對於 C 程式設計語言，也會定義常數 MQFMT_EVENT_ARRAY; 此值與 MQFMT_EVENT 相同，但卻是字元陣列而非字串。

MQFMT_IMS
訊息資料以 IMS 資訊標頭 MQIIH 開頭，後面接著應用程式資料。 應用程式資料的格式名稱由 MQIIH 結構中的 *Format* 欄位提供。

如需搭配使用 MQGET 與 MQGMO_CONVERT 時如何處理 MQIIH 結構的詳細資料，請參閱 第 351 頁的『格式 (MQCHAR8)』 及 第 351 頁的『ReplyTo 格式 (MQCHAR8)』。

對於 C 程式設計語言，也會定義常數 MQFMT_IMS_ARRAY; 其值與 MQFMT_IMS 相同，但卻是字元陣列而非字串。

MQFMT_IMS_VAR_STRING
訊息是 IMS 變數字串，它是 `llzzccc` 格式的字串，其中:

ll
是一個 2 位元組長度欄位，指定 IMS 變數字串項目的總長度。 此長度等於 `ll` (2 個位元組) 的長度加上 `zz` (2 個位元組) 的長度加上字串本身的長度。 `ll` 是 *Encoding* 欄位指定的編碼中的 2 個位元組二進位整數。

zz
是一個 2 位元組欄位，包含對 IMS 有意義的旗標。 `zz` 是由兩個 MQBYTE 欄位所組成的位元組字串，在傳輸時不會從傳送端變更為接收端 (亦即， `zz` 不會進行任何轉換)。

ccc
是包含 ll-4 字元的可變長度字串。 ccc 是在 *CodedCharSetId* 欄位指定的字集中。

在 z/OS 上，訊息資料可以由一連串 IMS 變數字串組成，每一個字串都是 llzzccc 格式。 連續 IMS 變數字串之間不得跳過位元組。 這表示如果第一個字串有奇數長度，則第二個字串會不對齊，也就是說，它不會在 2 的倍數的界限上開始。 在需要對齊基本資料類型的機器上建構這類字串時，請小心。

在 MQGET 呼叫上使用 MQGMO_CONVERT 選項，以轉換格式為 MQFMT_IMS_VAR_STRING 的訊息。

對於 C 程式設計語言，也會定義常數 MQFMT_IMS_VAR_STRING_ARRAY; 此值與 MQFMT_IMS_VAR_STRING 相同，但卻是字元陣列而非字串。

MQFMT_MD_EXTENSION
訊息資料以訊息描述子延伸 MQMDE 開頭，並選擇性地後接其他資料 (通常是應用程式訊息資料)。 MQMDE 後面的資料的格式名稱、字集及編碼由 MQMDE 中的 *Format*、 *CodedCharSetId* 及 *Encoding* 欄位提供。 如需此結構的詳細資料，請參閱 第 405 頁的『MQMDE-訊息描述子延伸』 。 如果在 MQGET 呼叫上指定 MQGMO_CONVERT 選項，則可以轉換此格式的訊息。

對於 C 程式設計語言，也會定義常數 MQFMT_MD_EXTENSION_ARRAY; 此值與 MQFMT_MD_EXTENSION 相同，但它是字元陣列而非字串。

MQFMT_PCF
訊息是使用者定義的訊息，符合可程式化指令格式 (PCF) 訊息的結構。 如果在 MQGET 呼叫上指定 MQGMO_CONVERT 選項，則可以轉換此格式的訊息。 如需使用可程式化指令格式訊息的相關資訊，請參閱 使用可程式化指令格式 。

對於 C 程式設計語言，也會定義常數 MQFMT_PCF_ARRAY; 此值與 MQFMT_PCF 相同，但卻是字元陣列而非字串。

MQFMT_REF_MSG_HEADER
訊息資料以參照訊息標頭 MQRMH 開頭，後面選擇性地接著其他資料。 資料的格式名稱、字集及編碼由 MQRMH 中的 *Format*、 *CodedCharSetId* 及 *Encoding* 欄位提供。 如需此結構的詳細資料，請參閱 第 475 頁的『MQRMH-參照訊息標頭』 。 如果在 MQGET 呼叫上指定 MQGMO_CONVERT 選項，則可以轉換此格式的訊息。

在下列環境中支援此格式: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows，以及連接至這些系統的 WebSphere MQ MQI 用戶端。

若為 C 程式設計語言，也會定義常數 MQFMT_REF_MSG_HEADER_ARRAY; 此值與 MQFMT_REF_MSG_HEADER 相同，但卻是字元陣列而非字串。

MQFMT_RF_HEADER
訊息資料以規則和格式化標頭 MQRFH 開頭，後面可選擇性地接著其他資料。 資料的格式名稱、字集及編碼 (如果有的話) 由 MQRFH 中的 *Format*、 *CodedCharSetId* 及 *Encoding* 欄位提供。 如果在 MQGET 呼叫上指定 MQGMO_CONVERT 選項，則可以轉換此格式的訊息。

對於 C 程式設計語言，也會定義常數 MQFMT_RF_HEADER_ARRAY; 此值與 MQFMT_RF_HEADER 相同，但卻是字元陣列而非字串。

MQFMT_RF_HEADER_2
訊息資料以 version-2 規則及格式化標頭 MQRFH2 開頭，並選擇性地後接其他資料。 選用資料 (如果有的話) 的格式名稱、字集和編碼由 MQRFH2 中的 *Format*、 *CodedCharSetId* 和 *Encoding* 欄位提供。 如果在 MQGET 呼叫上指定 MQGMO_CONVERT 選項，則可以轉換此格式的訊息。

對於 C 程式設計語言，也會定義常數 MQFMT_RF_HEADER_2_ARRAY ; 此值與 MQFMT_RF_HEADER_2 相同，但卻是字元陣列而非字串。

MQFMT_STRING
應用程式訊息資料可以是 SBCS 字串 (單位元組字集) 或 DBCS 字串 (雙位元組字集)。 如果在 MQGET 呼叫上指定 MQGMO_CONVERT 選項，則可以轉換此格式的訊息。

對於 C 程式設計語言，也會定義常數 MQFMT_STRING_ARRAY; 此值與 MQFMT_STRING 相同，但卻是字元陣列而非字串。

MQFMT_TRIGGER
訊息是觸發訊息，由 MQTM 結構說明; 如需此結構的詳細資料，請參閱 第 521 頁的『MQTM-觸發訊息』 。 如果在 MQGET 呼叫上指定 MQGMO_CONVERT 選項，則可以轉換此格式的訊息。

對於 C 程式設計語言，也會定義常數 MQFMT_TRIGGER_ARRAY; 此值與 MQFMT_TRIGGER 值相同，但它是字元陣列而非字串。

MQFMT_WORK_INFO_HEADER
訊息資料以工作資訊標頭 MQWIH 開頭，後面接著應用程式資料。 應用程式資料的格式名稱由 MQWIH 結構中的 *Format* 欄位提供。

在 z/OS 上，在 MQGET 呼叫上指定 MQGMO_CONVERT 選項，以在格式為 MQFMT_WORK_INFO_HEADER 的訊息中轉換 使用者資料 。 不過， MQWIH 結構本身一律會以佇列管理程式的字集及編碼方式傳回 (亦即，不論是否指定 MQGMO_CONVERT 選項，都會轉換 MQWIH 結構)。

若為 C 程式設計語言，也會定義常數 MQFMT_WORK_INFO_HEADER_ARRAY; 此值與 MQFMT_WORK_INFO_HEADER 相同，但卻是字元陣列而非字串。

MQFMT_XMIT_Q_HEADER
訊息資料以傳輸佇列標頭 MQXQH 開頭。 來自原始訊息的資料會立即遵循 MQXQH 結構。 原始訊息資料的格式名稱由 MQMD 結構中的 *Format* 欄位提供，這是傳輸佇列標頭 MQXQH 的一部分。 如需此結構的詳細資料，請參閱 第 539 頁的『MQXQH-傳輸佇列標頭』 。

對於 *Format* 為 MQFMT_XMIT_Q_HEADER 的訊息，不會產生 COA 及 COD 報告。

對於 C 程式設計語言，也會定義常數 MQFMT_XMIT_Q_HEDER_ARRAY; 此值與 MQFMT_XMIT_Q_HEADER 值相同，但它是字元陣列而非字串。

這是 MQGET 呼叫的輸出欄位，以及 MQPUT 和 MQPUT1 呼叫的輸入欄位。 此欄位的長度由 MQ_FORMAT_LENGTH 提供。 此欄位的起始值為 MQFMT_NONE。

GroupId (MQBYTE24)

這是一個位元組字串，用來識別實體訊息所屬的特定訊息群組或邏輯訊息。 如果訊息容許分段，也會使用 *GroupId* 。 在所有這些情況下， *GroupId* 都具有非空值，並且在 *MsgFlags* 欄位中設定下列一個以上旗標:

- MQMF_MSG_IN_GROUP
- MQMF_LAST_MSG_IN_GROUP
- MQMF_SEGMENT
- MQMF_LAST_SEGMENT
- 容許 MQMF_SEGMENTATION_ALLOWED

如果未設定任何這些旗標，則 *GroupId* 具有特殊空值 MQGI_NONE。

在下列情況下，應用程式不需要在 MQPUT 或 MQGET 呼叫上設定此欄位:

- 在 MQPUT 呼叫上，指定 MQPMO_LOGICAL_ORDER。
- 在 MQGET 呼叫上， 未 指定 MQMO_MATCH_GROUP_ID。

這些是針對未報告訊息的訊息使用這些呼叫的建議方式。 不過，如果應用程式需要更多控制，或呼叫是 MQPUT1，則應用程式必須確保 *GroupId* 設為適當的值。

只有在群組 ID 是唯一的時，才能正確處理訊息群組和區段。 因此， 應用程式不得產生自己的群組 *ID*; 相反地，應用程式必須執行下列其中一項:

- 如果指定 MQPMO_LOGICAL_ORDER ，佇列管理程式會自動為邏輯訊息的群組或區段中的第一個訊息產生唯一群組 ID ，並將該群組 ID 用於邏輯訊息的群組或區段中的其餘訊息，因此應用程式不需要採取任何特殊動作。 這是建議的程序。
- 如果 未 指定 MQPMO_LOGICAL_ORDER ，應用程式必須針對邏輯訊息的群組或區段中的訊息，在第一個 MQPUT 或 MQPUT1 呼叫中將 *GroupId* 設為 MQGI_NONE ，以要求佇列管理程式產生群組 ID。 然後，佇列管理程式在該呼叫的輸出上傳回的群組 ID 必須用於邏輯訊息的群組或區段中的其餘訊息。 如果訊息群組包含分段的訊息，則群組中的所有區段及訊息必須使用相同的群組 ID。

未指定 MQPMO_LOGICAL_ORDER 時，邏輯訊息的群組及區段中的訊息可以任何順序 (例如，反向順序) 放置，但群組 ID 必須由針對任何這些訊息發出的 第一個 MQPUT 或 MQPUT1 呼叫來配置。

在 MQPUT 和 MQPUT1 呼叫的輸入上，佇列管理程式會使用 佇列上的實體順序中說明的值。 在 MQPUT 和 MQPUT1 呼叫的輸出上，如果開啟的物件是單一佇列而非配送清單，則佇列管理程式會將此欄位設為隨訊息傳送的值，但如果開啟的物件是配送清單，則它會維持不變。 在後一種情況下，如果應用程式需要知道產生的群組 ID，則應用程式必須提供包含 *GroupId* 欄位的 MQPMR 記錄。

在 MQGET 呼叫的輸入上，佇列管理程式會使用 第 339 頁的表 506 中說明的值。 在 MQGET 呼叫的輸出上，佇列管理程式會將此欄位設為所擷取訊息的值。

下列是已定義的特殊值:

MQGI_NONE
未指定群組 ID。

欄位長度的值為二進位零。 這是用於不在群組中、不是邏輯訊息區段且不容許分段的訊息的值。

對於 C 程式設計語言，也會定義常數 MQGI_NONE_ARRAY; 此值與 MQGI_NONE 相同，但它是字元陣列而非字串。

此欄位的長度由 MQ_GROUP_ID_LENGTH 提供。 此欄位的起始值是 MQGI_NONE。 如果 *Version* 小於 MQMD_VERSION_2，則會忽略此欄位。

MsgFlags (MQLONG)
MsgFlags 是指定訊息屬性或控制其處理的旗標。

MsgFlags 分為下列種類:

- 分段旗標
- 狀態旗標

分段旗標: 當訊息對佇列而言太大時，嘗試將訊息放置在佇列上通常會失敗。 分段是一種技術，佇列管理程式或應用程式會將訊息分割成較小的片段 (稱為區段)，並將每一個區段作為個別實體訊息放置在佇列上。 擷取訊息的應用程式可以逐一擷取區段，或要求佇列管理程式將區段重新組合成 MQGET 呼叫所傳回的單一訊息。 後者是透過在 MQGET 呼叫上指定 MQGMO_COMPLETE_MSG 選項，並提供足夠大的緩衝區以容納完整訊息來達成。 (如需 MQGMO_COMPLETE_MSG 選項的詳細資料，請參閱 第 323 頁的『MQGMO-取得訊息選項』。) 訊息可以在傳送端佇列管理程式、中間佇列管理程式或目的地佇列管理程式中分段。

您可以指定下列其中一項來控制訊息的分段:

MQMF_SEGMENTATION_INHIBITED
此選項可防止佇列管理程式將訊息分成區段。 如果指定給已經是區段的訊息，則此選項會防止區段分成較小的區段。

此旗標的值為二進位零。 這是預設值。

容許 MQMF_SEGMENTATION_ALLOWED
此選項容許佇列管理程式將訊息分成區段。 如果指定給已經是區段的訊息，則此選項容許將區段分成較小的區段。 在未設定 MQMF_SEGMENT 或 MQMF_LAST_SEGSEGMENT 的情況下，可以設定 MQMF_SEGMENTATION_ALLOWED。

- 在 z/OS 上，佇列管理程式不支援訊息分段。 如果訊息對佇列而言太大， MQPUT 或 MQPUT1 呼叫會失敗，原因碼為 MQRC_MSG_TOO_BIG_FOR_Q。 不過，仍然可以指定 MQMF_SEGMENTATION_ALLOWED 選項，並容許在遠端佇列管理程式中分段訊息。

當佇列管理程式將訊息分段時，佇列管理程式會在隨每一個區段傳送的 MQMD 副本中開啟 MQMF_SEGMENT 旗標，但不會變更 MQPUT 或 MQPUT1 呼叫上應用程式所提供 MQMD 中這些旗標的設定。 對於邏輯訊息中的最後一個區段，佇列管理程式也會在隨區段一起傳送的 MQMD 中開啟 MQMF_LAST_SEGMENT 旗標。

註: 放置具有 MQMF_SEGMENTATION_ALLOWED 但沒有 MQPMO_LOGICAL_ORDER 的訊息時請小心。 如果訊息為:

- 不是區段，且

- 不在群組中，以及
- 未轉遞，

在 每一個 MQPUT 或 MQPUT1 呼叫之前，應用程式必須將 *GroupId* 欄位重設為 MQGI_NONE ，佇列管理程式才能為每一個訊息產生唯一群組 ID。 如果未執行此動作，則不相關的訊息可能具有相同的群組 ID ，這可能會導致後續處理不正確。 如需何時重設 *GroupId* 欄位的相關資訊，請參閱 *GroupId* 欄位及 MQPMO_LOGICAL_ORDER 選項的說明。

佇列管理程式會視需要將訊息分割成區段，使區段 (加上任何必要的標頭資料) 符合佇列。 不過，佇列管理程式所產生區段的大小有下限，且只有從訊息建立的最後一個區段可以小於此限制 (應用程式所產生區段的大小下限是一個位元組)。 佇列管理程式所產生的區段長度可能不相等。 佇列管理程式會依照下列方式來處理訊息:

- 使用者定義格式在 16 個位元組的倍數界限上分割; 佇列管理程式不會產生小於 16 個位元組 (最後一個區段除外) 的區段。
- MQFMT_STRING 以外的內建格式會在適合所呈現資料本質的點進行分割。 不過，佇列管理程式絕不會在 WebSphere MQ 標頭結構中間分割訊息。 這表示佇列管理程式無法進一步分割包含單一 MQ 標頭結構的區段，因此該訊息的可能區段大小下限大於 16 個位元組。

  佇列管理程式所產生的第二個或更新的區段以下列其中一項開始:

  – MQ 標頭結構
  – 應用程式訊息資料的開頭
  – 應用程式訊息資料的一部分
- MQFMT_STRING 會分割，而不考慮資料呈現的本質 (SBCS、DBCS 或混合 SBCS/DBCS)。 當字串是 DBCS 或混合 SBCS/DBCS 時，這可能會導致區段無法從一個字集轉換成另一個字集。 佇列管理程式絕不會將 MQFMT_STRING 訊息分割成小於 16 個位元組 (最後一個區段除外) 的區段。
- 佇列管理程式會設定每一個區段的 MQMD 中的 *Format*、 *CodedCharSetId* 及 *Encoding* 欄位，以正確地說明區段 開頭 所呈現的資料; 格式名稱是內建格式的名稱，或使用者定義格式的名稱。
- 已修改 *Offset* 大於零之區段的 MQMD 中的 *Report* 欄位。 對於每一種報告類型，如果報告選項是 MQRO_ * _WITH_DATA ，但區段不能包含任何前 100 個位元組的使用者資料 (亦即，可能呈現的任何 WebSphere MQ 標頭結構之後的資料) ，則報告選項會變更為 MQRO_ *。

佇列管理程式遵循上述規則，但在其他情況下無法預測地分割訊息; 請不要對訊息分割的位置進行假設。

對於 持續 訊息，佇列管理程式只能在工作單元內執行分段:

- 如果 MQPUT 或 MQPUT1 呼叫在使用者定義的工作單元內運作，則會使用該工作單元。 如果在分段程序期間呼叫失敗，佇列管理程式會移除因呼叫失敗而放置在佇列上的任何區段。 不過，失敗並不會阻止順利確定工作單元。
- 如果呼叫是在使用者定義工作單元之外運作，且沒有使用者定義工作單元存在，則佇列管理程式只會在呼叫期間建立工作單元。 如果呼叫成功，佇列管理程式會自動確定工作單元。 如果呼叫失敗，佇列管理程式會取消工作單元。
- 如果呼叫是在使用者定義工作單元之外運作，但存在使用者定義工作單元，則佇列管理程式無法執行分段。 如果訊息不需要分段，則呼叫仍會成功。 但如果訊息需要分段，則呼叫會失敗，原因碼為 MQRC_UOW_NOT_AVAILABLE。

對於 非持續 訊息，佇列管理程式不需要有可用的工作單元，即可執行分段。

在轉換可能分段的訊息中的資料時，請特別小心:

- 如果接收端應用程式在 MQGET 呼叫上轉換資料，並指定 MQGMO_COMPLETE_MSG 選項，則資料轉換結束程式會傳遞完整訊息給結束程式以進行轉換，且訊息分段的事實對結束程式很明顯。
- 如果接收端應用程式一次擷取一個區段，則會呼叫資料轉換結束程式來一次轉換一個區段。 因此，結束程式必須獨立於任何其他區段中的資料來轉換區段中的資料。

  如果訊息中資料的本質導致 16 位元組界限上資料的任意分段可能導致結束程式無法轉換的區段，或格式為 MQFMT_STRING 且字集為 DBCS 或混合 SBCS/DBCS ，則傳送應用程式必須建立並放置區段，並指定 MQMF_SEGMENTATION_INHIBITED 以抑制進一步分段。 以此方式，傳送端應用程式可以確保每一個區段包含足夠的資訊，以容許資料轉換結束程式順利轉換區段。

- 如果指定傳送端訊息通道代理程式 (MCA) 的傳送端轉換， MCA 只會轉換不是邏輯訊息區段的訊息; MCA 絕不會嘗試轉換屬於區段的訊息。

此旗標是 MQPUT 和 MQPUT1 呼叫的輸入旗標，以及 MQGET 呼叫的輸出旗標。 在後者呼叫中，佇列管理程式也會將旗標的值回應至 MQGMO 中的 *Segmentation* 欄位。

此旗標的起始值為 MQMF_SEGMENTATION_INHIBITED。

狀態旗標: 這些旗標指出實體訊息是否屬於訊息群組，是邏輯訊息的區段 (兩者或兩者皆非)。 可以在 MQPUT 或 MQPUT1 呼叫上指定下列一或多個項目，或由 MQGET 呼叫傳回:

MQMF_MSG_IN_GROUP
訊息是群組的成員。

MQMF_LAST_MSG_IN_GROUP
訊息是群組中的最後一個邏輯訊息。

如果設定此旗標，則佇列管理程式會在隨訊息一起傳送的 MQMD 副本中開啟 MQMF_MSG_IN_GROUP ，但不會變更 MQPUT 或 MQPUT1 呼叫上應用程式所提供的 MQMD 中這些旗標的設定。

群組只包含一個邏輯訊息是有效的。 如果是這種情況，則會設定 MQMF_LAST_MSG_IN_GROUP ，但 *MsgSeqNumber* 欄位的值為 1。

MQMF_SEGMENT
訊息是邏輯訊息的區段。

當指定 MQMF_SEGMENT 但未指定 MQMF_LAST_SEGMENT 時，區段中應用程式訊息資料的長度 (排除可能存在任何 WebSphere MQ 標頭結構的長度) 必須至少為 1。 如果長度為零，則 MQPUT 或 MQPUT1 呼叫會失敗，原因碼為 MQRC_SEGMENT_LENGTH_ZERO。

在 z/OS 上，如果將訊息放置在索引類型為 MQIT_GROUP_ID 的佇列上，則不支援此選項。

MQMF_LAST_SEGMENT
訊息是邏輯訊息的最後一個區段。

如果設定此旗標，則佇列管理程式會在隨訊息一起傳送的 MQMD 副本中開啟 MQMF_SEGMENT ，但不會變更 MQPUT 或 MQPUT1 呼叫上應用程式所提供 MQMD 中這些旗標的設定。

邏輯訊息只能由一個區段組成。 如果是這樣，則會設定 MQMF_LAST_SEGMENT ，但 *Offset* 欄位的值為零。

當指定 MQMF_LAST_SEGMENT 時，區段中應用程式訊息資料的長度 (排除 可能存在的任何標頭結構長度) 可以是零。

在 z/OS 上，如果將訊息放置在索引類型為 MQIT_GROUP_ID 的佇列上，則不支援此選項。

應用程式必須確保在放置訊息時正確設定這些旗標。 如果指定了 MQPMO_LOGICAL_ORDER ，或在佇列控點的之前 MQPUT 呼叫中指定了 MQPMO_LOGICAL_ORDER ，則旗標的設定必須與佇列管理程式為佇列控點保留的群組和區段資訊一致。 當指定 MQPMO_LOGICAL_ORDER 時，下列條件適用於佇列控點的 後續 MQPUT 呼叫:

- 如果沒有現行群組或邏輯訊息，則所有這些旗標 (及其組合) 都是有效的。
- 一旦指定 MQMF_MSG_IN_GROUP ，它必須保持開啟，直到指定 MQMF_LAST_MSG_IN_GROUP 為止。 如果未滿足此條件，則呼叫會失敗，原因碼為 MQRC_INCOMPLETE_GROUP。
- 一旦指定 MQMF_SEGMENT ，它必須維持在開啟狀態，直到指定 MQMF_LAST_SEGMENT 為止。 如果未滿足此條件，則呼叫會失敗，原因碼為 MQRC_INCOMPLETE_MSG。
- 在沒有 MQMF_MSG_IN_GROUP 的情況下指定 MQMF_SEGMENT 之後， MQMF_MSG_IN_GROUP 必須保持 關閉 ，直到指定 MQMF_LAST_SEGMENT 之後為止。 如果未滿足此條件，則呼叫會失敗，原因碼為 MQRC_INCOMPLETE_MSG。

佇列上的實體順序 顯示旗標的有效組合，以及用於各種欄位的值。

這些旗標是 MQPUT 和 MQPUT1 呼叫的輸入旗標，以及 MQGET 呼叫的輸出旗標。 在後者呼叫中，佇列管理程式也會將旗標的值回應至 MQGMO 中的 *GroupStatus* 及 *SegmentStatus* 欄位。

您無法搭配使用分組或分段的訊息與「發佈/訂閱」。

預設旗標: 可以指定下列旗標，以指出訊息具有預設屬性:

無 MQMF_NONE
無訊息旗標 (預設訊息屬性)。

這會抑制分段，並指出訊息不在群組中，且不是邏輯訊息的區段。 定義 MQMF_NONE 以協助程式說明文件。 此旗標並非預期與任何其他旗標一起使用，但由於其值為零，因此無法偵測此類使用。

MsgFlags 欄位會分割成子欄位; 如需詳細資料，請參閱 第 788 頁的『報告選項及訊息旗標』。

此欄位的起始值為 MQMF_NONE。 如果 *Version* 小於 MQMD_VERSION_2，則會忽略此欄位。

MsgId (MQBYTE24)

這是用來區分不同訊息的位元組字串。 一般而言，雖然佇列管理程式不允許使用相同的訊息 ID，但沒有兩個訊息應該具有相同的訊息 ID。 訊息 ID 是訊息的永久內容，在重新啟動佇列管理程式之後會持續保存。 因為訊息 ID 是位元組字串而非字串，所以當訊息從一個佇列管理程式流向另一個佇列管理程式時，訊息 ID 不會 在字集之間轉換。

對於 MQPUT 和 MQPUT1 呼叫，如果應用程式指定 MQMI_NONE 或 MQPMO_NEW_MSG_ID，則佇列管理程式會產生唯一的訊息 ID[2] 放置訊息時，將訊息放置在隨訊息一起傳送的訊息描述子中。 佇列管理程式也會在屬於傳送端應用程式的訊息描述子中傳回此訊息 ID。 應用程式可以使用此值來記錄特定訊息的相關資訊，以及回應應用程式其他部分的查詢。

如果要將訊息放入主題，佇列管理程式會根據需要為每一個發佈的訊息產生唯一訊息 ID。 如果由應用程式指定 MQPMO_NEW_MSG_ID，則佇列管理程式會產生要在輸出時傳回的唯一訊息 ID。 如果應用程式指定 MQMI_NONE，則 MQMD 中的 *MsgId* 欄位值在從呼叫返回時不會變更。

如需保留發佈資訊的詳細資料，請參閱 第 437 頁的『MQPMO 選項 (MQLONG)』 中 MQPMO_RETAIN 的說明。

如果將訊息放置到配送清單中，則佇列管理程式會根據需要產生唯一訊息 ID，但 MQMD 中的 *MsgId* 欄位值在從呼叫返回時保持不變，即使已指定 MQMI_NONE 或 MQPMO_NEW_MSG_ID。 如果應用程式需要知道佇列管理程式所產生的訊息 ID，則應用程式必須提供包含 *MsgId* 欄位的 MQPMR 記錄。

傳送端應用程式也可以指定 MQMI_NONE 以外的訊息 ID 值; 這會停止佇列管理程式產生唯一訊息 ID。 轉遞訊息的應用程式可以使用此方法來傳播原始訊息的訊息 ID。

佇列管理程式不會使用此欄位，除了:

- 如果要求，則產生唯一值，如上述
- 將值遞送至發出訊息取得要求的應用程式
- 將值複製到它針對此訊息所產生之任何報告訊息的 *CorrelId* 欄位 (視 *Report* 選項而定)

當佇列管理程式或訊息通道代理程式產生報告訊息時，它會以原始訊息的 *Report* 欄位 (MQRO_NEW_MSG_ID 或 MQRO_PASS_MSG_ID) 指定的方式設定 *MsgId* 欄位。 產生報告訊息的應用程式也必須執行此動作。

對於 MQGET 呼叫， *MsgId* 是可用來從佇列擷取特定訊息的五個欄位之一。 一般而言， MQGET 呼叫會傳回佇列上的下一個訊息，但只要以任何組合指定五個選取準則中的一或多個，即可取得特定訊息; 這些欄位如下:

- *MsgId*
- *CorrelId*
- *GroupId*
- *MsgSeqNumber*

---

[2] 佇列管理程式所產生的 *MsgId* 包含 4 個位元組的產品 ID (ASCII 或 EBCDIC 中的 AMQ ¬ 或 CSQ，其中 ¬ 代表空白字元)，後面接著唯一字串的產品特定實作。 在 WebSphere MQ 中，這包含佇列管理程式名稱的前 12 個字元，以及衍生自系統時鐘的值。 因此，所有可交互通訊的佇列管理程式都必須具有前 12 個字元不同的名稱，以確保訊息 ID 是唯一的。 產生唯一字串的能力也取決於未向後變更的系統時鐘。 若要消除佇列管理程式所產生的訊息 ID 複製應用程式所產生的訊息 ID 的可能性，應用程式必須避免產生其起始字元在 ASCII 或 EBCDIC (X'41 '至 X'49' 及 X'C1'至 X'C9') 中介於 A 到 I 範圍內的 ID。 不過，不會阻止應用程式產生起始字元在這些範圍內的 ID。

- *Offset*

應用程式會將其中一個以上欄位設為必要值，然後在 MQGMO 中的 *MatchOptions* 欄位中設定對應的 MQMO_ * 比對選項，以使用這些欄位作為選取準則。 只有在那些欄位中具有指定值的訊息才是可供擷取的候選項。 *MatchOptions* 欄位的預設值 (如果應用程式未變更的話) 是同時符合訊息 ID 和相關性 ID。

在 z/OS 上，您可以使用的選取準則受限於用於佇列的索引類型。 如需進一步詳細資料，請參閱 *IndexType* 佇列屬性。

一般而言，傳回的訊息是佇列上滿足選取準則的 第一個 訊息。 但如果指定 MQGMO_BROWSE_NEXT，則傳回的訊息是滿足選取準則的 下一個 訊息; 此訊息的掃描會從現行游標位置 後面 的訊息開始。

註: 佇列會循序掃描以尋找符合選取準則的訊息，因此擷取時間會比未指定選取準則時來得慢，尤其是在找到適當的訊息之前必須掃描許多訊息時。 這方面的例外情況如下:

- *CorrelId* 在 64 位元分散式平台上的 MQGET 呼叫，其中 *CorrelId* 索引不需要執行真正循序掃描。
- *IndexType* 在 z/OS 上的 MQGET 呼叫。

在這兩種情況下，都會改善擷取效能。

如需如何在各種狀況中使用選取準則的相關資訊，請參閱 第 339 頁的表 506 。

指定 MQMI_NONE 作為訊息 ID 具有與指定 MQMO_MATCH_MSG_ID 不 相同的效果，即 任何 訊息 ID 都符合。

如果在 MQGET 呼叫的 *GetMsgOpts* 參數中指定 MQGMO_MSG_UNDER_CURSOR 選項，則會忽略此欄位。

從 MQGET 呼叫傳回時， *MsgId* 欄位會設為所傳回訊息的訊息 ID (如果有的話)。

可以使用下列特殊值:

MQMI_NONE
未指定訊息 ID。

欄位長度的值為二進位零。

對於 C 程式設計語言，也會定義常數 MQMI_NONE_ARRAY; 此值與 MQMI_NONE 相同，但它是字元陣列而非字串。

這是 MQGET、MQPUT 及 MQPUT1 呼叫的輸入/輸出欄位。 此欄位的長度由 MQ_MSG_ID_LENGTH 提供。 此欄位的起始值為 MQMI_NONE。

MsgSeq 號碼 *(MQLONG)*
這是群組內邏輯訊息的序號。

序號從 1 開始，並針對群組中的每一個新邏輯訊息增加 1，最多為 999 999 999 999。 不在群組中的實際訊息的序號為 1。

在下列情況下，應用程式不需要在 MQPUT 或 MQGET 呼叫上設定此欄位:

- 在 MQPUT 呼叫上，指定 MQPMO_LOGICAL_ORDER。
- 在 MQGET 呼叫上， 未 指定 MQMO_MATCH_MSG_SEQ_NUMBER。

這些是針對未報告訊息的訊息使用這些呼叫的建議方式。 不過，如果應用程式需要更多控制，或呼叫是 MQPUT1，則應用程式必須確保 *MsgSeqNumber* 設為適當的值。

在 MQPUT 和 MQPUT1 呼叫的輸入上，佇列管理程式會使用 佇列上的實體順序中說明的值。 在 MQPUT 和 MQPUT1 呼叫的輸出上，佇列管理程式會將此欄位設為隨訊息一起傳送的值。

在 MQGET 呼叫的輸入中，佇列管理程式會使用 第 339 頁的表 506 中顯示的值。 在 MQGET 呼叫的輸出上，佇列管理程式會將此欄位設為所擷取訊息的值。

此欄位的起始值為 1。 如果 *Version* 小於 MQMD_VERSION_2，則會忽略此欄位。

MsgType (MQLONG)

這指出訊息的類型。 訊息類型分組如下:

MQMT_SYSTEM_FIRST
系統定義訊息類型的最低值。

MQMT_SYSTEM_LAST
系統定義訊息類型的最高值。

下列值目前定義在系統範圍內:

MQMT_DATAGRAM
訊息是不需要回覆的訊息。

MQMT_REQUEST
訊息是需要回覆的訊息。

在 *ReplyToQ* 欄位中指定要傳送回覆的佇列名稱。 *Report* 欄位指出如何設定回覆的 *MsgId* 和 *CorrelId* 。

MQMT_REPLY
此訊息是對先前要求訊息 (MQMT_REQUEST) 的回覆。 訊息必須傳送至要求訊息的 *ReplyToQ* 欄位所指示的佇列。 使用要求的 *Report* 欄位來控制如何設定回覆的 *MsgId* 和 *CorrelId* 。

註: 佇列管理程式不會施行要求/回覆關係; 這是應用程式責任。

MQMT_REPORT
訊息報告某些預期或非預期的出現項目，通常與某些其他訊息相關 (例如，收到包含無效資料的要求訊息)。 將訊息傳送至原始訊息之訊息描述子的 *ReplyToQ* 欄位所指示的佇列。 設定 *Feedback* 欄位，以指出報告的本質。 使用原始訊息的 *Report* 欄位來控制如何設定報告訊息的 *MsgId* 和 *CorrelId* 。

佇列管理程式或訊息通道代理程式所產生的報告訊息一律會傳送至 *ReplyToQ* 佇列，並依照上述說明來設定 *Feedback* 和 *CorrelId* 欄位。

也可以使用應用程式定義的值。 它們必須在下列範圍內:

MQMT_APPL_FIRST
應用程式定義訊息類型的最低值。

MQMT_APPL_LAST
應用程式定義訊息類型的最高值。

對於 MQPUT 和 MQPUT1 呼叫， *MsgType* 值必須在系統定義的範圍或應用程式定義的範圍內; 否則，呼叫會失敗，原因碼為 MQRC_MSG_TYPE_ERROR。

這是 MQGET 呼叫的輸出欄位，以及 MQPUT 和 MQPUT1 呼叫的輸入欄位。 此欄位的起始值為 MQMT_DATAGRAM。

偏移 *(MQLONG)*

這是實體訊息中的資料從資料構成部分的邏輯訊息開始算起的偏移 (以位元組為單位)。 此資料稱為 區段。 偏移在 0 到 999 999 999 的範圍內。 非邏輯訊息區段的實體訊息偏移為零。

在下列情況下，應用程式不需要在 MQPUT 或 MQGET 呼叫上設定此欄位:

- 在 MQPUT 呼叫上，指定 MQPMO_LOGICAL_ORDER。
- 在 MQGET 呼叫上， 未 指定 MQMO_MATCH_OFFSET。

這些是針對未報告訊息的訊息使用這些呼叫的建議方式。 不過，如果應用程式不符合這些條件，或呼叫是 MQPUT1，則應用程式必須確保 *Offset* 設為適當的值。

在 MQPUT 和 MQPUT1 呼叫的輸入上，佇列管理程式會使用 佇列上的實體順序中說明的值。 在 MQPUT 和 MQPUT1 呼叫的輸出上，佇列管理程式會將此欄位設為隨訊息一起傳送的值。

對於報告邏輯訊息區段的報告訊息，會使用 *OriginalLength* 欄位 (假設它不是 MQOL_UNDEFINED) 來更新佇列管理程式所保留區段資訊中的偏移。

在 MQGET 呼叫的輸入中，佇列管理程式會使用 第 339 頁的表 506 中顯示的值。 在 MQGET 呼叫的輸出上，佇列管理程式會將此欄位設為所擷取訊息的值。

此欄位的起始值為零。 如果 *Version* 小於 MQMD_VERSION_2，則會忽略此欄位。

OriginalLength (MQLONG)

此欄位僅與屬於區段的報告訊息相關。 它指定報告訊息相關的訊息區段長度; 它不指定區段形成部分的邏輯訊息長度，或報告訊息中資料的長度。

註: 為屬於區段的訊息產生報告訊息時，佇列管理程式及訊息通道代理程式會從原始訊息複製到報告訊息的 *GroupId*、 *MsgSeqNumber*、 *Offset* 及 *MsgFlags* 欄位 MQMD。 因此，報告訊息也是區段。 產生報告訊息的應用程式必須執行相同的動作，並正確設定 *OriginalLength* 欄位。

下列是已定義的特殊值:

未定義 MQOL_UNDEFINED
未定義訊息的原始長度。

OriginalLength 是 MQPUT 和 MQPUT1 呼叫上的輸入欄位，但只有在特定情況下，才會接受應用程式提供的值:

- 如果要放置的訊息是區段，而且也是報告訊息，則佇列管理程式會接受指定的值。 值必須為:
  - 如果區段不是最後一個區段，則大於零
  - 如果區段是最後一個區段，則不小於零
  - 不小於訊息中呈現的資料長度

  如果未滿足這些條件，則呼叫會失敗，原因碼為 MQRC_ORIGINAL_LENGTH_ERROR。
- 如果要放置的訊息是區段而非報告訊息，則佇列管理程式會忽略該欄位，並改用應用程式訊息資料的長度。
- 在所有其他情況下，佇列管理程式會忽略該欄位，並改用值 MQOL_UNDEFINED。

這是 MQGET 呼叫的輸出欄位。

此欄位的起始值為 MQOL_UNDEFINED。 如果 *Version* 小於 MQMD_VERSION_2，則會忽略此欄位。

持續性 *(MQLONG)*

這指出訊息是否在系統失敗及佇列管理程式重新啟動之後仍然存在。 對於 MQPUT 和 MQPUT1 呼叫，此值必須是下列其中一項:

MQPER_PERSISTENT
訊息在系統失敗及佇列管理程式重新啟動之後仍然存在。 一旦已放置訊息，且已確定放置訊息的工作單元 (如果將訊息放置為工作單元的一部分) ，則會將訊息保留在輔助儲存體上。 除非從佇列中移除訊息，且已確定取得訊息的工作單元 (如果擷取訊息作為工作單元的一部分) ，否則它會保留在該處。

當持續訊息傳送至遠端佇列時，儲存及轉遞機制會將訊息沿著目的地的路徑保留在每一個佇列管理程式上，直到已知訊息已到達下一個佇列管理程式為止。

持續訊息無法放置在:

- 暫時動態佇列數
- 對映至 CFLEVEL (2) 或以下的 CFSTRUCT 物件，或 CFSTRUCT 物件定義為 RECOVER (NO) 的共用佇列。

持續訊息可以放置在永久動態佇列及預先定義佇列上。

MQPER_NOT_PERSISTENT
此訊息通常無法在系統失敗或佇列管理程式重新啟動時存活。 即使在佇列管理程式重新啟動時，在輔助儲存體上找到完整的訊息副本，也會如此。

在 NPMCLASS (HIGH) 的情況下，非持續訊息會在正常佇列管理程式關閉並重新啟動之後繼續存在。

在共用佇列的情況下，非持續訊息會在佇列共用群組中重新啟動，但在共用佇列上用來儲存訊息的連結機能失敗之後，仍會繼續存在。

MQPER_PERSISTENCE_AS_Q_DEF

- 如果佇列是叢集佇列，則會從 目的地 佇列管理程式中定義的 *DefPersistence* 屬性取得訊息的持續性，該佇列管理程式擁有放置訊息之佇列的特定實例。 通常，叢集佇列的所有實例都具有相同的 *DefPersistence* 屬性值，雖然這不是強制要求。

當訊息放置在目的地佇列上時， *DefPersistence* 的值會複製到 *Persistence* 欄位中。 如果隨後變更 *DefPersistence* ，則不會影響已放置在佇列上的訊息。

- 如果佇列不是叢集佇列，則即使目的地佇列管理程式是遠端，也會從 本端 佇列管理程式中定義的 *DefPersistence* 屬性取得訊息持續性。

  如果佇列名稱解析路徑中有多個定義，則會從路徑中 第一個 定義的這個屬性值取得預設持續性。 這可以是：

  – 別名佇列
  – 本端佇列
  – 遠端佇列的本端定義
  – 佇列管理程式別名
  – 傳輸佇列 (例如， *DefXmitQName* 佇列)

  放置訊息時， *DefPersistence* 的值會複製到 *Persistence* 欄位。 如果隨後變更 *DefPersistence* ，則不會影響已放置的訊息。

持續及非持續訊息都可以存在於相同的佇列中。

在回覆訊息時，應用程式必須對回覆訊息使用要求訊息的持續性。

若為 MQGET 呼叫，傳回的值為 MQPER_PERSISTENT 或 MQPER_NOT_PERSISTENT。

這是 MQGET 呼叫的輸出欄位，以及 MQPUT 和 MQPUT1 呼叫的輸入欄位。 此欄位的起始值為 MQPER_PERSISTENCE_AS_Q_DEF。

優先順序 *(MQLONG)*

對於 MQPUT 和 MQPUT1 呼叫，該值必須大於或等於零; 零是最低優先順序。 也可以使用下列特殊值:

MQPRI_PRIORITY_AS_Q_DEF

- 如果佇列是叢集佇列，則訊息的優先順序會從 *DefPriority* 屬性中取得，該屬性定義在 目的地 佇列管理程式中，該佇列管理程式擁有放置訊息之佇列的特定實例。 通常，叢集佇列的所有實例都具有相同的 *DefPriority* 屬性值，雖然這不是強制要求。

  當訊息放置在目的地佇列上時， *DefPriority* 的值會複製到 *Priority* 欄位中。 如果隨後變更 *DefPriority* ，則不會影響已放置在佇列上的訊息。
- 如果佇列不是叢集佇列，則即使目的地佇列管理程式是遠端，也會從 本端 佇列管理程式中定義的 *DefPriority* 屬性取得訊息的優先順序。

  如果佇列名稱解析路徑中有多個定義，則會從路徑中 第一個 定義的這個屬性值取得預設優先順序。 這可以是：

  – 別名佇列
  – 本端佇列
  – 遠端佇列的本端定義
  – 佇列管理程式別名
  – 傳輸佇列 (例如， *DefXmitQName* 佇列)

  放置訊息時， *DefPriority* 的值會複製到 *Priority* 欄位。 如果隨後變更 *DefPriority* ，則不會影響已放置的訊息。

MQGET 呼叫傳回的值一律大於或等於零; 永不傳回 MQPRI_PRIORITY_AS_Q_DEF 值。

如果放置的訊息優先順序高於本端佇列管理程式所支援的上限 (此上限由 *MaxPriority* 佇列管理程式屬性提供) ，則佇列管理程式會接受該訊息，但會以佇列管理程式的優先順序上限放置在佇列上; MQPUT 或 MQPUT1 呼叫會完成，並傳回 MQCC_WARNING 及原因碼 MQRC_PRIORITY_EXCEEDS_MAXIMUM。 不過， *Priority* 欄位會保留放置訊息的應用程式所指定的值。

在 z/OS 上，如果將具有 MsgSeq 號碼 1 的訊息放置在訊息遞送順序為 MQMDS_PRIORITY 且索引類型為 MQIT_GROUP_ID 的佇列中，則佇列可能會處理具有不同優先順序的訊息。 如果訊息放置在優先順序為 0 或 1 的佇列上，則會如同其優先順序為 2 一樣進行處理。 這是因為放置在這種佇列上的訊息順序已最佳化，可

啟用有效的群組完整性測試。 如需訊息遞送順序 MQMDS_PRIORITY 及索引類型 MQIT_GROUP_ID 的相關資訊，請參閱 MsgDelivery 順序屬性。

在回覆訊息時，應用程式必須對回覆訊息使用要求訊息的優先順序。 在其他狀況中，指定 MQPRI_PRIORITY_AS_Q_DEF 可以在不變更應用程式的情況下執行優先順序調整。

這是 MQGET 呼叫的輸出欄位，以及 MQPUT 和 MQPUT1 呼叫的輸入欄位。 此欄位的起始值為 MQPRI_PRIORITY_AS_Q_DEF。

PutAppl 名稱 *(MQCHAR28)*
這是放置訊息的應用程式名稱，並且是訊息 原始環境定義 的一部分。 各平台的內容不同，各版本的內容也可能不同。

如需訊息環境定義的相關資訊，請參閱 第 364 頁的『MQMD 概觀』 和 訊息環境定義 。

PutApplName 的格式視 *PutApplType* 的值而定，並且可以從一個版次變更為另一個版次。 變更很少，但會在環境變更時發生。

當佇列管理程式設定此欄位 (亦即，針對 MQPMO_SET_ALL_CONTEXT 以外的所有選項) 時，它會將欄位設為環境所決定的值:

- 在 z/OS 上，佇列管理程式會使用:
  - 對於 z/OS 批次， JES JOB 卡中 8 個字元的工作名稱
  - 若為 TSO ， 7 個字元的 TSO 使用者 ID
  - 若為 CICS，則為 8 個字元的應用程式 ID ，後面接著 4 個字元的交易 ID
  - 對於 IMS，這是 8 個字元的 IMS 系統 ID ，後面接著 8 個字元的 PSB 名稱
  - 若為 XCF ，則為 8 個字元的 XCF 群組名稱，後面接著 16 個字元的 XCF 成員名稱
  - 對於佇列管理程式所產生的訊息，佇列管理程式名稱的前 28 個字元
  - 對於不含 CICS 的分散式佇列，通道起始程式的工作名稱為 8 個字元，後面接著模組放入無法傳送郵件之佇列的 8 個字元名稱，後面接著 8 個字元的作業 ID。

  名稱每一個都會在右側以空白填補，如同欄位其餘部分中的任何空格一樣。 如果有多個名稱，則它們之間沒有分隔字元。
- 在 Windows 系統上，佇列管理程式會使用:
  - 若為 CICS 應用程式，則為 CICS 交易名稱
  - 對於非 CICS 應用程式，執行檔完整名稱的最右側 28 個字元
- 在 IBM i 上，佇列管理程式會使用完整工作名稱。
- 在 UNIX 系統上，佇列管理程式會使用:
  - 若為 CICS 應用程式，則為 CICS 交易名稱
  - 若為非 CICS 應用程式， MQ 會要求作業系統提供處理程序的名稱。 這會以程式檔案名稱傳回，不含完整路徑。 然後 MQ 會將此程序名稱放置在 MQMD.PutApplName 欄位如下:

    AIX
    如果名稱小於或等於 28 個位元組，則會插入名稱，並在右側以空格填補。

    如果名稱大於 28 個位元組，則會插入名稱最左邊的 28 個位元組。

    Linux 和 Solaris
    如果名稱小於或等於 15 個位元組，則會插入名稱，並在右側以空格填補。

    如果名稱大於 15 個位元組，則會插入名稱最左側的 15 個位元組，並在右側以空格填補。

    HP-UX
    如果名稱小於或等於 14 個位元組，則會插入名稱，並在右側以空格填補。

    如果名稱大於 14 個位元組，則會插入名稱最左側的 14 個位元組，並在右側填補空格。

    例如，如果您執行 `/opt/mqm/samp/bin/amqsput QNAME QMNAME`，則 PutAppl 名稱為 `'amqsput'`。 在此 MQCHAR28 欄位中，填補有 21 個空格字元。 請注意，包括 `/opt/mqm/samp/bin` 在內的完整路徑未包括在 PutAppl 名稱中。

對於 MQPUT 及 MQPUT1 呼叫，如果在 *PutMsgOpts* 參數中指定 MQPMO_SET_ALL_CONTEXT，則這是輸入/輸出欄位。 會捨棄欄位中空值字元之後的任何資訊。 佇列管理程式會將空值字元及任何後續字元轉換為空白。 如果未指定 MQPMO_SET_ALL_CONTEXT，則輸入時會忽略此欄位，且此欄位是僅限輸出欄位。

PutAppl 類型 *(MQLONG)*

這是放置訊息的應用程式類型，並且是訊息 **原始環境定義** 的一部分。 如需訊息環境定義的相關資訊，請參閱 第 364 頁的『MQMD 概觀』 和 訊息環境定義。

PutApplType 可以具有下列其中一種標準類型。 您也可以定義自己的類型，但只能使用 MQAT_USER_FIRST 到 MQAT_USER_LAST 範圍內的值。

MQAT_AIX
AIX 應用程式 (與 MQAT_UNIX 的值相同)。

MQ 屬性分配管理系統
BROKER.

MQAT_CICS
CICS 交易。

MQAT_CICS_BRIDGE
CICS 橋接器。

MQAT_CICS_VSE
CICS/VSE 交易。

MQAT_DOS
PC DOS 上的 WebSphere MQ MQI 用戶端應用程式。

MQAT_DQM
分散式佇列管理程式代理程式。

Mqat_guardian
Tandem Guardian 應用程式 (與 MQAT_NSK 相同值)。

MQAT_IMS
IMS 應用程式。

MQAT_IMS_BRIDGE
IMS 橋接器。

MQAT_JAVA
Java。

MQAT_MVS
MVS 或 TSO 應用程式 (與 MQAT_ZOS 的值相同)。

MQAT_NOTES_AGENT
Lotus Notes 代理程式應用程式。

MQAT_NSK
HP Integrity NonStop Server 應用程式。

MQAT_OS390
OS/390 應用程式 (與 MQAT_ZOS 相同值)。

MQAT_OS400
IBM i 應用程式。

MQAT_QMGR
佇列管理程式。

MQAT_UNIX
UNIX 應用程式。

MQAT_VOS
Stratus VOS 應用程式。

MQ 視窗
16 位元 Windows 應用程式。

MQAT_WINDOWS_NT
32 位元 Windows 應用程式。

MQAT_WLM
z/OS 工作量管理程式應用程式。

MQAT_XCF
XCF。

MQAT_ZOS
z/OS 應用程式。

MQAT_DEFAULT
預設應用程式類型。

這是應用程式執行所在平台的預設應用程式類型。

註: 此常數的值是環境特定的。 因此，一律使用適用於將執行應用程式之平台的標頭、併入或複製檔案來編譯應用程式。

MQAT_UNKNOWN
使用此值指出應用程式類型不明，即使有其他環境定義資訊。

MQAT_USER_FIRST
使用者定義應用程式類型的最低值。

MQAT_USER_LAST
使用者定義應用程式類型的最高值。

也可以出現下列特殊值:

MQAT_NO_CONTEXT
當放置不含環境定義的訊息時 (亦即，指定 MQPMO_NO_CONTEXT 環境定義選項)，佇列管理程式會設定此值。

擷取訊息時，可以針對此值測試 *PutApplType*，以決定訊息是否具有環境定義 (如果任何其他環境定義欄位非空白，則建議使用 MQPMO_SET_ALL_CONTEXT 的應用程式永不將 *PutApplType* 設為 MQAT_NO_CONTEXT)。

當佇列管理程式因應用程式放置而產生此資訊時，該欄位會設為環境所決定的值。 在 IBM i 上，它設為 MQAT_OS400; 佇列管理程式在 IBM i 上永不使用 MQAT_CICS。

對於 MQPUT 及 MQPUT1 呼叫，如果在 *PutMsgOpts* 參數中指定 MQPMO_SET_ALL_CONTEXT，則這是輸入/輸出欄位。 如果未指定 MQPMO_SET_ALL_CONTEXT，則輸入時會忽略此欄位，且此欄位是僅限輸出欄位。

這是 MQGET 呼叫的輸出欄位。 此欄位的起始值是 MQAT_NO_CONTEXT。

PutDate (MQCHAR8)

這是放置訊息的日期，並且是訊息 **原始環境定義** 的一部分。 如需訊息環境定義的相關資訊，請參閱 第 364 頁的『MQMD 概觀』 和 訊息環境定義。

當佇列管理程式產生此欄位時，用於日期的格式為:

- YYYYMMDD

其中字元代表:

YYYY
年 (四個數字)

MM
月份 (01 到 12)

DD
日 (01 至 31)

「格林威治標準時間 (GMT)」用於 *PutDate* 及 *PutTime* 欄位，受精確設為 GMT 的系統時鐘所限制。

如果將訊息放置為工作單元的一部分，則日期是放置訊息的時間，而不是確定工作單元的日期。

對於 MQPUT 及 MQPUT1 呼叫，如果在 *PutMsgOpts* 參數中指定 MQPMO_SET_ALL_CONTEXT，則這是輸入/輸出欄位。 佇列管理程式不會檢查欄位的內容，但會捨棄欄位內空值字元之後的任何資訊。 佇列管理程式會將空值字元及任何後續字元轉換為空白。 如果未指定 MQPMO_SET_ALL_CONTEXT，則輸入時會忽略此欄位，且此欄位是僅限輸出欄位。

這是 MQGET 呼叫的輸出欄位。 此欄位的長度由 MQ_PUT_DATE_LENGTH 提供。 這個欄位的起始值在 C 中是空字串，在其他程式設計語言中是 8 個空白字元。

PutTime (MQCHAR8)

這是放置訊息的時間，並且是訊息 **原始環境定義** 的一部分。 如需訊息環境定義的相關資訊，請參閱 第 364 頁的『MQMD 概觀』 和 訊息環境定義。

當佇列管理程式產生此欄位時，所使用的時間格式為:

- HHMMSSTH

其中字元代表 (依序):

HH
小時 (00 到 23)

MM
分鐘 (00 到 59)

不銹鋼
秒 (00 至 59; 請參閱附註)

T
十分之一秒 (0 到 9)

H
百分之一秒 (0 到 9)

註: 如果系統時鐘已同步至非常精確的時間標準，則在極少數情況下，可能會在 *PutTime* 中傳回 60 或 61 秒。 將閏秒插入廣域時間標準時會發生這種情況。

「格林威治標準時間 (GMT)」用於 *PutDate* 及 *PutTime* 欄位，受精確設為 GMT 的系統時鐘所限制。

如果將訊息放置為工作單元的一部分，則時間是放置訊息的時間，而不是確定工作單元的時間。

對於 MQPUT 及 MQPUT1 呼叫，如果在 *PutMsgOpts* 參數中指定 MQPMO_SET_ALL_CONTEXT，則這是輸入/輸出欄位。 佇列管理程式不會檢查欄位的內容，但會捨棄欄位內空值字元之後的任何資訊。 佇列管理程式會將空值字元及任何後續字元轉換為空白。 如果未指定 MQPMO_SET_ALL_CONTEXT，則輸入時會忽略此欄位，且此欄位是僅限輸出欄位。

這是 MQGET 呼叫的輸出欄位。 此欄位的長度由 MQ_PUT_TIME_LENGTH 提供。 這個欄位的起始值在 C 中是空字串，在其他程式設計語言中是 8 個空白字元。

ReplyToQ (MQCHAR48)

這是發出訊息取得要求的應用程式向其傳送 MQMT_REPLY 和 MQMT_REPORT 訊息的訊息佇列名稱。 名稱是在 *ReplyToQMgr* 所識別的佇列管理程式上定義的佇列本端名稱。 雖然傳送端佇列管理程式在放置訊息時不會驗證此佇列，但此佇列不能是模型佇列。

對於 MQPUT 及 MQPUT1 呼叫，如果 *MsgType* 欄位具有值 MQMT_REQUEST，或 *Report* 欄位要求任何報告訊息，則此欄位不得為空白。 不過，不論訊息類型為何，都會將指定 (或替代) 的值傳遞給發出訊息取得要求的應用程式。

如果 *ReplyToQMgr* 欄位空白，本端佇列管理程式會在其自己的佇列定義中查閱 *ReplyToQ* 名稱。 如果存在具有此名稱之遠端佇列的本端定義，則所傳輸訊息中的 *ReplyToQ* 值會取代為遠端佇列定義中的 *RemoteQName* 屬性值，且當接收端應用程式對訊息發出 MQGET 呼叫時，會在訊息描述子中傳回此值。 如果遠端佇列的本端定義不存在，則 *ReplyToQ* 保持不變。

如果指定名稱，則可以包含尾端空白; 第一個空值字元及其後的字元會被視為空白。 否則，不會檢查名稱是否滿足佇列的命名規則; 如果在傳輸的訊息中取代了 *ReplyToQ*，則對於傳輸的名稱也是如此。 唯一的檢查是已指定名稱 (如果情況需要的話)。

如果不需要回覆目的地佇列，請將 *ReplyToQ* 欄位設為空白，或 (在 C 程式設計語言中) 設為空字串，或設為一個以上後接空字元的空白; 不要讓欄位維持未起始設定。

對於 MQGET 呼叫，佇列管理程式一律會傳回欄位長度以空白填補的名稱。

如果無法遞送需要報告訊息的訊息，且報告訊息也無法遞送至指定的佇列，則原始訊息和報告訊息都會進入無法傳送的郵件 (無法遞送的訊息) 佇列 (請參閱 第 701 頁的『佇列管理程式的屬性』中說明的 *DeadLetterQName* 屬性)。

這是 MQGET 呼叫的輸出欄位，以及 MQPUT 和 MQPUT1 呼叫的輸入欄位。 此欄位的長度由 MQ_Q_NAME_LENGTH 指定。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

ReplyTo 佇列管理程式 *(MQCHAR48)*

這是要傳送回覆訊息或報告訊息的佇列管理程式名稱。 *ReplyToQ* 是在此佇列管理程式上定義之佇列的本端名稱。

如果 *ReplyToQMgr* 欄位空白，本端佇列管理程式會在其佇列定義中查閱 *ReplyToQ* 名稱。 如果存在具有此名稱之遠端佇列的本端定義，則所傳輸訊息中的 *ReplyToQMgr* 值會取代為遠端佇列定義中的 *RemoteQMgrName* 屬性值，且當接收端應用程式對訊息發出 MQGET 呼叫時，會在訊息描述子中傳回此值。 如果遠端佇列的本端定義不存在，則隨訊息一起傳輸的 *ReplyToQMgr* 是本端佇列管理程式的名稱。

如果指定名稱，則可以包含尾端空白; 第一個空值字元及其後的字元會被視為空白。 否則，不會檢查名稱是否滿足佇列管理程式的命名規則，或傳送端佇列管理程式是否知道此名稱; 如果在傳輸的訊息中取代 *ReplyToQMgr* ，則此名稱也適用於傳輸的名稱。

如果不需要回覆目的地佇列，請將 *ReplyToQMgr* 欄位設為空白，或 (在 C 程式設計語言中) 設為空字串，或設為一個以上後接空字元的空白; 不要讓欄位維持未起始設定。

對於 MQGET 呼叫，佇列管理程式一律會傳回欄位長度以空白填補的名稱。

這是 MQGET 呼叫的輸出欄位，以及 MQPUT 和 MQPUT1 呼叫的輸入欄位。 此欄位的長度由 MQ_Q_MGR_NAME_LENGTH 提供。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

報告 *(MQLONG)*

報告訊息是另一則訊息的相關訊息，用來通知應用程式與原始訊息相關的預期或非預期事件。 *Report* 欄位可讓傳送原始訊息的應用程式指定需要哪些報告訊息、是否要將應用程式訊息資料併入其中，以及 (針對報告和回覆) 如何設定報告或回覆訊息中的訊息和相關性 ID。 可以要求下列任何或所有 (或無) 類型的報告訊息:

- 異常狀況
- 期限
- 到達時確認 (COA)
- 交貨時確認 (COD)
- 正面動作通知 (PAN)
- 負面動作通知 (NAN)

如果需要多種類型的報告訊息，或需要其他報告選項，則值可以是:

- 加在一起 (不要多次新增相同的常數) ，或
- 利用執行位元 OR 運算來組合（如果程式設計語言支援位元運算）。

接收報告訊息的應用程式可以檢查 MQMD 中的 *Feedback* 欄位，以判斷產生報告的原因; 如需詳細資料，請參閱 *Feedback* 欄位。

將訊息放置到主題時使用報告選項，可能會導致產生零個、一個或多個報告訊息，並將其傳送到應用程式。這是因為發佈訊息可能會傳送至零個、一個或多個訂閱應用程式。

異常狀況選項: 指定列出的其中一個選項，以要求異常狀況報告訊息。

MQRO_Exception
當訊息傳送至另一個佇列管理程式，且訊息無法遞送至指定的目的地佇列時，訊息通道代理程式會產生這種類型的報告。 例如，目的地佇列或中間傳輸佇列可能已滿，或訊息可能對佇列而言太大。

產生異常狀況報告訊息取決於原始訊息的持續性，以及原始訊息經過的訊息通道速度 (正常或快速):

- 對於所有持續訊息，以及透過正常訊息通道傳送的非持續訊息， 只有 在傳送端應用程式針對錯誤狀況指定的動作可以順利完成時，才會產生異常狀況報告。 傳送端應用程式可以指定下列其中一個動作，以在發生錯誤狀況時控制原始訊息的處置:
  - MQRO_DEAD_LETTER_Q (這會將原始訊息放置在無法傳送郵件的佇列上)。
  - MQRO_DISCARD_MSG (這會捨棄原始訊息)。

  如果傳送端應用程式指定的動作無法順利完成，則原始訊息會保留在傳輸佇列上，且不會產生異常狀況報告訊息。
- 對於透過快速訊息通道傳送的非持續訊息，會從傳輸佇列中移除原始訊息，並 即使 無法順利完成針對錯誤狀況指定的動作，也會產生異常狀況報告。 例如，如果指定 MQRO_DEAD_LETTER_Q ，但無法將原始訊息放置在無法傳送郵件的佇列上，因為該佇列已滿，則會產生異常狀況報告訊息，並捨棄原始訊息。

  如需一般及快速訊息通道的相關資訊，請參閱 非持續訊息速度 (NPMSPEED) 。

如果放置原始訊息的應用程式可以透過 MQPUT 或 MQPUT1 呼叫所傳回的原因碼同步收到問題通知，則不會產生異常狀況報告。

應用程式也可以傳送異常狀況報告，以指出無法處理訊息 (例如，因為它是借方交易，會導致帳戶超出其貸方限制)。

報告訊息不包含來自原始訊息的訊息資料。

請勿指定多個 MQRO_EXCEPTION、MQRO_EXCEPTION_WITH_DATA 及 MQRO_EXCEPTION_WITH_FULL_DATA。

MQRO_EXCEPTION_WITH_DATA
這與 MQRO_EXCEPTION 相同，不同之處在於報告訊息中包含原始訊息中應用程式訊息資料的前 100 個位元組。 如果原始訊息包含一個以上 MQ 標頭結構，則除了 100 個位元組的應用程式資料之外，還會包含在報告訊息中。

請勿指定多個 MQRO_EXCEPTION、MQRO_EXCEPTION_WITH_DATA 及 MQRO_EXCEPTION_WITH_FULL_DATA。

MQRO_EXCEPTION_WITH_FULL_DATA
需要具有完整資料的異常狀況報告。

這與 MQRO_EXCEPTION 相同，不同之處在於原始訊息中的所有應用程式訊息資料都包含在報告訊息中。

請勿指定多個 MQRO_EXCEPTION、MQRO_EXCEPTION_WITH_DATA 及 MQRO_EXCEPTION_WITH_FULL_DATA。

到期選項: 指定列出的其中一個選項，以要求到期報告訊息。

MQRO_EXPIRATION
如果在遞送至應用程式之前捨棄訊息，則佇列管理程式會產生此類型的報告，因為其到期時間已過 (請參閱 *Expiry* 欄位)。 如果未設定此選項，則在基於此原因捨棄訊息時 (即使您指定其中一個 MQRO_EXCEPTION_ * 選項) ，不會產生任何報告訊息。

報告訊息不包含來自原始訊息的訊息資料。

請勿指定多個 MQRO_EXPIRATION、MQRO_EXPIRATION_WITH_DATA 及 MQRO_EXPIRATION_WITH_FULL_DATA。

MQRO_EXPIRATION_WITH_DATA
這與 MQRO_EXPIRATION 相同，不同之處在於原始訊息中的前 100 個位元組應用程式訊息資料包括在報告訊息中。 如果原始訊息包含一個以上 MQ 標頭結構，則除了 100 個位元組的應用程式資料之外，還會包含在報告訊息中。

請勿指定多個 MQRO_EXPIRATION、MQRO_EXPIRATION_WITH_DATA 及 MQRO_EXPIRATION_WITH_FULL_DATA。

MQRO_EXPIRATION_WITH_FULL_DATA
這與 MQRO_EXPIRATION 相同，只是原始訊息中的所有應用程式訊息資料都包含在報告訊息中。

請勿指定多個 MQRO_EXPIRATION、MQRO_EXPIRATION_WITH_DATA 及 MQRO_EXPIRATION_WITH_FULL_DATA。

確認到達時選項: 指定列出的其中一個選項，以要求確認到達時報告訊息。

MQRO_COA
當訊息放置在目的地佇列上時，擁有目的地佇列的佇列管理程式會產生這種類型的報告。 報告訊息不包含來自原始訊息的訊息資料。

如果將訊息放置為工作單元的一部分，且目的地佇列是本端佇列，則只有在確定工作單元時，才能擷取佇列管理程式所產生的 COA 報告訊息。

如果訊息描述子中的 *Format* 欄位是 MQFMT_XMIT_Q_HEADER 或 MQFMT_DEAD_LETTER_HEADER，則不會產生 COA 報告。 如果訊息放置在傳輸佇列上，或無法遞送並放置在無法傳送郵件的佇列上，則這會防止產生 COA 報告。

如果是 IMS 橋接器佇列，則會在訊息到達 IMS 佇列 (從 IMS 收到確認通知) 時產生 COA 報告，而不是在訊息放入 MQ 橋接器佇列時產生。 這表示如果 IMS 非作用中，則在 IMS 啟動且訊息排入 IMS 佇列之前，不會產生任何 COA 報告。

執行使用 MQMD.Report= MQRO_COA 必須對回覆佇列具有 + passid 權限。 如果使用者沒有 + passid 權限，則 COA 報告訊息不會到達回覆佇列。 嘗試將報告訊息放置在無法傳送的郵件佇列上。

請勿指定 MQRO_COA、MQRO_COA_WITH_DATA 及 MQRO_COA_WITH_FULL_DATA 中的多個。

MQRO_COA_WITH_DATA
這與 MQRO_COA 相同，不同之處在於報告訊息中包含來自原始訊息的應用程式訊息資料的前 100 個位元組。 如果原始訊息包含一個以上 MQ 標頭結構，則除了 100 個位元組的應用程式資料之外，還會包含在報告訊息中。

請勿指定 MQRO_COA、MQRO_COA_WITH_DATA 及 MQRO_COA_WITH_FULL_DATA 中的多個。

MQ Ro_COA_WITH_FULL_DATA
這與 MQRO_COA 相同，不同之處在於原始訊息中的所有應用程式訊息資料都包含在報告訊息中。

請勿指定 MQRO_COA、MQRO_COA_WITH_DATA 及 MQRO_COA_WITH_FULL_DATA 中的多個。

確認遞送中選項: 指定列出的其中一個選項，以要求確認遞送中報告訊息。

MQRO_COD
當應用程式以從佇列中刪除訊息的方式從目的地佇列中擷取訊息時，佇列管理程式會產生這種類型的報告。 報告訊息不包含來自原始訊息的訊息資料。

如果擷取訊息作為工作單元的一部分，則會在相同的工作單元內產生報告訊息，因此在確定工作單元之前無法使用報告。 如果工作單元已取消，則不會傳送報告。

如果使用 MQGMO_MARK_SKIP_BACKOUT 選項擷取訊息，則不一定會產生 COD 報告。 如果已取消主要工作單元，但已確定次要工作單元，則會從佇列中移除訊息，但不會產生 COD 報告。

如果訊息描述子中的 *Format* 欄位是 MQFMT_DEAD_LETTER_HEADER，則不會產生 COD 報告。 這可防止在訊息無法遞送並置於無法傳送郵件的佇列時產生 COD 報告。

如果目的地佇列是 XCF 佇列，則 MQRO_COD 無效。

請勿指定 MQRO_COD、MQRO_COD_WITH_DATA 及 MQRO_COD_WITH_FULL_DATA 中的多個。

MQRO_COD_WITH_DATA
這與 MQRO_COD 相同，只不過原始訊息中的前 100 個位元組應用程式訊息資料包含在報告訊息中。 如果原始訊息包含一個以上 MQ 標頭結構，則除了 100 個位元組的應用程式資料之外，還會包含在報告訊息中。

如果在 MQGET 呼叫中指定原始訊息的 MQGMO_ACCEPT_TRUNCATED_MSG，且擷取的訊息已截斷，則放置在報告訊息中的應用程式訊息資料量取決於環境:

- 在 z/OS 上，它是下列項目的最小值:
  - 原始訊息的長度
  - 用來擷取訊息的緩衝區長度
  - 100 位元組。
- 在其他環境中，它是下列項目的最低值:
  - 原始訊息的長度
  - 100 位元組。

如果目的地佇列是 XCF 佇列，則 MQRO_COD_WITH_DATA 無效。

請勿指定 MQRO_COD、MQRO_COD_WITH_DATA 及 MQRO_COD_WITH_FULL_DATA 中的多個。

MQRO_COD_WITH_FULL_DATA
這與 MQRO_COD 相同，不同之處在於原始訊息中的所有應用程式訊息資料都包含在報告訊息中。

如果目的地佇列是 XCF 佇列，則 MQRO_COD_WITH_FULL_DATA 無效。

請勿指定 MQRO_COD、MQRO_COD_WITH_DATA 及 MQRO_COD_WITH_FULL_DATA 中的多個。

動作通知選項: 指定一或兩個列出的選項，以要求接收端應用程式傳送正面動作或負面動作報告訊息。

MQRO_PAN
此類型的報告是由擷取訊息並對其採取動作的應用程式所產生。 它指出已順利執行訊息中所要求的動作。 產生報告的應用程式會決定是否要將任何資料併入報告中。

除了將此要求傳送至擷取訊息的應用程式之外，佇列管理程式不會根據此選項採取任何動作。 如果適當的話，擷取應用程式必須產生報告。

MQRO_NAN
此類型的報告是由擷取訊息並對其採取動作的應用程式所產生。 它指出訊息中所要求的動作 未 順利執行。 產生報告的應用程式會決定是否要將任何資料併入報告中。 例如，您可能想要包含一些資料，指出無法執行要求的原因。

除了將此要求傳送至擷取訊息的應用程式之外，佇列管理程式不會根據此選項採取任何動作。 如果適當的話，擷取應用程式必須產生報告。

應用程式必須決定哪些條件對應於正面動作，哪些條件對應於負面動作。 不過，如果只局部執行要求，則會產生 NAN 報告，而不是 PAN 報告 (如果要求的話)。 每個可能的條件都必須對應於正面動作或負面動作，但不能同時對應兩者。

訊息 ID 選項: 指定其中一個列出的選項，以控制如何設定報告訊息 (或回覆訊息) 的 *MsgId* 。

MQRO_NEW_MSG_ID
這是預設動作，並指出如果產生報告或回覆作為此訊息的結果，則會針對報告或回覆訊息產生新的 *MsgId* 。

MQRO_PASS_MSG_ID
如果產生報告或回覆作為此訊息的結果，則此訊息的 *MsgId* 會複製到報告或回覆訊息的 *MsgId* 。

對於每一個接收發佈副本的訂閱者，發佈訊息的 *MsgId* 將會不同，因此每一個複製到報告或回覆訊息的 *MsgId* 將會不同。

如果未指定此選項，則會採用 MQRO_NEW_MSG_ID。

相關性 ID 選項: 指定列出的其中一個選項，以控制如何設定報告訊息 (或回覆訊息) 的 *CorrelId* 。

MQRO_COPY_MSG_ID_TO_CORREL_ID
這是預設動作，指出如果由於此訊息而產生報告或回覆，則此訊息的 *MsgId* 會複製到報告或回覆訊息的 *CorrelId* 。

對於每一個接收發佈副本的訂閱者，發佈訊息的 *MsgId* 將會不同，因此每一個報告或回覆訊息的 *MsgId* 複製到 *CorrelId* 將會不同。

MQRO_PASS_CORREL_ID
如果產生報告或回覆作為此訊息的結果，則此訊息的 *CorrelId* 會複製到報告或回覆訊息的 *CorrelId* 。

除非訂閱者使用 MQSO_SET_CORREL_ID 選項，並將 MQSD 中的 SubCorrelID 欄位設為 MQCI_NONE，否則發佈訊息的 *CorrelId* 是訂閱者特有的。 因此，每一個複製到報告或回覆訊息的 *CorrelId* 中的 *CorrelId* 可能各不相同。

如果未指定此選項，則會採用 MQRO_COPY_MSG_ID_TO_CORREL_ID。

回覆要求或產生報告訊息的伺服器必須檢查原始訊息中是否已設定 MQRO_PASS_MSG_ID 或 MQRO_PASS_CORREL_ID 選項。 如果是的話，伺服器必須採取針對那些選項所說明的動作。 如果都未設定，伺服器必須採取對應的預設動作。

處置選項: 指定列出的其中一個選項，以控制原始訊息在無法遞送至目的地佇列時的處置。 應用程式可以設定處置選項，與要求異常狀況報告無關。

MQRO_DEAD_LETTER_Q
這是預設動作，如果無法將訊息遞送至目的地佇列，則會將訊息放置在無法傳送郵件的佇列上。 在下列情況下會發生這種情況:

- 當放置原始訊息的應用程式無法透過 MQPUT 或 MQPUT1 呼叫所傳回的原因碼同步收到問題通知時。 如果傳送者要求異常狀況報告訊息，則會產生異常狀況報告訊息。
- 放置原始訊息的應用程式放置到主題時

MQRO_DISCARD_MSG
如果訊息無法遞送至目的地佇列，則會捨棄該訊息。 在下列情況下會發生這種情況:

- 當放置原始訊息的應用程式無法透過 MQPUT 或 MQPUT1 呼叫所傳回的原因碼同步收到問題通知時。 如果傳送者要求異常狀況報告訊息，則會產生異常狀況報告訊息。
- 放置原始訊息的應用程式放置到主題時

如果您要將原始訊息傳回給寄件者，而未將原始訊息放置在無法傳送郵件的佇列上，則寄件者必須指定 MQRO_DISCARD_MSG 與 MQRO_EXCEPTION_WITH_FULL_DATA。

MQRO_PASS_DISCARD_AND_EXPIRY
如果在訊息上設定此選項，並因此產生報告或回覆，則報告的訊息描述子會繼承:

- MQRO_DISCARD_MSG (如果已設定的話)。
- 訊息的剩餘到期時間 (如果這不是到期報告)。 如果這是到期報告，則到期時間設為 60 秒。

活動選項

MQRO_ACTIVITY
使用此值可讓您在整個佇列管理程式網路中追蹤 **任何** 訊息的路徑。 可在任何現行使用者訊息上指定報告選項，立即可讓您開始透過網路計算訊息的路徑。

如果產生訊息的應用程式無法開啟活動報告，則可以使用佇列管理程式管理者所提供的跨結束程式來開啟報告。

註:

1. 網路中能夠產生活動報告的佇列管理程式越少，路徑越詳細。
2. 可能很難以正確的順序放置活動報告，以判定所採取的路徑。
3. 活動報告可能找不到通往其所要求目的地的路徑。
4. 任何佇列管理程式都必須接受具有此報告選項集的訊息，即使它們不瞭解該選項也一樣。 這可讓您在任何使用者訊息上設定報告選項，即使它們是由非 6.0 版或更新版本的佇列管理程式所處理。
5. 如果處理程序 (佇列管理程式或使用者處理程序) 對具有此選項集的訊息執行活動，則它可以選擇產生並放置活動報告。

預設選項: 如果不需要報告選項，請指定下列選項:

MQRO_NONE
使用此值可指出尚未指定其他選項。 MQRO_NONE 定義為輔助程式說明文件。 此選項並非預期與任何其他選項搭配使用，但由於其值為零，因此無法偵測到此類使用。

一般資訊:

1. 傳送原始訊息的應用程式必須明確要求所有必要的報告類型。 例如，如果要求 COA 報告但未要求異常狀況報告，則會在將訊息置於目的地佇列時產生 COA 報告，但在訊息到達目的地佇列時，如果目的地佇列已滿，則不會產生異常狀況報告。 如果未設定 *Report* 選項，則佇列管理程式或訊息通道代理程式 (MCA) 不會產生任何報告訊息。

   即使本端佇列管理程式無法辨識部分報告選項，也可以指定這些選項; 當 目的地 佇列管理程式要處理此選項時，這很有用。 如需詳細資料，請參閱 第 788 頁的『報告選項及訊息旗標』 。

   如果要求報告訊息，則必須在 *ReplyToQ* 欄位中指定要將報告傳送至其中的佇列名稱。 當收到報告訊息時，可以檢查訊息描述子中的 *Feedback* 欄位來判斷報告的本質。
2. 如果產生報告訊息的佇列管理程式或 MCA 無法將報告訊息放置在回覆佇列上 (例如，因為回覆佇列或傳輸佇列已滿) ，則會將報告訊息放置在無法傳送郵件的佇列上。 如果該 也 失敗，或沒有無法傳送郵件的佇列，則所採取的動作取決於報告訊息的類型:
   - 如果報告訊息是異常狀況報告，則產生異常狀況報告的訊息會留在其傳輸佇列中; 這可確保訊息不會遺失。
   - 對於所有其他報告類型，會捨棄報告訊息，並正常地繼續處理。 這是因為原始訊息已安全遞送 (針對 COA 或 COD 報告訊息) ，或不再感興趣 (針對到期報告訊息)。

   一旦報告訊息順利放置在佇列 (目的地佇列或中間傳輸佇列) 上，該訊息就不再受到特殊處理; 就如同任何其他訊息一樣。
3. 產生報告時，會開啟 *ReplyToQ* 佇列，並使用 *UserIdentifier* 的權限將報告訊息放置在導致報告之訊息的 MQMD 中，但下列情況除外:
   - 接收 MCA 所產生的異常狀況報告，會以 MCA 在嘗試放置造成報告的訊息時所使用的任何權限來放置。
   - 當在產生報告的佇列管理程式上放置導致報告的訊息時，佇列管理程式所產生的 COA 報告會以任何權限放置。 例如，如果接收 MCA 使用 MCA 的使用者 ID 放置訊息，則佇列管理程式會使用 MCA 的使用者 ID 放置 COA 報告。

   產生報告的應用程式必須使用它們用來產生回覆的相同權限; 這通常是原始訊息中使用者 ID 的權限。

   如果報告必須傳送至遠端目的地，傳送者和接收者可以決定是否接受它，就像他們對其他訊息一樣。
4. 如果要求含有資料的報告訊息:
   - 報告訊息一律以原始訊息傳送者所要求的資料量來產生。 如果報告訊息對回覆佇列而言太大，則會進行上述處理; 報告訊息永不截斷，以適合回覆佇列。
   - 如果原始訊息的 *Format* 是 MQFMT_XMIT_Q_HEADER ，則報告中包含的資料不包括 MQXQH。 報告資料以原始訊息中 MQXQH 以外資料的第一個位元組開始。 不論佇列是否為傳輸佇列，都會發生這種情況。
5. 如果在回覆佇列中收到 COA、COD 或到期報告訊息，則可保證原始訊息已送達、已遞送或已過期 (視情況而定)。 不過，如果要求一或多個這些報告訊息，但 未 收到，則無法假設反向，因為可能發生下列其中一項:
   a. 因為鏈結已關閉，所以保留報告訊息。
   b. 因為在中間傳輸佇列或回覆佇列中存在封鎖狀況 (例如，佇列已滿或禁止放置) ，所以會保留報告訊息。
   c. 報告訊息位於無法傳送郵件的佇列上。
   d. 當佇列管理程式嘗試產生報告訊息時，它既無法將它放置在適當的佇列上，也無法放置在無法傳送郵件的佇列上，因此無法產生報告訊息。
   e. 在所報告的動作 (到達、遞送或到期) 與產生對應的報告訊息之間，發生佇列管理程式失敗。 (如果應用程式擷取工作單元內的原始訊息，則 COD 報告訊息不會發生這種情況，因為 COD 報告訊息是在相同工作單元內產生。)

   基於上述原因 1、2 及 3 ，可以使用相同方式保留異常狀況報告訊息。 不過，當 MCA 無法產生異常狀況報告訊息 (報告訊息無法放置在回覆佇列或無法傳送郵件的佇列上) 時，原始訊息會保留在傳送端的傳輸佇列上，且通道會關閉。 不論是否要在通道的傳送端或接收端產生報告訊息，都會發生此情況。
6. 如果暫時封鎖原始訊息 (導致產生異常狀況報告訊息，並將原始訊息放置在無法傳送郵件的佇列上) ，則會清除封鎖，然後應用程式會從無法傳送郵件的佇列讀取原始訊息，並將它重新放置到其目的地，可能會發生下列情況:

- 即使已產生異常狀況報告訊息，原始訊息最終仍會順利到達其目的地。
- 針對單一原始訊息會產生多個異常狀況報告訊息，因為原始訊息稍後可能會遇到另一個封鎖。

放置到主題時報告訊息:

1. 將訊息放置到主題時可以產生報告。 此訊息將傳送給主題的所有訂閱者，可能是零、一或多個。 在選擇使用報告選項時，應該考慮這一點，因為結果可能會產生許多報告訊息。
2. 將訊息放入主題時，可能會有許多要提供訊息副本的目的地佇列。 如果其中部分目的地佇列有問題 (例如佇列已滿) ，則 MQPUT 的順利完成取決於 NPMSGDLV 或 PMSGDLV 的設定 (視訊息的持續性而定)。 如果設定為訊息遞送至目的地佇列必須成功 (例如，它是可延續訂閱者的持續訊息，且 PMSGDLV 設為 ALL 或 ALLDURR) ，則成功定義為符合下列其中一項準則:
   - 順利放入訂閱者佇列
   - 如果訂閱者佇列無法取得訊息，則使用 MQRO_DEAD_LETTER_Q 並順利放入無法傳送郵件的佇列
   - 如果訂閱者佇列無法取得訊息，則使用 MQRO_DISCARD_MSG。

報告訊息區段的訊息:

1. 可以針對容許分段的訊息要求報告訊息 (請參閱 MQMF_SEGMENTATION_ALLOWED 旗標的說明)。 如果佇列管理程式發現需要將訊息分段，則可以針對後續遇到相關條件的每一個區段產生報告訊息。 應用程式必須準備接收所要求的每一種報告訊息類型的多個報告訊息。 使用報告訊息中的 *GroupId* 欄位，將多個報告與原始訊息的群組 ID 產生關聯，並由 *Feedback* 欄位識別每一個報告訊息的類型。
2. 如果使用 MQGMO_LOGICAL_ORDER 來擷取區段的報告訊息，請注意後續 MQGET 呼叫可能會傳回 不同類型 的報告。 例如，如果針對佇列管理程式所分段的訊息同時要求 COA 及 COD 報告，則報告訊息的 MQGET 呼叫可能會以無法預期的方式交錯傳回 COA 及 COD 報告訊息。 請使用 MQGMO_COMPLETE_MSG 選項 (選擇性地搭配 MQGMO_ACCEPT_TRUNCATED_MSG) 來避免此情況。 MQGMO_COMPLETE_MSG 會導致佇列管理程式重新組合具有相同報告類型的報告訊息。 例如，第一個 MQGET 呼叫可能會重新組合與原始訊息相關的所有 COA 訊息，第二個 MQGET 呼叫可能會重新組合所有 COD 訊息。 第一個重新組合的類型取決於佇列上第一個出現的報告訊息類型。
3. 本身放置區段的應用程式可以為每一個區段指定不同的報告選項。 不過，請注意下列要點:
   - 如果使用 MQGMO_COMPLETE_MSG 選項來擷取區段，則佇列管理程式只會允許使用 第一個 區段中的報告選項。
   - 如果逐一擷取區段，且大部分區段都有其中一個 MQRO_COD_ * 選項，但至少有一個區段沒有，則您無法使用 MQGMO_COMPLETE_MSG 選項來擷取具有單一 MQGET 呼叫的報告訊息，或使用 MQGMO_ALL_SEGMENTS_AVAILABLE 選項來偵測所有報告訊息何時到達。
4. 在 MQ 網路中，佇列管理程式可以具有不同的功能。 如果區段的報告訊息是由不支援分段的佇列管理程式或 MCA 所產生，依預設，佇列管理程式或 MCA 不會在報告訊息中包含必要的區段資訊，這可能會導致難以識別導致產生報告的原始訊息。 請使用報告訊息來要求資料，即指定適當的 MQRO_ * _WITH_DATA 或 MQRO_ * _WITH_FULL_DATA 選項，以避免此困難。 不過，請注意，如果指定 MQRO_ * _WITH_DATA ，且報告訊息是由不支援分段的佇列管理程式或 MCA 所產生，則 可能會傳回少於 100 個位元組的應用程式訊息資料給擷取報告訊息的應用程式。

報告訊息的訊息描述子內容: 當佇列管理程式或訊息通道代理程式 (MCA) 產生報告訊息時，它會將訊息描述子中的欄位設為下列值，然後以正常方式放置訊息。

MQMD 中的欄位	**使用的值**
StrucId	MQMD_STRUC_ID
Version	MQMD_VERSION_2
Report	MQRO_NONE
MsgType	MQMT_REPORT
Expiry	MQEI_UNLIMITED
Feedback	適合報告的本質 (MQFB_COA、MQFB_COD、MQFB_EXPIRATION 或 MQRC_ * 值)
Encoding	從原始訊息描述子複製

MQMD 中的欄位	使用的值
CodedCharSetId	從原始訊息描述子複製
Format	從原始訊息描述子複製
Priority	從原始訊息描述子複製
Persistence	從原始訊息描述子複製
MsgId	如原始訊息描述子中的報告選項所指定
CorrelId	如原始訊息描述子中的報告選項所指定
BackoutCount	0
ReplyToQ	空白
ReplyToQMgr	佇列管理程式的名稱
UserIdentifier	如 MQPMO_PASS_IDENTITY_CONTEXT 選項所設定
AccountingToken	如 MQPMO_PASS_IDENTITY_CONTEXT 選項所設定
ApplIdentityData	如 MQPMO_PASS_IDENTITY_CONTEXT 選項所設定
PutApplType	MQAT_QMGR，或適用於訊息通道代理程式
PutApplName	佇列管理程式名稱或訊息通道代理程式名稱的前 28 個位元組。 對於 IMS 橋接器所產生的報告訊息，此欄位包含訊息相關之 IMS 系統的 XCF 群組名稱及 XCF 成員名稱。
PutDate	傳送報告訊息的日期
PutTime	傳送報告訊息的時間
ApplOriginData	空白
GroupId	從原始訊息描述子複製
MsgSeqNumber	從原始訊息描述子複製
Offset	從原始訊息描述子複製
MsgFlags	從原始訊息描述子複製
OriginalLength	如果不是 MQOL_UNDEFINED，則從原始訊息描述子複製，否則設為原始訊息資料的長度

建議產生報告的應用程式設定類似的值，但下列項目除外:

- *ReplyToQMgr* 欄位可以設為空白 (放置訊息時，佇列管理程式會將此變更為本端佇列管理程式的名稱)。
- 使用已用於回覆的選項 (通常是 MQPMO_PASS_IDENTITY_CONTEXT) 來設定環境定義欄位。

分析報告欄位: *Report* 欄位包含子欄位; 因此，需要檢查訊息傳送者是否要求特定報告的應用程式必須使用第 789 頁的『分析報告欄位』中說明的其中一項技術。

這是 MQGET 呼叫的輸出欄位，以及 MQPUT 和 MQPUT1 呼叫的輸入欄位。 此欄位的起始值為 MQRO_NONE。

StrucId (MQCHAR4)

這是結構 ID，且必須是:

MQMD_STRUC_ID
訊息描述子結構的 ID。

對於 C 程式設計語言，也會定義常數 MQMD_STRUC_ID_ARRAY; 這與 MQMD_STRUC_ID 具有相同的值，但它是字元陣列而非字串。

這一律是輸入欄位。 此欄位的起始值是 MQMD_STRUC_ID。

UserIdentifier (MQCHAR12)

這是訊息 **身分環境定義** 的一部分。 如需訊息環境定義的相關資訊，請參閱 第 364 頁的『MQMD 概觀』 和 訊息環境定義 。

UserIdentifier 指定產生訊息之應用程式的使用者 ID。 佇列管理程式會將此資訊視為字元資料，但不會定義其格式。

收到訊息之後，請使用後續 MQOPEN 或 MQPUT1 呼叫之 *ObjDesc* 參數的 *AlternateUserId* 欄位中的 *UserIdentifier* ，來執行 *UserIdentifier* 使用者的授權檢查，而不是執行開啟的應用程式。

當佇列管理程式為 MQPUT 或 MQPUT1 呼叫產生此資訊時:

- 在 z/OS 上，如果指定 MQOO_ALTERNATE_USER_AUTHORITY 或 MQPMO_ALTERNATE_USER_AUTHORITY 選項，則佇列管理程式會使用來自 MQOPEN 或 MQPUT1 呼叫之 *ObjDesc* 參數的 *AlternateUserId* 。 如果未指定相關選項，則佇列管理程式會使用從環境判定的使用者 ID。
- 在其他環境中，佇列管理程式一律使用從環境決定的使用者 ID。

從環境判定使用者 ID 時:

- 在 z/OS 上，佇列管理程式會使用:
  - 對於 MVS (批次) ，來自 JES JOB 卡或已啟動作業的使用者 ID
  - 對於 TSO ，使用者 ID 在工作提交期間延伸到工作
  - 對於 CICS，與作業相關聯的使用者 ID
  - 對於 IMS，使用者 ID 視應用程式類型而定:

    - 時間長度:

      - 非訊息 BMP 區域
      - 非訊息 IFP 區域
      - 未 發出成功 GU 呼叫的訊息 BMP 及訊息 IFP 區域

      佇列管理程式會使用來自區域 JES JOB 卡的使用者 ID 或 TSO 使用者 ID。 如果這些是空白或空值，則會使用程式規格區塊 (PSB) 的名稱。

    - 時間長度:

      - 已 發出成功 GU 呼叫的訊息 BMP 及訊息 IFP 區域
      - MPP 區域

      佇列管理程式使用下列其中一項:

      - 與訊息相關聯的登入使用者 ID
      - 邏輯終端機 (LTERM) 名稱
      - 來自區域 JES JOB 卡的使用者 ID
      - TSO 使用者 ID
      - PSB 名稱
- 在 IBM i 上，佇列管理程式會使用與應用程式工作相關聯的使用者設定檔名稱。
- 在 UNIX 系統上，佇列管理程式會使用:
  - 應用程式的登入名稱
  - 程序的有效使用者 ID (如果沒有可用的登入)
  - 與交易相關聯的使用者 ID (如果應用程式是 CICS 交易)
- 在 Windows 系統上，佇列管理程式會使用已登入使用者名稱的前 12 個字元。

此欄位通常是由佇列管理程式所產生的輸出欄位，但對於 MQPUT 或 MQPUT1 呼叫，您可以將此欄位設為輸入/輸出欄位，並指定 UserIdentification 欄位，而不是讓佇列管理程式產生此資訊。 如果您不想要佇列管理程式為 MQPUT 或 MQPUT1 呼叫產生 UserIdentifier 欄位，請在 PutMsgOpts 參數中指定

MQPMO_SET_IDENTITY_CONTEXT 或 MQPMO_SET_ALL_CONTEXT，並在 UserIdentifier 欄位中指定使用者 ID。

對於 MQPUT 及 MQPUT1 呼叫，如果在 *PutMsgOpts* 參數中指定 MQPMO_SET_IDENTITY_CONTEXT 或 MQPMO_SET_ALL_CONTEXT，則這是輸入/輸出欄位。 會捨棄欄位中空值字元之後的任何資訊。 佇列管理程式會將空值字元及任何後續字元轉換為空白。 如果未指定 MQPMO_SET_IDENTITY_CONTEXT 或 MQPMO_SET_ALL_CONTEXT，則輸入時會忽略此欄位，且此欄位是僅限輸出的欄位。

順利完成 MQPUT 或 MQPUT1 呼叫之後，此欄位包含在將訊息放入佇列時隨訊息一起傳輸的 *UserIdentifier*。 這將是隨訊息保留的 *UserIdentifier* 值 (如需保留的發佈資訊的詳細資料，請參閱 MQPMO_RETAIN 的說明)，但在將訊息當作發佈資訊傳送給訂閱者時不會用作 *UserIdentifier*，因為它們提供值來置換傳送給訂閱者的所有發佈資訊中的 *UserIdentifier*。 如果訊息沒有環境定義，則欄位會完全空白。

這是 MQGET 呼叫的輸出欄位。 此欄位的長度由 MQ_USER_ID_LENGTH 提供。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 12 個空白字元。

版本 *(MQLONG)*

這是結構版本號碼，且必須是下列其中一項:

MQMD_VERSION_1
Version-1 訊息描述子結構。

所有環境都支援此版本。

MQMD_VERSION_2
Version-2 訊息描述子結構。

所有 WebSphere MQ V6.0 及更新版本環境中都支援此版本，以及連接至這些系統的 WebSphere MQ MQI 用戶端。

註: 使用 version-2 MQMD 時，佇列管理程式會對可能存在於應用程式訊息資料開頭的任何 MQ 標頭結構執行其他檢查; 如需進一步詳細資料，請參閱 MQPUT 呼叫的用法注意事項。

僅存在於結構最新版本中的欄位會在欄位說明中如此識別。 下列常數指定現行版本的版本號碼:

MQMD_CURRENT_VERSION
訊息描述子結構的現行版本。

這一律是輸入欄位。 此欄位的起始值為 MQMD_VERSION_1。

MQMD 的起始值及語言宣告

表 *515: MQMD for MQMD* 中欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQMD_STRUC_ID	'MD'
Version	MQMD_VERSION_1	1
Report	MQRO_NONE	0
MsgType	MQMT_DATAGRAM	8
Expiry	MQEI_UNLIMITED	-1
Feedback	MQFB_NONE	0
Encoding	MQENC_NATIVE	取決於環境
CodedCharSetId	MQCCSI_Q_MGR	0
Format	MQFMT_NONE	空白
Priority	MQPRI_PRIORITY_AS_Q_DEF	-1
Persistence	MQPER_PERSISTENCE_AS_Q_DEF	2

表 515: *MQMD for MQMD* 中欄位的起始值 (繼續)

欄位名稱	常數名稱	常數值
MsgId	MQMI_NONE	空值
CorrelId	MQCI_NONE	空值
BackoutCount	無	0
ReplyToQ	無	空字串或空白
ReplyToQMgr	無	空字串或空白
UserIdentifier	無	空字串或空白
AccountingToken	MQACT_NONE	空值
ApplIdentityData	無	空字串或空白
PutApplType	MQAT_NO_CONTEXT	0
PutApplName	無	空字串或空白
PutDate	無	空字串或空白
PutTime	無	空字串或空白
ApplOriginData	無	空字串或空白
GroupId	MQGI_NONE	空值
MsgSeqNumber	無	1
Offset	無	0
MsgFlags	無 MQMF_NONE	0
OriginalLength	未定義 MQOL_UNDEFINED	-1

附註:

1. 空值字串或空白值表示 C 中的空值字串，以及其他程式設計語言中的空白字元。
2. 在 C 程式設計語言中，巨集變數 MQMD_DEFAULT 包含上面列出的值。 您可以用下列方式來提供結構中欄位的起始值:

```
MQMD MyMD = {MQMD_DEFAULT};
```

C 宣告

```
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 */
};
```

COBOL 宣告

```
**   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.
```

PL/I 宣告

```
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 ReplyToQMgr      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 */
```

High Level Assembler 宣告

```
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)
```

Visual Basic 宣告

```
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
```

MQMDE-訊息描述子延伸

下表彙總結構中的欄位。

表 *516: MQMDE* 中的欄位

欄位	說明	主題
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
StrucLength	MQMDE 結構的長度	StrucLength
Encoding	MQMDE 之後的資料的數字編碼	編碼
CodedCharSetId	MQMDE 之後的資料的字集 ID	CodedCharSetId
Format	MQMDE 之後的資料的格式名稱	格式
Flags	一般旗標	旗標

表 516: *MQMDE* 中的欄位 (繼續)		
欄位	**說明**	**主題**
GroupId	群組 ID	GroupId
MsgSeqNumber	群組內的邏輯訊息的序號	MsgSeqNumber
Offset	從邏輯訊息開始的實體訊息中的資料偏移	偏移
MsgFlags	訊息旗標	MsgFlags
OriginalLength	原始訊息的長度	OriginalLength

MQMDE 概觀

可用性: 所有 WebSphere MQ 系統，以及連接至這些系統的 WebSphere MQ 用戶端。

目的:MQMDE 結構說明有時在應用程式訊息資料之前出現的資料。 此結構包含存在於 version-2 MQMD 中，但不存在於 version-1 MQMD 中的那些 MQMD 欄位。

格式名稱:MQFMT_MD_EXTENSION。

字集及編碼:MQMDE 中的資料必須是本端佇列管理程式的字集及編碼; 這些是由 *CodedCharSetId* 佇列管理程式屬性及 C 程式設計語言的 MQENC_NATIVE 所提供。

在下列欄位中，將 MQMDE 的字集及編碼設為 *CodedCharSetId* 及 *Encoding* 欄位:

- MQMD (如果 MQMDE 結構是在訊息資料的開頭) ，或
- MQMDE 結構之前的標頭結構 (所有其他案例)。

如果 MQMDE 不在佇列管理程式的字集和編碼中，則會接受 MQMDE ，但不允許使用，亦即會將 MQMDE 視為訊息資料。

註: 在 Windows 上，使用 Micro Focus COBOL 編譯的應用程式會使用不同於佇列管理程式編碼的 MQENC_NATIVE 值。 雖然 MQPUT、 MQPUT1 及 MQGET 呼叫的 MQMD 結構中的數值欄位必須採用 Micro Focus COBOL 編碼，但 MQMDE 結構中的數值欄位必須採用佇列管理程式的編碼。 後者是由 C 程式設計語言的 MQENC_NATIVE 所提供，值為 546。

用法: 使用 version-2 MQMD 的應用程式將不會遇到 MQMDE 結構。 不過，特殊化應用程式以及繼續使用 version-1 MQMD 的應用程式在某些狀況下可能會遇到 MQMDE。 在下列情況下可能會發生 MQMDE 結構:

- 在 MQPUT 和 MQPUT1 呼叫上指定
- 由 MQGET 呼叫傳回
- 在傳輸佇列上的訊息中

MQPUT 及 MQPUT1 呼叫上指定的 MQMDE: 在 MQPUT 及 MQPUT1 呼叫上，如果應用程式提供 version-1 MQMD ，則應用程式可以選擇性地以 MQMDE 作為訊息資料的字首，將 MQMD 中的 *Format* 欄位設為 MQFMT_MD_EXTENSION ，以指出 MQMDE 存在。 如果應用程式未提供 MQMDE ，則佇列管理程式會採用 MQMDE 中欄位的預設值。 佇列管理程式使用的預設值與結構的起始值相同; 請參閱 第 409 頁的表 518。

如果應用程式提供 version-2 MQMD ，且 會以 MQMDE 作為應用程式訊息資料的字首，則會依照 第 406 頁的表 517 所示來處理結構。

表 517: 在 *MQPUT* 或 *MQPUT1 for MQMDE* 上指定 *MQMDE* 時的佇列管理程式動作			
MQMD 版本	**version-2 欄位的值**	**MQMDE 中對應欄位的值**	**佇列管理程式所採取的動作**
1	-	有效	允許使用 MQMDE
2	預設值	有效	允許使用 MQMDE
2	非預設值	有效	將 MQMDE 視為訊息資料
1 或 2	任意	無效	通話失敗，帶有適當的原因碼

表 517: 在 *MQPUT* 或 *MQPUT1 for MQMDE* 上指定 *MQMDE* 時的佇列管理程式動作 (繼續)			
MQMD 版本	**version-2 欄位的值**	**MQMDE 中對應欄位的值**	**佇列管理程式所採取的動作**
1 或 2	任意	MQMDE 使用錯誤的字集或編碼，或是不受支援的版本	將 MQMDE 視為訊息資料
註: 在 z/OS 上，如果應用程式使用 MQMDE 指定 version-1 MQMD ，則只有在佇列具有 MQIT_GROUP_ID 的 *IndexType* 時，佇列管理程式才會驗證 MQMDE。			

有一個特殊情況。 如果應用程式使用 version-2 MQMD 來放置屬於區段的訊息 (亦即，已設定 MQMF_SEGMENT 或 MQMF_LAST_SEGMENT 旗標) ，且 MQMD 中的格式名稱是 MQFMT_DEAD_LETTER_HEADER ，則佇列管理程式會產生 MQMDE 結構，並在 MQDLH 結構及其後面的資料 之間 插入它。 在佇列管理程式保留訊息的 MQMD 中， version-2 欄位會設為其預設值。

存在於 version-2 MQMD 中但不存在於 version-1 MQMD 的數個欄位是 MQPUT 及 MQPUT1 上的輸入/輸出欄位。 不過，在 MQPUT 及 MQPUT1 呼叫的輸出上，佇列管理程式 不會 在 MQMDE 的對等欄位中傳回任何值; 如果應用程式需要這些輸出值，則必須使用 version-2 MQMD。

MQGET 呼叫傳回的 MQMDE: 在 MQGET 呼叫上，如果應用程式提供 version-1 MQMD ，則佇列管理程式會以 MQMDE 作為傳回訊息的字首，但只有在 MQMDE 中的一個以上欄位具有非預設值時。 佇列管理程式會將 MQMD 中的 *Format* 欄位設為值 MQFMT_MD_EXTENSION ，以指出 MQMDE 存在。

如果應用程式在 *Buffer* 參數啟動時提供 MQMDE ，則會忽略 MQMDE。 從 MQGET 呼叫返回時，它會由訊息的 MQMDE 取代 (如果需要的話) ，或由應用程式訊息資料改寫 (如果不需要 MQMDE)。

如果 MQGET 呼叫傳回 MQMDE ，則 MQMDE 中的資料通常是在佇列管理程式的字集及編碼中。 不過，在下列情況下， MQMDE 可能採用其他字集及編碼:

- MQMDE 被視為 MQPUT 或 MQPUT1 呼叫上的資料 (請參閱 第 406 頁的表 517 ，以瞭解可能導致此情況的情況)。
- 從 TCP 連線所連接的遠端佇列管理程式收到訊息，且未正確設定接收訊息通道代理程式 (MCA)。

註: 在 Windows 上，使用 Micro Focus COBOL 編譯的應用程式使用不同於佇列管理程式編碼的 MQENC_NATIVE 值 (請參閱上文)。

傳輸佇列上訊息中的 MQMDE: 傳輸佇列上訊息的字首為 MQXQH 結構，其中包含 version-1 MQMD。 MQMDE 也可能存在，位於 MQXQH 結構與應用程式訊息資料之間，但通常只有在 MQMDE 中的一或多個欄位具有非預設值時才會存在。

MQXQH 結構與應用程式訊息資料之間也可能出現其他 MQ 標頭結構。 例如，當出現無法傳送郵件的標頭 MQDLH ，且訊息不是區段時，順序為:

- MQXQH (包含 version-1 MQMD)
- MQMDE
- MQDLH
- 應用程式訊息資料

MQMDE 的欄位

MQMDE 結構包含下列欄位; 這些欄位按 **字母順序**進行說明:

CodedCharSetId (MQLONG)

這會指定遵循 MQMDE 結構之資料的字集 ID; 它不適用於 MQMDE 結構本身中的字元資料。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。 佇列管理程式不會檢查此欄位是否有效。 可以使用下列特殊值:

MQCCSI_INHERIT
此結構 後面 的資料中的字元資料與此結構的字集相同。

佇列管理程式會將訊息中所傳送結構的這個值變更為結構的實際字集 ID。 如果沒有發生錯誤， MQGET 呼叫不會傳回值 MQCCSI_INHERIT。

如果 MQMD 中的 *PutApplType* 欄位值為 MQAT_BROKER ，則無法使用 MQCCSI_INHERIT。

此值在下列環境中受支援: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows，以及連接至這些系統的 WebSphere MQ 用戶端。

此欄位的起始值為 MQCCSI_UNDEFINED。

編碼 *(MQLONG)*

這指定遵循 MQMDE 結構之資料的數值編碼; 它不適用於 MQMDE 結構本身中的數值資料。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。 佇列管理程式不會檢查欄位是否有效。 如需資料編碼的相關資訊，請參閱 第 363 頁的『MQMD-訊息描述子』 中說明的 *Encoding* 欄位。

此欄位的起始值為 MQENC_NATIVE。

旗標 *(MQLONG)*

可以指定下列旗標:

MQMDEF_NONE
沒有旗標。

此欄位的起始值是 MQMDEF_NONE。

格式 *(MQCHAR8)*

這會指定遵循 MQMDE 結構之資料的格式名稱。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。 佇列管理程式不會檢查此欄位是否有效。 如需格式名稱的相關資訊，請參閱 第 363 頁的『MQMD-訊息描述子』 中說明的 *Format* 欄位。

此欄位的起始值為 MQFMT_NONE。

GroupId (MQBYTE24)

請參閱 第 363 頁的『MQMD-訊息描述子』中說明的 *GroupId* 欄位。 此欄位的起始值是 MQGI_NONE。

MsgFlags (MQLONG)

請參閱 第 363 頁的『MQMD-訊息描述子』中說明的 *MsgFlags* 欄位。 此欄位的起始值為 MQMF_NONE。

MsgSeq 號碼 *(MQLONG)*

請參閱 第 363 頁的『MQMD-訊息描述子』中說明的 *MsgSeqNumber* 欄位。 此欄位的起始值為 1。

偏移 *(MQLONG)*

請參閱 第 363 頁的『MQMD-訊息描述子』中說明的 *Offset* 欄位。 此欄位的起始值為 0。

OriginalLength (MQLONG)

請參閱 第 363 頁的『MQMD-訊息描述子』中說明的 *OriginalLength* 欄位。 此欄位的起始值為 MQOL_UNDEFINED。

StrucId (MQCHAR4)

值必須為:

MQMDE_STRUC_ID
訊息描述子延伸結構的 ID。

對於 C 程式設計語言，也會定義常數 MQMDE_STRUC_ARRAY; 此值與 MQMDE_STRUC_ID 相同，但卻是字元陣列而非字串。

此欄位的起始值是 MQMDE_STRUC_ID。

StrucLength (MQLONG)

這是 MQMDE 結構的長度; 已定義下列值:

MQMDE_LENGTH_2
version-2 訊息描述子延伸結構的長度。

此欄位的起始值為 MQMDE_LENGTH_2。

版本 *(MQLONG)*

這是結構版本號碼; 值必須是:

MQMDE_VERSION_2
Version-2 訊息描述子延伸結構。

下列常數指定現行版本的版本號碼:

MQMDE_CURRENT_VERSION
訊息描述子延伸結構的現行版本。

此欄位的起始值為 MQMDE_VERSION_2。

MQMDE 的起始值和語言宣告

表 *518: MQMDE* 的 *MQMDE* 中欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQMDE_STRUC_ID	'MDE¬'
Version	MQMDE_VERSION_2	2
StrucLength	MQMDE_LENGTH_2	72
Encoding	MQENC_NATIVE	取決於環境
CodedCharSetId	未定義 MQCCSI_UNDEFINED	0
Format	MQFMT_NONE	空白
Flags	MQMDEF_NONE	0
GroupId	MQGI_NONE	空值
MsgSeqNumber	無	1
Offset	無	0
MsgFlags	無 MQMF_NONE	0
OriginalLength	未定義 MQOL_UNDEFINED	-1

附註:

1. 符號 ¬ 代表單一空白字元。
2. 在 C 程式設計語言中，巨集變數 MQMDE_DEFAULT 包含上面列出的值。 您可以用下列方式來提供結構中欄位的起始值:

```
MQMDE MyMDE = {MQMDE_DEFAULT};
```

C 宣告

```
typedef struct tagMQMDE MQMDE;
struct tagMQMDE {
```

```
  MQCHAR4   StrucId;         /* Structure identifier */
  MQLONG    Version;         /* Structure version number */
  MQLONG    StrucLength;     /* 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 */
};
```

COBOL 宣告

```
**   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-STRUCLENGTH    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.
```

PL/I 宣告

```
dcl
 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 */
```

High Level Assembler 宣告

```
MQMDE                DSECT
```

```
MQMDE_STRUCID         DS   CL4   Structure identifier
MQMDE_VERSION         DS   F     Structure version number
MQMDE_STRUCLENGTH     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)
```

Visual Basic 宣告

```
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
```

MQMHBO-緩衝區選項的訊息控點

下表彙總結構中的欄位。 MQMHBO 結構-緩衝區選項的訊息控點

表 *519: MQMHBO* 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
Options	控制 MQMHBUF 動作的選項	選項

MQMHBO 概觀

可用性: 所有 WebSphere MQ 系統及 WebSphere MQ MQI 用戶端。

目的:MQMHBO 結構可讓應用程式指定選項，以控制如何從訊息控點產生緩衝區。 此結構是 MQMHBUF 呼叫上的輸入參數。

字集及編碼:MQMHBO 中的資料必須位於應用程式的字集及應用程式的編碼 (MQENC_NATIVE)。

MQMHBO 的欄位

緩衝區選項結構欄位的訊息控點

MQMHBO 結構包含下列欄位; 這些欄位按 **字母順序**說明:

選項 *(MQLONG)*
緩衝區選項結構的訊息控點-選項欄位

這些選項控制 MQMHBUF 的動作。

您必須指定下列選項:

MQMHBO_PROPERTIES_IN_MQRFH2
將內容從訊息控點轉換成緩衝區時，請將它們轉換成 MQRFH2 格式。

您也可以選擇性地指定下列值。 如果必要值可以是:

- 加在一起 (不要多次新增相同的常數) ，或
- 利用執行位元 OR 運算來組合（如果程式設計語言支援位元運算）。

MQMHBO_DELETE_PROPERTIES
新增至緩衝區的內容會從訊息控點中刪除。 如果呼叫失敗，則不會刪除任何內容。

這一律是輸入欄位。 此欄位的起始值是 MQMHBO_PROPERTIES_IN_MQRFH2。

StrucId (MQCHAR4)
緩衝區選項結構的訊息控點- StrucId 欄位

這是結構 ID。 值必須為:

MQMHBO_STRUC_ID
緩衝區選項結構的訊息控點 ID。

對於 C 程式設計語言，也會定義常數 MQMHBO_STRUC_ID_ARRAY; 此值與 MQMHBO_STRUC_ID 相同，但卻是字元陣列而非字串。

這一律是輸入欄位。 此欄位的起始值是 MQMHBO_STRUC_ID。

版本 *(MQLONG)*
緩衝區選項結構的訊息控點-版本欄位

這是結構版本號碼。 值必須為:

MQMHBO_VERSION_1
訊息控點至緩衝區選項結構的版本號碼。

下列常數指定現行版本的版本號碼:

MQMHBO_CURRENT_VERSION
訊息控點至緩衝區選項結構的現行版本。

這一律是輸入欄位。 此欄位的起始值為 MQMHBO_VERSION_1。

MQMHBO 的起始值和語言宣告
緩衝區結構的訊息控點-起始值

表 *520: MQMHBO* 中欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQMHBO_STRUC_ID	'MHBO'
Version	MQMHBO_VERSION_1	1
Options	MQMHBO_PROPERTIES_IN_MQRFH2	

表 *520: MQMHBO* 中欄位的起始值 *(*繼續*)*

欄位名稱	常數名稱	常數值

附註:

1. 空值字串或空白值表示 C 中的空值字串，以及其他程式設計語言中的空白字元。
2. 在 C 程式設計語言中，巨集變數 MQMHBO_DEFAULT 包含上述值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQMHBO MyMHBO = {MQMHBO_DEFAULT};
```

C 宣告
緩衝區選項結構的訊息控點-C 語言宣告

```
typedef struct tagMQMHBO MQMHBO;
struct tagMQMHBO {
  MQCHAR4  StrucId;        /* Structure identifier */
  MQLONG   Version;        /* Structure version number */
  MQLONG   Options;        /* Options that control the action of
                              MQMHBUF */
};
```

COBOL 宣告
訊息控點至緩衝區選項結構-COBOL 語言宣告

```
**   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.
```

PL/I 宣告
緩衝區選項結構的訊息控點-PL/I 語言宣告

```
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 */
```

High Level Assembler 宣告
訊息控點至緩衝區選項結構-組譯器語言宣告

```
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)
```

MQOD-物件描述子

下表彙總結構中的欄位。

欄位	說明	主題
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
ObjectType	物件類型	ObjectType
ObjectName	物件名稱	ObjectName
ObjectQMgrName	物件佇列管理程式名稱	ObjectQMgrName
DynamicQName	動態佇列名稱	DynamicQName
AlternateUserId	替代使用者 ID	AlternateUserid
註: 如果 *Version* 小於 MQOD_VERSION_2，則會忽略其餘欄位。		
RecsPresent	存在的物件記錄數	RecsPresent
KnownDestCount	順利開啟的本端佇列數	KnownDestCount
UnknownDestCount	順利開啟的遠端佇列數	UnknownDestCount
InvalidDestCount	無法開啟的佇列數	InvalidDestCount
ObjectRecOffset	從 MQOD 開始的第一個物件記錄的偏移	ObjectRec 偏移
ResponseRecOffset	從 MQOD 開始的第一個回應記錄的偏移	ResponseRec 偏移
ObjectRecPtr	第一個物件記錄的位址	ObjectRecPtr
ResponseRecPtr	第一個回應記錄的位址	ResponseRecPtr
註: 如果 *Version* 小於 MQOD_VERSION_3，則會忽略其餘欄位。		
AlternateSecurityId	替代安全 ID	AlternateSecurityId
ResolvedQName	解析的佇列名稱	ResolvedQName
ResolvedQMgrName	解析的佇列管理程式名稱	ResolvedQMgr 名稱
註: 如果 *Version* 小於 MQOD_VERSION_4，則會忽略其餘欄位。		
ObjectString	長物件名稱	ObjectString
SelectionString	選擇字串	SelectionString
ResObjectString	已解析長物件名稱	ResObject 字串
ResolvedType	已解析物件類型	ResolvedType

MQOD 概觀

可用性: 所有 WebSphere MQ 系統，以及連接至那些系統的 WebSphere MQ MQI 用戶端。

目的:MQOD 結構用來依名稱指定物件。 有效的物件類型如下:

- 佇列或配送清單
- 名稱清單
- 程序定義
- 佇列管理程式
- 主題

此結構是 MQOPEN 和 MQPUT1 呼叫上的輸入/輸出參數。

版本:MQOD 的現行版本是 MQOD_VERSION_4。 您要在數個環境之間連接的應用程式必須確保所有相關環境都支援所需版本的 MQOD。 僅存在於結構最新版本中的欄位在接下來的說明中如此識別。

為支援的程式設計語言提供的標頭、COPY 和 INCLUDE 檔案包含環境支援的 MQOD 最新版本，但 *Version* 欄位的起始值設為 MQOD_VERSION_1。 若要使用不在 version-1 結構中的欄位，應用程式必須將 *Version* 欄位設為所需版本的版本號碼。

若要開啟發佈清單， *Version* 必須是 MQOD_VERSION_2 或更高版本。

字集和編碼:MQOD 中的資料必須採用 *CodedCharSetId* 佇列管理程式屬性所提供的字集，以及 MQENC_NATIVE 所提供本端佇列管理程式的編碼。 不過，如果應用程式以 MQ MQI 用戶端身分執行，則該結構必須採用用戶端的字集及編碼。

MQOD 的欄位

MQOD 結構包含下列欄位; 這些欄位按 **字母順序**說明:

AlternateSecurityID (MQBYTE40)

這是隨 *AlternateUserId* 傳遞至授權服務的安全 ID，容許執行適當的授權檢查。 只有在下列情況下，才會使用 *AlternateSecurityId* :

- MQOPEN 呼叫上指定 MQOO_ALTERNATE_USER_AUTHORITY，或
- 在 MQPUT1 呼叫上指定 MQPMO_ALTERNATE_USER_AUTHORITY，

及 *AlternateUserId* 欄位不是完全空白，直到第一個空值字元或欄位結尾。

在 Windows 上， *AlternateSecurityId* 可用來提供 Windows 安全 ID (SID)，以唯一識別 *AlternateUserId*。 透過使用 LookupAccountName() Windows API 呼叫，可以從 Windows 系統取得使用者的 SID。

在 z/OS 上，會忽略這個欄位。

AlternateSecurityId 欄位具有下列結構:

- 第一個位元組是二進位整數，包含下列有效資料的長度; 該值會排除長度位元組本身。 如果沒有安全 ID，則長度為零。
- 第二個位元組指出存在的安全 ID 類型; 可能的值如下:

  MQSIDT_NT_SECURITY_ID
  Windows 安全 ID。

  無 MQSIDT_NONE
  沒有安全 ID。
- 第三個及後續的位元組，直到第一個位元組所定義的長度為止，包含安全 ID 本身。
- 欄位中的剩餘位元組會設為二進位零。

您可以使用下列特殊值:

MQSID_NONE
未指定安全 ID。

欄位長度的值為二進位零。

對於 C 程式設計語言，也會定義常數 MQSID_NONE_ARRAY; 此值與 MQSID_NONE 相同，但卻是字元陣列而非字串。

這是輸入欄位。 此欄位的長度由 MQ_SECURITY_ID_LENGTH 提供。 此欄位的起始值是 MQSID_NONE。 如果 *Version* 小於 MQOD_VERSION_3，則會忽略此欄位。

AlternateUserID (MQCHAR12)

如果您指定 MQOPEN 呼叫的 MQOO_ALTERNATE_USER_AUTHORITY，或指定 MQPUT1 呼叫的 MQPMO_ALTERNATE_USER_AUTHORITY，則此欄位包含替代使用者 ID，用來檢查開啟的授權，以取代目前執行應用程式的使用者 ID。 不過，部分檢查仍以現行使用者 ID 執行 (例如，環境定義檢查)。

如果指定 MQOO_ALTERNATE_USER_AUTHORITY 或 MQPMO_ALTERNATE_USER_AUTHORITY，且此欄位完全空白，直到第一個空值字元或欄位結尾，則只有在不需要使用者授權即可使用指定的選項開啟此物件時，開啟才會成功。

如果未指定 MQOO_ALTERNATE_USER_AUTHORITY 或 MQPMO_ALTERNATE_USER_AUTHORITY，則會忽略此欄位。

指出的環境中存在下列差異:

- 在 z/OS 上，只會使用 *AlternateUserId* 的前 8 個字元來檢查開啟的授權。 不過，現行使用者 ID 必須獲得授權，才能指定這個特定的替代使用者 ID; 這項檢查會使用替代使用者 ID 的所有 12 個字元。 使用者 ID 只能包含外部安全管理程式所容許的字元。

  如果為佇列指定了 *AlternateUserId*，則在放置訊息時，佇列管理程式隨後可以使用該值。 如果 MQPUT 或 MQPUT1 呼叫上指定的 MQPMO_ * _CONTEXT 選項會導致佇列管理程式產生身分環境定義資訊，則佇列管理程式會將 *AlternateUserId* 放置在訊息 MQMD 的 *UserIdentifier* 欄位中，以取代現行使用者 ID。
- 在其他環境中， *AlternateUserId* 僅用於對所開啟物件的存取控制檢查。 如果物件是佇列，則 *AlternateUserId* 不會影響使用該佇列控點所傳送訊息的 MQMD 中 *UserIdentifier* 欄位的內容。

這是輸入欄位。 此欄位的長度由 MQ_USER_ID_LENGTH 提供。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 12 個空白字元。

DynamicQName (MQCHAR48)

這是要由 MQOPEN 呼叫建立的動態佇列名稱。 這只有在 *ObjectName* 指定模型佇列的名稱時才會相關; 在所有其他情況下， *DynamicQName* 會被忽略。

名稱中有效的字元與 *ObjectName* 中有效的字元相同，但星號也有效。 如果 *ObjectName* 是模型佇列的名稱，則空白的名稱 (或在第一個空值字元之前只出現空白的名稱) 無效。

如果名稱中的最後一個非空白字元是星號 (*)，佇列管理程式會以字串取代星號，以保證為佇列產生的名稱在本端佇列管理程式中是唯一的。 為了容許足夠的字元數目，星號僅在位置 1 到 33 中有效。 星號後面不得有空白或空字元以外的字元。

在第一個字元位置中出現星號是有效的，在此情況下，名稱只包含佇列管理程式所產生的字元。

在 z/OS 上，請勿在第一個字元位置使用星號的名稱，因為在自動產生完整名稱的佇列上不會進行安全檢查。

這是輸入欄位。 此欄位的長度由 MQ_Q_NAME_LENGTH 指定。 此欄位的起始值由環境決定:

- 在 z/OS 上，值為 'CSQ.*'。
- 在其他平台上，此值為 'AMQ.*'。

此值是 C 中以空值結尾的字串，以及其他程式設計語言中以空白填補的字串。

InvalidDest 計數 *(MQLONG)*

這是配送清單中無法順利開啟的佇列數。 如果存在的話，當開啟不在配送清單中的單一佇列時，也會設定這個欄位。

註: If present, this field is set 僅 if the *CompCode* parameter on the MQOPEN or MQPUT1 call is MQCC_OK or MQCC_WARNING; it is 不 set if the *CompCode* parameter is MQCC_FAILED.

這是輸出欄位。 此欄位的起始值為 0。 如果 *Version* 小於 MQOD_VERSION_1，則會忽略此欄位。

KnownDest 計數 *(MQLONG)*

這是配送清單中解析為本端佇列且已順利開啟的佇列數。 計數不包括解析為遠端佇列的佇列 (即使最初使用本端傳輸佇列來儲存訊息)。 如果存在的話，當開啟不在配送清單中的單一佇列時，也會設定這個欄位。

這是輸出欄位。 此欄位的起始值為 0。 如果 *Version* 小於 MQOD_VERSION_1，則會忽略此欄位。

ObjectName (MQCHAR48)

這是 *ObjectQMgrName* 所識別之佇列管理程式上定義的物件本端名稱。 名稱可以包含下列字元:

- 大寫英文字母 (A 到 Z)
- 小寫英文字母 (a 到 z)
- 數字 (0 到 9)
- 句點 (.)、正斜線 (/)、底線 (_)、百分比 (%)

名稱不能包含前導或內含空白，但可以包含尾端空白。 請使用空值字元來指出名稱中有效資料的結尾; 空值及其後面的任何字元都會被視為空白。 下列限制適用於指出的環境:

- 在使用 EBCDIC Katakana 的系統上，無法使用小寫字元。
- 在 z/OS 上:
  - 避免名稱以底線開頭或結尾; 作業和控制台無法處理它們。
  - 百分比字元對 RACF 具有特殊意義。 如果使用 RACF 作為外部安全管理程式，則名稱不得包含百分比。 如果有的話，當使用 RACF 通用設定檔時，這些名稱不會包含在任何安全檢查中。
- 在 IBM i 上，當在指令上指定時，包含小寫字元、正斜線或百分比的名稱必須以引號括住。 對於在結構中作為欄位或在呼叫中作為參數出現的名稱，不得指定這些引號。

完整主題名稱可以從兩個不同的欄位來建置: *ObjectName* 和 *ObjectString*。 如需如何使用這兩個欄位的詳細資料，請參閱 第 503 頁的『使用主題字串』。

下列要點適用於指出的物件類型:

- 如果 *ObjectName* 是模型佇列的名稱，則佇列管理程式會使用模型佇列的屬性來建立動態佇列，並在 *ObjectName* 欄位中傳回所建立佇列的名稱。 模型佇列只能在 MQOPEN 呼叫上指定; 模型佇列在 MQPUT1 呼叫上無效。
- 如果 *ObjectName* 是具有 TARGTYPE (TOPIC) 之別名佇列的名稱，則會先對指名的別名佇列進行安全檢查; 當使用別名佇列時，這是正常的。 當安全檢查順利完成時， MQOPEN 呼叫會繼續，且其行為與 MQOT_TOPIC 上的 MQOPEN 呼叫一樣; 這包括對管理主題物件進行安全檢查。
- 如果 *ObjectName* 和 *ObjectQMgrName* 識別本端佇列管理程式所屬的佇列共用群組所擁有的共用佇列，則本端佇列管理程式上也不能有同名的佇列定義。 如果有這類定義 (本端佇列、別名佇列、遠端佇列或模型佇列) ，則呼叫會失敗，原因碼為 MQRC_OBJECT_NOT_UNIQUE。
- 如果開啟的物件是配送清單 (亦即， *RecsPresent* 存在且大於零) ，則 *ObjectName* 必須是空白或空字串。 如果未滿足此條件，則呼叫會失敗，原因碼為 MQRC_OBJECT_NAME_ERROR。
- 如果 *ObjectType* 是 MQOT_Q_MGR ，則適用特殊規則; 在此情況下，名稱必須完全空白，直到第一個空值字元或欄位結尾。

當 *ObjectName* 是模型佇列的名稱時，這是 MQOPEN 呼叫的輸入/輸出欄位，在所有其他情況下則是僅限輸入欄位。 此欄位的長度由 MQ_Q_NAME_LENGTH 指定。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

ObjectQMgr 名稱 *(MQCHAR48)*

這是在其中定義 *ObjectName* 物件的佇列管理程式名稱。 名稱中的有效字元與 *ObjectName* 的有效字元相同 (請參閱 第 416 頁的『ObjectName (MQCHAR48)』)。 直到第一個空值字元或欄位結尾都是完全空白的名稱，表示應用程式所連接的佇列管理程式 (本端佇列管理程式)。

下列要點適用於指出的物件類型:

- 如果 *ObjectType* 是 MQOT_TOPIC、MQOT_NAMELIST、MQOT_PROCESS 或 MQOT_Q_MGR ，則 *ObjectQMgrName* 必須是空白或本端佇列管理程式的名稱。
- 如果 *ObjectName* 是模型佇列的名稱，則佇列管理程式會使用模型佇列的屬性來建立動態佇列，並在 *ObjectQMgrName* 欄位中傳回在其上建立佇列的佇列管理程式名稱; 這是本端佇列管理程式的名稱。 模型佇列只能在 MQOPEN 呼叫上指定; 模型佇列在 MQPUT1 呼叫上無效。
- 如果 *ObjectName* 是叢集佇列的名稱，而 *ObjectQMgrName* 是空白，則佇列管理程式 (或叢集工作量結束程式，如果已安裝的話) 會選擇使用 MQOPEN 呼叫所傳回的佇列控點所傳送訊息的目的地，如下所示:
  - 如果指定 MQOO_BIND_ON_OPEN ，則佇列管理程式會在處理 MQOPEN 呼叫時選取叢集佇列的特定實例，並將使用此佇列控點放置的所有訊息傳送至該實例。

– 如果指定 MQOO_BIND_NOT_FIXED，則佇列管理程式可以針對使用此佇列控點的每一個後續 MQPUT 呼叫，選擇目的地佇列的不同實例 (位於叢集中的不同佇列管理程式上)。

如果應用程式需要將訊息傳送至叢集佇列的 特定 實例 (亦即，位於叢集中特定佇列管理程式上的佇列實例)，則應用程式必須在 *ObjectQMgrName* 欄位中指定該佇列管理程式的名稱。 這會強制本端佇列管理程式將訊息傳送至指定的目的地佇列管理程式。

- 如果 *ObjectName* 是共用佇列的名稱，由本端佇列管理程式所屬的佇列共用群組所擁有，*ObjectQMgrName* 可以是佇列共用群組的名稱、本端佇列管理程式的名稱或空白; 訊息會放置在相同佇列中的任何指定值。

  佇列共用群組僅在 z/OS 上受支援。
- 如果 *ObjectName* 是遠端佇列共用群組所擁有的共用佇列名稱 (亦即，本端佇列管理程式 不屬於 的佇列共用群組)，則 *ObjectQMgrName* muat 是佇列共用群組的名稱。 您可以使用屬於該群組的佇列管理程式名稱，但當訊息到達佇列共用群組時，如果該特定佇列管理程式無法使用，這可能會延遲訊息。
- 如果要開啟的物件是配送清單 (即 *RecsPresent* 大於零)，則 *ObjectQMgrName* 必須是空白或空字串。 如果未滿足此條件，則呼叫會失敗，原因碼為 MQRC_OBJECT_Q_MGR_NAME_ERROR。

當 *ObjectName* 是模型佇列的名稱時，這是 MQOPEN 呼叫的輸入/輸出欄位，在所有其他情況下則是僅限輸入欄位。 此欄位的長度由 MQ_Q_MGR_NAME_LENGTH 提供。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

ObjectRec 偏移 *(MQLONG)*

這是從 MQOD 結構開始算起第一個 MQOR 物件記錄的偏移 (以位元組為單位)。 偏移可以是正數或負數。 只有在開啟配送清單時，才會使用 *ObjectRecOffset* 。 如果 *RecsPresent* 為零，則會忽略該欄位。

開啟配送清單時，必須提供一個以上 MQOR 物件記錄的陣列，才能在配送清單中指定目的地佇列的名稱。 作法有兩種:

- 使用偏移欄位 *ObjectRecOffset*。

  在此情況下，應用程式必須宣告其自己的結構包含 MQOD，後面接著 MQOR 記錄陣列 (具有所需數目的陣列元素)，並將 *ObjectRecOffset* 設為從 MQOD 開始算起陣列中第一個元素的偏移。 請確定此偏移是正確的，且具有可在 MQLONG 內容納的值 (最嚴格的程式設計語言是 COBOL，其有效範圍是 -999 999 999 至 +999 999 999)。

  對於不支援指標資料類型的程式設計語言，或以不可攜至不同環境 (例如 COBOL 程式設計語言) 的方式來實作指標資料類型的程式設計語言，請使用 *ObjectRecOffset* 。
- 使用指標欄位 *ObjectRecPtr*。

  在此情況下，應用程式可以與 MQOD 結構分開宣告 MQOR 結構的陣列，並將 *ObjectRecPtr* 設為陣列的位址。

  以可攜至不同環境 (例如， C 程式設計語言) 的方式，將 *ObjectRecPtr* 用於支援指標資料類型的程式設計語言。

不論您選擇何種技術，請使用 *ObjectRecOffset* 和 *ObjectRecPtr* 之一; 如果兩者都是零，或兩者都不是零，則呼叫會失敗，原因碼為 MQRC_OBJECT_RECORDS_ERROR。

這是輸入欄位。 此欄位的起始值為 0。 如果 *Version* 小於 MQOD_VERSION_2，則會忽略此欄位。

ObjectRecPtr (MQPTR)

這是第一個 MQOR 物件記錄的位址。 只有在開啟配送清單時，才會使用 *ObjectRecPtr* 。 如果 *RecsPresent* 為零，則會忽略該欄位。

您可以使用 *ObjectRecPtr* 或 *ObjectRecOffset* 來指定物件記錄，但不能同時指定兩者; 如需詳細資料，請參閱上述 *ObjectRecOffset* 欄位的說明。 如果您不使用 *ObjectRecPtr*，請將它設為空值指標或空值位元組。

這是輸入欄位。 在那些支援指標的程式設計語言中，此欄位的起始值是空值指標，否則是全空值位元組字串。 如果 *Version* 小於 MQOD_VERSION_2，則會忽略此欄位。

註: 在程式設計語言不支援指標資料類型的平台上，此欄位宣告為適當長度的位元組字串，起始值為 all-null 位元組字串。

ObjectString (MQCHARV)
ObjectString 欄位指定長物件名稱。

這會指定要使用的長物件名稱。 只有 *ObjectType* 的特定值才會參照這個欄位，所有其他值都會忽略這個欄位。 如需哪些值指出使用此欄位的詳細資料，請參閱 *ObjectType* 的說明。

如果未正確指定 *ObjectString* ，則根據如何使用 MQCHARV 結構的說明，或者如果它超出長度上限，則呼叫會失敗，原因碼為 MQRC_OBJECT_STRING_ERROR。

這是輸入欄位。 此結構中欄位的起始值與 MQCHARV 結構中欄位的起始值相同。

完整主題名稱可以從兩個不同的欄位來建置: *ObjectName* 和 *ObjectString*。 如需如何使用這兩個欄位的詳細資料，請參閱 第 503 頁的『使用主題字串』。

ObjectType (MQLONG)

在物件描述子中命名的物件類型。 可能的值為:

MQOT_CLNTCONN_CHANNEL
用戶端連線通道。 在 *ObjectName* 欄位中找到物件的名稱。

MQOT_Q
佇列。 在 *ObjectName* 欄位中找到物件的名稱。

MQOT_NAMELIST
名單。 在 *ObjectName* 欄位中找到物件的名稱

MQ 處理程序
程序定義。 在 *ObjectName* 欄位中找到物件的名稱

MQOT_Q_MGR
佇列管理程式。 在 *ObjectName* 欄位中找到物件的名稱

MQOT_TOPIC
主題。 完整主題名稱可以從兩個不同的欄位來建置: *ObjectName* 和 *ObjectString*。
如需如何使用這兩個欄位的詳細資料，請參閱 第 503 頁的『使用主題字串』。

這一律是輸入欄位。 此欄位的起始值是 MQOT_Q。

RecsPresent (MQLONG)

這是應用程式所提供的 MQOR 物件記錄數。 如果此數字大於零，則表示正在開啟配送清單，其中 *RecsPresent* 是清單中目的地佇列的數目。 配送清單只能包含一個目的地。

RecsPresent 的值不得小於零，如果大於零 *ObjectType* 則必須是 MQOT_Q; 如果未滿足這些條件，則呼叫會失敗，原因碼為 MQRC_RECS_PRESENT_ERROR。

在 z/OS 上，此欄位必須為零。

這是輸入欄位。 此欄位的起始值為 0。 如果 *Version* 小於 MQOD_VERSION_2，則會忽略此欄位。

ResObject 字串 (MQCHARV)
ResObject 字串欄位是在佇列管理程式解析 *ObjectName* 欄位中提供的名稱之後的長物件名稱。

僅針對參照主題物件的主題及佇列別名傳回此欄位。

如果在 *ObjectString* 中提供長物件名稱，但在 *ObjectName* 中未提供任何內容，則在此欄位中傳回的值與在 *ObjectString* 中提供的值相同。

如果省略此欄位 (即 ResObjectString.VSBufSize 為零) ，則不會傳回 *ResObjectString* ，但會在 ResObjectString.VSLength 中傳回長度。

如果緩衝區長度 (在 ResObjectStrng.VSBufSize 中提供) 小於完整 *ResObjectString*，則會截斷字串，並傳回所提供緩衝區中可容納的最右側字元數。

如果指定 *ResObjectString* 不正確，則根據如何使用 MQCHARV 結構的說明，或如果它超出長度上限，則呼叫會失敗，原因碼為 MQRC_RES_OBJECT_STRING_ERROR。

ResolvedQMgr 名稱 (MQCHAR48)

這是本端佇列管理程式解析名稱之後的目的地佇列管理程式名稱。 傳回的名稱是擁有 *ResolvedQName* 所識別之佇列的佇列管理程式名稱。 *ResolvedQMgrName* 可以是本端佇列管理程式的名稱。

如果 *ResolvedQName* 是本端佇列管理程式所屬佇列共用群組所擁有的共用佇列，則 *ResolvedQMgrName* 是佇列共用群組的名稱。 如果佇列由某個其他佇列共用群組所擁有，則 *ResolvedQName* 可以是佇列共用群組的名稱或佇列共用群組成員的佇列管理程式名稱 (傳回值的本質由本端佇列管理程式中存在的佇列定義決定)。

只有在物件是開啟用於瀏覽、輸入或輸出 (或任何組合) 的單一佇列時，才會傳回非空白值。 如果開啟的物件是下列任何一項，則 *ResolvedQMgrName* 會設為空白:

- 不是佇列
- 佇列，但未開啟以進行瀏覽、輸入或輸出
- 指定 MQOO_BIND_NOT_FIXED (或在 *DefBind* 佇列屬性具有 MQBND_BIND_NOT_FIXED 值時生效的 MQOO_BIND_AS_Q_DEF) 的叢集佇列
- 配送清單

這是輸出欄位。 此欄位的長度由 MQ_Q_NAME_LENGTH 指定。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。 如果 *Version* 小於 MQOD_VERSION_3，則會忽略此欄位。

ResolvedQName (MQCHAR48)

這是本端佇列管理程式解析名稱之後的目的地佇列名稱。 傳回的名稱是存在於 *ResolvedQMgrName* 所識別之佇列管理程式上的佇列名稱。

只有在物件是開啟用於瀏覽、輸入或輸出 (或任何組合) 的單一佇列時，才會傳回非空白值。 如果開啟的物件是下列任何一項，則 *ResolvedQName* 會設為空白:

- 不是佇列
- 佇列，但未開啟以進行瀏覽、輸入或輸出
- 配送清單
- 參照主題物件的別名佇列 (請改為參閱 ResObjectString)。
- 解析為主題物件的別名佇列。

這是輸出欄位。 此欄位的長度由 MQ_Q_NAME_LENGTH 指定。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。 如果 *Version* 小於 MQOD_VERSION_3，則會忽略此欄位。

ResolvedType (MQLONG)
正在開啟的已解析 (基本) 物件類型。

可能值包括:

MQOT_Q
已解析的物件是佇列。 當直接開啟佇列或開啟指向佇列的別名佇列時，此值適用。

MQOT_TOPIC
已解析的物件是主題。 當直接開啟主題或開啟指向主題物件的別名佇列時，此值適用。

MQOT_NONE
解析的類型既不是佇列，也不是主題。

ResponseRec 偏移 (MQLONG)

這是從 MQOD 結構開始算起第一個 MQRR 回應記錄的偏移 (以位元組為單位)。 偏移可以是正數或負數。 只有在開啟配送清單時，才會使用 *ResponseRecOffset* 。 如果 *RecsPresent* 為零，則會忽略該欄位。

當開啟發佈清單時，您可以提供一或多個 MQRR 回應記錄的陣列，以識別無法開啟的佇列 (MQRR 中的 *CompCode* 欄位) ，以及每次失敗的原因 (MQRR 中的 *Reason* 欄位)。 在回應記錄陣列中傳回資料的順序，與在物件記錄陣列中出現佇列名稱的順序相同。 只有在呼叫的結果混合時 (亦即，某些佇列已順利開啟，而其他佇列則失敗，或所有佇列皆失敗，但因不同原因) ，佇列管理程式才會設定回應記錄; 呼叫中的原因碼 MQRC_MULTIPLE_REASONS 會指出此情況。 如果相同的原因碼套用至所有佇列，則會在 MQOPEN 或 MQPUT1 呼叫的 *Reason* 參數中傳回該原因，且不會設定回應記錄。 回應記錄是選用的，但如果提供它們，則必須有 *RecsPresent* 個。

透過在 *ResponseRecOffset* 中指定偏移，或在 *ResponseRecPtr* 中指定位址，可以使用與物件記錄相同的方式提供回應記錄; 如需如何執行此動作的詳細資料，請參閱上方 *ObjectRecOffset* 的說明。 不過，只能使用 *ResponseRecOffset* 和 *ResponseRecPtr* 中的一個以上; 如果兩者都不是零，則呼叫會失敗，原因碼為 MQRC_RESPONSE_RECORDS_ERROR。

對於 MQPUT1 呼叫，這些回應記錄用於傳回將訊息傳送至配送清單中佇列時所發生錯誤的相關資訊，以及開啟佇列時所發生錯誤的相關資訊。 只有在來自佇列的完成碼是 MQCC_OK 或 MQCC_WARNING 時，來自佇列之放置作業的完成碼及原因碼才會取代來自該佇列之開啟作業的完成碼。

這是輸入欄位。 此欄位的起始值為 0。 如果 *Version* 小於 MQOD_VERSION_2，則會忽略此欄位。

ResponseRecPtr (MQPTR)

這是第一個 MQRR 回應記錄的位址。 只有在開啟配送清單時，才會使用 *ResponseRecPtr* 。 如果 *RecsPresent* 為零，則會忽略該欄位。

使用 *ResponseRecPtr* 或 *ResponseRecOffset* 來指定回應記錄，但不能同時指定兩者; 如需詳細資料，請參閱上方 *ResponseRecOffset* 欄位的說明。 如果您不使用 *ResponseRecPtr*，請將它設為空值指標或空值位元組。

這是輸入欄位。 在那些支援指標的程式設計語言中，此欄位的起始值是空值指標，否則是全空值位元組字串。 如果 *Version* 小於 MQOD_VERSION_2，則會忽略此欄位。

註: 在程式設計語言不支援指標資料類型的平台上，此欄位宣告為適當長度的位元組字串，起始值為 all-null 位元組字串。

SelectionString (MQCHARV)
此字串用來提供從佇列中擷取訊息時所使用的選取準則。

在下列情況下，不得提供 *SelectionString* :

- 如果 *ObjectType* 不是 MQOT_Q
- 如果正在開啟的佇列未使用其中一個 MQOO_BROWSE 或 MQOO_INPUT_ * 選項來開啟

如果在這些情況下提供 *SelectionString* ，則呼叫會失敗，原因碼為 MQRC_SELECTOR_INVALID_FOR_TYPE。

如果未正確指定 *SelectionString* ，根據如何使用 第 262 頁的『MQCHARV-可變長度字串』 結構的說明，或如果它超出長度上限，則呼叫會失敗，原因碼為 MQRC_SELECTION_STRING_ERROR。 *SelectionString* 的長度上限為 MQ_SELECTOR_LENGTH。

SelectionString 用法在 選取器中說明。

StrucId (MQCHAR4)

這是結構 ID; 值必須是:

MQOD_STRUC_ID
物件描述子結構的 ID。

對於 C 程式設計語言，也會定義常數 MQOD_STRUC_ARRAY; 此值與 MQOD_STRUC_ID 相同，但它是字元陣列而非字串。

這一律是輸入欄位。 此欄位的起始值是 MQOD_STRUC_ID。

UnknownDest 計數 *(MQLONG)*

這是配送清單中解析為遠端佇列且已順利開啟的佇列數。 如果存在的話，當開啟不在配送清單中的單一佇列時，也會設定這個欄位。

這是輸出欄位。 此欄位的起始值為 0。 如果 *Version* 小於 MQOD_VERSION_1，則會忽略此欄位。

版本 *(MQLONG)*

這是結構版本號碼; 值必須是下列其中一項:

MQOD_VERSION_1
Version-1 物件描述子結構。

MQOD_VERSION_2
Version-2 物件描述子結構。

MQOD_VERSION_3
Version-3 物件描述子結構。

MQOD_VERSION_4
Version-4 物件描述子結構。

所有 WebSphere MQ V7.0 環境都支援所有版本。

僅存在於結構最新版本中的欄位會在欄位說明中如此識別。 下列常數指定現行版本的版本號碼:

MQOD_CURRENT_VERSION
物件描述子結構的現行版本。

這一律是輸入欄位。 此欄位的起始值為 MQOD_VERSION_1。

MQOD 的起始值和語言宣告

欄位名稱	常數名稱	常數值
StrucId	MQOD_STRUC_ID	'OD¬¬'
Version	MQOD_VERSION_1	1
ObjectType	MQOT_Q	1
ObjectName	無	空字串或空白
ObjectQMgrName	無	空字串或空白
DynamicQName	無	'CSQ.*' on z/OS; 否則 'AMQ.*'
AlternateUserId	無	空字串或空白
RecsPresent	無	0
KnownDestCount	無	0
UnknownDestCount	無	0
InvalidDestCount	無	0
ObjectRecOffset	無	0
ResponseRecOffset	無	0
ObjectRecPtr	無	空值指標或空值位元組
ResponseRecPtr	無	空值指標或空值位元組
AlternateSecurityId	MQSID_NONE	空值
ResolvedQName	無	空字串或空白
ResolvedQMgrName	無	空字串或空白
ObjectString	MQCHARV_DEFAULT	如 MQCHARV 所定義
SelectionString	MQCHARV_DEFAULT	如 MQCHARV 所定義
ResObjectString	MQCHARV_DEFAULT	如 MQCHARV 所定義
ResolvedType	MQOT_NONE	0

欄位名稱	常數名稱	常數值

附註:

1. 符號 ¬ 代表單一空白字元。
2. 空值字串或空白值表示 C 中的空值字串，以及其他程式設計語言中的空白字元。
3. 在 C 程式設計語言中，巨集變數 MQOD_DEFAULT 包含上面列出的值。 您可以用下列方式來提供結構中欄位的起始值:

```
MQOD MyOD = {MQOD_DEFAULT};
```

C 宣告

```
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
                                      oject type */
  /* Ver:4 */
};
```

COBOL 宣告

```
** 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-KNOWNDESTCOUNT             PIC S9(9) BINARY.
** Number of remote queues opened successfully
```

```
 15  MQOD-UNKNOWNDESTCOUNT            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-OBJECTRECPTR                POINTER.
** Address of first response record
 15  MQOD-RESPONSERECPTR              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.
```

PL/I 宣告

```
dcl
 1 MQOD based,
  3 StrucId             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 */
```

High Level Assembler 宣告

```
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_UNKNOWNDESTCOUNT          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_OBJECTRECPTR              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_VSBUFSIZE 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_VSBUFSIZE 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)
```

Visual Basic 宣告

```
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
```

MQOR-物件記錄

下表彙總結構中的欄位。

表 *521: MQOR* 中的欄位		
欄位	**說明**	**主題**
ObjectName	物件名稱	ObjectName
ObjectQMgrName	物件佇列管理程式名稱	ObjectQMgrName

MQOR 概觀

可用性: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows，以及連接至這些系統的 WebSphere MQ MQI 用戶端。

目的: 使用 MQOR 結構來指定單一目的地佇列的佇列名稱及佇列管理程式名稱。 MQOR 是 MQOPEN 和 MQPUT1 呼叫的輸入結構。

字集及編碼:MQOR 中的資料必須採用 *CodedCharSetId* 佇列管理程式屬性所提供的字集，以及 MQENC_NATIVE 所提供本端佇列管理程式的編碼。 不過，如果應用程式以 MQ MQI 用戶端身分執行，則該結構必須採用用戶端的字集及編碼。

用法: 透過在 MQOPEN 呼叫上提供這些結構的陣列，您可以開啟佇列清單; 此清單稱為 配送清單。 如果順利開啟佇列，則會將使用該 MQOPEN 呼叫所傳回之佇列控點的每一則訊息放置在清單中的每一個佇列上。

MQOR 的欄位

MQOR 結構包含下列欄位; 這些欄位按 **字母順序**說明:

ObjectName (MQCHAR48)

這與 MQOD 結構中的 *ObjectName* 欄位相同 (如需詳細資料，請參閱 MQOD) ， 不同之處如下:

- 它必須是佇列的名稱。
- 它不能是模型佇列的名稱。

這一律是輸入欄位。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

ObjectQMgr 名稱 *(MQCHAR48)*

這與 MQOD 結構中的 *ObjectQMgrName* 欄位相同 (如需詳細資料，請參閱 MQOD)。

這一律是輸入欄位。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

MQOR 的起始值和語言宣告

表 *522: MQOR* 中 *MQOR* 的欄位起始值

欄位名稱	常數名稱	常數值
ObjectName	無	空字串或空白
ObjectQMgrName	無	空字串或空白

附註:

1. 空值字串或空白值表示 C 中的空值字串，以及其他程式設計語言中的空白字元。
2. 在 C 程式設計語言中，巨集變數 MQOR_DEFAULT 包含上面列出的值。 您可以用下列方式來提供結構中欄位的起始值:

```
MQOR MyOR = {MQOR_DEFAULT};
```

C 宣告

```
typedef struct tagMQOR MQOR;
struct tagMQOR {
  MQCHAR48  ObjectName;      /* Object name */
  MQCHAR48  ObjectQMgrName;  /* Object queue manager name */
};
```

COBOL 宣告

```
**   MQOR structure
  10 MQOR.
**    Object name
   15 MQOR-OBJECTNAME     PIC X(48).
**    Object queue manager name
   15 MQOR-OBJECTQMGRNAME PIC X(48).
```

PL/I 宣告

```
dcl
 1 MQOR based,
  3 ObjectName     char(48), /* Object name */
  3 ObjectQMgrName char(48); /* Object queue manager name */
```

Visual Basic 宣告

```
Type MQOR
  ObjectName     As String*48 'Object name'
  ObjectQMgrName As String*48 'Object queue manager name'
End Type
```

MQPD-內容描述子

下表彙總結構中的欄位。

表 *523: MQPD* 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
Options	選項	選項
Support	訊息內容的必要支援	支援
Context	內容所屬的訊息環境定義	環境定義
CopyOptions	複製內容所屬的選項	CopyOptions

MQPD 概觀

可用性: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows、 z/OS 及 WebSphere MQ MQI 用戶端。

目的: **MQPD** 用來定義內容的屬性。 結構是 MQSETMP 呼叫的輸入/輸出參數，以及 MQINQMP 呼叫的輸出參數。

字集及編碼: **MQPD** 中的資料必須在應用程式的字集及應用程式的編碼中 (**MQENC_NATIVE**)。

MQPD 的欄位

MQPD 結構包含下列欄位; 這些欄位按 **字母順序**進行說明:

環境定義 *(MQLONG)*

這說明內容所屬的訊息環境定義。

當佇列管理程式收到包含 WebSphere MQ 定義內容的訊息，且佇列管理程式將該內容視為不正確時，佇列管理程式會更正 *Context* 欄位的值。

可以指定下列選項:

MQPD_USER_CONTEXT
內容與使用者環境定義相關聯。

不需要特殊授權即可使用 MQSETMP 呼叫來設定與使用者環境定義相關聯的內容。

在 WebSphere MQ 7.0 版佇列管理程式上，會依照 MQOO_SAVE_ALL_CONTEXT 的說明來儲存與使用者環境定義相關聯的內容。 指定 MQPMO_PASS_ALL_CONTEXT 的 MQPUT 呼叫會導致將內容從已儲存的環境定義複製到新訊息中。

如果不需要先前說明的選項，則可以使用下列選項:

MQPD_NO_CONTEXT
內容未與訊息環境定義相關聯。

無法辨識的值被拒絕， *Reason* 程式碼為 MQRC_PD_ERROR

這是 MQSETMP 呼叫的輸入/輸出欄位，以及 MQINQMP 呼叫的輸出欄位。 此欄位的起始值為 MQPD_NO_CONTEXT。

CopyOptions (MQLONG)

這說明內容應該複製到哪一種類型的訊息。 這是已辨識 WebSphere MQ 定義內容的僅限輸出欄位; WebSphere MQ 會設定適當的值。

當佇列管理程式接收包含 WebSphere MQ 定義內容的訊息時，如果佇列管理程式將該內容視為不正確，則佇列管理程式會更正 *CopyOptions* 欄位的值。

您可以指定下列一或多個選項，如果您需要多個選項，則值可以是:

- 相加（請不要將同一常數相加多次），或者
- 利用執行位元 OR 運算來組合（如果程式設計語言支援位元運算）。

MQ COPY_FORWARD
此內容會複製到正在轉遞的訊息中。

MQCOPY_PUBLISH
當發佈訊息時，此內容會複製到訂閱者所接收的訊息中。

MQCOPY_REPLY
此內容會複製到回覆訊息中。

MQCOPY_REPORT
此內容會複製到報告訊息中。

MQCOPY_ALL
此內容會複製到所有類型的後續訊息。

預設選項: 可以指定下列選項來提供預設複製選項集:

MQCOPY_DEFAULT
此內容會在發佈訊息時複製到正在轉遞的訊息、報告訊息或訂閱者所接收的訊息中。

這相當於指定選項 MQCOPY_FORWARD 的組合，加上 MQCOPY_REPORT ，加上 MQCOPY_PUBLISH。

如果不需要上述任何選項，請使用下列選項:

MQCOPY_NONE
使用此值指出未指定其他副本選項; 以程式化方式，此內容與後續訊息之間不存在任何關係。 一律會針對訊息描述子內容傳回此訊息。

這是 MQSETMP 呼叫的輸入/輸出欄位，以及 MQINQMP 呼叫的輸出欄位。 此欄位的起始值為 MQCOPY_DEFAULT。

選項 *(MQLONG)*

值必須為:

MQPD_NONE
未指定選項

這一律是輸入欄位。 此欄位的起始值為 MQPD_NONE。

StrucId (MQCHAR4)

這是結構 ID; 值必須是:

MQPD_STRUC_ID
內容描述子結構的 ID。

對於 C 程式設計語言，也會定義常數 **MQPD_STRUC_ID_ARRAY** ; 其值與 **MQPD_STRUC_ID** 相同，但卻是字元陣列而非字串。

這一律是輸入欄位。 此欄位的起始值為 **MQPD_STRUC_ID**。

支援 *(MQLONG)*

此欄位說明佇列管理程式需要訊息內容的支援層次，才能將包含此內容的訊息放入佇列。 這僅適用於 WebSphere MQ 定義的內容; 其他所有內容的支援是選用的。

當佇列管理程式已知 WebSphere MQ 定義的內容時，此欄位會自動設為正確的值。 如果無法辨識內容，則會指派 MQPD_SUPPORT_OPTIONAL。 當佇列管理程式收到包含 WebSphere MQ 定義內容的訊息，且佇列管理程式將該內容視為不正確時，佇列管理程式會更正 *Support* 欄位的值。

在已設定 MQCMHO_NO_VALIDATION 選項的訊息控點上使用 MQSETMP 呼叫來設定 WebSphere MQ 定義的內容時， *Support* 會變成輸入欄位。 這可讓應用程式放置具有正確值的 WebSphere MQ 定義內容，其中已連接的佇列管理程式不支援該內容，但預期在另一個佇列管理程式上處理訊息。

值 MQPD_SUPPORT_OPTIONAL 一律指派給不是 WebSphere MQ 定義內容的內容。

如果支援訊息內容的 WebSphere MQ 7.0 佇列管理程式收到包含無法辨識 *Support* 值的內容，則會將該內容視為:

- 如果 MQPD_REJECT_UNSUPPORT_MASK 中包含任何無法辨識的值，則已指定 MQPD_SUPPORT_REQUIRED。
- 如果 MQPD_ACCEPT_UNSUPPORT_IF_XMIT_MASK 中包含任何無法辨識的值，則已指定 MQPD_SUPPORT_REQUIRED_IF_LOCAL
- 否則會指定 MQPD_SUPPORT_OPTIONAL。

在設定 MQCMHO_NO_VALIDATION 選項的訊息控點上使用 MQSETMP 呼叫時， MQINQMP 呼叫會傳回下列其中一個值，或者可以指定其中一個值:

MQPD_SUPPORT_OPTIONAL
即使不支援此內容，佇列管理程式也會接受此內容。 可以捨棄此內容，以便訊息流向不支援訊息內容的佇列管理程式。 此值也會指派給非 WebSphere MQ 定義的內容。

MQPD_SUPPORT_REQUIRED
需要支援內容。 訊息被不支援 WebSphere MQ 定義內容的佇列管理程式拒絕。 MQPUT 或 MQPUT1 呼叫失敗，完成碼為 MQCC_FAILED ，原因碼為 MQRC_UNSUPPORTED_PROPERTY。

MQPD_SUPPORT_REQUIRED_IF_LOCAL
如果訊息指定給本端佇列，則不支援 WebSphere MQ 定義內容的佇列管理程式會拒絕該訊息。 MQPUT 或 MQPUT1 呼叫失敗，完成碼為 MQCC_FAILED ，原因碼為 MQRC_UNSUPPORTED_PROPERTY。

如果訊息以遠端佇列管理程式為目的地，則 MQPUT 或 MQPUT1 呼叫會成功。

這是 MQINQMP 呼叫上的輸出欄位，以及 MQSETMP 呼叫上的輸入欄位 (如果在建立訊息控點時已設定 MQCMHO_NO_VALIDATION 選項)。 此欄位的起始值為 MQPD_SUPPORT_OPTIONAL。

版本 *(MQLONG)*

這是結構版本號碼; 值必須是:

MQPD_VERSION_1
Version-1 內容描述子結構。

下列常數指定現行版本的版本號碼:

MQPD_CURRENT_VERSION
內容描述子結構的現行版本。

這一律是輸入欄位。 此欄位的起始值為 **MQPD_VERSION_1**。

MQPD 的起始值及語言宣告

表 *524: MQPD* 中欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQPD_STRUC_ID	'PD'
Version	MQPD_VERSION_1	1
Options	MQPD_NONE	0
Support	MQPD_SUPPORT_OPTIONAL	0
Context	MQPD_NO_CONTEXT	0
CopyOptions	MQCOPY_DEFAULT	0

表 *524: MQPD* 中欄位的起始值 (繼續)

欄位名稱	常數名稱	常數值

附註:

1. 在 C 程式設計語言中，巨集變數 MQPD_DEFAULT 包含上述值。 您可以用下列方式來提供結構中欄位的起始值:

```
MQPD MyPD = {MQPD_DEFAULT};
```

C 宣告

```
typedef struct tagMQPD MQPD;
struct tagMQPD {
  MQCHAR4  StrucId;     /* 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 */
};
```

COBOL 宣告

```
**   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.
```

PL/I 宣告

```
dcl
 1 MQPD 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 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 */
```

High Level Assembler 宣告

```
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)
```

MQPMO-放置訊息選項

下表彙總結構中的欄位。

表 *525: MQPMO* 結構		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
Options	控制 MQPUT 及 MQPUT1 動作的選項	選項
Timeout	已保留	逾時
Context	輸入佇列的物件控點	環境定義
KnownDestCount	順利傳送至本端佇列的訊息數	KnownDestCount
UnknownDestCount	順利傳送至遠端佇列的訊息數	UnknownDestCount
InvalidDestCount	無法傳送的訊息數	InvalidDestCount
ResolvedQName	解析的目的地佇列名稱	ResolvedQName
ResolvedQMgrName	解析的目的地佇列管理程式名稱	ResolvedQMgr 名稱
註: 如果 *Version* 小於 MQPMO_VERSION_2，則會忽略其餘欄位。		
RecsPresent	存在的放置訊息記錄或回應記錄數	RecsPresent
PutMsgRecFields	指出哪些 MQPMR 欄位存在的旗標	PutMsgRecFields
PutMsgRecOffset	從 MQPMO 開始的第一個放置訊息記錄的偏移	PutMsgRecOffset
ResponseRecOffset	從 MQPMO 開始的第一個回應記錄的偏移	ResponseRec 偏移
PutMsgRecPtr	第一個放置訊息記錄的位址	PutMsgRecPtr
ResponseRecPtr	第一個回應記錄的位址	ResponseRecPtr
註: 如果 *Version* 小於 MQPMO_VERSION_3，則會忽略其餘欄位。		
OriginalMsgHandle	原始訊息控點	OriginalMsg 控點
NewMsgHandle	新建訊息控點	NewMsg 控點
Action	要執行的放置類型，以及 *OriginalMsgHandle* 欄位指定的原始訊息與 *NewMsgHandle* 欄位指定的新訊息之間的關係	動作:
PubLevel	發佈設定為目標的訂閱層次	PubLevel

MQPMO 概觀

可用性: 所有 WebSphere MQ 系統，以及連接至這些系統的 WebSphere MQ 用戶端。

目的:MQPMO 結構可讓應用程式指定選項，以控制如何將訊息放置在佇列上或發佈至主題。 此結構是 MQPUT 及 MQPUT1 呼叫的輸入/輸出參數。

版本:MQPMO 的現行版本是 MQPMO_VERSION_3。 某些欄位只能在某些 MQPMO 版本中使用。 如果您需要在數個環境之間連接應用程式，則必須確保 MQPMO 的版本在所有環境之間一致。 僅存在於特定結構版本中的欄位在 第 432 頁的『MQPMO-放置訊息選項』 及欄位說明中被識別為如此。

為支援的程式設計語言提供的標頭、COPY 和 INCLUDE 檔案包含環境支援的 MQPMO 最新版本，但 *Version* 欄位的起始值設為 MQPMO_VERSION_1。 若要使用不在 version-1 結構中的欄位，應用程式必須將 *Version* 欄位設為所需版本的版本號碼。

字集及編碼:MQPMO 中的資料必須採用 *CodedCharSetId* 佇列管理程式屬性所提供的字集，以及 MQENC_NATIVE 所提供本端佇列管理程式的編碼。 不過，如果應用程式以 MQ MQI 用戶端身分執行，則該結構必須採用用戶端的字集及編碼。

MQPMO 的欄位

MQPMO 結構包含下列欄位; 這些欄位按 **字母順序**進行說明:

動作 *(MQLONG)*
這會指定要執行的放置類型，以及 OriginalMsgHandle 欄位指定的原始訊息與 NewMsgHandle 欄位指定的新訊息之間的關係。 佇列管理程式會根據指定的「動作」值來選擇訊息的內容。

您可以選擇在 MQPUT 或 MQPUT1 呼叫上使用 MsgDesc 參數來提供訊息描述子的內容。 或者，可以不提供 MsgDesc 參數，或者透過在 MQPMO 結構的「選項」欄位中包括 MQPMO_MD_FOR_OUTPUT_ONLY 來指定它是僅輸出。

如果未提供 MsgDesc 參數，或者已指定為僅輸出，則根據本主題中說明的規則，會從 MQPMO 的訊息控點欄位移入新訊息的訊息描述子。

在編寫訊息描述子之後， 控制環境定義資訊 中說明的環境定義設定及傳遞活動會生效。

如果指定不正確的動作值，則呼叫會失敗，原因碼為 MQRC_ACTION_ERROR。

可以指定下列任何一個動作:

MQACTP_NEW
正在放置新訊息，且程式未指定與前一個訊息的關係。 訊息描述子如下所示:

- 如果在 MQPUT 或 MQPUT1 呼叫上提供 MsgDesc ，且 MQPMO_MD_FOR_OUTPUT_ONLY 不在 MQPMO.Options，這是用來作為未經修改的訊息描述子。
- 如果未提供 MsgDesc ，或 MQPMO_MD_FOR_OUTPUT_ONLY 位於 MQPMO.Options 佇列管理程式會使用 OriginalMsgHandle 和 NewMsgHandle 中的內容組合來產生訊息描述子。 在新訊息控點上明確設定的任何訊息描述子欄位，優先於原始訊息控點中的訊息描述子欄位。

訊息資料取自 MQPUT 或 MQPUT1 緩衝區參數。

MQACTP_FORWARD
正在轉遞先前擷取的訊息。 原始訊息控點指定先前擷取的訊息。

新的訊息控點指定對原始訊息控點中內容 (包括訊息描述子中的任何內容) 的任何修改。

訊息描述子如下所示:

- 如果在 MQPUT 或 MQPUT1 呼叫上提供 MsgDesc ，且 MQPMO_MD_FOR_OUTPUT_ONLY 不在 MQPMO.Options，這是用來作為未經修改的訊息描述子。
- 如果未提供 MsgDesc ，或 MQPMO_MD_FOR_OUTPUT_ONLY 位於 MQPMO.Options 佇列管理程式會使用 OriginalMsgHandle 和 NewMsgHandle 中的內容組合來產生訊息描述子。 在新訊息控點上明確設定的任何訊息描述子欄位，優先於原始訊息控點中的訊息描述子欄位。
- 如果在 MQPMO.Options，則會遵守這些選項。

訊息內容編製如下:

- 原始訊息控點中具有 MQPD.CopyOptions
- 新訊息控點中的所有內容。 對於新訊息控點中與原始訊息控點中具有相同名稱之內容的每一個內容，會從新訊息控點取得值。 當新訊息控點中的內容與原始訊息控點中的內容同名，但內容值為空值時，此規則的唯一例外情況是特殊情況。 在此情況下，會從訊息中移除內容。

要轉遞的訊息資料是從 MQPUT 或 MQPUT1 緩衝區參數取得。

MQACTP_REPLY
正在回覆先前擷取的訊息。 原始訊息控點指定先前擷取的訊息。

新的訊息控點指定對原始訊息控點中內容 (包括訊息描述子中的任何內容) 的任何修改。

訊息描述子如下所示:

- 如果在 MQPUT 或 MQPUT1 呼叫上提供 MsgDesc，且 MQPMO_MD_FOR_OUTPUT_ONLY 不在 MQPMO.Options，這是用來作為未經修改的訊息描述子。
- 如果未提供 MsgDesc，或 MQPMO_MD_FOR_OUTPUT_ONLY 位於 MQPMO.Options 選項之後會選擇起始訊息描述子欄位，如下所示:

表 *526:* 回覆訊息控點轉換

MQMD 中的欄位	**使用的值**
報告	如果 MQRO_PASS_DISCARD_AND_EXPIRY 及 MQRO_DISCARD_MSG 已設定: MQRO_DISCARD_MSG 否則 MQRO_NONE
MsgType	MQMT_REPLY
期限	如果 MQRO_PASS_DISCARD_AND_EXPIRY 已設定: 從輸入訊息複製 否則 MQEI_UNLIMITED
意見	MQFB_NONE
MsgId	如果已設定 MQPMO_NEW_MSG_ID: 產生新的訊息 ID else if MQRO_PASS_MSG_ID is set: 從輸入訊息複製 否則 MQMI_NONE

CorrelId	如果已設定 MQPMO_NEW_CORREL_ID:
	產生新的相關性 ID
	else if MQRO_COPY_MSG_ID_TO_CORREL_ID is set:
	從下列項目的 MsgId 欄位複製
	輸入訊息
	else if MQRO_PASS_CORREL_ID is set:
	從的 CorrelId 欄位複製
	輸入訊息
	否則
	MQCI_NONE

BackoutCount	0
ReplyToQ	空白
回覆目的地佇列管理程式	空白
GroupId	MQGI_NONE
MsgSeqNumber	1
偏移	0
MsgFlags	無 MQMF_NONE
OriginalLength	未定義 MQOL_UNDEFINED

- 然後，新的訊息控點會修改訊息描述子-任何明確設定為新訊息控點中內容的訊息描述子欄位，優先於上述訊息描述子欄位。

訊息內容編製如下:

- 原始訊息控點中在 MQPD.CopyOptions
- 新訊息控點中的所有內容。 對於新訊息控點中與原始訊息控點中具有相同名稱之內容的每一個內容，會從新訊息控點取得值。 當新訊息控點中的內容與原始訊息控點中的內容同名，但內容值為空值時，此規則的唯一例外情況是特殊情況。 在此情況下，會從訊息中移除內容。

要轉遞的訊息資料取自 MQPUT/MQPUT1 緩衝區參數。

MQACTP_REPORT
由於先前擷取的訊息而產生報告。 原始訊息控點指定導致產生報告的訊息。

新的訊息控點指定對原始訊息控點中內容 (包括訊息描述子中的任何內容) 的任何修改。

訊息描述子如下所示:

- 如果在 MQPUT 或 MQPUT1 呼叫上提供 MsgDesc ， 且 MQPMO_MD_FOR_OUTPUT_ONLY 不在 MQPMO.Options，這是用來作為未經修改的訊息描述子。
- 如果未提供 MsgDesc ， 或 MQPMO_MD_FOR_OUTPUT_ONLY 位於 MQPMO.Options 選項之後會選擇起始訊息描述子欄位，如下所示:

表 *527:* 報告訊息控點轉換

MQMD 中的欄位	**使用的值**
報告	如果 MQRO_PASS_DISCARD_AND_EXPIRY 及 已設定 MQRO_DISCARD_MSG: MQRO_DISCARD_MSG 否則 MQRO_NONE
MsgType	MQMT_REPORT
期限	如果 MQRO_PASS_DISCARD_AND_EXPIRY 已設定: 從輸入訊息複製 否則 MQEI_UNLIMITED
MsgId	如果已設定 MQPMO_NEW_MSG_ID: 產生新的訊息 ID else if MQRO_PASS_MSG_ID is set: 從輸入訊息複製 否則 MQMI_NONE

CorrelId	如果已設定 MQPMO_NEW_CORREL_ID:
	產生新的相關性 ID
	else if MQRO_COPY_MSG_ID_TO_CORREL_ID is set:
	從下列項目的 MsgId 欄位複製
	輸入訊息
	else if MQRO_PASS_CORREL_ID is set:
	從的 CorrelId 欄位複製
	輸入訊息
	否則
	MQCI_NONE

BackoutCount	0
ReplyToQ	空白

表 *527:* 報告訊息控點轉換 *(*繼續*)*	
MQMD 中的欄位	**使用的值**
回覆目的地佇列管理程式	空白
OriginalLength	設為 *BufferLength*

- 然後，新的訊息控點會修改訊息描述子-任何明確設定為新訊息控點中內容的訊息描述子欄位，優先於上述訊息描述子欄位。

訊息內容編製如下:

- 原始訊息控點中在 MQPD.CopyOptions
- 新訊息控點中的所有內容。 對於新訊息控點中與原始訊息控點中具有相同名稱之內容的每一個內容，會從新訊息控點取得值。 當新訊息控點中的內容與原始訊息控點中的內容同名，但內容值為空值時，此規則的唯一例外情況是特殊情況。 在此情況下，會從訊息中移除內容。

產生的 MQMD 中的「意見回饋」欄位代表要產生的報告。 回饋值 MQFB_NONE 會導致 MQPUT 或 MQPUT1 呼叫失敗，原因碼為 MQRC_FEEDBACK_ERROR。

若要選擇報告訊息的使用者資料， WebSphere MQ 會查閱結果 MQMD 中的「報告」及「意見回饋」欄位，以及 MQPUT 或 MQPUT1 呼叫的「緩衝區」及 BufferLength 參數。

- 如果意見回饋是 MQFB_COA、MQFB_COD 或 MQFB_EXPIRATION ，則會檢查「報告」的值。
- 如果下列任何情況為真，則會使用「緩衝區」中長度為 BufferLength 的完整訊息資料。
  - 回饋是 MQFB_EXPIRATION 且報告包含 MQRO_EXPIRATION_WITH_FULL_DATA
  - 回饋是 MQFB_COD 且報告包含 MQRO_COD_WITH_FULL_DATA
  - 回饋是 MQFB_COA 且報告包含 MQRO_COA_WITH_FULL_DATA
- 如果下列任何情況為真，則會使用「緩衝區」中訊息的前 100 個位元組 (如果小於 100 ，則為 BufferLength)
  - 回饋為 MQFB_EXPIRATION 且報告包含 MQRO_EXPIRATION_WITH_DATA
  - 意見回饋是 MQFB_COD ，且報告包含 MQRO_COD_WITH_DATA
  - 回饋是 MQFB_COA 且報告包含 MQRO_COA_WITH_DATA
- 如果意見回饋是 MQFB_EXPIRATION、MQFB_COD 或 MQFB_COA ，且報告不包含與該「回復」值相關的 * _WITH_FULL_DATA 或 * _WITH_DATA 選項，則訊息中不會包含任何使用者資料。
- 如果「回饋」採用不同於上面列出的值，則會正常使用「緩衝區」及 BufferLength 。

下表顯示使用者資料的衍生:

表 *528:* 使用者資料來源			
	MQFB_COA	**MQFB_COD**	**MQFB_EXPIRATION**
MQRO_EXPIRATION_WITH_FULL_DATA	無	無	緩衝區 (緩衝區長度)
MQRO_COD_WITH_FULL_DATA	無	緩衝區 (緩衝區長度)	無
MQRO_COA_WITH_FULL_DATA	緩衝區 (緩衝區長度)	無	無
MQRO_EXPIRATION_WITH_DATA	無	無	緩衝區 (前 100 個位元組)
MQRO_COD_WITH_DATA	無	緩衝區 (前 100 個位元組)	無
MQRO_COA_WITH_DATA	緩衝區 (前 100 個位元組)	無	無

環境定義 *(MQHOBJ)*

如果指定 MQPMO_PASS_IDENTITY_CONTEXT 或 MQPMO_PASS_ALL_CONTEXT，則此欄位必須包含輸入佇列控點，從中取得要與放置訊息相關聯的環境定義資訊。

如果未指定 MQPMO_PASS_IDENTITY_CONTEXT 或 MQPMO_PASS_ALL_CONTEXT，則會忽略此欄位。

這是輸入欄位。 此欄位的起始值為 0。

InvalidDest 計數 *(MQLONG)*

這是無法傳送至配送清單中佇列的訊息數。 計數包括無法開啟的佇列，以及已順利開啟但放置作業失敗的佇列。 當將訊息放入不在配送清單中的單一佇列時，也會設定此欄位。

註: 如果 MQPUT 或 MQPUT1 呼叫上的 *CompCode* 參數是 MQCC_OK 或 MQCC_WARNING; 如果 *CompCode* 參數是 MQCC_FAILED，但在應用程式碼中不依賴此參數，則可以設定此欄位。

這是輸出欄位。 此欄位的起始值為 0。 如果 *Version* 小於 MQPMO_VERSION_1，則不會設定此欄位。

在 z/OS 上未定義此欄位，因為不支援配送清單。

KnownDest 計數 *(MQLONG)*

這是現行 MQPUT 或 MQPUT1 呼叫已順利傳送至配送清單中屬於本端佇列的佇列的訊息數。 此計數不包括傳送至解析為遠端佇列之佇列的訊息 (即使最初使用本端傳輸佇列來儲存訊息)。 當將訊息放入不在配送清單中的單一佇列時，也會設定此欄位。

這是輸出欄位。 此欄位的起始值為 0。 如果 *Version* 小於 MQPMO_VERSION_1，則不會設定此欄位。

在 z/OS 上未定義此欄位，因為不支援配送清單。

NewMsg 控點 *(MQHMSG)*

這是將訊息置於「動作」欄位值的選用控點。 它會定義訊息的內容，並置換 *OriginalMsgHandle* 的值 (如果有指定的話)。

從 **MQPUT** 或 **MQPUT1** 呼叫返回時，控點的內容會反映實際放置的訊息。

這是輸入欄位。 此欄位的起始值為 **MQHM_NONE**。 如果版本小於 **MQPMO_VERSION_3**，則會忽略此欄位。

MQPMO 選項 *(MQLONG)*
「選項」欄位控制 **MQPUT** 和 **MQPUT1** 呼叫的作業。

範圍選項。 您可以指定任何 MQPMO 選項或不指定任何 MQPMO 選項。 如果需要多個選項，您為選項指定的值可以透過下列方式使用:

- 可以新增值。 請勿多次新增相同的常數。
- 如果程式設計語言支援按位元運算，則可以使用按位元 OR 運算來結合值。

會記下無效的組合; 任何其他組合都是有效的。

下列選項控制所傳送發佈的範圍:

MQPMO_SCOPE_QMGR

發佈只會傳送給已在此佇列管理程式上訂閱的訂閱者。 發佈不會轉遞至任何已訂閱此佇列管理程式的遠端發佈/訂閱佇列管理程式，這會置換已使用 PUBSCOPE 主題屬性設定的任何行為。

註: 如果未設定，發佈範圍是由 PUBSCOPE 主題屬性所決定。

發佈選項。 下列選項控制將訊息發佈至主題的方式:

MQPMO_SUPPRESS_REPLYTO

在此發佈的 MQMD 的 *ReplyToQ* 和 *ReplyToQMgr* 欄位中指定的任何資訊都不會傳遞給訂閱者。 如果此選項與需要 *ReplyToQ* 的報告選項搭配使用，則呼叫會失敗，並產生 MQRC_MISSING_REPLY_TO_Q。

MQPMO_RETAIN

佇列管理程式會保留所傳送的發佈資訊。 此保留可讓訂閱者在發佈此發佈的時間之後，使用 MQSUBRQ 呼叫來要求此發佈的副本。 它也容許將發佈傳送至在建立此發佈之後進行訂閱的應用程式 (除非他們選擇不使用選項 MQSO_NEW_PUBLICATIONS_ONLY 來傳送它)。 如果傳送已保留的發佈資訊給應用程式，則該發佈資訊的 MQIsRetained 訊息內容會指出該應用程式。

在主題樹狀結構的每一個節點上只能保留一個發佈資訊。 因此，如果任何其他應用程式已發佈此主題的保留發佈資訊，則會將它取代為此發佈資訊。 因此，最好避免有多個發佈者保留相同主題的訊息。

當訂閱者要求保留的發佈資訊時，所使用的訂閱可能在主題中包含萬用字元，在此情況下，許多保留的發佈資訊可能相符 (在主題樹狀結構中的各種節點上)，且數個發佈資訊可能會傳送至發出要求的應用程式。 如需詳細資料，請參閱 第 698 頁的『MQSUBRQ-訂閱要求』 呼叫的說明。

如需保留的發佈資訊如何與訂閱層次互動的相關資訊，請參閱 截取發佈資訊。

如果使用此選項，且無法保留發佈，則不會發佈訊息，且呼叫會失敗並產生 MQRC_PUT_NOT_RETAINED。

MQPMO_NOT_OWN_SUBS

告訴佇列管理程式，應用程式不想要將其任何發佈傳送至它所擁有的訂閱。 如果連線控點相同，則訂閱會被視為由相同的應用程式所擁有。

MQPMO_WARN_IF_NO_SUBS_MATCHED

如果沒有訂閱符合發佈，則會傳回 MQCC_WARNING 的完成碼 (*CompCode*)，以及原因碼 MQRC_NO_SUBS_MATCHED。

如果 put 作業傳回 MQRC_NO_SUBS_MATCHED，則發佈不會遞送至任何訂閱。 不過，如果在 put 作業上指定 MQPMO_RETAIN 選項，則訊息會保留並遞送至任何後續定義的相符訂閱。

如果符合下列任何條件，則主題上的訂閱會符合發佈:

- 訊息會遞送至訂閱佇列
- 訊息已遞送至訂閱佇列，但佇列的問題表示無法將訊息放置到佇列中，因此會將訊息放置在無法傳送的郵件佇列中或捨棄。
- 定義遞送結束程式會暫停將訊息遞送至訂閱

如果符合下列任何條件，則主題上的訂閱不符合發佈:

- 訂閱具有不符合發佈的選取字串
- 訂閱已指定 MQSO_PUBLICATION_ON_REQUEST 選項
- 未遞送發佈，因為在 put 作業上指定了 MQPMO_NOT_OWN_SUBS 選項，且訂閱符合發佈者的身分

同步點選項。 下列選項與工作單元內 MQPUT 或 MQPUT1 呼叫的參與相關:

MQPMO_SYNCPOINT

要求是在正常工作單元通訊協定內運作。 在確定工作單元之前，在工作單元之外看不到訊息。 如果工作單元已取消，則會刪除訊息。

如果未指定 MQPMO_SYNCPOINT 和 MQPMO_NO_SYNCPOINT，則工作單元通訊協定中放置要求的併入是由執行佇列管理程式的環境來決定，而不是由執行應用程式的環境來決定。 在 z/OS 上，放置要求是在工作單元內。 在所有其他環境中，放置要求不在工作單元內。

由於這些差異，您要連接的應用程式不得容許此選項預設; 請明確指定 MQPMO_SYNCPOINT 或 MQPMO_NO_SYNCPOINT。

請勿將 MQPMO_SYNCPOINT 與 MQPMO_NO_SYNCPOINT 一起指定。

MQPMO_NO_SYNCPOINT

要求是在正常工作單元通訊協定之外運作。 訊息會立即可用，且無法藉由取消工作單元來刪除它。

如果未指定 MQPMO_NO_SYNCPOINT 和 MQPMO_SYNCPOINT，則工作單元通訊協定中放置要求的併入是由執行佇列管理程式的環境決定，而不是由執行應用程式的環境決定。 在 z/OS 上，放置要求是在工作單元內。 在所有其他環境中，放置要求不在工作單元內。

由於這些差異，您要連接的應用程式不得容許此選項預設; 請明確指定 MQPMO_SYNCPOINT 或 MQPMO_NO_SYNCPOINT。

請勿將 MQPMO_NO_SYNCPOINT 與 MQPMO_SYNCPOINT 一起指定。

訊息 ID 和相關性 ID 選項。 下列選項要求佇列管理程式產生新的訊息 ID 或相關性 ID:

MQPMO_NEW_MSG_ID
佇列管理程式會將 MQMD 中 *MsgId* 欄位的內容取代為新的訊息 ID。 此訊息 ID 隨訊息一起傳送，並在 MQPUT 或 MQPUT1 呼叫輸出時傳回給應用程式。

將訊息放入配送清單時也可以指定 MQPMO_NEW_MSG_ID 選項; 如需詳細資料，請參閱 MQPMR 結構中 *MsgId* 欄位的說明。

使用此選項可解除應用程式在每一個 MQPUT 或 MQPUT1 呼叫之前，將 *MsgId* 欄位重設為 MQMI_NONE 的需求。

MQPMO_NEW_CORREL_ID
佇列管理程式會將 MQMD 中的 *CorrelId* 欄位內容取代為新的相關性 ID。 此相關性 ID 隨訊息一起傳送，並在 MQPUT 或 MQPUT1 呼叫輸出時傳回給應用程式。

將訊息放入配送清單時，也可以指定 MQPMO_NEW_CORREL_ID 選項; 如需詳細資料，請參閱 MQPMR 結構中 *CorrelId* 欄位的說明。

在應用程式需要唯一相關性 ID 的情況下， MQPMO_NEW_CORREL_ID 很有用。

群組和區段選項。 下列選項與處理邏輯訊息群組及區段中的訊息相關。 請閱讀下列定義，以協助您瞭解選項。

小心: 您無法將分段或分組的訊息與「發佈/訂閱」搭配使用。

實體訊息
是可以放置在佇列上或從佇列中移除的最小資訊單元; 它通常對應於在單一 MQPUT、 MQPUT1 或 MQGET 呼叫上指定或擷取的資訊。 每一個實體訊息都有自己的訊息描述子 (MQMD)。 通常，實體訊息是由訊息 ID (MQMD 中的 *MsgId* 欄位) 的不同值來識別，但佇列管理程式不會強制這樣做。

邏輯訊息
邏輯訊息是單一單元的應用程式資訊，僅適用於非 z/OS 平台。 在沒有系統限制的情況下，邏輯訊息與實體訊息相同。 但當邏輯訊息非常大時，系統限制可能會建議或需要將邏輯訊息分割成兩個以上實體訊息，稱為 區段。

已分段的邏輯訊息包含兩個以上具有相同非空值群組 ID (MQMD 中的 *GroupId* 欄位) 及相同訊息序號 (MQMD 中的 *MsgSeqNumber* 欄位) 的實體訊息。 透過區段偏移 (MQMD 中的 *Offset* 欄位) 的不同值來識別區段，這會提供實體訊息中從邏輯訊息中資料開始算起的資料偏移。 因為每一個區段都是實體訊息，所以邏輯訊息中的區段通常具有不同的訊息 ID。

尚未分段但傳送應用程式已允許分段的邏輯訊息也具有非空值群組 ID ，雖然在此情況下，如果邏輯訊息不屬於訊息群組，則只有一個具有該群組 ID 的實體訊息。 除非邏輯訊息屬於訊息群組，否則傳送應用程式禁止分段的邏輯訊息具有空值群組 ID (MQGI_NONE)。

訊息群組
訊息群組是一組具有相同非空值群組 ID 的一或多個邏輯訊息。 群組中的邏輯訊息由訊息序號的不同值識別，該序號是 1 到 *n* 範圍內的整數，其中 *n* 是群組中邏輯訊息的數目。 如果已分段一個以上邏輯訊息，則群組中有 *n* 個以上實體訊息。

MQPMO_LOGICAL_ORDER
這個選項告訴佇列管理程式應用程式如何將訊息放入邏輯訊息的群組和區段中。 只能在 MQPUT 呼叫上指定它; 它在 MQPUT1 呼叫上無效。

如果指定 MQPMO_LOGICAL_ORDER，則表示應用程式使用連續的 MQPUT 呼叫來執行以下動作:

1. 按區段偏移的遞增順序（從 0 開始，無間隙），將區段放置在每個邏輯訊息中。
2. 先將所有區段放置在一個邏輯訊息中，然後再將區段放置在下一個邏輯訊息中。

3. 按訊息序號的遞增順序（從 1 開始，無間隙），將邏輯訊息放置在每個訊息群組中。IBM WebSphere MQ 自動增加訊息序號。
4. 先將所有邏輯訊息放置在一個訊息群組中，然後再將邏輯訊息放置在下一個訊息群組中。

如需 MQPMO_LOGICAL_ORDER 的詳細資訊，請參閱 邏輯和實體排序

環境定義選項。 下列選項控制訊息環境定義的處理:

MQPMO_NO_CONTEXT
身分和原始環境定義都設為表示沒有環境定義。 這表示 MQMD 中的環境定義欄位設為:

- 字元欄位空白
- 位元組欄位的空值
- 數值欄位為零

MQPMO_DEFAULT_CONTEXT
針對身分和原點，訊息會有相關聯的預設環境定義資訊。 佇列管理程式會在訊息描述子中設定環境定義欄位，如下所示:

MQMD 中的欄位	使用的值
UserIdentifier	可能的話，由環境決定; 否則設為空白。
AccountingToken	可能的話，從環境判定; 否則設為 MQACT_NONE。
ApplIdentityData	設為空白。
PutApplType	從環境判定。
PutApplName	可能的話，由環境決定; 否則設為空白。
PutDate	設為放置訊息的日期。
PutTime	設為放置訊息的時間。
ApplOriginData	設為空白。

如需訊息環境定義的相關資訊，請參閱 訊息環境定義。

如果未指定環境定義選項，這些是預設值和動作。

MQPMO_PASS_IDENTITY_CONTEXT
訊息將具有與其相關聯的環境定義資訊。 身分環境定義取自 *Context* 欄位中指定的佇列控點。 佇列管理程式會以 MQPMO_DEFAULT_CONTEXT 的相同方式產生原始環境定義資訊 (如需值，請參閱前述表格)。 如需訊息環境定義的相關資訊，請參閱 訊息環境定義。

對於 MQPUT 呼叫，佇列必須已使用 MQOO_PASS_IDENTITY_CONTEXT 選項 (或暗示它的選項) 開啟。 對於 MQPUT1 呼叫，會使用 MQOO_PASS_IDENTITY_CONTEXT 選項執行與 MQOPEN 呼叫相同的授權檢查。

MQPMO_PASS_ALL_CONTEXT
訊息將具有與其相關聯的環境定義資訊。 環境定義取自 *Context* 欄位中指定的佇列控點。 如需訊息環境定義的相關資訊，請參閱 控制環境定義資訊。

對於 MQPUT 呼叫，佇列必須已使用 MQOO_PASS_ALL_CONTEXT 選項 (或暗示它的選項) 開啟。 對於 MQPUT1 呼叫，會使用 MQOO_PASS_ALL_CONTEXT 選項執行與 MQOPEN 呼叫相同的授權檢查。

MQPMO_SET_IDENTITY_CONTEXT
訊息將具有與其相關聯的環境定義資訊。 應用程式在 MQMD 結構中指定身分環境定義。 佇列管理程式會以 MQPMO_DEFAULT_CONTEXT 的相同方式產生原始環境定義資訊 (如需值，請參閱前述表格)。 如需訊息環境定義的相關資訊，請參閱 訊息環境定義。

對於 MQPUT 呼叫，佇列必須已使用 MQOO_SET_IDENTITY_CONTEXT 選項 (或默示它的選項) 開啟。 對於 MQPUT1 呼叫，會使用 MQOO_SET_IDENTITY_CONTEXT 選項執行與 MQOPEN 呼叫相同的授權檢查。

MQPMO_SET_ALL_CONTEXT
訊息將具有與其相關聯的環境定義資訊。 應用程式在 MQMD 結構中指定身分、原點及使用者環境定義。 如需訊息環境定義的相關資訊，請參閱 訊息環境定義。

對於 MQPUT 呼叫，必須已使用 MQOO_SET_ALL_CONTEXT 選項開啟佇列。 對於 MQPUT1 呼叫，會使用 MQOO_SET_ALL_CONTEXT 選項執行與 MQOPEN 呼叫相同的授權檢查。

您只能指定其中一個 MQPMO_ * _CONTEXT 環境定義選項。 如果您指定 none，則會採用 MQPMO_DEFAULT_CONTEXT。

內容選項。 下列選項與訊息的內容相關:

MQPMO_MD_FOR_OUTPUT_ONLY
訊息描述子參數只能用於輸出，以傳回所放置訊息的訊息描述子。 與 **MQPMO** 結構的 *NewMsgHandle*、*OriginalMsgHandle* 或這兩個欄位相關聯的訊息描述子欄位必須用於輸入。

如果未提供有效的訊息控點，則呼叫會失敗，原因碼為 **MQRC_MD_ERROR**。

放置回應選項。 下列選項控制傳回給 MQPUT 或 MQPUT1 呼叫的回應。 您只能指定下列其中一個選項。 如果未指定 MQPMO_ASYNC_RESPONSE 及 MQPMO_SYNC_RESPONSE，則會採用 MQPMO_RESPONSE_AS_Q_DEF 或 MQPMO_RESPONSE_AS_TOPIC_DEF。

MQPMO_ASYNC_RESPONSE
MQPMO_ASYNC_RESPONSE 選項會要求 MQPUT 或 MQPUT1 作業完成，而不需要應用程式等待佇列管理程式完成呼叫。 使用這個選項可以增進傳訊效能，特別是對於使用用戶端連結的應用程式。 應用程式可以使用 MQSTAT 動詞定期檢查在任何先前的非同步呼叫期間是否發生錯誤。

使用此選項，只保證在 MQMD 中完成下列欄位;

- ApplIdentityData
- PutApplType
- PutApplName
- ApplOriginData

此外，如果將 MQPMO_NEW_MSG_ID 或 MQPMO_NEW_CORREL_ID 中的任一或兩者指定為選項，則傳回的 MsgId 和 CorrelId 也會完成。 (可以透過指定空白 MsgId 欄位隱含地指定 MQPMO_NEW_MSG_ID)。

只會完成之前指定的欄位。 未定義通常會在 MQMD 或 MQPMO 結構中傳回的其他資訊。

當要求 MQPUT1 的非同步放置回應時，未定義 MQOD 結構中所傳回的 ResolvedQName 和 ResolvedQMgr 名稱。

當要求 MQPUT 或 MQPUT1 的非同步放置回應時， CompCode 及 MQCC_OK 和 MQRC_NONE 的原因不一定表示訊息已順利放入佇列。 開發使用非同步放置回應且需要確認訊息已放置到佇列的 MQI 應用程式時，您必須檢查來自放置作業的 CompCode 及「原因碼」，並使用 MQSTAT 來查詢非同步錯誤資訊。

雖然不會立即傳回每一個個別 MQPUT 或 MQPUT1 呼叫的成功或失敗，但稍後可以透過呼叫 MQSTAT 來判斷非同步呼叫下發生的第一個錯誤。

如果無法使用非同步放置回應遞送同步點下的持續訊息，且您嘗試確定交易，則確定會失敗，且會取消交易，完成碼為 MQCC_FAILED 且原因為 MQRC_BACKED_OUT。 應用程式可以呼叫 MQSTAT 來判斷先前 MQPUT 或 MQPUT1 失敗的原因。

MQPMO_SYNC_RESPONSE
指定此 PUT 回應類型可確保一律同步發出 MQPUT 或 MQPUT1 作業。 如果放置作業成功，則 MQMD 和 MQPMO 中的所有欄位都會完成。

此選項可確保同步回應，而不考慮佇列或主題物件上定義的預設放置回應值。

MQPMO_RESPONSE_AS_Q_DEF
如果針對 MQPUT 呼叫指定此值，則使用的放置回應類型會從應用程式第一次開啟佇列上指定的 DEFPRESP 值中取得。 如果用戶端應用程式連接至早於 7.0 版之層次的佇列管理程式，則其行為會如同已指定 MQPMO_SYNC_RESPONSE 一樣。

如果針對 MQPUT1 呼叫指定此選項，則在將要求傳送至伺服器之前， DEFPRESP 屬性值不明。 依預設，如果 MQPUT1 呼叫使用 MQPMO_SYNCPOINT，則其行為與 MQPMO_ASYNC_RESPONSE 相同，如果使用 MQPMO_NO_SYNCPOINT，則其行為與 MQPMO_SYNC_RESPONSE 相同。 不過，您可以透

過在用戶端配置檔中設定 Put1DefaultAlwaysSync 內容來置換此預設行為，請參閱 用戶端配置檔的 CHANNELS 段落。

MQPMO_RESPONSE_AS_TOPIC_DEF
MQPMO_RESPONSE_AS_TOPIC_DEF 是與主題物件搭配使用之 MQPMO_RESPONSE_AS_Q_DEF 的同義字。

其他選項。 下列選項控制授權檢查、佇列管理程式靜止時發生的情況，以及解析佇列和佇列管理程式名稱:

MQPMO_ALTERNATE_USER_AUTHORITY
MQPMO_ALTERNATE_USER_AUTHORITY 指出 MQPUT1 呼叫的 *ObjDesc* 參數中的 *AlternateUserId* 欄位包含使用者 ID，用來驗證將訊息放置在佇列上的權限。 只有在 *AlternateUserId* 獲授權以指定的選項開啟佇列時，不論執行應用程式的使用者 ID 是否獲授權開啟佇列，呼叫才會成功。 (這不適用於指定的環境定義選項，不過，一律會根據執行應用程式的使用者 ID 來檢查這些選項。)

此選項僅適用於 MQPUT1 呼叫。

MQPMO_FAIL_IF_QUIESCING
如果佇列管理程式處於靜止狀態，此選項會強制 MQPUT 或 MQPUT1 呼叫失敗。

在 z/OS 上，如果連線 (適用於 CICS 或 IMS 應用程式) 處於靜止狀態，則此選項也會強制 MQPUT 或 MQPUT1 呼叫失敗。

呼叫會傳回完成碼 MQCC_FAILED，原因碼為 MQRC_Q_MGR_QUIESCING 或 MQRC_CONNECTION_QUIESCING。

MQPMO_RESOLVE_LOCAL_Q
Use this option to fill *ResolvedQName* in the MQPMO structure with the name of the local queue to which the message is put, and *ResolvedQMgrName* with the name of the local queue manager that hosts the local queue. 如需 MQPMO_RESOLVE_LOCAL_Q 的相關資訊，請參閱主題 MQOO_RESOLVE_LOCAL_Q。

如果您已獲授權放入佇列中，則具有在 MQPUT 呼叫上指定此旗標的必要權限; 不需要特殊權限。

預設選項。 如果您不需要任何說明的選項，請使用下列選項:

MQPMO_NONE
使用這個值來指出未指定其他選項；所有選項都採用其預設值。 MQPMO_NONE 定義為輔助程式文件; 此選項不預期與任何其他選項搭配使用，但由於其值為零，因此無法偵測到此類使用。

MQPMO_NONE 是輸入欄位。 *Options* 欄位的起始值是 MQPMO_NONE。

OriginalMsg 控點 *(MQHMSG)*

這是訊息的選用控點。 先前可能已從佇列中擷取。 此控點的使用遵循 *Action* 欄位的值; 另請參閱 NewMsg 控點。

MQPUT 或 **MQPUT1** 呼叫不會變更原始訊息控點的內容。

這是輸入欄位。 此欄位的起始值為 **MQHM_NONE**。 如果版本小於 **MQPMO_VERSION_3**，則會忽略此欄位。

PubLevel (MQLONG)
此欄位的起始值為 9。 此發佈設為目標的訂閱層次。 只有 SubLevel 最高值小於或等於此值的訂閱才會接收此發佈。 此值必須在 0 到 9 的範圍內; 0 是最低層次。 不過，如果已保留發佈，則無法再供較高層次的訂閱者使用，因為它在 PubLevel 1 上重新發佈。

如需相關資訊，請參閱 截取出版品 。

PutMsgRecFields (MQLONG)

此欄位包含的旗標指出應用程式提供的放置訊息記錄中存在哪些 MQPMR 欄位。 只有在將訊息放入配送清單時，才使用 *PutMsgRecFields* 。 如果 *RecsPresent* 是零，或 *PutMsgRecOffset* 和 *PutMsgRecPtr* 兩者都是零，則會忽略此欄位。

對於呈現的欄位，佇列管理程式會針對每一個目的地使用對應放置訊息記錄中欄位的值。 對於不存在的欄位，佇列管理程式會使用 MQMD 結構中的值。

使用下列一或多個旗標，以指出放置訊息記錄中存在哪些欄位:

MQPMRF_MSG_ID
訊息 ID 欄位存在。

MQPMRF_CORREL_ID
存在相關性 ID 欄位。

MQPMRF_GROUP_ID
存在群組 ID 欄位。

MQPMRF_FEEDBACK
意見回饋欄位存在。

MQPMRF_ACCOUNTING_TOKEN
存在 accounting-token 欄位。

如果您指定此旗標，請在 *Options* 欄位中指定 MQPMO_SET_IDENTITY_CONTEXT 或 MQPMO_SET_ALL_CONTEXT; 如果未滿足此條件，則呼叫會失敗，原因碼為 MQRC_PMO_RECORD_FLAGS_ERROR。

如果沒有 MQPMR 欄位，則可以指定下列項目:

MQPMRF_NONE
沒有任何放置訊息記錄欄位。

如果指定此值，則 *RecsPresent* 必須為零，或 *PutMsgRecOffset* 及 *PutMsgRecPtr* 兩者都必須為零。

MQPMRF_NONE 定義為輔助程式文件。 此常數並非預期與任何其他常數一起使用，但由於其值為零，因此無法偵測此類使用。

如果 *PutMsgRecFields* 包含無效的旗標，或提供了放置訊息記錄，但 *PutMsgRecFields* 具有值 MQPMRF_NONE ，則呼叫會失敗，原因碼為 MQRC_PMO_RECORD_FLAGS_ERROR。

這是輸入欄位。 此欄位的起始值為 MQPMRF_NONE。 如果 *Version* 小於 MQPMO_VERSION_2，則會忽略此欄位。

PutMsgRecOffset (MQLONG)

這是從 MQPMO 結構開始第一個 MQPMR 放置訊息記錄的偏移 (以位元組為單位)。 偏移可以是正數或負數。 只有在將訊息放入配送清單時，才會使用 *PutMsgRecOffset* 。 如果 *RecsPresent* 為零，則會忽略該欄位。

將訊息放入配送清單時，可以提供一或多個 MQPMR 放置訊息記錄的陣列，以個別指定每個目的地的訊息特定內容; 這些內容如下:

- 訊息 ID
- 相關性 ID
- 群組 ID
- 回饋值
- 帳戶記號

您不需要指定所有這些內容，但無論您選擇哪一個子集，請以正確的順序指定欄位。 如需進一步詳細資料，請參閱 MQPMR 結構的說明。

通常，當開啟配送清單時，放置訊息記錄必須與 MQOD 指定的物件記錄一樣多; 每一個放置訊息記錄都會提供對應物件記錄所識別之佇列的訊息內容。 配送清單中無法開啟的佇列仍必須在陣列中的適當位置為它們配置放置訊息記錄，雖然在此情況下會忽略訊息內容。

放置訊息記錄數可能與物件記錄數不同。 如果放置訊息記錄少於物件記錄，則會從訊息描述子 MQMD 中的對應欄位取得沒有放置訊息記錄之目的地的訊息內容。 如果放置訊息記錄比物件記錄多，則不會使用多餘的 (雖然仍可能存取它們)。 放置訊息記錄是選用的，但如果提供它們，則必須有 *RecsPresent* 個。

透過在 *PutMsgRecOffset* 中指定偏移，或在 *PutMsgRecPtr* 中指定位址，以提供與 MQOD 中物件記錄類似的放置訊息記錄; 如需如何執行此動作的詳細資料，請參閱 第 413 頁的『MQOD-物件描述子』中說明的 *ObjectRecOffset* 欄位。

不能使用 *PutMsgRecOffset* 和 *PutMsgRecPtr* 中的多個; 如果兩者都不是零，則呼叫會失敗，原因碼為 MQRC_PUT_MSG_RECORDS_ERROR。

這是輸入欄位。 此欄位的起始值為 0。 如果 *Version* 小於 MQPMO_VERSION_2，則會忽略此欄位。

PutMsgRecPtr (MQPTR)

這是第一個 MQPMR 放置訊息記錄的位址。 只有在將訊息放入配送清單時，才使用 *PutMsgRecPtr* 。 如果 *RecsPresent* 為零，則會忽略該欄位。

您可以使用 *PutMsgRecPtr* 或 *PutMsgRecOffset* 來指定放置訊息記錄，但不能同時使用兩者; 如需詳細資料，請參閱上述 *PutMsgRecOffset* 欄位的說明。 如果您不使用 *PutMsgRecPtr*，請將它設為空值指標或空值位元組。

這是輸入欄位。 在那些支援指標的程式設計語言中，此欄位的起始值是空值指標，否則是全空值位元組字串。 如果 *Version* 小於 MQPMO_VERSION_2，則會忽略此欄位。

註: 在程式設計語言不支援指標資料類型的平台上，此欄位宣告為適當長度的位元組字串，起始值為 all-null 位元組字串。

RecsPresent (MQLONG)

這是應用程式所提供的 MQPMR 放置訊息記錄或 MQRR 回應記錄數目。 只有在將訊息放入配送清單時，此數字才可以大於零。 放置訊息記錄和回應記錄是選用的; 應用程式不需要提供任何記錄，也可以選擇只提供一種類型的記錄。 不過，如果應用程式提供這兩種類型的記錄，則必須提供每一種類型的 *RecsPresent* 記錄。

RecsPresent 的值不需要與配送清單中的目的地數目相同。 如果提供太多記錄，則不會使用過多記錄; 如果提供太少記錄，則會將預設值用於那些沒有放置訊息記錄之目的地的訊息內容 (請參閱 *PutMsgRecOffset*)。

如果 *RecsPresent* 小於零，或大於零，但未將訊息放入配送清單，則呼叫會失敗，原因碼為 MQRC_RECS_PRESENT_ERROR。

這是輸入欄位。 此欄位的起始值為 0。 如果 *Version* 小於 MQPMO_VERSION_2，則會忽略此欄位。

ResolvedQMgr 名稱 (MQCHAR48)

這是本端佇列管理程式執行名稱解析之後的目的地佇列管理程式名稱。 傳回的名稱是擁有 *ResolvedQName* 所識別之佇列的佇列管理程式名稱，且可以是本端佇列管理程式的名稱。

如果 *ResolvedQName* 是本端佇列管理程式所屬佇列共用群組所擁有的共用佇列，則 *ResolvedQMgrName* 是佇列共用群組的名稱。 如果佇列由某個其他佇列共用群組所擁有，則 *ResolvedQName* 可以是佇列共用群組的名稱或佇列共用群組成員的佇列管理程式名稱 (傳回值的本質由本端佇列管理程式中存在的佇列定義決定)。

只有在物件是單一佇列時，才會傳回非空白值; 如果物件是配送清單或主題，則傳回的值未定義。

這是輸出欄位。 此欄位的長度由 MQ_Q_MGR_NAME_LENGTH 提供。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

ResolvedQName (MQCHAR48)

這是在本端佇列管理程式執行名稱解析之後的目的地佇列名稱。 傳回的名稱是存在於 *ResolvedQMgrName* 所識別之佇列管理程式上的佇列名稱。

只有在物件是單一佇列時，才會傳回非空白值; 如果物件是配送清單或主題，則傳回的值未定義。

這是輸出欄位。 此欄位的長度由 MQ_Q_NAME_LENGTH 指定。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

ResponseRec 偏移 (MQLONG)

這是從 MQPMO 結構開始第一個 MQRR 回應記錄的偏移 (以位元組為單位)。 偏移可以是正數或負數。 只有在將訊息放入配送清單時，才會使用 *ResponseRecOffset* 。 如果 *RecsPresent* 為零，則會忽略該欄位。

將訊息放入配送清單時，您可以提供一或多個 MQRR 回應記錄的陣列，以識別訊息未順利傳送至其中的佇列 (MQRR 中的 *CompCode* 欄位) ，以及每次失敗的原因 (MQRR 中的 *Reason* 欄位)。 因為佇列無法開啟，或因為放置作業失敗，所以可能未傳送訊息。 只有在呼叫的結果混合時 (亦即，部分訊息已順利傳送，而其他訊息則失敗，或所有訊息皆失敗，但因不同原因) ，佇列管理程式才會設定回應記錄; 呼叫的原因碼 MQRC_MULTIPLE_REASONS 會指出此情況。 如果相同的原因碼套用至所有佇列，則會在 MQPUT 或 MQPUT1 呼叫的 *Reason* 參數中傳回該原因，且不會設定回應記錄。

在開啟配送清單時，通常會有與 MQOD 指定的物件記錄一樣多的回應記錄; 必要時，會將每一個回應記錄設為完成碼及原因碼，以放入對應物件記錄所識別的佇列。 配送清單中無法開啟的佇列仍必須在陣列中的適當位置為其配置回應記錄，雖然它們設定為開啟作業而非放置作業所產生的完成碼及原因碼。

回應記錄數目可能與物件記錄數目不同。 如果回應記錄少於物件記錄，應用程式可能無法識別放置作業失敗的所有目的地，或失敗的原因。 如果回應記錄比物件記錄多，則不會使用多餘的回應記錄 (雖然仍可能存取它們)。 回應記錄是選用的，但如果提供它們，則必須有 *RecsPresent* 個。

透過在 *ResponseRecOffset* 中指定偏移或在 *ResponseRecPtr* 中指定位址，以類似 MQOD 中物件記錄的方式提供回應記錄; 如需如何執行此動作的詳細資料，請參閱 第 413 頁的『MQOD-物件描述子』中說明的 *ObjectRecOffset* 欄位。 不過，請不要使用 *ResponseRecOffset* 和 *ResponseRecPtr* 中的一個以上; 如果兩者都不是零，則呼叫會失敗，原因碼為 MQRC_RESPONSE_RECORDS_ERROR。

對於 MQPUT1 呼叫，此欄位必須為零。 這是因為回應資訊 (如果要求的話) 會在物件描述子 MQOD 指定的回應記錄中傳回。

這是輸入欄位。 此欄位的起始值為 0。 如果 *Version* 小於 MQPMO_VERSION_2，則會忽略此欄位。

ResponseRecPtr (MQPTR)

這是第一個 MQRR 回應記錄的位址。 只有在將訊息放入配送清單時，才會使用 *ResponseRecPtr* 。 如果 *RecsPresent* 為零，則會忽略該欄位。

使用 *ResponseRecPtr* 或 *ResponseRecOffset* 來指定回應記錄，但不能同時指定兩者; 如需詳細資料，請參閱上方 *ResponseRecOffset* 欄位的說明。 如果您不使用 *ResponseRecPtr* ，請將它設為空值指標或空值位元組。

對於 MQPUT1 呼叫，此欄位必須是空值指標或空值位元組。 這是因為回應資訊 (如果要求的話) 會在物件描述子 MQOD 指定的回應記錄中傳回。

這是輸入欄位。 在那些支援指標的程式設計語言中，此欄位的起始值是空值指標，否則是全空值位元組字串。 如果 *Version* 小於 MQPMO_VERSION_2，則會忽略此欄位。

註: 在程式設計語言不支援指標資料類型的平台上，此欄位宣告為適當長度的位元組字串，起始值為 all-null 位元組字串。

StrucId (MQCHAR4)

這是結構 ID; 值必須是:

MQPMO_STRUC_ID
放置訊息選項結構的 ID。

對於 C 程式設計語言，也會定義常數 MQPMO_STRUC_ID_ARRAY; 此值與 MQPMO_STRUC_ID 相同，但卻是字元陣列而非字串。

這一律是輸入欄位。 此欄位的起始值是 MQPMO_STRUC_ID。

逾時 *(MQLONG)*

這是保留欄位; 其值不顯著。 此欄位的起始值為 -1。

UnknownDest 計數 *(MQLONG)*

這是現行 MQPUT 或 MQPUT1 呼叫已順利傳送至配送清單中解析為遠端佇列的佇列的訊息數。 佇列管理程式暫時保留在配送清單表單中的訊息，會視為那些配送清單包含的個別目的地數目。 當將訊息放入不在配送清單中的單一佇列時，也會設定此欄位。

這是輸出欄位。 此欄位的起始值為 0。 如果 *Version* 小於 MQPMO_VERSION_1，則不會設定此欄位。

在 z/OS 上未定義此欄位，因為不支援配送清單。

版本 *(MQLONG)*
結構版本號碼。

此值必須是下列其中一個：

MQPMO_VERSION_1
Version-1 放置訊息選項結構。

所有環境都支援此版本。

MQPMO_VERSION_2
Version-2 放置訊息選項結構。

下列環境支援此版本: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows，以及連接至這些系統的 WebSphere MQ MQI 用戶端。

MQPMO_VERSION_3
Version-3 放置訊息選項結構。

所有環境都支援此版本。

僅存在於結構最新版本中的欄位會在欄位說明中如此識別。 下列常數指定現行版本的版本號碼:

MQPMO_CURRENT_VERSION
放置訊息選項結構的現行版本。

這一律是輸入欄位。 此欄位的起始值為 MQPMO_VERSION_1。

MQPMO 的起始值及語言宣告

表 *529: MQPMO* 中欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQPMO_STRUC_ID	'PMO¬'
Version	MQPMO_VERSION_1	1
Options	MQPMO_NONE	0
Timeout	無	-1
Context	無	0
KnownDestCount	無	0
UnknownDestCount	無	0
InvalidDestCount	無	0
ResolvedQName	無	空字串或空白
ResolvedQMgrName	無	空字串或空白
RecsPresent	無	0
PutMsgRecFields	MQPMRF_NONE	0
PutMsgRecOffset	無	0
ResponseRecOffset	無	0

表 529: *MQPMO* 中欄位的起始值 (繼續)

欄位名稱	常數名稱	常數值
PutMsgRecPtr	無	空值指標或空值位元組
ResponseRecPtr	無	空值指標或空值位元組
OriginalMsgHandle	MQHM_NONE	0
NewMsgHandle	MQHM_NONE	0
Action	MQACTP_NEW	0
PubLevel	無	9

附註:

1. 符號 ¬ 代表單一空白字元。
2. 空值字串或空白值表示 C 中的空值字串，以及其他程式設計語言中的空白字元。
3. 在 C 程式設計語言中，巨集變數 MQPMO_DEFAULT 包含上述值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQPMO MyPMO = {MQPMO_DEFAULT};
```

C 宣告

```
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 */
  MQHOBJ    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 */
};
```

COBOL 宣告

```
**   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-KNOWNDESTCOUNT    PIC S9(9) BINARY.
**    Number of messages sent successfully to remote queues
   15 MQPMO-UNKNOWNDESTCOUNT  PIC S9(9) BINARY.
**    Number of messages that could not be sent
   15 MQPMO-INVALIDDESTCOUNT  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-PUTMSGRECPTR      POINTER.
**    Address of first response record
   15 MQPMO-RESPONSERECPTR    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.
```

PL/I 宣告

```
dcl
 1 MQPMO 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 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 */
```

High Level Assembler 宣告

```
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_KNOWNDESTCOUNT    DS   F     Number of messages sent successfully
*                                  to local queues
MQPMO_UNKNOWNDESTCOUNT  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
                        ORG  MQPMO
MQPMO_AREA              DS   CL(MQPMO_LENGTH)
```

Visual Basic 宣告

```
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
```

MQPMR-放置訊息記錄

下表彙總結構中的欄位。

表 530: MQPMR 中的欄位		
欄位	**說明**	**主題**
MsgId	訊息 ID	MsgId
CorrelId	相關性 ID	CorrelId
GroupId	群組 ID	GroupId
Feedback	回饋碼或原因碼	意見
AccountingToken	帳戶記號	AccountingToken

MQPMR 概觀

可用性: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows，以及連接至這些系統的 WebSphere MQ 用戶端。

目的: 將訊息放入配送清單時，請使用 MQPMR 結構來指定單一目的地的各種訊息內容。 MQPMR 是 MQPUT 及 MQPUT1 呼叫的輸入/輸出結構。

字集及編碼:MQPMR 中的資料必須採用 *CodedCharSetId* 佇列管理程式屬性所提供的字集，以及 MQENC_NATIVE 所提供本端佇列管理程式的編碼。 不過，如果應用程式以 MQ 用戶端身分執行，則結構必須採用用戶端的字集及編碼。

用法: 藉由在 MQPUT 或 MQPUT1 呼叫上提供這些結構的陣列，您可以為配送清單中的每一個目的地佇列指定不同的值。 部分欄位僅為輸入，其他欄位則為輸入/輸出。

註: 此結構不常見，因為它沒有固定的佈置。 此結構中的欄位是選用的，每一個欄位是否存在由 MQPMO 中 *PutMsgRecFields* 欄位中的旗標指示。 呈現 **的欄位必須依下列順序**出現:

- *MsgId*
- *CorrelId*
- *GroupId*
- *Feedback*
- *AccountingToken*

不存在的欄位在記錄中不佔用任何空間。

因為 MQPMR 沒有固定的佈置，所以標頭、COPY 及 INCLUDE 檔案中未提供其定義，以用於支援的程式設計語言。 應用程式設計師必須建立包含應用程式所需欄位的宣告，並在 *PutMsgRecFields* 中設定旗標以指出存在的欄位。

MQPMR 的欄位

MQPMR 結構包含下列欄位; 這些欄位按 **字母順序**說明:

AccountingToken (MQBYTE32)

這是結算記號，用於傳送至佇列的訊息，其名稱是由 MQOPEN 或 MQPUT1 呼叫上提供之 MQOR 結構陣列中的對應元素所指定。 其處理方式與 MQMD 中用於放置至單一佇列的 *AccountingToken* 欄位相同。 如需此欄位內容的相關資訊，請參閱 第 363 頁的『MQMD-訊息描述子』 中 *AccountingToken* 的說明。

如果此欄位不存在，則會使用 MQMD 中的值。

這是輸入欄位。

CorrelId (MQBYTE24)

這是相關性 ID ，用於傳送至佇列的訊息，其名稱由 MQOPEN 或 MQPUT1 呼叫上提供之 MQOR 結構陣列中的對應元素所指定。 其處理方式與 MQMD 中用於放置至單一佇列的 *CorrelId* 欄位相同。

如果 MQPMR 記錄中沒有此欄位，或 MQPMR 記錄少於目的地，則 MQMD 中的值會用於那些沒有 MQPMR 記錄包含 *CorrelId* 欄位的目的地。

如果指定 MQPMO_NEW_CORREL_ID，則不論它們是否具有 MQPMR 記錄，都會產生 單一 新的相關性 ID，並用於配送清單中的所有目的地。 這與 MQPMO_NEW_MSG_ID 的處理方式不同 (請參閱 *MsgId* 欄位)。

這是輸入/輸出欄位。

回饋 *(MQLONG)*

這是要用於傳送至佇列的訊息的回饋碼，其名稱由 MQOPEN 或 MQPUT1 呼叫上提供的 MQOR 結構陣列中的對應元素指定。 其處理方式與 MQMD 中用於放置至單一佇列的 *Feedback* 欄位相同。

如果此欄位不存在，則會使用 MQMD 中的值。

這是輸入欄位。

GroupId (MQBYTE24)

GroupId 是群組 ID，用於傳送至佇列的訊息，其名稱是由 MQOPEN 或 MQPUT1 呼叫上所提供 MQOR 結構陣列中的對應元素所指定。 其處理方式與 MQMD 中用於放置至單一佇列的 *GroupId* 欄位相同。

如果 MQPMR 記錄中沒有此欄位，或 MQPMR 記錄少於目的地，則 MQMD 中的值會用於那些沒有 MQPMR 記錄包含 *GroupId* 欄位的目的地。 該值的處理方式如 佇列上的實體順序中所記載，但有下列差異:

- GroupId 是從 QMName 及時間戳記建立。 因此，也要保留 GroupId 唯一的佇列管理程式名稱。 此外，請勿將佇列管理程式機器上的時鐘設回。
- 在將使用新群組 ID 的情況下，佇列管理程式會為每一個目的地產生不同的群組 ID (亦即，沒有兩個目的地具有相同的群組 ID)。
- 在將使用欄位中的值的情況下，呼叫會失敗，原因碼為 MQRC_GROUP_ID_ERROR

這是輸入/輸出欄位。

MsgId (MQBYTE24)

這是訊息 ID，用於傳送至佇列的訊息，其名稱由 MQOPEN 或 MQPUT1 呼叫上提供的 MQOR 結構陣列中的對應元素所指定。 其處理方式與 MQMD 中用於放置至單一佇列的 *MsgId* 欄位相同。

如果 MQPMR 記錄中沒有此欄位，或 MQPMR 記錄少於目的地，則 MQMD 中的值會用於那些沒有 MQPMR 記錄包含 *MsgId* 欄位的目的地。 如果該值為 MQMI_NONE，則會針對 每一個 這些目的地產生新的訊息 ID (亦即，沒有兩個目的地具有相同的訊息 ID)。

如果指定 MQPMO_NEW_MSG_ID，則不論它們是否具有 MQPMR 記錄，都會針對配送清單中的所有目的地產生新的訊息 ID。 這不同於 MQPMO_NEW_CORREL_ID 的處理方式 (請參閱 *CorrelId* 欄位)。

這是輸入/輸出欄位。

MQPMR 的起始值和語言宣告

未定義此結構的起始值，因為標頭、COPY 及 INCLUDE 檔案中未提供受支援程式設計語言的結構宣告。 範例宣告顯示如果所有欄位都是必要的，如何宣告結構。

C 宣告

```
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 */
};
```

COBOL 宣告

```
**   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).
```

PL/I 宣告

```
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 */
```

Visual Basic 宣告

```
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
```

MQRFH-規則和格式化標頭

本節說明規則和格式化標頭、它包含哪些欄位，以及這些欄位的起始值。

MQRFH 概觀

可用性: 所有 WebSphere MQ 系統，以及連接至這些系統的 WebSphere MQ MQI 用戶端。

目的:MQRFH 結構定義規則和格式化標頭的佈置。 使用此標頭以名稱/值配對形式傳送字串資料。

格式名稱:MQFMT_RF_HEADER。

字集及編碼: MQRFH 結構中的欄位 (包括 *NameValueString*) 位於字集及編碼中，如果 MQRFH 位於應用程式訊息資料開頭，則由 MQRFH 之前的標頭結構中的 *CodedCharSetId* 及 *Encoding* 欄位提供，或由 MQMD 結構中的那些欄位提供。

對於佇列名稱中有效的字元，字集必須是具有單位元組字元的字集。

MQRFH 的欄位

MQRFH 結構包含下列欄位; 這些欄位按 **字母順序**進行說明:

CodedCharSetId (MQLONG)

這會指定 *NameValueString* 之後的資料字集 ID; 它不適用於 MQRFH 結構本身中的字元資料。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。 可以使用下列特殊值:

MQCCSI_INHERIT
此結構 後面 的資料中的字元資料與此結構的字集相同。

佇列管理程式會將訊息中所傳送結構的這個值變更為結構的實際字集 ID。 如果沒有發生錯誤， MQGET 呼叫不會傳回值 MQCCSI_INHERIT。

如果 MQMD 中的 *PutApplType* 欄位值為 MQAT_BROKER ，則無法使用 MQCCSI_INHERIT。

此欄位的起始值為 MQCCSI_UNDEFINED。

編碼 *(MQLONG)*

這指定遵循 *NameValueString* 的資料的數值編碼; 它不適用於 MQRFH 結構本身的數值資料。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。

此欄位的起始值為 MQENC_NATIVE。

旗標 *(MQLONG)*

可以指定下列項目:

MQRFH_NONE
沒有旗標。

此欄位的起始值為 MQRFH_NONE。

格式 *(MQCHAR8)*

這會指定 *NameValueString* 之後的資料格式名稱。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。 此欄位的編碼規則與 MQMD 中 *Format* 欄位的編碼規則相同。

此欄位的起始值為 MQFMT_NONE。

NameValue 字串 *(MQCHARn)*

這是包含名稱/值配對的可變長度字串，格式如下:

```
name1 value1 name2 value2 name3 value3 ...
```

每一個名稱或值必須以一個以上空白字元與相鄰名稱或值區隔; 這些空白並不重要。 名稱或值可以包含有效的空白，方法是在名稱或值前面加上雙引號並加上字尾; 將左雙引號與右雙引號之間的所有字元視為有效。 在下列範例中，名稱是 FAMOUS_WORDS，值是 Hello World:

```
FAMOUS_WORDS "Hello World"
```

名稱或值可以包含空值字元以外的任何字元 (作為 *NameValueString* 的定界字元)。 不過，為了協助交互作業能力，應用程式可以將名稱限制為下列字元:

- 第一個字元: 大寫或小寫英文字母 (A 到 Z 或 a 到 z) 或底線。
- 後續字元: 大寫或小寫英文字母、十進位數 (0 到 9)、底線、連字號或點。

如果名稱或值包含一個以上雙引號，則名稱或值必須以雙引號括住，且字串內的每一個雙引號都必須加倍:

```
Famous_Words "The program displayed ""Hello World"""
```

名稱和值區分大小寫，亦即小寫字母不視為與大寫字母相同。 例如， FAMOUS_WORDS 和 Famous_Words 是兩個不同的名稱。

NameValueString 的長度 (以位元組為單位) 等於 *StrucLength* 減去 MQRFH_STRUC_LENGTH_FIXED。 為了避免在某些環境中轉換使用者資料時發生問題，請將此長度設為四的倍數。 以空白填補 *NameValueString* ，或在字串中最後一個有效字元之後加上空值字元，以提早終止它。 空值字元及其後面的位元組會被忽略，直到指定的 *NameValueString* 長度為止。

註: 因為此欄位的長度不固定，所以會從提供給受支援程式設計語言之結構的宣告中省略該欄位。

StrucId (MQCHAR4)

這是結構 ID; 值必須是:

MQRFH_STRUC_ID
規則及格式化標頭結構的 ID。

對於 C 程式設計語言，也會定義常數 MQRFH_STRUC_ID_ARRAY; 此值與 MQRFH_STRUC_ID 相同，但卻是字元陣列而非字串。

此欄位的起始值是 MQRFH_STRUC_ID。

StrucLength (MQLONG)

這是 MQRFH 結構的長度 (以位元組為單位)，包括結構結尾的 *NameValueString* 欄位。 此長度 不 包括 *NameValueString* 欄位後面的任何使用者資料。

為了避免在某些環境中轉換使用者資料時發生問題， *StrucLength* 必須是四的倍數。

下列常數提供結構 固定 部分的長度，即不包括 *NameValueString* 欄位的長度:

MQRFH_STRUC_LENGTH_FIXED
MQRFH 結構的固定部分長度。

此欄位的起始值為 MQRFH_STRUC_LENGTH_FIXED。

版本 *(MQLONG)*

這是結構版本號碼; 值必須是:

MQRFH_VERSION_1
Version-1 規則和格式化標頭結構。

此欄位的起始值為 MQRFH_VERSION_1。

MQRFH 的起始值和語言宣告

表 *531: MQRFH* 中 *MQRFH* 欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQRFH_STRUC_ID	'RFH¬'
Version	MQRFH_VERSION_1	1
StrucLength	MQRFH_STRUC_LENGTH_FIXED	32
Encoding	MQENC_NATIVE	取決於環境
CodedCharSetId	未定義 MQCCSI_UNDEFINED	0
Format	MQFMT_NONE	空白
Flags	MQRFH_NONE	0

附註:

1. 符號 ¬ 代表單一空白字元。
2. 在 C 程式設計語言中，巨集變數 MQRFH_DEFAULT 包含上述值。 您可以用下列方式來提供結構中欄位的起始值:

```
MQRFH MyRFH = {MQRFH_DEFAULT};
```

C 宣告

```
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 */
};
```

COBOL 宣告

```
**   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.
```

PL/I 宣告

```
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 */
```

High Level Assembler 宣告

```
MQRFH                 DSECT
MQRFH_STRUCID         DS   CL4  Structure identifier
MQRFH_VERSION         DS   F    Structure version number
MQRFH_STRUCLENGTH     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)
```

Visual Basic 宣告

```
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
```

MQRFH2 -規則和格式化標頭 2

本節說明規則和格式化標頭 2、它包含哪些欄位，以及那些欄位的起始值。

MQRFH2 概觀

可用性

所有 WebSphere MQ 系統，以及連接至這些系統的 WebSphere MQ MQI 用戶端。

用途

MQRFH2 標頭以 MQRFH 標頭為基礎，但它容許在不轉換的情況下傳輸 Unicode 字串，而且它可以包含數值資料類型。

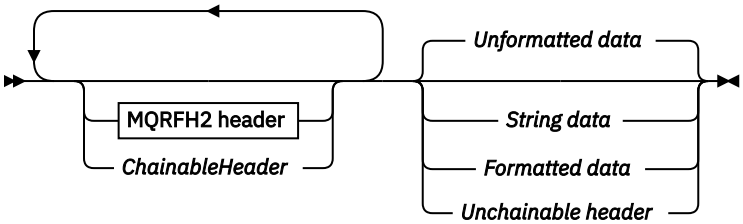
MQRFH2 結構定義 version-2 版規則及格式化標頭的格式。 使用此標頭來傳送已使用 XML 型語法編碼的資料。 訊息可以包含系列中的兩個以上 MQRFH2 結構，使用者資料可以選擇性地在系列中最後一個 MQRFH2 結構之後。

格式名稱

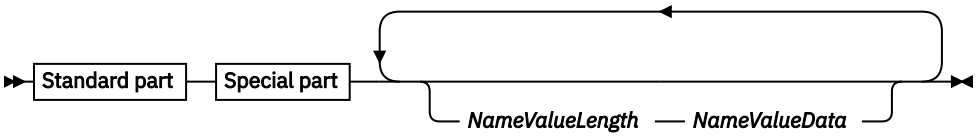
MQFMT_RF_HEADER_2

Syntax

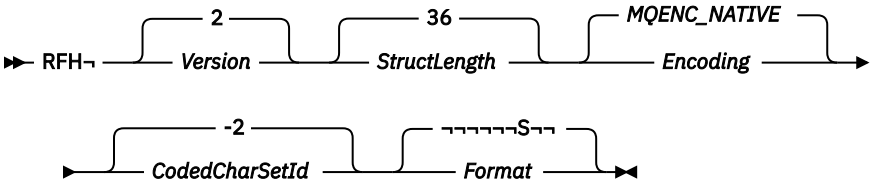
WebSphere MQ Message



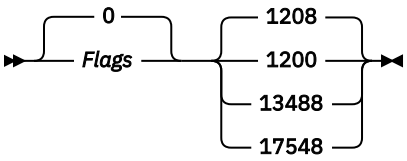
MQRFH2 header



Standard part



Special part



字集和編碼

特殊規則適用於 MQRFH2 結構所使用的字集及編碼:

- *NameValueData* 以外的欄位是在字集及編碼中，如果 MQRFH2 是在應用程式訊息資料的開頭，則由標頭結構中 MQRFH2 之前的 *CodedCharSetId* 及 *Encoding* 欄位所提供，或由 MQMD 結構中的那些欄位所提供。

  對於佇列名稱中有效的字元，字集必須是具有單位元組字元的字集。

  在 MQGET 呼叫上指定 MQGMO_CONVERT 時，佇列管理程式會將 *NameValueData* 以外的 MQRFH2 欄位轉換為所要求的字集及編碼。

- *NameValueData* 在 *NameValueCCSID* 欄位提供的字集中。 僅列出的 Unicode 字集對 *NameValueCCSID* 有效; 如需詳細資料，請參閱 *NameValueCCSID* 的說明。

  部分字集具有視編碼而定的表示法。 如果 *NameValueCCSID* 是這些字集之一，則 *NameValueData* 必須採用與 MQRFH2 中其他欄位相同的編碼。

  在 MQGET 呼叫上指定 MQGMO_CONVERT 時，佇列管理程式會將 *NameValueData* 轉換為所要求的編碼，但不會變更其字集。

MQRFH2 的欄位

MQRFH2 結構包含下列欄位; 這些欄位按 **字母順序**說明:

CodedCharSetId (MQLONG)

這指定最後一個 *NameValueData* 欄位之後的資料字集 ID; 它不適用於 MQRFH2 結構本身中的字元資料。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。 可以使用下列特殊值:

MQCCSI_INHERIT
此結構 後面 的資料中的字元資料與此結構的字集相同。

佇列管理程式會將訊息中所傳送結構的這個值變更為結構的實際字集 ID。 如果沒有發生錯誤， MQGET 呼叫不會傳回值 MQCCSI_INHERIT。

如果 MQMD 中的 *PutApplType* 欄位值為 MQAT_BROKER ，則無法使用 MQCCSI_INHERIT。

此欄位的起始值為 MQCCSI_INHERIT。

編碼 *(MQLONG)*

這會指定最後一個 *NameValueData* 欄位之後的資料數值編碼; 它不適用於 MQRFH2 結構本身中的數值資料。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。

此欄位的起始值為 MQENC_NATIVE。

旗標 *(MQLONG)*

此欄位的起始值為 MQRFH_NONE。 MQRFH_NONE 必須指定。

MQRFH_NONE
沒有旗標。

MQRFH_INTERNAL
MQRFH2 標頭包含內部設定內容。

MQRFH_INTERNAL 供佇列管理程式使用。

前 16 個位元 (MQRFH_FLAGS_RESTRICTED_MASK) 保留給佇列管理程式集合的旗標。 使用者可能設定的旗標定義在最後 16 個位元中。

格式 *(MQCHAR8)*

這會指定最後一個 *NameValueData* 欄位之後的資料格式名稱。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。 此欄位的編碼規則與 MQMD 中 *Format* 欄位的編碼規則相同。

此欄位的起始值為 MQFMT_NONE。

NameValueCCSID (MQLONG)

這會在 *NameValueData* 欄位中指定資料的編碼字集 ID。 這不同於 MQRFH2 結構中其他字串的字集，且可能不同於結構結尾最後一個 *NameValueData* 欄位後面的資料字集 (如果有的話)。

NameValueCCSID 必須具有下列其中一個值:

CCSID	**意義**
1200	UCS-2 開放式
13488	UCS-2 2.0 子集
17584	UCS-2 2.1 子集 (包括歐元符號)
1208	UTF-8

對於 UCS-2 字集， *NameValueData* 的編碼 (位元組順序) 必須與 MQRFH2 結構中其他欄位的編碼相同。 不支援代理字元 (X'D800'至 X'DFFF')。

註: 如果 *NameValueCCSID* 沒有上述其中一個值，且 MQRFH2 結構在 MQGET 呼叫上需要轉換，則呼叫會完成，原因碼為 MQRC_SOURCE_CCSID_ERROR ，且會傳回未轉換的訊息。

此欄位的起始值為 1208。

NameValue 資料 *(MQCHARn)*
NameValueData 是一個可變長度欄位，包含包含訊息內容名稱/值配對的資料夾。 資料夾是包含使用 XML 類似語法編碼之資料的可變長度字串。 字串的長度 (以位元組為單位) 由 *NameValueData* 欄位之前的 *NameValueLength* 欄位提供。 長度必須是 4 的倍數。

NameValueLength 和 *NameValueData* 欄位是選用的，但如果有的話，它們必須成對出現且相鄰。 欄位配對可以根據需要重複多次，例如:

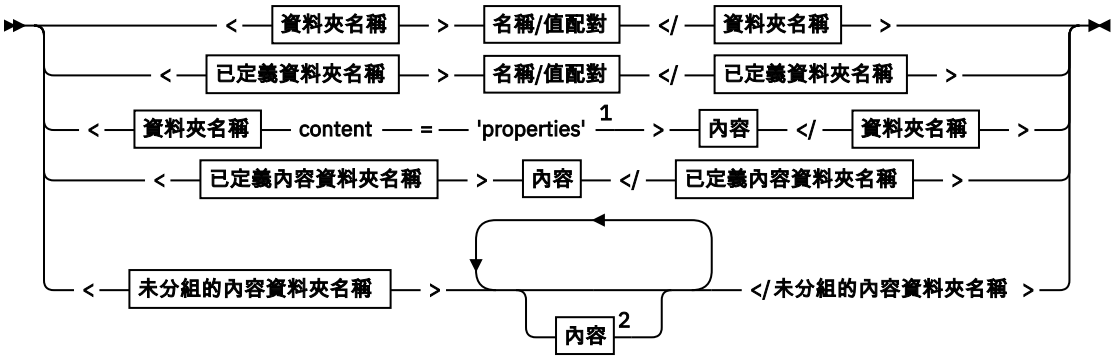
```
length1 data1 length2 data2 length3 data3
```

NameValueData 不會 轉換為 MQGET 呼叫上指定的字集。 即使使用有效的 MQGMO_CONVERT 選項擷取訊息， *NameValueData* 也會保留在其原始字集中。 不過， *NameValueData* 會 轉換為 MQGET 呼叫上指定的編碼。

註: 因為這些欄位是選用的，所以會從針對所支援各種程式設計語言所提供的結構宣告中省略它們。

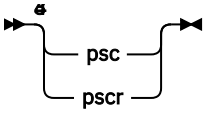
註: 語法圖中使用術語 "defined" 和 "reserved" 。 "已定義" 表示 IBM WebSphere MQ 使用該名稱。 "保留" 表示保留名稱供 WebSphere MQ 未來使用。

NameValueData 語法

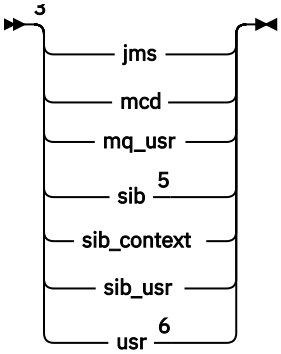


資料夾名稱

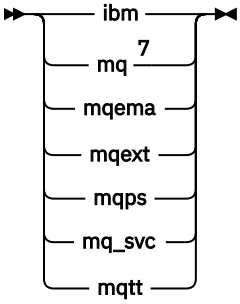
已定義資料夾名稱



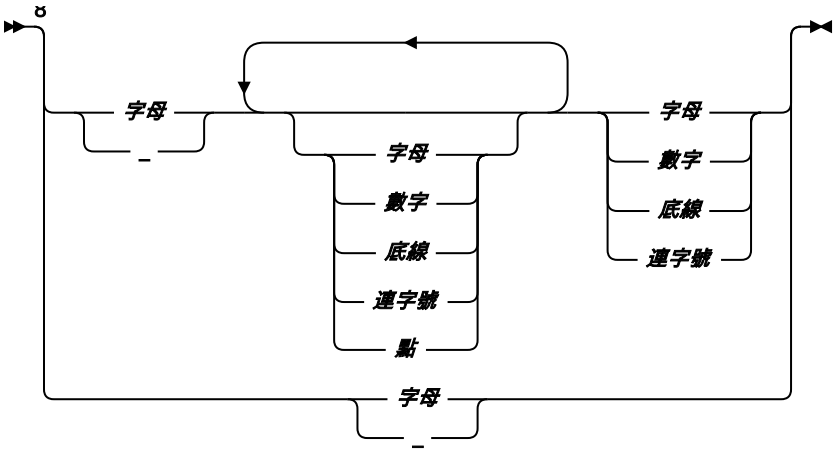
已定義內容資料夾名稱



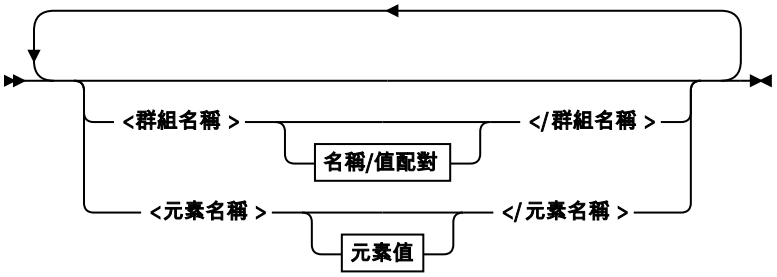
未分組的內容資料夾名稱



名稱



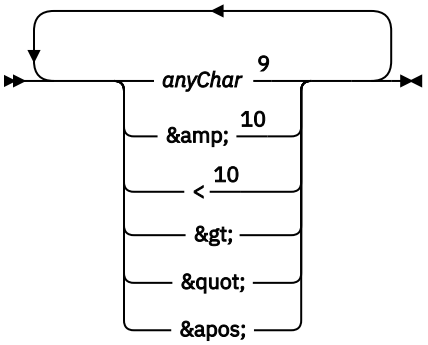
名稱/值配對



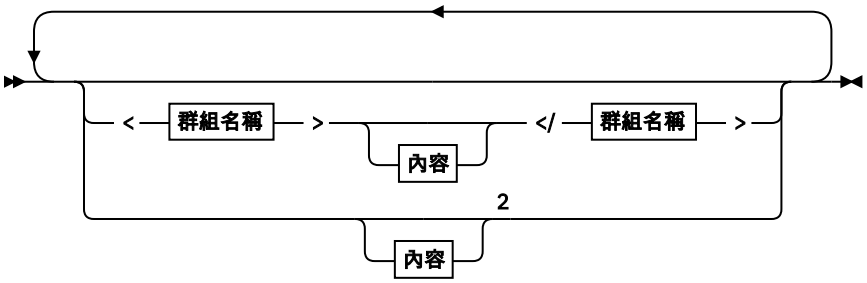
群組名稱

元素名稱

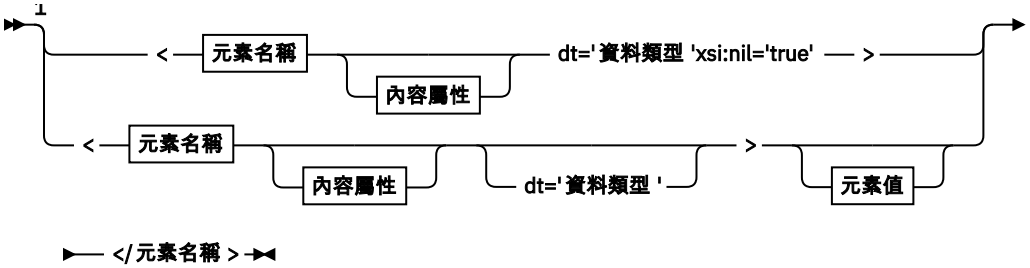
元素值



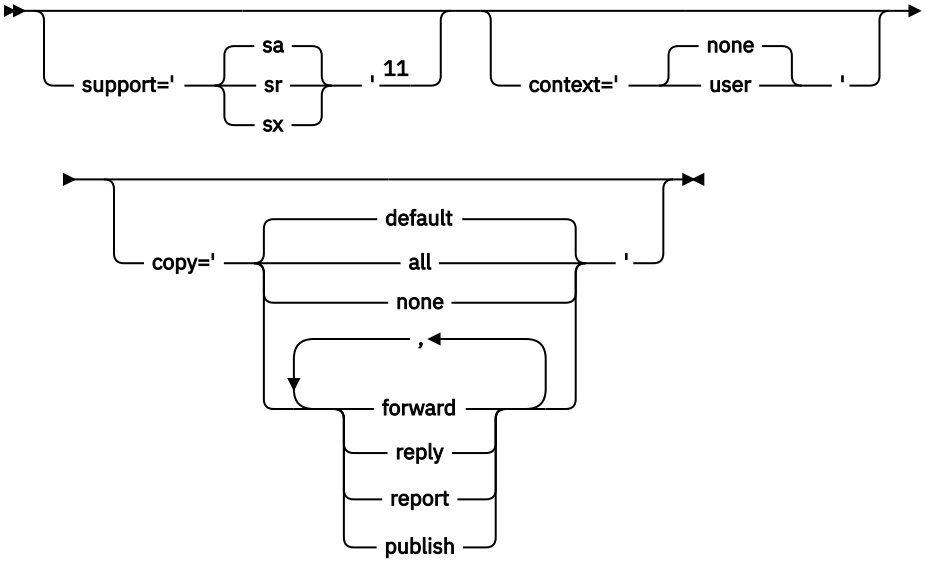
內容



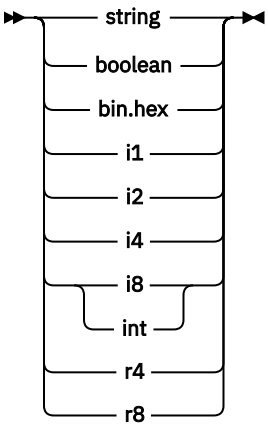
內容



內容屬性



資料類型



註:

[1] 雙引號或單引號是有效的。

[2] 請勿使用無效的內容名稱; 請參閱 第 471 頁的『內容名稱無效』。 請只將保留內容名稱用於其定義的目的; 請參閱 第 471 頁的『已定義的內容名稱』。

[3] 名稱必須是小寫。

[4] 僅支援一個 psc 和 pscr 資料夾。

[5] 只有第一個 MQRFH2 標頭中的內容才有意義。 WebSphere Application Server 服務 Integration Bus 會忽略後續 MQRFH2 標頭中的 sib、 sib_context 及 sib_usr 資料夾。

[6] MQRFH2 中不得存在多個 usr 資料夾。 usr 資料夾中的內容不得多次出現。

[7] 只有第一個 mq 資料夾中的內容才有意義。 如果資料夾是 UTF-8，則只支援單位元組 UTF-8 字元。 唯一的空格字元是 Unicode U+0020。

[8] 有效字元定義在 W3C XML 規格中，且基本上由 Unicode 種類 Ll, Lu, Lo, Lt, Nl, Mc, Mn, Lm, 及 Nd 組成。
[9] 所有字元都很重要。 前導和尾端空白是元素值的一部分。
[10] 請勿使用無效字元; 請參閱 第 471 頁的『無效字元』。 請使用 ESC 序列，而不是這些無效字元。
[11] 支援內容屬性僅適用於 mq 資料夾

資料夾名稱

NameValueData 包含單一資料夾。 若要建立多個資料夾，請建立多個 *NameValueData* 欄位。 您可以在訊息內的單一 MQRFH2 標頭中建立多個 *NameValueData* 欄位。 或者，您可以建立多個鏈結的 MQRFH2 標頭，每一個都包含多個 *NameValueData* 欄位。

MQRFH2 標頭的順序及 *NameValueData* 欄位的順序對資料夾的邏輯內容沒有任何不同。 如果同一資料夾在訊息中出現多次，則資料夾會剖析為整體。 如果在相同資料夾的多個實例中出現相同的內容，則會將它剖析為清單。

MQRFH2 的正確剖析不受資料夾實際儲存在訊息中的替代方式影響。

有四個資料夾未遵循此規則。 只會剖析 mq、 sib、 sib_context 和 sib_usr 資料夾的第一個實例。

如果相同的內容在鏈結的 MQRFH2 標頭的合併內容中出現多次，則只會剖析內容的第一個實例。 如果使用 API 呼叫 (例如 MQSETMP) 來設定內容，並由應用程式直接新增至 MQRFH2 ，則會優先使用 API 呼叫。

資料夾名稱是包含名稱/值配對或群組的資料夾名稱。 群組和名稱/值配對可以在資料夾樹狀結構中的相同層次混合; 請參閱 第 462 頁的圖 1。 請勿結合群組名稱與元素名稱; 請參閱 第 462 頁的圖 2

```
<group1><nvp1>value</nvp1></group1><group2><nvp2>value</nvp2></group2>
<group3><nvp1>value</nvp1></group3><nvp3>value</nvp3>
```

圖 *1:* 正確使用群組及名稱/值配對

```
<group1><nvp1>value</nvp1>value</group1>
```

圖 *2:* 不正確使用群組及名稱/值配對

請勿使用無效或保留的資料夾名稱; 請參閱 第 471 頁的『無效的路徑名稱』 和 第 470 頁的『保留資料夾或內容資料夾名稱』。 僅基於已定義的目的使用已定義的資料夾名稱; 請參閱 第 463 頁的『已定義資料夾名稱』。

如果您將屬性 'content=properties' 新增至資料夾名稱標籤，該資料夾會變成內容資料夾; 請參閱 第 462 頁的圖 3。

```
<myFolder></myfolder>
<myPropertyFolder contents='properties'></myPropertyFolder>
```

圖 *3:* 資料夾和內容資料夾的範例

資料夾名稱區分大小寫。 資料夾名稱和內容資料夾名稱共用相同的名稱空間。 它們必須有不同的名稱。 第 462 頁的圖 4 中的 Folder1 必須不同於 第 463 頁的圖 5 中的 Folder2 。

```
<Folder1><NVP1>value</NVP1></Folder1>
```

圖 *4: Folder1* 名稱空間

```
<Folder2 content='properties'><Property1>value</Property1></Folder2>
```

圖 *5: Folder2* 名稱空間

不同資料夾中的群組、內容及名稱/值配對具有不同的名稱空間。 第 463 頁的圖 5 中的 Property1 與 第 463 頁的圖 6 中的 Property1 是不同的內容。

```
<Folder3 content='properties'><Property1>value</Property1></Folder3>
```

圖 *6: Folder3* 名稱空間

在兩個重要方面，內容資料夾與非內容資料夾不同:

1. 內容資料夾包含內容，非內容資料夾包含名稱/值配對。 資料夾在語法上略有不同。
2. 使用定義的介面 (例如 MQI 內容或 JMS 訊息內容) 來存取訊息內容。 這些介面可確保 MQRFH2 中的內容資料夾形式完整。 形式完整的內容資料夾可在不同平台及不同版次上的佇列管理程式之間交互作業。

訊息內容 MQI 是讀取及寫入 MQRFH2 的強大方式，可避免正確剖析 MQRFH2 的困難。

已定義資料夾名稱

定義的資料夾名稱是保留給 WebSphere MQ 或其他產品使用的資料夾名稱。 請勿建立同名的資料夾，且不要將您自己的名稱/值配對新增至資料夾。 已定義的資料夾為 psc 和 pscr。

psc 和 pscr 由排入佇列的發佈/訂閱使用。

放置了 MQMF_SEGMENT 或 MQMF_SEGMENTATION_ALLOWED 的分段訊息不能包含具有已定義資料夾名稱的 MQRFH2 。 MQPUT 失敗，原因碼為 2443， MQRC_SEGMENTATION_NOT_ALLOWED。

已定義內容資料夾名稱

已定義的內容資料夾名稱是 IBM WebSphere MQ 或另一個產品所使用的內容資料夾名稱。 如需資料夾及其內容的名稱，請參閱 內容資料夾。 定義的內容資料夾名稱是 WebSphere MQ 所保留的所有資料夾名稱的子集; 請參閱 第 470 頁的『保留資料夾或內容資料夾名稱』。

儲存在已定義內容資料夾中的任何元素都是內容。 儲存在已定義的內容資料夾中的元素不得有 content='properties' 屬性。

您只能將內容新增至已定義的內容資料夾 usr、 mq_usr 及 sib_usr。 在其他內容資料夾 (例如 mq 和 sib) 中， WebSphere MQ 會忽略或擲出它無法辨識的內容。

每一個已定義內容資料夾的說明會列出 IBM WebSphere MQ 已定義可供應用程式使用的內容。 部分內容是透過設定或取得 JMS 內容來間接存取，部分則是使用 MQSETMP 及 MQINQMP MQI 呼叫直接存取。

已定義的內容資料夾也包含 IBM WebSphere MQ 已保留但應用程式沒有存取權的其他內容。 未列出保留內容的名稱。 usr、 mq_usr 及 sib_usr 內容資料夾中沒有保留內容。 但不要建立內容名稱無效的內容; 請參閱 第 471 頁的『內容名稱無效』。

內容資料夾

jms

jms 包含 JMS 標頭欄位，以及無法在 MQMD 中完整表達的 JMSX 內容。 jms 資料夾一律呈現在 JMS MQRFH2 中。

表 *532: jms* 內容名稱、同義字、資料類型及資料夾			
內容同義字	**內容名稱**	**資料類型**	**資料夾**
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>
JMSPriority	jms.Pri	i4	<jms><Pri>*priority*</Pri></jms>
JMSReplyTo	jms.Rto	string	<jms><Rto>*replyToURI*</Rto></jms>
JMSTimeStamp	jms.Tms	i8	<jms><Tms>*timestamp*</Tms></jms>
JMSXGroupID	jms.Gid	string	<jms><Gid>*groupId*</Gid></jms>
JMSXGroupSeq	jms.Seq	i4	<jms><Seq>*messageSequenceNo*</Seq></jms>

請勿在 jms 資料夾中新增您自己的內容。

mcd

mcd 包含說明訊息格式的內容。 例如，訊息服務網域 Msd 內容將 JMS 訊息識別為 JMSTextMessage、JMSBytesMessage、JMSStreamMessage、JMSMapMessage、JMSObjectMessage 或空值。

mcd 資料夾一律存在於包含 MQRFH2 的 JMS 訊息中。

它一律存在於包含從 WebSphere Message Broker 傳送之 MQRFH2 的訊息中。 它會說明訊息的網域、格式、類型及訊息集。

表 *533: mcd* 內容名稱、同義字、資料類型及資料夾			
內容同義字	**內容名稱**	**資料類型**	**資料夾**
	mcd.Msd	string	<mcd><Msd>*messageDomain*</Msd></mcd>
	mcd.Set	string	<mcd><Set>*messageDomain*</Set></mcd>
	mcd.Type	string	<mcd><Type>*messageDomain*</Type></mcd>
	mcd.Fmt	string	<mcd><Fmt>*messageDomain*</Fmt></mcd>

請勿在 mcd 資料夾中新增您自己的內容。

mq_usr

mq_usr 包含未公開為 JMS 使用者定義內容的應用程式定義內容。 不符合 JMS 需求的內容可以放在這個資料夾中。

您可以在 mq_usr 資料夾中建立內容。 您在 mq_usr 中建立的內容類似於您在新資料夾中使用 content='properties' 屬性所建立的內容。

sib

sib 包含 WebSphere Application Server 服務整合匯流排 (WAS/SIB) 系統訊息內容。 sib 內容不會以 JMS 內容形式向 IBM WebSphere MQ JMS 應用程式公開，因為它們不是受支援的類型。 例如，部分 sib 內容無法公開為 JMS 內容，因為它們是位元組陣列。 部分 sib 內容會以 JMS_IBM_* 內容形式向 WAS/SIB 應用程式公開; 這些包括正向和反向遞送路徑內容。

請勿在 sib 資料夾中新增您自己的內容。

sib_context

sib_context 包含未向 WAS/SIB 使用者應用程式公開或作為 JMS 內容的 WAS/SIB 系統訊息內容。 sib_context 包含用於 Web 服務的安全和交易式內容。

請勿在 sib_context 資料夾中新增您自己的內容。

sib_usr

sib_usr 包含未公開為 JMS 使用者內容的 WAS/SIB 使用者訊息內容，因為它們不是支援的類型。 sib_usr 在 SIMessage 介面中向 WAS/SIB 應用程式公開; 請參閱 開發服務整合。

sib_usr 內容的類型必須是 bin.hex，且值必須採用正確的格式。 如果 IBM WebSphere MQ 應用程式以錯誤格式將 bin.hex 類型化元素寫入資料夾，則應用程式會收到 IOException。 如果內容的資料類型不是 bin.hex ，應用程式會收到 ClassCastException。

請勿嘗試利用這個資料夾，將 JMS 使用者內容提供給 WAS/SIB; 請改用 usr 資料夾。

您可以在 sib_usr 資料夾中建立內容。

usr

usr 包含與訊息相關聯的應用程式定義 JMS 內容。 僅當應用程式已設定應用程式定義的內容時，才會呈現 usr 資料夾。

usr 是預設內容資料夾。 如果設定內容時沒有資料夾名稱，則會將它放置在 usr 資料夾中。

表 *534: usr* 內容名稱、同義字、資料類型及資料夾.

Web 服務內容值在 MQRFH2 SOAP 設定 中說明

內容同義字	內容名稱	資料類型	資料夾
	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>

您可以在 usr 資料夾中建立內容。

放置有 MQMF_SEGMENT 或 MQMF_SEGMENTATION_ALLOWED 的分段訊息不能包含具有已定義內容資料夾名稱的 MQRFH2。 MQPUT 失敗，原因碼為 2443， MQRC_SEGMENTATION_NOT_ALLOWED。

未分組的內容資料夾名稱

ibm

ibm 包含僅供 IBM WebSphere MQ 使用的內容。

表 *535: ibm* 內容名稱、同義字、資料類型及資料夾

內容同義字	內容名稱	資料類型	資料夾
	ibm.rfp	string	<ibm><rfp>fingerprint</rfp></ibm>

請勿在 ibm 資料夾中新增您自己的內容。

mq

mq 包含僅供 IBM WebSphere MQ 使用的內容。

下列限制適用於 mq 資料夾中的內容:

- MQ 只會處理訊息中第一個重要 mq 資料夾中的內容; 訊息中任何其他 mq 資料夾中的內容都會被忽略。
- 資料夾中只接受單位元組 UTF-8 字元。 資料夾中的多位元組字元可能會導致剖析失敗，並拒絕訊息。
- 請勿在資料夾中使用跳出字串。 跳出字串會被視為元素的實際值。
- 資料夾內只會將 Unicode 字元 U+0020 視為空格。 所有其他字元都會視為重要字元，且會導致剖析資料夾失敗，以及拒絕訊息。

如果剖析 mq 資料夾失敗，或資料夾未遵守這些限制，則會拒絕訊息，原因碼為 2527, MQRC_RFH_RESTRICTED_FORMAT_ERR。

請勿在 mq 資料夾中新增您自己的內容。

mqema

mqema 包含僅供 WebSphere Application Server 使用的內容。 資料夾已取代為 mqext。

請勿在 mqema 資料夾中新增您自己的內容。

mqext

mqext 包含僅供 WebSphere Application Server 使用的內容。 只有在應用程式至少已設定其中一個 IBM 定義的內容時，資料夾才會存在。

表 *536: mqext* 內容名稱、同義字、資料類型及資料夾

內容同義字	內容名稱	資料類型	資料夾
JMSArmCorrelator	mqext.Arm	string	<mqext><Arm>*armCorrelator*</Arm></mqext>
JMSRMCorrelator	mqext.Wrm	string	<mqext><Wrm>*wrmCorrelator*</Wrm></mqext>

請勿在 mqext 資料夾中新增您自己的內容。

mqps

mqps 包含僅供 IBM WebSphere MQ 發佈/訂閱使用的內容。 只有在應用程式已設定至少其中一個整合的發佈/訂閱內容時，該資料夾才存在。

表 *537: mqps* 內容名稱、同義字、資料類型及資料夾			
內容同義字	**內容名稱**	**資料類型**	**資料夾**
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>
MQPubStrIntData	mqpse.Sid	string	<mqps><Sid>*publicationData*</Sid></mqps>
MQPubFormat	mqpse.Pfmt	i8	<mqps><Pfmt>*messageFormat*</Pfmt></mqps>

請勿在 mqps 資料夾中新增您自己的內容。

mq_svc

mq_svc 包含 SupportPac MA93 使用的內容。

請勿在 mq_svc 資料夾中新增您自己的內容。

mqtt

mqtt 包含 IBM WebSphere MQ Telemetry 使用的內容

表 *538: mqtt* 內容名稱、同義字、資料類型及資料夾			
內容同義字	**內容名稱**	**資料類型**	**資料夾**
	mqtt.clientId	string	<mqtt><clientId>*topicString*</clientId></mqtt>
	mqtt.qos	i4	<mqtt><qos>*qualityOfService*</qos></mqtt>
	mqtt.msgid	string	<mqtt><msgid>*messageIdentifier*</msgid></mqtt>

請勿在 mqtt 資料夾中新增您自己的內容。

放置了 MQMF_SEGMENT 或 MQMF_SEGMENTATION_ALLOWED 的分段訊息不能包含具有未分組內容資料夾名稱的 MQRFH2。 MQPUT 失敗，原因碼為 2443， MQRC_SEGMENTATION_NOT_ALLOWED。

名稱/值配對

在語法圖中， "名稱/值配對" 說明一般資料夾的內容。 一般資料夾包含群組和元素。 元素是名稱/值配對。 群組包含元素及其他群組。

就樹狀結構而言，元素是葉節點，群組是內部節點。 內部節點和資料夾 (根節點) 可以包含內部節點和葉節點的混合。 節點不能同時為內部節點和葉節點; 請參閱 第 462 頁的圖 2。

內容

在語法圖中， "內容" 說明內容資料夾的內容。 內容資料夾包含群組及內容。 內容是具有選用資料類型屬性的名稱/值配對。 群組包含內容及其他群組。

就樹狀結構而言，內容是葉節點，群組是內部節點。 內部節點和內容資料夾 (即根節點) 可以包含內部節點和葉節點的混合。 節點不能同時為內部節點和葉節點; 請參閱 第 462 頁的圖 2。

內容

訊息內容是內容資料夾中的名稱/值配對。 它可以選擇性地包含資料類型屬性及內容屬性; 如需範例，請參閱 第 468 頁的圖 7。 如果省略資料類型屬性，則內容類型為 `string`。

```
<pf><p1 dt='i8' >value</p1></pf>
```

圖 *7:* 資料類型屬性

訊息內容的名稱是其完整路徑名稱，並以類似 XML 的 <> 語法取代。 例如，`myPropertyFolder1.myGroup1.myGroup2.myProperty1` 對映至 第 468 頁的圖 8 中的 *NameValueData* 字串。 字串已格式化以更容易讀取。

```
<myPropertyFolder1>
    <myGroup1>
        <myGroup2>
            <myProperty1>value</myProperty1>
        </myGroup2>
    </myGroup1>
</myPropertyFolder1>
```

圖 *8:* 單一內容名稱對映

內容資料夾可以包含多個內容。 例如， 第 468 頁的圖 9 中的內容對映至 第 468 頁的圖 10 中的內容資料夾

```
myPropertyFolder1.myProperty4
myPropertyFolder1.myGroup1.myGroup2.myProperty1
myPropertyFolder1.myGroup1.myGroup2.myProperty2
myPropertyFolder1.myGroup1.myProperty3
```

圖 *9:* 多個具有相同根名稱的內容

```
<myPropertyFolder1>
    <myProperty4>value</myProperty4>
    <myGroup1>
        <myGroup2>
            <myProperty1>value</myProperty1>
            <myProperty2>value</myProperty2>
        </myGroup2>
        <myProperty3>value</myProperty3>
    </myGroup1>
</myPropertyFolder1>
```

圖 *10:* 多個內容名稱對映

名稱

名稱必須以 字母 或 底線開頭。 它不得包含 冒號，不能以 句點 結尾，且只能包含 字母、 數字、 底線、 連字號及 *Dots*。 有效字元定義在 W3C XML 規格中，且基本上由 Unicode 種類 `Ll, Lu, Lo, Lt, Nl, Mc, Mn, Lm,` 及 `Nd` 組成。

內容或名稱/值配對的完整路徑不得違反 第 471 頁的『無效的路徑名稱』中說明的規則。 路徑限制為 4095 個位元組，不得包含 Unicode 相容性字元，且不得以字串 `XML` 開頭。

群組名稱

群組名稱的語法與名稱相同。 群組名稱是選用的。 內容及名稱/值配對可以放置在資料夾的根目錄中。 如果有助於組織內容和名稱/值配對，請使用群組。

元素名稱

元素名稱的語法與名稱相同。

元素值

元素值包括 <*Element name*> 標籤與 </*Element name*>之間的所有空格。 請勿在值中使用兩個字元 < 和 &。 然後取代為 < 和 `&`。

內容屬性

內容屬性對映內容描述子欄位: 對映如下:

支援

sa
`MQPD_SUPPORT_OPTIONAL`

sr
`MQPD_SUPPORT_REQUIRED`

sx
`MQPD_SUPPORT_REQUIRED_IF_LOCAL`

環境定義

none
`MQPD_NO_CONTEXT`

user
`MQPD_USER _CONTEXT`

CopyOptions

forward
`MQPD_COPY_FORWARD`

reply
`MQPD_COPY_REPLY`

report
`MQPD_COPY_REPORT`

publish
`MQPD_COPY_PUBLISH`

all
`MQPD_COPY_ALL`

請勿與其他選項一起使用 `all`。

default
`MQPD_COPY_DEFAULT`

請勿與其他選項一起使用 default。 default 與 forward + report + publish

none
MQPD_COPY_NONE

請勿與其他選項一起使用 none。

支援 內容屬性僅適用於 mq 資料夾中的內容。

環境定義 和 CopyOptions 內容屬性適用於所有內容資料夾。

資料類型

MQRFH2 資料類型對映至訊息內容類型，如下所示:

表 *539:* 資料類型對映

MQRFH2 資料類型	**訊息內容類型**
bin.hex	MQBYTE[]
boolean	MQBOOL
i1	MQINT8
i2	MQINT16
i4	MQINT32
i8	MQINT64
r4	MQFLOAT32
r8	MQFLOAT64
string	MQCHAR[]

任何沒有資料類型的元素都會假設為 string 類型。

元素屬性 xsi:nil='true'指出空值。 請勿將屬性 xsi:nil='false' 用於非空值。 例如，下列內容具有空值:

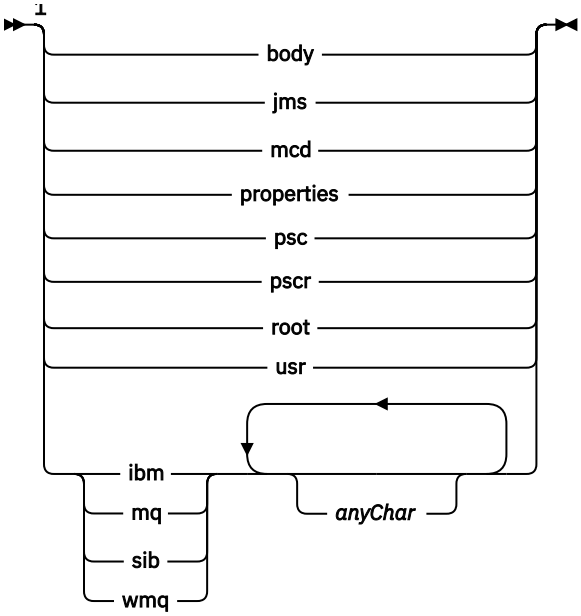
```
<NullProperty
xsi:nil='true'></NullProperty>
```

位元組或字串內容可以有空值。 空值由具有零長度元素值的 MQRFH2 元素表示。 例如，下列內容具有空值:

```
<EmptyProperty></EmptyProperty>
```

保留資料夾或內容資料夾名稱

限制資料夾或內容資料夾的名稱不能以下列任何字串開頭。 字首保留給 IBM 所建立的資料夾或內容名稱。

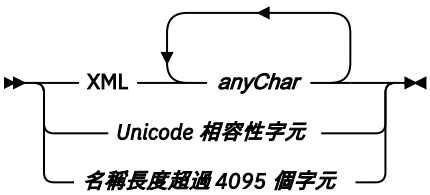


註:

[1] 保留的資料夾或內容名稱包含大小寫字母的任何混合。

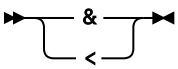
無效的路徑名稱

將名稱/值配對或內容的完整路徑限制為不包含下列任何字串。



無效字元

一律使用 ESC 序列 & 和 <，而不是文字 "&" 和 "<"。

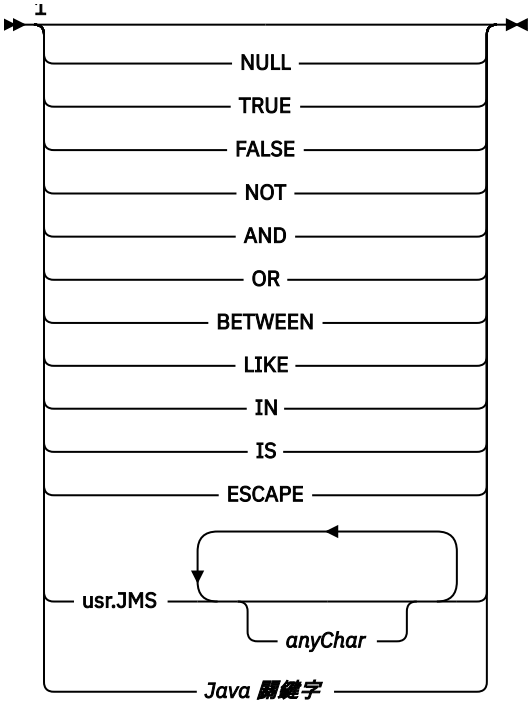


已定義的內容名稱

定義的內容名稱是 WebSphere MQ 或其他產品所定義且由 IBM WebSphere MQ 和使用者應用程式使用的內容名稱。 定義的內容只存在於定義的內容資料夾中。 已定義的內容名稱在內容資料夾的說明中說明; 請參閱內容資料夾。

內容名稱無效

請勿建構符合下列規則的內容名稱。 此規則適用於命名內容的完整內容路徑，而不只是適用於內容元素名稱。



註:

1 無效的內容名稱可以包含大寫和小寫的任意組合。

NameValue 長度 *(MQLONG)*
對應 NameValueData 欄位的長度

這會在 *NameValueData* 欄位中指定資料的長度 (以位元組為單位)。 *NameValueLength* 必須是 4 的倍數。

註: *NameValueLength* 和 *NameValueData* 欄位是選用的，但如果有的話，它們必須成對出現且相鄰。 欄位配對可以根據需要重複多次，例如:

```
length1 data1 length2 data2 length3 data3
```

因為這些欄位是選用的，所以會從針對所支援各種程式設計語言所提供的結構宣告中省略它們。

StrucId (MQCHAR4)

這是結構 ID; 值必須是:

MQRFH_STRUC_ID
規則及格式化標頭結構的 ID。

對於 C 程式設計語言，也會定義常數 MQRFH_STRUC_ID_ARRAY; 此值與 MQRFH_STRUC_ID 相同，但卻是字元陣列而非字串。

此欄位的起始值是 MQRFH_STRUC_ID。

StrucLength (MQLONG)

這是 MQRFH2 結構的長度 (以位元組為單位) ，包括結構尾端的 *NameValueLength* 和 *NameValueData* 欄位。 它適用於結構結尾的多個 *NameValueLength* 和 *NameValueData* 欄位配對，順序如下:

```
length1, data1, length2, data2, ...
```

StrucLength 不 包含 可能在結構結尾最後一個 *NameValueData* 欄位之後的任何使用者資料。

為了避免在某些環境中轉換使用者資料時發生問題， *StrucLength* 必須是四的倍數。

下列常數提供結構 固定 部分的長度，即不包括 *NameValueLength* 及 *NameValueData* 欄位的長度:

MQRFH_STRUC_LENGTH_FIXED_2
MQRFH2 結構固定部分的長度。

此欄位的起始值為 MQRFH_STRUC_LENGTH_FIXED_2。

版本 *(MQLONG)*

這是結構版本號碼; 值必須是:

MQRFH_VERSION_2
Version-2 規則和格式化標頭結構。

此欄位的起始值為 MQRFH_VERSION_2。

MQRFH2 的起始值和語言宣告

表 *540: MQRFH2* 中 *MQRFH2* 欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQRFH_STRUC_ID	'RFH¬'
Version	MQRFH_VERSION_2	2
StrucLength	MQRFH_STRUC_LENGTH_FIXED_2	36
Encoding	MQENC_NATIVE	取決於環境
CodedCharSetId	MQCCSI_INHERIT	-2
Format	MQFMT_NONE	空白
Flags	MQRFH_NONE	0
NameValueCCSID	無	1208

附註:

1. 符號 ¬ 代表單一空白字元。
2. 在 C 程式設計語言中，巨集變數 MQRFH2_DEFAULT 包含上述值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQRFH2 MyRFH2 = {MQRFH2_DEFAULT};
```

C 宣告

```
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 */
};
```

COBOL 宣告

```
**   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-STRUCLENGTH    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
   15 MQRFH2-FLAGS          PIC S9(9) BINARY.
**    Character set identifier of NAMEVALUEDATA
   15 MQRFH2-NAMEVALUECCSID PIC S9(9) BINARY.
```

PL/I 宣告

```
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 */
```

High Level Assembler 宣告

```
MQRFH                 DSECT
MQRFH_STRUCID         DS   CL4  Structure identifier
MQRFH_VERSION         DS   F    Structure version number
MQRFH_STRUCLENGTH     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)
```

Visual Basic 宣告

```
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
```

MQRMH-參照訊息標頭

下表彙總結構中的欄位。

表 *541: MQRMH* 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
StrucLength	MQRMH 的總長度，包括固定欄位結尾的字串，但不包括主體資料	StrucLength
Encoding	主體資料的數字編碼	編碼
CodedCharSetId	主體資料的字集 ID	CodedCharSetId
Format	主體資料的格式名稱	格式
Flags	參照訊息旗標	旗標
ObjectType	物件類型	ObjectType
ObjectInstanceId	物件實例 ID	ObjectInstanceID
SrcEnvLength	來源環境資料的長度	SrcEnv 長度
SrcEnvOffset	來源環境資料的偏移	SrcEnv 偏移
SrcNameLength	來源物件名稱的長度	SrcName 長度
SrcNameOffset	來源物件名稱的偏移	SrcName 偏移
DestEnvLength	目的地環境資料的長度	DestEnv 長度
DestEnvOffset	目的地環境資料的偏移	DestEnv 偏移
DestNameLength	目的地物件名稱的長度	DestName 長度
DestNameOffset	目的地物件名稱的偏移	DestName 偏移
DataLogicalLength	主體資料的長度	DataLogical 長度
DataLogicalOffset	主體資料的低偏移	DataLogical 偏移
DataLogicalOffset2	主體資料的高偏移	DataLogicalOffset2

MQRMH 概觀

可用性: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows，以及連接至這些系統的 WebSphere MQ 用戶端。

目的:MQRMH 結構定義參照訊息標頭的格式。 此標頭與使用者撰寫的訊息通道結束程式搭配使用，以將極大量資料 (稱為 大量資料) 從一個佇列管理程式傳送至另一個佇列管理程式。 與一般傳訊相比的差異是大量資料不是儲存在佇列上; 而是只有大量資料的 參照 儲存在佇列上。 這可減少少數極大訊息耗盡 MQ 資源的可能性。

格式名稱:MQFMT_REF_MSG_HEADER。

字集和編碼:MQRMH 中的字元資料，以及偏移欄位所定址的字串，必須在本端佇列管理程式的字集中; 這是由 *CodedCharSetId* queue-manager 屬性所提供。 MQRMH 中的數值資料必須採用原生機器編碼; 這是由 C 程式設計語言的 MQENC_NATIVE 值所提供。

在下列欄位中，將 MQRMH 的字集及編碼設為 *CodedCharSetId* 及 *Encoding* 欄位:

- MQMD (如果 MQRMH 結構是在訊息資料的開頭) ，或
- 在 MQRMH 結構之前的標頭結構 (所有其他觀察值)。

用法: 應用程式放置由 MQRMH 組成的訊息，但省略大量資料。 當訊息通道代理程式 (MCA) 從傳輸佇列讀取訊息時，會呼叫使用者提供的訊息結束程式來處理參照訊息標頭。 在 MCA 透過通道將訊息傳送至下一個佇列管理程式之前，結束程式可以將 MQRMH 結構所識別的大量資料附加至參照訊息。

在接收端，等待參照訊息的訊息結束程式必須存在。 收到參照訊息時，結束程式必須從訊息中 MQRMH 之後的大量資料建立物件，然後在沒有大量資料的情況下傳遞參照訊息。 稍後，應用程式可以從佇列中讀取參照訊息 (不含大量資料) 來擷取參照訊息。

一般而言， MQRMH 結構是訊息中的全部。 不過，如果訊息位於傳輸佇列上，則在 MQRMH 結構之前會有一或多個其他標頭。

參照訊息也可以傳送至配送清單。 在此情況下，當訊息位於傳輸佇列時， MQDH 結構及其相關記錄位於 MQRMH 結構之前。

註: 請勿將參照訊息當作分段訊息來傳送，因為訊息結束程式無法正確處理它。

資料轉換: 基於資料轉換目的，轉換 MQRMH 結構包括來源環境資料、來源物件名稱、目的地環境資料及目的地物件名稱的轉換。 在資料轉換之後，會捨棄結構開頭 *StrucLength* 位元組內的任何其他位元組，或具有未定義的值。 只要符合下列所有條件，即會轉換大量資料:

- 執行資料轉換時，大量資料會出現在訊息中。
- MQRMH 中 *Format* 欄位的值不是 MQFMT_NONE。
- 已存在具有指定格式名稱的使用者撰寫資料轉換結束程式。

不過，請注意，當訊息位於佇列時，通常大量資料 不會 出現在訊息中，因此大量資料會由 MQGMO_CONVERT 選項進行轉換。

MQRMH 的欄位

MQRMH 結構包含下列欄位; 這些欄位按 **字母順序**進行說明:

CodedCharSetId (MQLONG)

這指定大量資料的字集 ID; 它不適用於 MQRMH 結構本身中的字元資料。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。 可以使用下列特殊值:

MQCCSI_INHERIT
此結構 後面 的資料中的字元資料與此結構的字集相同。

佇列管理程式會將訊息中所傳送結構的這個值變更為結構的實際字集 ID。 如果沒有發生錯誤， MQGET 呼叫不會傳回值 MQCCSI_INHERIT。

如果 MQMD 中 *PutApplType* 欄位的值是 MQAT_BROKER ，請勿使用 MQCCSI_INHERIT。

此值在下列環境中受支援: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows，以及連接至這些系統的 WebSphere MQ 用戶端。

此欄位的起始值為 MQCCSI_UNDEFINED。

DataLogical 長度 *(MQLONG)*

DataLogicalLength 欄位指定 MQRMH 結構所參照的大量資料的長度。

如果大量資料實際存在於訊息中，則資料從 MQRMH 結構開始的 *StrucLength* 位元組偏移開始。 整個訊息的長度減去 *StrucLength* 會提供大量資料呈現的長度。

如果訊息中有資料， *DataLogicalLength* 會指定相關的資料量。 正常情況下， *DataLogicalLength* 的值會與訊息中呈現的資料長度相同。

如果 MQRMH 結構代表物件中的剩餘資料 (從指定的邏輯偏移開始) ，您可以使用 *DataLogicalLength* 的值零，前提是大量資料實際上不存在於訊息中。

如果不存在任何資料，則 MQRMH 結尾與訊息結尾一致。

此欄位的起始值為 0。

DataLogical 偏移 (MQLONG)

此欄位指定大量資料從物件開始的低偏移，大量資料構成物件的一部分。 從物件開頭開始的大量資料偏移稱為 邏輯偏移。 這 不是 從 MQRMH 結構開始算起的大量資料實體偏移; 該偏移由 *StrucLength* 提供。

為了容許使用參照訊息傳送大型物件，邏輯偏移會分成兩個欄位，而實際邏輯偏移是由這兩個欄位的總和所提供:

- *DataLogicalOffset* 代表當邏輯偏移除以 1 000 000 000 時所取得的餘數。 因此，它是 0 到 999 999 999 範圍內的值。
- *DataLogicalOffset2* 代表當邏輯偏移除以 1000 000 000 時所取得的結果。 因此，它是邏輯偏移中存在的 1000 000 000 的完整倍數。 倍數是在 0 到 999 999 999 的範圍內。

此欄位的起始值為 0。

DataLogicalOffset2 (MQLONG)

此欄位指定大量資料從物件開始的高偏移量，大量資料構成該物件的一部分。 它是 0 到 999 999 999 範圍內的值。 如需詳細資料，請參閱 *DataLogicalOffset*。

此欄位的起始值為 0。

DestEnv 長度 (MQLONG)

這是目的地環境資料的長度。 如果此欄位為零，則沒有目的地環境資料，並忽略 *DestEnvOffset* 。

DestEnv 偏移 (MQLONG)

此欄位指定目的地環境資料從 MQRMH 結構開始的偏移。 如果參照訊息的建立者知道目的地環境資料，則該資料可以由該建立者指定。 例如，在 Windows 上，目的地環境資料可能是要儲存大量資料之物件的目錄路徑。 不過，如果建立者不知道目的地環境資料，則由使用者提供的訊息結束程式負責判斷任何需要的環境資訊。

目的地環境資料的長度由 *DestEnvLength* 提供; 如果此長度為零，則沒有目的地環境資料，且會忽略 *DestEnvOffset* 。 如果存在的話，目的地環境資料必須從結構開始完全位於 *StrucLength* 個位元組內。

應用程式不得假設目的地環境資料與 *SrcEnvOffset*、 *SrcNameOffset* 及 *DestNameOffset* 欄位所定址的任何資料連續。

此欄位的起始值為 0。

DestName 長度 (MQLONG)

目的地物件名稱的長度。 如果此欄位為零，則沒有目的地物件名稱，且會忽略 *DestNameOffset* 。

DestName 偏移 (MQLONG)

此欄位指定目的地物件名稱從 MQRMH 結構開始的偏移。 如果參照訊息的建立者知道該資料，則該參照訊息的建立者可以指定目的地物件名稱。 不過，如果建立者不知道目的地物件名稱，則由使用者提供的訊息結束程式負責識別要建立或修改的物件。

目的地物件名稱的長度由 *DestNameLength* 提供; 如果此長度為零，則沒有目的地物件名稱，且會忽略 *DestNameOffset* 。 如果存在的話，目的地物件名稱必須從結構開頭開始完全位於 *StrucLength* 個位元組內。

應用程式不得假設目的地物件名稱與 *SrcEnvOffset*、 *SrcNameOffset* 及 *DestEnvOffset* 欄位所定址的任何資料連續。

此欄位的起始值為 0。

編碼 *(MQLONG)*

這指定大量資料的數值編碼; 它不適用於 MQRMH 結構本身中的數值資料。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。

此欄位的起始值為 MQENC_NATIVE。

旗標 *(MQLONG)*

這些是參照訊息旗標。 下列是已定義的旗標:

MQRMHF_LAST
此旗標指出參照訊息代表或包含參照物件的最後一部分。

MQRMHF_NOT_LAST
參照訊息不包含或代表物件的最後部分。 MQRMHF_NOT_LAST 輔助程式說明文件。 此選項並非預期與任何其他選項搭配使用，但由於其值為零，因此無法偵測到此類使用。

此欄位的起始值為 MQRMHF_NOT_LAST。

格式 *(MQCHAR8)*

這會指定大量資料的格式名稱。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。 此欄位的編碼規則與 MQMD 中 *Format* 欄位的編碼規則相同。

此欄位的起始值為 MQFMT_NONE。

ObjectInstanceID (MQBYTE24)

使用此欄位來識別物件的特定實例。 如果不需要，請將它設為下列值:

MQOII_NONE
未指定物件實例 ID。 欄位長度的值為二進位零。

對於 C 程式設計語言，也會定義常數 MQOII_NONE_ARRAY; 此值與 MQOII_NONE 相同，但卻是字元陣列而非字串。

此欄位的長度由 MQ_OBJECT_INSTANCE_ID_LENGTH 提供。 此欄位的起始值為 MQOII_NONE。

ObjectType (MQCHAR8)

這是訊息結束程式可用來辨識其支援之參照訊息類型的名稱。 名稱必須符合上述 *Format* 欄位的相同規則。

此欄位的起始值為 8 個空白。

SrcEnv 長度 *(MQLONG)*

來源環境資料的長度。 如果此欄位為零，則沒有來源環境資料，且會忽略 *SrcEnvOffset* 。

此欄位的起始值為 0。

SrcEnv 偏移 *(MQLONG)*

此欄位指定來源環境資料從 MQRMH 結構開始的偏移。 如果參照訊息的建立者知道來源環境資料，則該資料可以由建立者指定。 例如，在 Windows 上，來源環境資料可能是包含大量資料之物件的目錄路徑。 不過，如果建立者不知道來源環境資料，使用者提供的訊息結束程式必須判斷任何需要的環境資訊。

來源環境資料的長度由 *SrcEnvLength* 提供; 如果此長度為零，則沒有來源環境資料，且會忽略 *SrcEnvOffset* 。 如果存在的話，來源環境資料必須從結構開始完全位於 *StrucLength* 個位元組內。

應用程式不得假設環境資料在結構中最後一個固定欄位之後立即開始，或它與 *SrcNameOffset*、*DestEnvOffset* 及 *DestNameOffset* 欄位所處理的任何資料連續。

此欄位的起始值為 0。

SrcName 長度 *(MQLONG)*

來源物件名稱的長度。 如果此欄位為零，則沒有來源物件名稱，且會忽略 *SrcNameOffset* 。

此欄位的起始值為 0。

SrcName 偏移 *(MQLONG)*

此欄位指定來源物件名稱從 MQRMH 結構開始的偏移。 如果建立者知道該資料，則參照訊息的建立者可以指定來源物件名稱。 不過，如果建立者不知道來源物件名稱，則使用者提供的訊息結束程式必須識別要存取的物件。

來源物件名稱的長度由 *SrcNameLength* 提供; 如果此長度為零，則沒有來源物件名稱，且會忽略 *SrcNameOffset* 。 如果存在的話，來源物件名稱必須從結構開始完全位於 *StrucLength* 個位元組內。

應用程式不得假設來源物件名稱與 *SrcEnvOffset*、 *DestEnvOffset* 及 *DestNameOffset* 欄位所定址的任何資料連續。

此欄位的起始值為 0。

StrucId (MQCHAR4)

這是結構 ID; 值必須是:

MQRMH_STRUC_ID
參照訊息標頭結構的 ID。

對於 C 程式設計語言，也會定義常數 MQRMH_STRUC_ID_ARRAY; 此值與 MQRMH_STRUC_ID 相同，但卻是字元陣列而非字串。

此欄位的起始值是 MQRMH_STRUC_ID。

StrucLength (MQLONG)
MQRMH 的總長度，包括固定欄位結尾的字串，但不包括大量資料。

此欄位的起始值為零。

版本 *(MQLONG)*

結構版本號碼。 值必須為:

MQRMH_VERSION_1
Version-1 參照訊息標頭結構。

下列常數指定現行版本的版本號碼:

MQRMH_CURRENT_VERSION
參照訊息標頭結構的現行版本。

此欄位的起始值為 MQRMH_VERSION_1。

MQRMH 的起始值和語言宣告

表 *542: MQRMH* 中 *MQRMH* 欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQRMH_STRUC_ID	'RMH¬'
Version	MQRMH_VERSION_1	1
StrucLength	無	0
Encoding	MQENC_NATIVE	取決於環境
CodedCharSetId	未定義 MQCCSI_UNDEFINED	0
Format	MQFMT_NONE	空白

表 542: *MQRMH* 中 *MQRMH* 欄位的起始值 (繼續)

欄位名稱	常數名稱	常數值
Flags	MQRMHF_NOT_LAST	0
ObjectType	無	空白
ObjectInstanceId	MQOII_NONE	空值
SrcEnvLength	無	0
SrcEnvOffset	無	0
SrcNameLength	無	0
SrcNameOffset	無	0
DestEnvLength	無	0
DestEnvOffset	無	0
DestNameLength	無	0
DestNameOffset	無	0
DataLogicalLength	無	0
DataLogicalOffset	無	0
DataLogicalOffset2	無	0

附註:

1. 符號 ¬ 代表單一空白字元。
2. 在 C 程式設計語言中，巨集變數 MQRMH_DEFAULT 包含上面列出的值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQRMH MyRMH = {MQRMH_DEFAULT};
```

C 宣告

```
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 */
};
```

COBOL 宣告

```
**   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-DATALOGICALLENGTH  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.
```

PL/I 宣告

```
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 */
```

High Level Assembler 宣告

```
MQRMH                     DSECT
MQRMH_STRUCID             DS   CL4   Structure identifier
MQRMH_VERSION             DS   F     Structure version number
MQRMH_STRUCLENGTH         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_CODEDCHARSETID      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_DATALOGICALLENGTH   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)
```

Visual Basic 宣告

```
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 MQBYTE24 '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
```

MQRR-回應記錄

下表彙總結構中的欄位。

表 *543: MQRR* 中的欄位		
欄位	**說明**	**主題**
CompCode	佇列的完成碼	CompCode
Reason	佇列的原因碼	原因

MQRR 概觀

可用性: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows，以及連接至這些系統的 WebSphere MQ 用戶端。

目的: 當目的地是配送清單時，使用 MQRR 結構來接收單一目的地佇列的開啟或放置作業所產生的完成碼及原因碼。 MQRR 是 MQOPEN、MQPUT 及 MQPUT1 呼叫的輸出結構。

字集及編碼:MQRR 中的資料必須採用 *CodedCharSetId* 佇列管理程式屬性所提供的字集，以及 MQENC_NATIVE 所提供本端佇列管理程式的編碼。 不過，如果應用程式以 MQ MQI 用戶端身分執行，則該結構必須採用用戶端的字集及編碼。

用法: 透過在 MQOPEN 和 MQPUT 呼叫上或在 MQPUT1 呼叫上提供這些結構的陣列，您可以在呼叫結果混合時，即在清單中部分佇列呼叫成功但其他佇列呼叫失敗時，判定配送清單中所有佇列的完成碼和原因碼。 來自呼叫的原因碼 MQRC_MULTIPLE_REASONS 指出佇列管理程式已設定回應記錄 (如果由應用程式提供)。

MQRR 的欄位

MQRR 結構包含下列欄位; 這些欄位按 **字母順序**進行說明:

CompCode (MQLONG)

這是在 MQOPEN 或 MQPUT1 呼叫所提供的 MQOR 結構陣列中，對應元素所指定名稱之佇列的開啟或放置作業所產生的完成碼。

這一律是輸出欄位。 此欄位的起始值是 MQCC_OK。

原因 *(MQLONG)*

這是因佇列的開啟或放置作業所產生的原因碼，該佇列的名稱是由 MQOPEN 或 MQPUT1 呼叫上所提供 MQOR 結構陣列中的對應元素所指定。

這一律是輸出欄位。 此欄位的起始值為 MQRC_NONE。

MQRR 的起始值及語言宣告

表 *544: MQRR* 的 *MQRR* 中欄位的起始值		
欄位名稱	**常數名稱**	**常數值**
CompCode	MQCC_OK	0
Reason	MQRC_NONE	0

附註:

1. 在 C 程式設計語言中，巨集變數 MQRR_DEFAULT 包含上面列出的值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQRR MyRR = {MQRR_DEFAULT};
```

C 宣告

```
typedef struct tagMQRR MQRR;
struct tagMQRR {
  MQLONG  CompCode;  /* Completion code for queue */
```

```
  MQLONG  Reason;    /* Reason code for queue */
};
```

COBOL 宣告

```
**   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.
```

PL/I 宣告

```
dcl
 1 MQRR based,
  3 CompCode fixed bin(31), /* Completion code for queue */
  3 Reason   fixed bin(31); /* Reason code for queue */
```

Visual Basic 宣告

```
Type MQRR
  CompCode As Long 'Completion code for queue'
  Reason   As Long 'Reason code for queue'
End Type
```

MQSCO-SSL 配置選項

下表彙總結構中的欄位。

表 *545:* 欄位位於*(F) MQSCO.*

MQSCO 中的欄位清單 (依版本)，以及說明欄位之主題的鏈結。

欄位	說明	主題
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
KeyRepository	金鑰儲存庫的位置	KeyRepository
CryptoHardware	加密硬體的詳細資料	CryptoHardware
AuthInfoRecCount	存在的 MQAIR 記錄數	AuthInfoRecCount
AuthInfoRecOffset	從 MQSCO 開始的第一個 MQAIR 記錄的偏移	AuthInfoRecOffset
AuthInfoRecPtr	第一個 MQAIR 記錄的位址	AuthInfoRecPtr
註: 如果 *Version* 小於 MQSCO_VERSION_2，則會忽略下列兩個欄位。		
KeyResetCount	SSL 秘密金鑰重設計數	KeyReset 計數
FipsRequired	在 WebSphere MQ 中使用 FIPS 認證的加密演算法	第 487 頁的『FipsRequired (MQLONG)』
註: 如果 *Version* 小於 MQSCO_VERSION_3，則會忽略下列欄位。		
EncryptionPolicySuiteB	僅使用 Suite B 加密演算法	EncryptionPolicySuiteB
註: 如果 *Version* 小於 MQSCO_VERSION_4，則會忽略下列欄位。		

表 *545:* 欄位位於*(F) MQSCO.*

MQSCO 中的欄位清單 (依版本)，以及說明欄位之主題的鏈結。

*(*繼續*)*

欄位	說明	主題
CertificateValPolicy	憑證驗證原則	CertificateVal 原則

相關參考

第 282 頁的『MQCNO-連接選項』
下表彙總結構中的欄位。

第 485 頁的『MQSCO 概觀』
可用性: AIX、 HP-UX、 IBM i、Solaris Linux 及 Windows 用戶端。

第 485 頁的『MQSCO 的欄位』

第 488 頁的『MQSCO 的起始值和語言宣告』

MQSCO 概觀

可用性: AIX、 HP-UX、 IBM i、Solaris Linux 及 Windows 用戶端。

目的:MQSCO 結構 (與 MQCD 結構中的 SSL 欄位一起使用) 可讓作為 WebSphere MQ MQI 用戶端執行的應用程式指定配置選項，當通道通訊協定為 TCP/IP 時，控制用戶端連線使用 SSL 的配置選項。 此結構是 MQCONNX 呼叫的輸入參數。

如果用戶端通道的通道通訊協定不是 TCP/IP，則會忽略 MQSCO 結構。

字集及編碼:MQSCO mustis 中的資料位於 *CodedCharSetId* 佇列管理程式屬性所提供的字集，以及 MQENC_NATIVE 所提供本端佇列管理程式的編碼。

MQSCO 的欄位

MQSCO 結構包含下列欄位; 這些欄位按 **字母順序**說明:

AuthInfoRecCount (MQLONG)

這是 *AuthInfoRecPtr* 或 *AuthInfoRecOffset* 欄位所處理的鑑別資訊 (MQAIR) 記錄數。 如需相關資訊，請參閱第 242 頁的『MQAIR-鑑別資訊記錄』。 值必須為零或大於零。 如果值無效，則呼叫會失敗，原因碼為 MQRC_AUTH_INFO_REC_COUNT_ERROR。

這是輸入欄位。 此欄位的起始值為 0。

AuthInfoRecOffset (MQLONG)

這是從 MQSCO 結構開始算起第一個鑑別資訊記錄的偏移 (以位元組為單位)。 偏移可以是正數或負數。 如果 *AuthInfoRecCount* 為零，則會忽略該欄位。

您可以使用 *AuthInfoRecOffset* 或 *AuthInfoRecPtr* 來指定 MQAIR 記錄，但不能同時指定兩者; 如需詳細資料，請參閱 *AuthInfoRecPtr* 欄位的說明。

這是輸入欄位。 此欄位的起始值為 0。

AuthInfoRecPtr (PMQAIR)

這是第一個鑑別資訊記錄的位址。 如果 *AuthInfoRecCount* 為零，則會忽略該欄位。

您可以使用下列兩種方式之一來提供 MQAIR 記錄的陣列:

- 使用指標欄位 *AuthInfoRecPtr*

  在此情況下，應用程式可以宣告與 MQSCO 結構分開的 MQAIR 記錄陣列，並將 *AuthInfoRecPtr* 設為陣列的位址。

考量將 *AuthInfoRecPtr* 用於以可攜至不同環境 (例如， C 程式設計語言) 的方式支援指標資料類型的程式設計語言。

- 使用偏移欄位 *AuthInfoRecOffset*

  在此情況下，應用程式必須宣告包含 MQSCO 的複合結構，後面接著 MQAIR 記錄陣列，並將 *AuthInfoRecOffset* 設為從 MQSCO 結構開始算起陣列中第一筆記錄的偏移。 請確定此值是正確的，且具有可在 MQLONG 內容納的值 (最嚴格的程式設計語言是 COBOL ，其有效範圍是 -999 999 999 至 +999 999 999)。

  對於不支援指標資料類型的程式設計語言，或以不可攜至不同環境 (例如 COBOL 程式設計語言) 的方式來實作指標資料類型的程式設計語言，請考慮使用 *AuthInfoRecOffset* 。

不論您選擇何種技術，都只能使用 *AuthInfoRecPtr* 和 *AuthInfoRecOffset* 之一; 如果兩者都不是零，則呼叫會失敗，原因碼為 MQRC_AUTH_INFO_REC_ERROR。

這是輸入欄位。 在那些支援指標的程式設計語言中，此欄位的起始值是空值指標，否則是全空值位元組字串。

註: 在程式設計語言不支援指標資料類型的平台上，此欄位宣告為適當長度的位元組字串。

CertificateVal 原則 (MQLONG)

此欄位指定使用的憑證驗證原則類型。 欄位可以設為下列其中一個值:

全部 MQ_CERT_VAL_POLICY_ANY
套用 Secure Socket Library 所支援的每一個憑證驗證原則。 如果有任何原則將憑證鏈視為有效，請接受憑證鏈。

MQ_CERT_VAL_POLICY_RFC5280
僅套用 RFC5280 相容憑證驗證原則。 此設定提供比 ANY 設定更嚴格的驗證，但拒絕部分較舊的數位憑證。

此欄位的起始值為 MQ_CERT_VAL_POLICY_ANY

CryptoHardware (MQCHAR256)
此欄位提供連接至用戶端系統之加密硬體的配置詳細資料。

將欄位設為下列格式的字串，或將它保留空白或空值:

```
GSK_PKCS11=<the PKCS #11 driver path and file name>;<the PKCS #11
token label>;<the PKCS #11 token password>;<symmetric cipher setting>;
```

若要使用符合 PKCS #11 介面的加密硬體，例如 IBM 4960 或 IBM 4764 ，必須指定 PKCS #11 驅動程式路徑、PKCS #11 記號標籤及 PKCS #11 記號密碼字串，每一個都以分號終止。

PKCS #11 驅動程式路徑是提供 PKCS #11 卡支援之共用程式庫的絕對路徑。 PKCS #11 驅動程式檔名是共用程式庫的名稱。 PKCS #11 路徑和檔名所需的值範例如下:

```
/usr/lib/pkcs11/PKCS11_API.so
```

PKCS #11 記號標籤必須完全為小寫。 如果您已使用大小寫混合的記號標籤來配置硬體，請使用這個小寫標籤來重新配置它。

如果不需要加密硬體配置，請將欄位設為空白或空值。

如果該值短於欄位長度，請以空字元終止該值，或以空白填補該值至欄位長度。 如果值無效，或在用來配置加密硬體時導致失敗，則呼叫會失敗，原因碼為 MQRC_CRYPTO_HARDWARE_ERROR。

這是輸入欄位。 此欄位的長度由 MQ_SSL_CRYPTO_HARDWARE_LENGTH 提供。 此欄位的起始值是 C 中的空字串，以及其他程式設計語言中的空白字元。

EncryptionPolicySuiteB(MQLONG)

此欄位指定是否使用套組 B 相容加密法，以及使用的強度層次。 此值可以是下列之一或多個:

- MQ_SUITE_B_NONE

不使用套組 B 相容加密法。

- MQ_SUITE_B_128_BIT

  使用套組 B 128 位元強度安全。

- MQ_SUITE_B_192_BIT

  使用套組 B 192 位元強度安全。

註: 搭配使用 MQ_SUITE_B_NONE 與此欄位中的任何其他值無效。

FipsRequired (MQLONG)
WebSphere MQ 可以配置加密硬體，以便使用的加密法模組是硬體產品所提供的; 根據使用中的加密硬體產品，這些模組可以 FIPS 認證為特定層次。 使用此欄位來指定在 WebSphere MQ 提供的軟體中提供加密法時，只使用 FIPS 認證的演算法。

當安裝 WebSphere MQ 時，也會安裝 SSL 加密法的實作，以提供一些 FIPS 認證的模組。

值可以是:

MQSSL_FIPS_NO
這是預設值。 設為此值時:

- 可以使用特定平台上支援的任何 CipherSpec 。
- 如果在不使用加密硬體的情況下執行，則下列 CipherSpecs 會在 WebSphere MQ 平台上使用 FIPS 140-2 認證加密法執行:
  - TLS_RSA_WITH_3DES_EDE_CBC_SHA
  - TLS_RSA_WITH_AES_128_CBC_SHA
  - TLS_RSA_WITH_AES_256_CBC_SHA

MQSSL_FIPS_YES
當設為此值時，除非您使用加密硬體來執行加密法，否則您可以確定

- 在套用至此用戶端連線的 CipherSpec 中，只能使用 FIPS 認證的加密演算法。
- 只有在使用下列其中一個「密碼規格」時，入埠及出埠 SSL 通道連線才會成功:
  - TLS_RSA_WITH_3DES_EDE_CBC_SHA
  - TLS_RSA_WITH_AES_128_CBC_SHA
  - TLS_RSA_WITH_AES_256_CBC_SHA

附註:

1. CipherSpec TLS_RSA_WITH_3DES_EDE_CBC_SHA 已淘汰。
2. 可能的話，如果已配置僅 FIPS CipherSpecs ，則 MQI 用戶端會拒絕指定非 FIPS CipherSpec 與 MQRC_SSL_INITIALIZATION_ERROR 的連線。 WebSphere MQ 不保證拒絕所有這類連線，您必須負責判斷您的 WebSphere MQ 配置是否符合 FIPS 標準。

distributed *KeyRepository (MQCHAR256)*

此欄位僅適用於在 UNIX、 Linux 及 Windows 系統上執行的 WebSphere MQ MQI 用戶端。 它指定在其中儲存金鑰及憑證的金鑰資料庫檔的位置。 金鑰資料庫檔必須具有 `zzz.kdb` 格式的檔名，其中 `zzz` 可由使用者選取。 *KeyRepository* 欄位包含此檔案的路徑，以及檔名詞幹 (檔名直至但不包括最終 `.kdb` 的所有字元)。 `.kdb` 檔案字尾會自動新增。

每一個金鑰資料庫檔都有相關聯的 密碼隱藏檔。 這會保留用來容許對金鑰資料庫進行程式化存取的已編碼密碼。 密碼隱藏檔必須位於相同的目錄中，且與金鑰資料庫具有相同的檔案系統，且必須以字尾 `.sth` 結尾。

例如，如果 *KeyRepository* 欄位具有值 `/xxx/yyy/key` ，則金鑰資料庫檔必須是 `/xxx/yyy/key.kdb` ，密碼隱藏檔必須是 `/xxx/yyy/key.sth` ，其中 `xxx` 和 `yyy` 代表目錄名稱。

如果該值短於欄位長度，請以空字元終止該值，或以空白填補該值至欄位長度。 未勾選此值; 如果存取金鑰儲存庫時發生錯誤，則呼叫會失敗，原因碼為 MQRC_KEY_REPOSITORY_ERROR。

若要從 WebSphere MQ MQI 用戶端執行 SSL 連線，請將 *KeyRepository* 設為有效的金鑰資料庫檔名。

這是輸入欄位。 此欄位的長度由 MQ_SSL_KEY_REPOSITORY_LENGTH 指定。 此欄位的起始值是 C 中的空字串，以及其他程式設計語言中的空白字元。

KeyReset 計數 *(MQLONG)*
這代表在重新協議秘密金鑰之前，在 SSL 或 TLS 交談內傳送及接收的未加密位元組總數。

位元組數包括 MCA 所傳送的控制資訊。

如果您指定 1 位元組到 32 KB 範圍內的 SSL 或 TLS 秘密金鑰重設計數，則 SSL 或 TLS 通道將使用 32 KB 的秘密金鑰重設計數。 這是為了避免對小型 SSL 或 TLS 秘密金鑰重設值進行過多金鑰重設的處理成本。

這是輸入欄位。 此值是 0 到 999 999 999 範圍內的數字，預設值為 0。 使用值 0 表示永不重新協議秘密金鑰。

StrucId (MQCHAR4)

這是結構 ID; 值必須是:

MQSCO_STRUC_ID
SSL 配置選項結構的 ID。

對於 C 程式設計語言，也會定義常數 MQSCO_STRUC_ID_ARRAY; 此值與 MQSCO_STRUC_ID 相同，但卻是字元陣列而非字串。

這一律是輸入欄位。 此欄位的起始值是 MQSCO_STRUC_ID。

版本 *(MQLONG)*

這是結構版本號碼; 值必須是:

MQSCO_VERSION_1
Version-1 SSL 配置選項結構。

MQSCO_VERSION_2
Version-2 SSL 配置選項結構。

MQSCO_VERSION_3
Version-3 SSL 配置選項結構。

MQSCO_VERSION_4
Version-4 SSL 配置選項結構。

下列常數指定現行版本的版本號碼:

MQSO_CURRENT_VERSION
SSL 配置選項結構的現行版本。

這一律是輸入欄位。 此欄位的起始值是 MQSCO_VERSION_1。

MQSCO 的起始值和語言宣告

表 *546: MQSCO* 中欄位的起始值.

MQSCO 中欄位及其起始值的說明

欄位名稱	常數名稱	常數值
StrucId	MQSCO_STRUC_ID	'SCO¬'
Version	MQSCO_CURRENT_VERSION	1
KeyRepository	無	空字串或空白
CryptoHardware	無	空字串或空白
AuthInfoRecCount	無	0

表 546: *MQSCO* 中欄位的起始值.

MQSCO 中欄位及其起始值的說明

(繼續)

欄位名稱	常數名稱	常數值
AuthInfoRecOffset	無	0
AuthInfoRecPtr	無	空值指標或空值位元組
KeyResetCount	MQSCO_RESET_COUNT_DEFAULT	0
FipsRequired	MQSSL_FIPS_NO	0
EncryptionPolicySuiteB	MQ_SUITE_B_NONE, MQ_SUITE_B_NOT_AVAILABLE, MQ_SUITE_B_NOT_AVAILABLE, MQ_SUITE_B_NOT_AVAILABLE	1, 0, 0, 0
CertificateValPolicy	MQ_CERT_VAL_POLICY_DEFAULT	0

Notes:

1. The symbol ¬ represents a single blank character.
2. In the C programming language, the macro variable MQSCO_DEFAULT contains the values listed above. Use it in the following way to provide initial values for the fields in the structure:

```
MQSCO MySCO = {MQSCO_DEFAULT};
```

C 宣告

```
typedef struct tagMQSCO MQSCO;
struct tagMQSCO {
   MQCHAR4    StrucId;                   /* Structure identifier */
   MQLONG     Version;                   /* Structure version number */
   MQCHAR256  KeyRepository;             /* Location of SSL 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 */
/* Ver:2 */
                                         /* algorithms */
   MQLONG     EncryptionPolicySuiteB[4]; /* Use only Suite B */
/* Ver:3 */
                                         /* cryptographic algorithms */
   MQLONG     CertificateValPolicy;      /* Certificate validation */
                                         /* policy */
/* Ver:4 */
```

COBOL 宣告

```
**   MQSCO structure
  10 MQSCO.
**    Structure identifier
```

```
   15 MQSCO-STRUCID                  PIC X(4).
**    Structure version number
   15 MQSCO-VERSION                  PIC S9(9) BINARY.
**    Location of SSL 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
```

PL/I 宣告

```
dcl
 1 MQSCO based,
  3 StrucId                    char(4),       /* Structure identifier */
  3 Version                    fixed bin(31), /* Structure version number */
  3 KeyRepository              char(256),     /* Location of SSL 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 */
```

Visual Basic 宣告

```
Type MQSCO
  StrucId           As String*4   'Structure identifier'
  Version           As Long       'Structure version number'
  KeyRepository     As String*256 'Location of SSL key repository'
  CryptoHardware    As String*256 'Cryptographic hardware configuration'
                                  'string'
  AuthInfoRecCount  As Long       'Number of MQAIR records present'
  AuthInfoRecOffset As Long       'Offset of first MQAIR record from'
                                  'start of MQSCO structure'
  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
```

MQSD-訂閱描述子

下表彙總結構中的欄位。

欄位	說明	主題
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
Options	選項	選項
ObjectName	物件名稱	ObjectName
AlternateUserId	替代使用者 ID	AlternateUserid
AlternateSecurityId	替代安全 ID	AlternateSecurityId
SubExpiry	訂閱期限	SubExpiry
ObjectString	物件字串	ObjectString
SubName	訂閱名稱	SubName
SubUserData	訂閱使用者資料	SubUserData
SubCorrelId	訂閱相關性 ID	SubCorrelId
PubPriority	發佈優先順序	PubPriority
PubAccountingToken	發佈統計記號	PubAccounting 記號
PubAppIdentityData	發佈應用程式身分資料	PubAppIdentityData
SelectionString	提供選取準則的字串	SelectionString
SubLevel	訂閱層次	SubLevel
ResObjectString	長物件名稱	ResObject 字串

MQSD 概觀

可用性: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows、 z/OS，以及連接至這些系統的 WebSphere MQ MQI 用戶端。

目的:MQSD 結構用來指定所建立訂閱的詳細資料。

此結構是 MQSUB 呼叫上的輸入/輸出參數。 如需相關資訊，請參閱 MQSUB 使用注意事項。

受管理訂閱: 如果應用程式不需要使用特定佇列作為符合其訂閱之發佈的目的地，則可以使用受管理訂閱特性。 如果應用程式選擇使用受管理訂閱，則佇列管理程式會透過提供物件控點作為 MQSUB 呼叫的輸出，將已發佈訊息傳送至的目的地通知訂閱者。 如需相關資訊，請參閱 Hobj (MQHOBJ)-輸入/輸出。

移除訂閱時，在下列情況下，佇列管理程式也會承諾清除尚未從受管理目的地擷取的訊息:

- 當移除訂閱時 (搭配使用 MQCLOSE 與 MQCO_REMOVE_SUB) ，且受管理 Hobj 已關閉。
- 隱含表示使用不可延續訂閱 (MQSO_NON_DURABLE) 來中斷與應用程式的連線
- 因為訂閱已過期且受管理 Hobj 已關閉而移除訂閱時到期。

您必須將受管理訂閱與不可延續訂閱搭配使用，這樣才能進行清除，且已關閉不可延續訂閱的訊息不會佔用佇列管理程式中的空間。 可延續訂閱也可以使用受管理目的地。

版本:MQSD 的現行版本是 MQSD_VERSION_1。

字集及編碼:MQSD 中的資料必須採用 *CodedCharSetId* 佇列管理程式屬性所提供的字集，以及 MQENC_NATIVE 所提供本端佇列管理程式的編碼。 不過，如果應用程式以 MQ MQI 用戶端身分執行，則該結構必須採用用戶端的字集及編碼。

MQSD 的欄位

MQSD 結構包含下列欄位; 這些欄位按字母順序說明:

AlternateSecurityID (MQBYTE40)

這是隨 AlternateUserID 傳遞至授權服務以容許執行適當授權檢查的安全 ID。

只有在指定 MQSO_ALTERNATE_USER_AUTHORITY 且 AlternateUserID 欄位不是完全空白時，才會使用 AlternateSecurityID，直到第一個空值字元或欄位結尾。

使用 MQSO_RESUME 從 MQSUB 呼叫返回時，此欄位保持不變。

如需相關資訊，請參閱 MQOD 資料類型中 第 415 頁的『AlternateSecurityID (MQBYTE40)』 的說明。

AlternateUserID (MQCHAR12)

如果您指定 MQSO_ALTERNATE_USER_AUTHORITY，則此欄位包含替代使用者 ID，用來檢查訂閱及輸出至目的地佇列 (在 MQSUB 呼叫的 *Hobj* 參數中指定) 的授權，以取代目前執行應用程式的使用者 ID。

如果成功，則在此欄位中指定的使用者 ID 會記錄為擁有訂閱的使用者 ID，以取代目前執行應用程式的使用者 ID。

如果指定 MQSO_ALTERNATE_USER_AUTHORITY，且此欄位完全空白，直到第一個空值字元或欄位結尾，則只有在不需要使用者授權即可使用指定的選項或輸出的目的地佇列來訂閱此主題時，訂閱才能成功。

如果未指定 MQSO_ALTERNATE_USER_AUTHORITY，則會忽略此欄位。

指出的環境中存在下列差異:

- 在 z/OS 上，只會使用 AlternateUserID 的前 8 個字元來檢查訂閱的授權。 不過，現行使用者 ID 必須獲得授權，才能指定這個特定的替代使用者 ID; 這項檢查會使用替代使用者 ID 的所有 12 個字元。 使用者 ID 只能包含外部安全管理程式所容許的字元。

使用 MQSO_RESUME 從 MQSUB 呼叫返回時，此欄位保持不變。

這是輸入欄位。 此欄位的長度由 MQ_USER_ID_LENGTH 提供。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 12 個空白字元。

ObjectName (MQCHAR48)

這是本端佇列管理程式上定義的主題物件名稱。

名稱可以包含下列字元:

- 大寫英文字母 (A 到 Z)
- 小寫英文字母 (a 到 z)
- 數字 (0 到 9)
- 句點 (.)、正斜線 (/)、底線 (_)、百分比 (%)

名稱不能包含前導或內含空白，但可以包含尾端空白。 請使用空值字元來指出名稱中有效資料的結尾; 空值及其後面的任何字元都會被視為空白。 下列限制適用於指出的環境:

- 在使用 EBCDIC Katakana 的系統上，無法使用小寫字元。
- 在 z/OS 上:
  - 避免名稱以底線開頭或結尾; 作業和控制台無法處理它們。
  - 百分比字元對 RACF 具有特殊意義。 如果使用 RACF 作為外部安全管理程式，則名稱不得包含百分比。 如果有的話，當使用 RACF 通用設定檔時，這些名稱不會包含在任何安全檢查中。
- 在 IBM i 上，當在指令上指定時，包含小寫字元、正斜線或百分比的名稱必須以引號括住。 對於在結構中作為欄位或在呼叫中作為參數出現的名稱，不得指定這些引號。

ObjectName 用來形成完整主題名稱。

完整主題名稱可以從兩個不同的欄位來建置: *ObjectName* 和 *ObjectString*。 如需如何使用這兩個欄位的詳細資料，請參閱 第 503 頁的『使用主題字串』。

如果找不到 *ObjectName* 欄位所識別的物件，則即使 *ObjectString* 中指定了字串，呼叫也會失敗，原因碼為 MQRC_UNKNOWN_OBJECT_NAME。

使用 MQSO_RESUME 選項從 MQSUB 呼叫傳回時，此欄位保持不變。

此欄位的長度由 MQ_TOPIC_NAME_LENGTH 提供。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

如果使用 MQSO_ALTER 選項變更現有訂閱，則無法變更訂閱的主題物件名稱。 可以省略此欄位及 *ObjectString* 欄位。 如果有提供它們，它們必須解析成相同的完整主題名稱。 如果沒有，則呼叫會失敗，並產生 MQRC_TOPIC_NOT_ALTERABLE。

ObjectString (MQCHARV)

這是要使用的長物件名稱。

ObjectString 用來形成完整主題名稱。

完整主題名稱可以從兩個不同的欄位來建置: *ObjectName* 和 *ObjectString*。 如需如何使用這兩個欄位的詳細資料，請參閱 第 503 頁的『使用主題字串』。

ObjectString 的長度上限為 10240。

如果未正確指定 *ObjectString* ，根據如何使用 MQCHARV 結構的說明，或如果它超出長度上限，則呼叫會失敗，原因碼為 MQRC_OBJECT_STRING_ERROR。

這是輸入欄位。 此結構中欄位的起始值與 MQCHARV 結構中欄位的起始值相同。

如果 *ObjectString* 中有萬用字元，則可以使用 MQSD 的「選項」欄位中指定的「萬用字元」選項來控制這些萬用字元的解譯。

使用 MQSO_RESUME 選項從 MQSUB 呼叫傳回時，此欄位保持不變。 如果提供緩衝區，則會在 *ResObjectString* 欄位中傳回所使用的完整主題名稱。

如果使用 MQSO_ALTER 選項變更現有訂閱，則無法變更所訂閱主題物件的完整名稱。 可以省略此欄位及 *ObjectName* 欄位。 如果提供它們，則必須解析為相同的完整主題名稱，否則呼叫會失敗，並產生 MQRC_TOPIC_NOT_ALTERABLE。

選項 *(MQLONG)*
這會提供選項來控制 MQSUB 呼叫的動作。

您必須至少指定下列其中一個選項:

- MQSO_ALTER
- MQ 回復
- MQSO_CREATE

您為選項指定的值可以透過下列方式使用:

- 可以新增值。 請勿多次新增相同的常數。
- 如果程式設計語言支援按位元運算，則可以使用按位元 OR 運算來結合值。

本主題指出無效的組合; 任何其他組合都是有效的。

存取或建立選項: 存取及建立選項控制是否建立訂閱，或是否傳回或變更現有訂閱。 您必須至少指定其中一個選項。 表格會顯示存取及建立選項的有效組合。

選項組合	**附註**
MQSO_CREATE	如果訂閱不存在，則建立訂閱。 如果訂閱存在，則此組合會失敗。
MQ 回復	回復現有訂閱。 如果不存在訂閱，則此組合會失敗。

選項組合	附註
MQSO_CREATE + MQSO_RESUME	如果訂閱不存在，則會建立訂閱，如果存在，則會回復相符的訂閱。 當在執行多次的應用程式中使用此組合時，此組合非常有用。
MQSO_ALTER (請參閱附註)	回復現有訂閱，變更任何欄位以符合 MQSD 中指定的欄位。 如果不存在訂閱，則此組合會失敗。
MQSO_CREATE + MQSO_ALTER (請參閱附註)	如果不存在訂閱，則建立訂閱並回復相符的訂閱，如果存在，則變更任何欄位以符合 MQSD 中指定的欄位。 在應用程式中使用此組合時，如果要確保其訂閱在繼續之前處於特定狀態，則此組合非常有用。
附註: 指定 MQSO_ALTER 的選項也可以指定 MQSO_RESUME，但此組合對單獨指定 MQSO_ALTER 沒有其他影響。 MQSO_ALTER 暗示 MQSO_RESUME，因為呼叫 MQSUB 來變更訂閱暗示也會回復訂閱。 相反的情況並不正確，不過: 回復訂閱並不表示要變更它。	

MQSO_CREATE

為指定的主題建立新的訂閱。 如果存在使用相同 *SubName* 的訂閱，則呼叫會失敗，且 MQRC_SUB_ALREADY_EXISTS。 結合 MQSO_CREATE 選項與 MQSO_RESUME 可以避免此失敗。 並非一律需要 *SubName* 。 如需詳細資料，請參閱該欄位的說明。

結合 MQSO_CREATE 與 MQSO_RESUME 會針對指定的 *SubName* ，將控點傳回預先存在的訂閱; 如果沒有現有的訂閱，則會使用 MQSD 中提供的所有欄位來建立新的訂閱。

MQSO_CREATE 也可以與 MQSO_ALTER 結合，以產生類似的效果。

MQ 回復

將控點傳回給符合 *SubName* 所指定之預先存在的訂閱。 未對相符訂閱屬性進行任何變更， MQSD 結構中的輸出會傳回這些屬性。 只會使用下列 MQSD 欄位: StrucId、Version、Options、 AlternateUserId 和 AlternateSecurityId ，以及 SubName。

如果不存在符合完整訂閱名稱的訂閱，則呼叫會失敗，原因碼為 MQRC_NO_SUBSCRIPTION。 結合 MQSO_CREATE 選項與 MQSO_RESUME 可以避免此失敗。

訂閱的使用者 ID 是建立訂閱的使用者 ID ，或者如果稍後已由不同的使用者 ID 變更它，則它是最近成功變更的使用者 ID。 如果使用 AlternateUserID ，且容許該使用者使用替代使用者 ID ，則會將替代使用者 ID 記錄為建立訂閱的使用者 ID ，而不是用來建立訂閱的使用者 ID。

如果存在不使用 MQSO_ANY_USERID 選項建立的相符訂閱，且訂閱的使用者 ID 與要求訂閱控點之應用程式的使用者 ID 不同，則呼叫會失敗，原因碼為 MQRC_IDENTITY_MISMATCH。

如果相符訂閱存在且目前正在使用中，則呼叫會失敗，並出現 MQRC_SUBSCRIPTION_IN_USE。

如果 SubName 中指定的訂閱不是要從應用程式回復或變更的有效訂閱，則呼叫會失敗，並出現 MQRC_INVALID_SUBSCRIPTION。

MQSO_ALTER 暗示 MQSO_RESUME ，因此您不需要將它與該選項結合。 不過，結合這兩個選項並不會導致錯誤。

MQSO_ALTER

將控點傳回預先存在的訂閱，且其完整訂閱名稱符合 *SubName* 中的名稱所指定。 訂閱中任何不同於 MQSD 中所指定之屬性的屬性都會在訂閱中變更，除非該屬性不允許變更。 詳細資料記錄在每一個屬性的說明中，並彙總在下表中。 如果您嘗試變更無法變更的屬性，或變更已設定 MQSO_IMMUTABLE 選項的訂閱，則呼叫會失敗，原因碼如下表所示。

如果符合完整訂閱名稱的訂閱不存在，則呼叫會失敗，原因碼為 MQRC_NO_SUBSCRIPTION。 您可以結合 MQSO_CREATE 選項與 MQSO_ALTER 來避免此失敗。

將 MQSO_CREATE 與 MQSO_ALTER 結合，會針對指定的 *SubName* 將控點傳回預先存在的訂閱; 如果沒有現有的訂閱，則會使用 MQSD 中提供的所有欄位來建立新的訂閱。

訂閱的使用者 ID 是建立訂閱的使用者 ID，或者如果稍後被不同的使用者 ID 變更，則它是最近成功變更的使用者 ID。 如果使用 AlternateUserID，且容許該使用者使用替代使用者 ID，則會將替代使用者 ID 記錄為建立訂閱的使用者 ID，而不是用來建立訂閱的使用者 ID。

如果存在已建立但沒有 MQSO_ANY_USERID 選項的相符訂閱，且訂閱的使用者 ID 與要求訂閱控點之應用程式的使用者 ID 不同，則呼叫會失敗，原因碼為 MQRC_IDENTITY_MISMATCH。

如果相符訂閱存在且目前正在使用中，則呼叫會失敗，並出現 MQRC_SUBSCRIPTION_IN_USE。

如果 SubName 中指定的訂閱不是要從應用程式回復或變更的有效訂閱，則呼叫會失敗，並出現 MQRC_INVALID_SUBSCRIPTION。

下表顯示 MQSO_ALTER 變更 MQSD 及 MQSUB 中屬性值的能力。

表 *547: MQSD* 及 *MQSUB* 中可變更的屬性

資料類型描述子或函數呼叫	欄位名稱	可以使用 MQSO_ALTER 來變更此屬性	原因碼
MQSD	延續性選項	否	MQRC_DURABILITY_NOT_ALTERABLE
MQSD	目的地選項	是	無
MQSD	登錄選項	是 (請參閱附註 第 495 頁的『1』)	如果您嘗試變更 MQSO_GROUP_SUB，則 MQRC_GROUPING_NOT_ALTERABLE
MQSD	發佈選項	是 (請參閱附註 第 495 頁的『2』)	無
MQSD	萬用字元選項	否	MQRC_TOPIC_NOT_ALTERABLE
MQSD	其他選項	否 (請參閱附註 第 495 頁的『3』)	無
MQSD	ObjectName	否	MQRC_TOPIC_NOT_ALTERABLE
MQSD	AlternateUserid	否 (請參閱附註 第 495 頁的『4』)	無
MQSD	AlternateSecurityId	否 (請參閱附註 第 495 頁的『4』)	無
MQSD	SubExpiry	是	無
MQSD	ObjectString	否	MQRC_TOPIC_NOT_ALTERABLE
MQSD	SubName	否 (請參閱附註 第 495 頁的『5』)	無
MQSD	SubUserData	是	無
MQSD	SubCorrelId	是 (請參閱附註 第 495 頁的『6』)	在分組訂閱中時 MQRC_GROUPING_NOT_ALTERABLE
MQSD	PubPriority	是	無
MQSD	PubAccounting 記號	是	無
MQSD	PubApplIdentityData	是	無
MQSD	SubLevel	否	MQRC_SUBLEVEL_NOT_ALTERABLE
MQSUB	HOBJ	是 (請參閱附註 第 495 頁的『6』)	在分組訂閱中時 MQRC_GROUPING_NOT_ALTERABLE

附註:

1. 無法變更 MQSO_group_SUB。
2. 無法變更 MQSO_NEW_PUBLICATIONS_ONLY，因為它不是訂閱的一部分
3. 這些選項不是訂閱的一部分
4. 此屬性不是訂閱的一部分
5. 此屬性是所變更訂閱的身分
6. 可變更，除非是已分組子項的一部分 (MQSO_GROUP_SUB)

延續性選項: 下列選項控制訂閱的可延續程度。 您只能指定下列其中一個選項。 如果您使用 MQSO_ALTER 選項來變更現有訂閱，則無法變更訂閱的延續性。 使用 MQSO_RESUME 從 MQSUB 呼叫返回時，會設定適當的延續性選項。

MQ 可延續
要求保留此主題的訂閱，直到使用 MQCLOSE 搭配 MQCO_REMOVE_SUB 選項明確移除此主題為止。 如果未明確移除此訂閱，則即使在關閉此應用程式與佇列管理程式的連線之後，仍會保留此訂閱。

如果對定義為不容許可延續訂閱的主題要求可延續訂閱，則呼叫會失敗，且 MQRC_DURABILITY_NOT_ALLOWED。

MQSO_NON_DURABLE
要求在關閉佇列管理程式的應用程式連線時移除此主題的訂閱 (如果尚未明確移除的話)。 MQSO_NON_DURABLE 與 MQSO_durable 選項相反，且定義為輔助程式文件。 如果都未指定，則它是預設值。

目的地選項: 下列選項控制將已訂閱之主題的發佈資訊傳送至其中的目的地。 如果使用 MQSO_ALTER 選項變更現有訂閱，則可以變更用於訂閱發佈的目的地。 使用 MQSO_RESUME 從 MQSUB 呼叫傳回時，如果適當的話，會設定此選項。

受管理 MQSO_MANAGED

要求將發佈資訊傳送至的目的地由佇列管理程式管理。

在 *Hobj* 中傳回的物件控點代表佇列管理程式受管理佇列，並與後續的 MQGET、MQCB、MQINQ 或 MQCLOSE 呼叫搭配使用。

未指定 MQSO_MANAGED 時，無法在 *Hobj* 參數中提供從前一個 MQSUB 呼叫傳回的物件控點。

MQSO_NO_MULTICAST

要求將發佈傳送至的目的地不是多重播送群組位址。 只有在與 MQSO_MANAGED 選項結合時，此選項才有效。 在 *Hobj* 參數中提供佇列控點時，多重播送無法用於此訂閱，且選項無效。

如果使用 MCAST(ONLY) 設定將主題定義為僅容許多重播送訂閱，則呼叫會失敗，原因碼為 MQRC_MULTICAST_REQUIRED。

範圍選項: 下列選項控制所建立訂閱的範圍。 如果使用 MQSO_ALTER 選項變更現有訂閱，則無法變更此訂閱範圍選項。 使用 MQSO-RESUME 從 MQSUB 呼叫傳回時，會設定適當的 scope 選項。

MQSO_SCOPE_QMGR
此訂閱僅在本端佇列管理程式上進行。 不會將任何 Proxy 訂閱配送至網路中的其他佇列管理程式。 只有在此佇列管理程式中發佈的發佈會傳送至這個訂閱者。 這會置換使用 SUBSCOPE 主題屬性所設定的任何行為。

註: 如果未設定，訂閱範圍由 SUBSCOPE 主題屬性決定。

登錄選項: 下列選項控制對此訂閱的佇列管理程式所進行登錄的詳細資料。 如果使用 MQSO_ALTER 選項變更現有訂閱，則可以變更這些登錄選項。 使用 MQSO_RESUME 從 MQSUB 呼叫返回時，會設定適當的登錄選項。

MQ 群組 _ 訂閱

此訂閱將與使用相同佇列並指定相同相關性 ID 之相同 SubLevel 的其他訂閱一起分組，以便主題的任何發佈會導致將多個發佈訊息提供給訂閱群組，因為所使用的主題字串集重疊，只會導致將一個訊息遞送至佇列。 如果未使用此選項，則每一個符合的唯一訂閱 (由 SubName 識別) 都會隨附發佈副本，這可能表示多個發佈副本可能會放置在多個訂閱所共用的佇列上。

只有群組中最重要的訂閱才會隨附發佈的副本。 最重要的訂閱是根據完整主題名稱，直到找到萬用字元為止。 如果在群組內混合使用萬用字元架構，則只有萬用字元的位置很重要。 建議您不要在共用相同佇列的訂閱群組內結合不同的萬用字元架構。

建立新的分組訂閱時，它仍必須具有唯一的 SubName，但如果它符合群組中現有訂閱的完整主題名稱，則呼叫會失敗，並出現 MQRC_DUPLICATE_GROUP_SUB。

如果群組中最重要的訂閱也指定 MQSO_NOT_OWN_PUBS ，且這是來自相同應用程式的發佈，則不會將任何發佈遞送至佇列。

變更使用此選項所建立的訂閱時，無法變更暗示 MQSUB 呼叫 (代表佇列及佇列管理程式名稱) 的分組、Hobj 及 SubCorrelID 的欄位。 嘗試變更它們會導致呼叫失敗，並產生 MQRC_GROUPING_NOT_ALTERABLE。

此選項必須與 MQSO_SET_CORREL_ID 結合，且具有未設為 MQCI_NONE 且無法與 MQSO_MANAGED 結合的 SubCorrelID。

MQSO_ANY_USERID

指定 MQSO_ANY_USERID 時，訂閱者的身分不受限於單一使用者 ID。 這可讓任何使用者在具有適當權限時變更或回復訂閱。 一次只能有單一使用者具有訂閱。 嘗試回復使用另一個應用程式目前正在使用的訂閱，會導致呼叫失敗，並產生 MQRC_SUBSCRIPTION_IN_USE。

若要將此選項新增至現有訂閱， MQSUB 呼叫 (使用 MQSO_ALTER) 必須來自與原始訂閱本身相同的使用者 ID。

如果 MQSUB 呼叫參照已設定 MQSO_ANY_USERID 的現有訂閱，且使用者 ID 與原始訂閱不同，則只有在新使用者 ID 有權訂閱主題時，呼叫才會成功。 順利完成時，此訂閱者的未來發佈會放入訂閱者佇列中，並在發佈訊息中設定新的使用者 ID。

請勿同時指定 MQSO_ANY_USERID 和 MQSO_FIXED_USERID。 如果都未指定，則預設值為 MQSO_FIXED_USERID。

MQSO_FIXED_USERID

當指定 MQSO_FIXED_USERID 時，只能由最後一個使用者 ID 變更或回復訂閱，以變更訂閱。 如果訂閱未變更，則它是建立訂閱的使用者 ID。

如果 MQSUB 動詞參照已設定 MQSO_ANY_USERID 的現有訂閱，並將使用 MQSO_ALTER 的訂閱變更為使用選項 MQSO_FIXED_USERID ，則訂閱的使用者 ID 現在會固定在這個新的使用者 ID。 只有在新使用者 ID 具有訂閱主題的權限時，呼叫才會成功。

如果記錄為擁有訂閱的使用者 ID 以外的使用者 ID 嘗試回復或變更 MQSO_FIXED_USERID 訂閱，則呼叫會因 MQRC_IDENTITY_MISMATCH 而失敗。 可以使用 DISPLAY SBSTATUS 指令來檢視訂閱的擁有使用者 ID。

請勿同時指定 MQSO_ANY_USERID 和 MQSO_FIXED_USERID。 如果都未指定，則預設值為 MQSO_FIXED_USERID。

發佈選項: 下列選項控制將發佈傳送給此訂閱者的方式。 如果使用 MQSO_ALTER 選項變更現有訂閱，則可以變更這些發佈選項。

MQSO_NOT_OWN_PUBS
告訴分配管理系統，應用程式不想要看到它自己的任何發佈。 如果連線控點相同，則會將發佈視為源自相同的應用程式。 使用 MQSO_RESUME 從 MQSUB 呼叫傳回時，如果適當的話，會設定此選項。

僅 MQSO_NEW_PUBLICATIONS_ONLY
建立此訂閱時，不會傳送目前保留的發佈，只會傳送新的發佈。 只有在指定 MQSO_CREATE 時，此選項才適用。 訂閱的任何後續變更都不會變更發佈的流程，因此任何保留在主題上的發佈都已傳送至訂閱者作為新的發佈。

如果指定此選項時沒有指定 MQSO_CREATE ，則呼叫會失敗，並產生 MQRC_OPTIONS_ERROR。 使用 MQSO_RESUME 從 MQSUB 呼叫傳回時，即使使用此選項建立訂閱，也不會設定此選項。

如果未使用此選項，則會將先前保留的訊息傳送至提供的目的地佇列。 如果此動作由於錯誤 (MQRC_RETAINED_MSG_Q_ERROR 或 MQRC_RETAINED_NOT_DELIVERED) 而失敗，則建立訂閱會失敗。

MQSO_PUBLICATIONS_ON_REQUEST
設定此選項指出訂閱者將在必要時特別要求資訊。 佇列管理程式不會將自發的訊息傳送至訂閱者。 每次使用前一個 MQSUB 呼叫中的 Hsub 控點進行 MQSUBRQ 呼叫時，都會將保留的發佈資訊 (如果在主題中指定萬用字元，則可能有多個發佈資訊) 傳送給訂閱者。 由於使用此選項進行 MQSUB 呼叫，因此不會傳送任何發佈。 使用 MQSO_RESUME 從 MQSUB 呼叫傳回時，如果適當的話，會設定此選項。

此選項與大於 1 的 SubLevel 組合無效。

先讀選項: 下列選項控制是否在要求非持續訊息的應用程式之前，將非持續訊息傳送至應用程式。

MQSO_READ_AHEAD_AS_Q_DEF
如果 MQSUB 呼叫使用受管理控點，則與所訂閱主題相關聯之模型佇列的預設先讀屬性會決定在應用程式要求訊息之前，是否將訊息傳送至應用程式。

這是預設值。

MQSO_NO_READ_AHEAD
如果 MQSUB 呼叫使用受管理控點，則在應用程式要求訊息之前，不會將訊息傳送至應用程式。

MQSO_READ_AHEAD
如果 MQSUB 呼叫使用受管理控點，則在應用程式要求訊息之前，可能會先將訊息傳送至應用程式。

註:

下列附註適用於先讀選項:

1. 只能指定其中一個選項。 如果同時指定 MQOO_READ_AHEAD 和 MQOO_NO_READ_AHEAD ，則會傳回原因碼 MQRC_OPTIONS_ERROR。 只有在指定 MQSO_MANAGED 時，這些選項才適用。
2. 當傳遞先前已開啟的佇列時，它們不適用於 MQSUB。 當要求時，可能未啟用先讀。 在第一個 MQGET 呼叫上使用的 MQGET 選項可能會阻止啟用先讀。 此外，當用戶端連接至不支援先讀的佇列管理程式時，也會停用先讀。 如果應用程式不是作為 WebSphere MQ 用戶端執行，則會忽略這些選項。

萬用字元選項: 下列選項控制如何在 MQSD 的 ObjectString 欄位中提供的字串中解譯萬用字元。 您只能指定下列其中一個選項。 如果使用 MQSO_ALTER 選項變更現有訂閱，則無法變更這些萬用字元選項。 使用 MQSO_RESUME 從 MQSUB 呼叫返回時，會設定適當的萬用字元選項。

MQSO_WILDCARD_CHAR
萬用字元只會處理主題字串內的字元。

下表顯示 MQSO_WILDCARD_CHAR 所定義的行為。

特殊字元	行為
正斜線 (/)	不重要，只是另一個角色
星號 (*)	萬用字元，零個以上字元
問號 (?)	萬用字元， 1 個字元
百分比符號 (%)	跳出字元，容許在字串中使用字元 (*)、(?) 或 (%) ， 而不解譯為特殊字元，例如 (% *)、(%?) 或 (%%)。

例如，發佈下列主題:

```
/level0/level1/level2/level3/level4
```

使用下列主題來比對訂閱者:

```
*
/*
/ level0/level1/level2/level3/*
/ level0/level1/*/level3/level4
/ level0/level1/le?el2/level3/level4
```

註: 當使用 MQRFH1 格式化訊息以進行發佈/訂閱時，使用萬用字元會確切提供 WebSphere MQ V6 和 WebSphere MB V6 中所提供的意義。 建議不要用於新撰寫的應用程式，且僅用於先前針對該版本執行且尚未變更為使用預設萬用字元行為的應用程式，如 MQSO_WILDCARD_TOPIC 中所述。

MQSO_WILDCARD_TOPIC

萬用字元只會對主題字串內的主題元素進行操作。 如果未選擇任何項目，則這是預設行為。

下表顯示 MQSO_WILDCARD_TOPIC 所需的行為:

特殊字元	行為
(/)	主題層次分隔字元

特殊字元	行為
# 記號	萬用字元: 多個主題層次
加號 (+)	萬用字元: 單一主題層次
附註: 如果 (+) 和 (#) 與主題層次內的其他字元 (包括本身) 混合在一起，則不會將它們視為萬用字元。 在下列字串中，(#) 和 (+) 字元被視為一般字元。 `level0/level1/#+/level3/level#`	

例如，發佈下列主題:

```
/level0/level1/level2/level3/level4
```

使用下列主題來比對訂閱者:

```
#
/#
/ level0/level1/level2/level3/#
/ level0/level1/+/level3/level4
```

註: 當使用 MQRFH2 格式化訊息來發佈/訂閱時，使用萬用字元會提供 WebSphere Message Broker 第 6 版所提供的意義。

其他選項: 下列選項控制發出 API 呼叫而非訂閱的方式。 使用 MQSO_RESUME 從 MQSUB 呼叫傳回時，這些選項不會變更。 如需詳細資料，請參閱 第 492 頁的『AlternateUserID (MQCHAR12)』 。

MQSO_ALTERNATE_USER_AUTHORITY

AlternateUserID 欄位包含用來驗證此 MQSUB 呼叫的使用者 ID。 只有在此 AlternateUserID 獲授權開啟具有指定存取選項的物件時，不論執行應用程式的使用者 ID 是否獲授權開啟該物件，呼叫才會成功。

MQSO_SET_CORREL_ID

訂閱將使用 *SubCorrelId* 欄位中提供的相關性 ID。 如果未指定此選項，佇列管理程式會在訂閱時自動建立相關性 ID ，並在 *SubCorrelId* 欄位中傳回給應用程式。 如需相關資訊，請參閱 第 501 頁的『SubCorrelID (MQBYTE24)』 。

此選項無法與 MQSO_MANAGED 結合。

MQSO_SET_IDENTITY_CONTEXT

訂閱將使用 *PubAccountingToken* 和 *PubApplIdentityData* 欄位中提供的帳戶記號和應用程式身分資料。

如果指定此選項，則會執行相同的授權檢查，如同使用 MQOPEN 呼叫搭配 MQOO_SET_IDENTITY_CONTEXT 來存取目的地佇列一樣，但也使用 MQSO_MANAGED 選項時除外，在此情況下，目的地佇列上沒有授權檢查。

如果未指定此選項，則傳送至此訂閱者的發佈具有與其相關聯的預設環境定義資訊，如下所示:

MQMD 中的欄位	使用的值
UserIdentifier	建立訂閱時與訂閱相關聯的使用者 ID。
AccountingToken	可能的話，從環境判定; 可能的話，設為 MQACT_NONE。
ApplIdentityData	設為空白

此選項僅適用於 MQSO_CREATE 及 MQSO_ALTER。 如果與 MQSO_RESUME 一起使用，則會忽略 *PubAccountingToken* 及 *PubApplIdentityData* 欄位，因此此選項沒有作用。

如果未使用先前訂閱提供身分環境定義資訊的這個選項來變更訂閱，則會針對已變更的訂閱產生預設環境定義資訊。

如果訂閱容許不同的使用者 ID 與選項 MQSO_ANY_USERID 搭配使用，則會由不同的使用者 ID 回復，並為現在擁有訂閱的新使用者 ID 產生預設身分環境定義，且會遞送包含新身分環境定義的任何後續發佈。

MQSO_FAIL_IF_QUIESCING

如果佇列管理程式處於靜止狀態，則 MQSUB 呼叫會失敗。 在 z/OS 上，對於 CICS 或 IMS 應用程式，如果連線處於靜止狀態，此選項也會強制 MQSUB 呼叫失敗。

PubAccounting 記號 (MQBYTE32)

這是符合此訂閱之所有發佈訊息的「訊息描述子 (MQMD)」 *AccountingToken* 欄位中的值。 *AccountingToken* 是訊息身分環境定義的一部分。 如需訊息環境定義的相關資訊，請參閱 訊息環境定義。 如需 MQMD 中 *AccountingToken* 欄位的相關資訊，請參閱 第 366 頁的『AccountingToken (MQBYTE32)』

您可以對 *PubAccountingToken* 欄位使用下列特殊值:

MQACT_NONE

未指定帳戶記號。

欄位長度的值為二進位零。

對於 C 程式設計語言，也會定義常數 MQACT_NONE_ARRAY; 此值與 MQACT_NONE 相同，但它是字元陣列而非字串。

如果未指定選項 MQSO_SET_IDENTITY_CONTEXT，則佇列管理程式會產生結算記號作為預設環境定義資訊，且此欄位是一個輸出欄位，其中包含將在針對此訂閱發佈的每一則訊息中設定的 *AccountingToken*。

如果指定選項 MQSO_SET_IDENTITY_CONTEXT，則使用者會產生帳戶記號，且此欄位是輸入欄位，其中包含要在此訂閱的每一個發佈中設定的 *AccountingToken*。

此欄位的長度由 MQ_ACCOUNTING_TOKEN_LENGTH 提供。 此欄位的起始值是 MQACT_NONE。

如果使用 MQSO_ALTER 選項變更現有訂閱，則可以變更任何未來發佈訊息中的 *AccountingToken* 值。

使用 MQSO_RESUME 從 MQSUB 呼叫傳回時，此欄位會設為用於訂閱的現行 *AccountingToken*。

PubApplIdentityData (MQCHAR32)

這是所有符合此訂閱之發佈訊息的「訊息描述子 (MQMD)」 *ApplIdentityData* 欄位中的值。 *ApplIdentityData* 是訊息身分環境定義的一部分。 如需訊息環境定義的相關資訊，請參閱 訊息環境定義。 如需 MQMD 中 *ApplIdentityData* 欄位的相關資訊，請參閱 第 368 頁的『ApplIdentity 資料 (MQCHAR32)』

如果未指定選項 MQSO_SET_IDENTITY_CONTEXT，則在針對此訂閱發佈的每一則訊息中設定的 *ApplIdentityData* 為空白，作為預設環境定義資訊。

如果指定選項 MQSO_SET_IDENTITY_CONTEXT，則使用者會產生 *PubApplIdentityData*，且此欄位是輸入欄位，其中包含要在此訂閱的每一個發佈中設定的 *ApplIdentityData*。

此欄位的長度由 MQ_APPL_IDENTITY_DATA_LENGTH 提供。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 32 個空白字元。

如果使用 MQSO_ALTER 選項變更現有訂閱，則可以變更任何未來發佈訊息的 *ApplIdentityData*。

使用 MQSO_RESUME 從 MQSUB 呼叫傳回時，此欄位會設為用於訂閱的現行 *ApplIdentityData*。

PubPriority (MQLONG)

這是符合此訂閱之所有發佈訊息的「訊息描述子 (MQMD)」 *Priority* 欄位中的值。 如需 MQMD 中 *Priority* 欄位的相關資訊，請參閱 第 387 頁的『優先順序 (MQLONG)』。

值必須大於或等於零; 零是最低優先順序。 也可以使用下列特殊值:

MQPRI_PRIORITY_AS_Q_DEF

如果在 MQSUB 呼叫的 *Hobj* 欄位中提供訂閱佇列，且不是受管理控點，則會從這個佇列的 *DefPriority* 屬性取得訊息的優先順序。 如果佇列是叢集佇列，或佇列名稱解析路徑中有多個定義，當發佈訊息放入佇列時，會依照 第 387 頁的『優先順序 (MQLONG)』的說明來決定優先順序。

如果 MQSUB 呼叫使用受管理控點，則會從與所訂閱主題相關聯之模型佇列的 *DefPriority* 屬性取得訊息的優先順序。

已發佈 MQPRI_PRIORITY_AS_PUBLISHED

訊息的優先順序是原始發佈的優先順序。 這是欄位的起始值。

如果使用 MQSO_ALTER 選項變更現有訂閱，則可以變更任何未來發佈訊息的 *Priority* 。

使用 MQSO_RESUME 從 MQSUB 呼叫返回時，此欄位會設為用於訂閱的現行優先順序。

ResObject 字串 (MQCHARV)
這是在佇列管理程式解析 *ObjectName* 中提供的名稱之後的長物件名稱。

如果在 *ObjectString* 中提供長物件名稱，但在 *ObjectName* 中未提供任何內容，則在此欄位中傳回的值與在 *ObjectString* 中提供的值相同。

如果省略此欄位 (即 ResObjectString.VSBufSize 為零) ，則不會傳回 *ResObjectString* ，但會在 ResObjectString.VSLength 中傳回長度。 如果長度短於完整 ResObject 字串，則會截斷它，並傳回符合所提供長度的最右側字元數。

如果指定 *ResObjectString* 不正確，則根據如何使用 MQCHARV 結構的說明，或如果它超出長度上限，則呼叫會失敗，原因碼為 MQRC_RES_OBJECT_STRING_ERROR。

SelectionString (MQCHARV)
這是用來提供從主題訂閱訊息時所用的選取準則的字串。

如果提供緩衝區，且在 VSBufSize 中有正的緩衝區長度，則會在 MQSUB 呼叫使用 MQSO_RESUME 選項輸出時傳回此可變長度欄位。 如果在呼叫中未提供任何緩衝區，則在 MQCHARV 的 VSLength 欄位中只會傳回選取字串的長度。 如果提供的緩衝區小於傳回欄位所需的空間，則在提供的緩衝區中只會傳回 VSBufSize 個位元組。

如果未正確指定 *SelectionString* ，根據如何使用 第 262 頁的『MQCHARV-可變長度字串』 結構的說明，或如果它超出長度上限，則呼叫會失敗，原因碼為 MQRC_SELECTION_STRING_ERROR。

選取器 中說明 SelectionString 用法。

StrucId (MQCHAR4)

這是結構 ID; 值必須是:

MQSD_STRUC_ID
訂閱描述子結構的 ID。

對於 C 程式設計語言，也會定義常數 MQSD_STRUC_ID_ARRAY; 此值與 MQSD_STRUC_ID 相同，但卻是字元陣列而非字串。

這一律是輸入欄位。 此欄位的起始值是 MQSD_STRUC_ID。

SubCorrelID (MQBYTE24)
此欄位包含符合此訂閱的所有發佈共用的相關性 ID。

小心: 相關性 ID 只能在發佈/訂閱叢集中的佇列管理程式之間傳遞，不能在階層之間傳遞。

傳送以符合此訂閱的所有發佈在訊息描述子中包含此相關性 ID。 如果多個訂閱從相同佇列取得其發佈，則依相關性 ID 使用 MQGET 只容許取得特定訂閱的發佈。 此相關性 ID 可以由佇列管理程式或使用者產生。

如果未指定選項 MQSO_SET_CORREL_ID，則佇列管理程式會產生相關性 ID，且此欄位是一個輸出欄位，其中包含將在針對此訂閱發佈的每一則訊息中設定的相關性 ID。 產生的相關性 ID 包含 4 個位元組的產品 ID (ASCII 或 EBCDIC 中的 AMQX 或 CSQM)，後面接著唯一字串的產品特定實作。

如果指定選項 MQSO_SET_CORREL_ID，則使用者會產生相關性 ID，且此欄位是輸入欄位，其中包含要在此訂閱的每一個發佈中設定的相關性 ID。 在此情況下，如果欄位包含 MQCI_NONE，則在針對此訂閱發佈的每一則訊息中設定的相關性 ID，是由訊息原始放置所建立的相關性 ID。

如果指定選項 MQSO_GROUP_SUB，且指定的相關性 ID 與使用相同佇列及重疊主題字串的現有分組訂閱相同，則只有群組中最重要的訂閱才會隨附發佈的副本。

此欄位的長度由 MQ_CORREL_ID_LENGTH 提供。 此欄位的起始值為 MQCI_NONE。

如果您使用 MQSO_ALTER 選項來變更現有訂閱，且此欄位是輸入欄位，則可以變更訂閱相關性 ID，除非訂閱是分組訂閱，亦即，已使用選項 MQSO_GROUP_SUB 建立訂閱，在此情況下，無法變更訂閱相關性 ID。

使用 MQSO_RESUME 從 MQSUB 呼叫返回時，此欄位會設為訂閱的現行相關性 ID。

SubExpiry (MQLONG)

這是訂閱到期之前的時間 (以十分之一秒為單位)。 過了此間隔之後，沒有其他發佈將符合此訂閱。 一旦訂閱到期，就不再將發佈傳送至佇列。 不過，已存在的出版品不會受到任何影響。 *SubExpiry* 不會影響發佈期限。

可辨識下列特殊值:

MQEI_UNLIMITED
訂閱具有無限制的有效期限。

如果使用 MQSO_ALTER 選項變更現有訂閱，則可以變更訂閱的期限。

使用 MQSO_RESUME 選項從 MQSUB 呼叫傳回時，此欄位會設為訂閱的原始到期，而不是剩餘到期時間。

SubLevel (MQLONG)
這是與訂閱相關聯的層次。 只有在此訂閱位於最高 SubLevel 值小於或等於發佈時所使用 PubLevel 的訂閱集中時，才會將發佈遞送至此訂閱。 不過，如果已保留發佈，則無法再供較高層次的訂閱者使用，因為它在 PubLevel 1 上重新發佈。

值必須在 0 到 9 的範圍內。 零是最低層次。

此欄位的起始值為 1。

如需相關資訊，請參閱 截取出版品 。

如果使用 MQSO_ALTER 選項變更現有訂閱，則無法變更 SubLevel 。

不容許使用選項 MQSO_PUBLICATIONS_ON_REQUEST 結合 SubLevel 與大於 1 的值。

使用 MQSO_RESUME 從 MQSUB 呼叫返回時，此欄位會設為用於訂閱的現行層次。

SubUser 資料 *(MQCHARV)*

這會指定訂閱使用者資料。 在此欄位中的訂閱上提供的資料將併入作為傳送至此訂閱的每個發佈資訊的 MQSubUser 資料訊息內容。

SubUserData 的長度上限為 10240。

如果未正確指定 *SubUserData*，則根據如何使用 MQCHARV 結構的說明，或者如果它超出長度上限，則呼叫會失敗，原因碼為 MQRC_SUB_USER_DATA_ERROR。

這是輸入欄位。 此結構中欄位的起始值與 MQCHARV 結構中欄位的起始值相同。

如果使用 MQSO_ALTER 選項變更現有訂閱，則可以變更訂閱使用者資料。

如果提供緩衝區且 *VSBufLen* 中有正的緩衝區長度，則會在 MQSUB 呼叫使用 MQSO_RESUME 選項輸出時傳回此可變長度欄位。 如果在呼叫中未提供任何緩衝區，則 MQCHARV 的 *VSLength* 欄位中只會傳回訂閱

使用者日期的長度。 如果提供的緩衝區小於傳回欄位所需的空間，則只會在提供的緩衝區中傳回 *VSBufLen* 個位元組。

SubName (MQCHARV)

這會指定訂閱名稱。 只有在 *Options* 指定選項 MQSO_DURABLE 時，才需要此欄位，但如果提供的話，佇列管理程式也會使用 MQSO_NON_DURABLE。

如果指定，則 *SubName* 在佇列管理程式中必須是唯一的，因為它是用來識別訂閱的方法。

SubName 的長度上限為 10240。

此欄位有兩個用途。 對於 MQSO_DURABLE 訂閱，您可以使用此欄位來識別訂閱，以便如果您已關閉訂閱的控點 (使用 MQCO_KEEP_SUB 選項) 或已中斷與佇列管理程式的連線，則可以在建立訂閱之後回復該訂閱。 這是使用 MQSUB 呼叫搭配 MQSO_RESUME 選項來完成。 它也會顯示在 DISPLAY SBSTATUS 中 SUBNAME 欄位的訂閱管理視圖中。

如果未正確指定 *SubName* ，則會根據如何使用 MQCHARV 結構的說明，在需要時予以遺漏 (即 *SubName*)。 *VSLength* 是零) ，或者如果超出長度上限，則呼叫會失敗，原因碼為 MQRC_SUB_NAME_ERROR。

這是輸入欄位。 此結構中欄位的起始值與 MQCHARV 結構中欄位的起始值相同。

如果使用 MQSO_ALTER 選項變更現有訂閱，則無法變更訂閱名稱，因為它是用來尋找所參照訂閱的識別欄位。 在 MQSUB 呼叫的輸出上使用 MQSO_RESUME 選項不會變更它。

版本 *(MQLONG)*

這是結構版本號碼; 值必須是:

MQSD_VERSION_1
Version-1 訂閱描述子結構。

下列常數指定現行版本的版本號碼:

MqsD_CURRENT_VERSION
訂閱描述子結構的現行版本。

這一律是輸入欄位。 此欄位的起始值為 MQSD_VERSION_1。

使用主題字串

主題是從主題物件中所識別的子主題，以及應用程式所提供的子主題來建構。 您可以使用子主題作為主題名稱，或結合它們以形成新的主題名稱。

在 MQI 程式中， MQOPEN 會建立完整主題名稱。 它由兩個在發佈/訂閱 MQI 呼叫中使用的欄位組成，依列出的順序:

1. 主題物件的 **TOPICSTR** 屬性，在 **ObjectName** 欄位中指定。
2. **ObjectString** 參數定義應用程式所提供的子主題。

產生的主題字串會在 **ResObjectString** 參數中傳回。

如果每一個欄位的第一個字元不是空白或空字元，且欄位長度大於零，則這些欄位會被視為存在。 如果只存在其中一個欄位，則會使用未變更的欄位作為主題名稱。 如果任一欄位都沒有值，則呼叫會失敗，原因碼為 MQRC_UNKNOWN_OBJECT_NAME ，如果完整主題名稱無效，則為 MQRC_TOPIC_STRING_ERROR。

如果這兩個欄位都存在，則會在產生的合併主題名稱的兩個元素之間插入 '/' 字元。

第 503 頁的表 548 顯示主題字串連結的範例:

表 *548:* 主題字串連結範例

TOPICSTR	**ObjectString**	**完整主題名稱**	**備註**
足球/評分	' '	足球/評分	TOPICSTR 單獨使用
' '	足球/評分	足球/評分	ObjectString 單獨使用

表 548: 主題字串連結範例 (繼續)			
TOPICSTR	**ObjectString**	**完整主題名稱**	**備註**
football	評分	足球/評分	在連結點新增 '/' 字元
football	/Scores	足球//Scores	在兩個字串之間產生「空節點」
/Football	評分	/Football/Scores	主題以「空節點」開頭

'/' 字元被視為特殊字元，提供 主題樹狀結構 中完整主題名稱的結構，且不得用於任何其他原因，因為主題樹狀結構會受到影響。 主題 "/Football" 與主題 "Football" 不同。

下列萬用字元是特殊字元:

- 加號 '+'
- # 記號 '#'
- 星號 '*'
- 問號 '?'

這些字元不會被視為無效，但您必須確保瞭解其使用方式。 您可能偏好在發佈時不要在主題字串中使用這些字元。 在主題層次內使用 '#' 或 '+' 與其他字元 (包括本身) 混合的主題字串發佈，可以使用任一萬用字元架構來訂閱。 在主題字串上發佈並以 '#' 或 '+' 作為兩個 '/' 字元之間的唯一字元，會產生無法由應用程式使用萬用字元架構 MQSO_WILDCARD_TOPIC 明確訂閱的主題字串。 此狀況會導致應用程式取得超出預期的發佈數。

程式碼 Snippet 範例

此程式碼 Snippet 擷取自範例程式 Example 2: Publisher to a variable topic，結合主題物件與變數主題字串。

```
MQOD     td = {MQOD_DEFAULT};  /* Object Descriptor            */
td.ObjectType = MQOT_TOPIC;    /* Object is a topic            */
td.Version = MQOD_VERSION_4;   /* Descriptor needs to be V4    */
strncpy(td.ObjectName, topicName,  MQ_TOPIC_NAME_LENGTH);
td.ObjectString.VSPtr = topicString;
td.ObjectString.VSLength = (MQLONG)strlen(topicString);
td.ResObjectString.VSPtr = resTopicStr;
td.ResObjectString.VSBufSize = sizeof(resTopicStr)-1;
MQOPEN(Hconn, &td, MQOO_OUTPUT | MQOO_FAIL_IF_QUIESCING, &Hobj, &CompCode, &Reason);
```

MQSD 的起始值和語言宣告

欄位名稱	**常數名稱**	**常數值**
StrucId	MQSD_STRUC_ID	'SD¬¬'
Version	MQSD_VERSION_1	1
Options	MQSO_NON_DURABLE	0
ObjectName	無	空字串或空白
AlternateUserId	無	空字串或空白
AlternateSecurityId	MQSID_NONE	空值
SubExpiry	MQEI_UNLIMITED	-1
ObjectString	無	MQCHARV 定義的名稱和值
SubName	無	MQCHARV 定義的名稱和值
SubUserData	無	MQCHARV 定義的名稱和值

欄位名稱	常數名稱	常數值
SubCorrelId	MQCI_NONE	空值
PubPriority	MQPRI_PRIORITY_AS_Q_DEF	-3
PubAccountingToken	MQACT_NONE	空值
PubApplIdentityData	無	空字串或空白
Selection String	無	MQCHARV 定義的名稱和值
SubLevel	無	1
ResObjectString	無	MQCHARV 定義的名稱和值

附註:

1. 符號 ¬ 代表單一空白字元。
2. 空值字串或空白值表示 C 中的空值字串，以及其他程式設計語言中的空白字元。
3. 在 C 程式設計語言中，巨集變數 MQSD_DEFAULT 包含上述值。 您可以用下列方式來提供結構中欄位的起始值:

```
MQSD MySD = {MQSD_DEFAULT};
```

C 宣告

```
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 */
};
```

COBOL 宣告

```
** 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.
```

PL/I 宣告

```
dcl
 1 MQSD based,
  3 StrucId             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 */
```

High Level Assembler 宣告

```
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)
```

MQSMPO-設定訊息內容選項

下表彙總結構中的欄位。

表 *549: MQSMPO* 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
Options	選項	選項
ValueEncoding	內容值編碼	ValueEncoding
ValueCCSID	內容值字集	ValueCCSID

MQSMPO 概觀

可用性: 所有 WebSphere MQ 系統及 WebSphere MQ 用戶端。

目的: **MQSMPO** 結構可讓應用程式指定控制如何設定訊息內容的選項。 結構是 **MQSETMP** 呼叫上的輸入參數。

字集及編碼: **MQSMPO** 中的資料必須在應用程式的字集及應用程式的編碼中 (**MQENC_NATIVE**)。

MQSMPO 的欄位

MQSMPO 結構包含下列欄位; 這些欄位按 **字母順序**進行說明:

選項 *(MQLONG)*

位置選項: 相較於內容游標，下列選項與內容的相對位置相關:

MQSMPO_SET_FIRST
設定符合指定名稱的第一個內容的值，或者如果它不存在，則在具有相符階層的所有其他內容之後新增內容。

MQSMPO_SET_PROP_UNDER_CURSOR
設定內容游標所指向的內容值。 內容 cursor 所指向的內容是前次使用 MQIMPO_INQ_FIRST 或 MQIMPO_INQ_NEXT 選項所查詢的內容。

在 MQGET 呼叫上重複使用訊息控點時，或在 MQPUT 呼叫上 MQGMO 或 MQPMO 結構的 *MsgHandle* 欄位中指定訊息控點時，會重設內容游標。

如果在尚未建立內容游標或已刪除內容游標所指向的內容指標時使用此選項，則呼叫會失敗，完成碼為 MQCC_FAILED ，原因碼為 MQRC_PROPERTY_NOT_AVAILABLE。

MQSMPO_SET_PROP_BEFORE_CURSOR
在內容游標指向的內容之前設定新內容。 內容 cursor 所指向的內容是前次使用 MQIMPO_INQ_FIRST 或 MQIMPO_INQ_NEXT 選項所查詢的內容。

在 MQGET 呼叫上重複使用訊息控點時，或在 MQPUT 呼叫上 MQGMO 或 MQPMO 結構的 *MsgHandle* 欄位中指定訊息控點時，會重設內容游標。

如果在尚未建立內容游標或已刪除內容游標所指向的內容指標時使用此選項，則呼叫會失敗，完成碼為 MQCC_FAILED ，原因碼為 MQRC_PROPERTY_NOT_AVAILABLE。

MQSMPO_SET_PROP_AFTER_CURSOR
在內容游標指向的內容之後設定新內容。 內容 cursor 所指向的內容是前次使用 MQIMPO_INQ_FIRST 或 MQIMPO_INQ_NEXT 選項所查詢的內容。

在 MQGET 呼叫上重複使用訊息控點時，或在 MQPUT 呼叫上 MQGMO 或 MQPMO 結構的 *MsgHandle* 欄位中指定訊息控點時，會重設內容游標。

如果在尚未建立內容游標或已刪除內容游標所指向的內容指標時使用此選項，則呼叫會失敗，完成碼為 MQCC_FAILED ，原因碼為 MQRC_PROPERTY_NOT_AVAILABLE。

MQSMPO_APPEND_PROPERTY
導致在具有相符階層的所有其他內容之後新增新內容。 如果至少存在一個符合指定名稱的內容，則會在該內容清單結尾之後新增內容。

此選項容許建立具有相同名稱的內容清單。

如果您不需要任何說明的選項，請使用下列選項:

MQSMPO_NONE
未指定選項。

這一律是輸入欄位。 此欄位的起始值是 MQSMPO_SET_FIRST。

StrucId (MQCHAR4)

這是結構 ID; 值必須是:

MQSMPO_STRUC_ID

設定訊息內容選項結構的 ID。

對於 C 程式設計語言，也會定義常數 **MQSMPO_STRUC_ID_ARRAY** ; 其值與 **MQSMPO_STRUC_ID** 相同，但卻是字元陣列而非字串。

這一律是輸入欄位。 此欄位的起始值為 **MQSMPO_STRUC_ID**。

ValueCCSID (MQLONG)

如果值是字串，要設定之內容值的字集。

這一律是輸入欄位。 此欄位的起始值為 **MQCCSI_APPL**。

ValueEncoding (MQLONG)

要設定的內容值的編碼 (如果值是數值的話)。

這一律是輸入欄位。 此欄位的起始值為 **MQENC_NATIVE**。

版本 *(MQLONG)*

這是結構版本號碼; 值必須是:

MQSMPO_VERSION_1
Version-1 設定訊息內容選項結構。

下列常數指定現行版本的版本號碼:

MQSMPO_CURRENT_VERSION
設定訊息內容選項結構的現行版本。

這一律是輸入欄位。 此欄位的起始值為 **MQSMPO_VERSION_1**。

MQSMPO 的起始值和語言宣告

表 *550: MQSMPO* 中欄位的起始值

欄位名稱	**常數名稱**	**常數值**
StrucId	MQSMPO_STRUC_ID	'SMPO'
Version	MQSMPO_VERSION_1	1
Options	MQSMPO_NONE	0
ValueEncoding	MQENC_NATIVE	取決於環境
ValueCCSID	MQCCSI_APPL	-3

附註:

1. 空值字串或空白值表示 C 中的空值字串，以及其他程式設計語言中的空白字元。
2. 在 C 程式設計語言中，巨集變數 MQSMPO_DEFAULT 包含上面列出的值。 您可以用下列方式來提供結構中欄位的起始值:

```
MQSMPO MySMPO = {MQSMPO_DEFAULT};
```

C 宣告

```
typedef struct tagMQSMPO MQSMPO;
struct tagMQSMPO {
  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 */
};
```

COBOL 宣告

```
**   MQSMPO structure
  10 MQSMPO.
**    Structure identifier
   15 MQSMPO-STRUCID       PIC X(4).
**    Structure version number
   15 MQSMPO-VERSION       PIC S9(9) BINARY.
**    Options that control the action of MQSETMP
   15 MQSMPO-OPTIONS       PIC S9(9) BINARY.
**    Encoding of VALUE
   15 MQSMPO-VALUEENCODING PIC S9(9) BINARY.
**    Character set identifier of VALUE
   15 MQSMPO-VALUECCSID    PIC S9(9) BINARY.
```

PL/I 宣告

```
dcl
 1 MQSMPO 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 */
```

High Level Assembler 宣告

```
MQSMPO                DSECT
MQSMPO_STRUCID        DS   CL4   Structure identifier
MQSMPO_VERSION        DS   F     Structure version number
MQSMPO_OPTIONS        DS   F     Options that control the action of
```

```
*                                MQSETMP
MQSMPO_VALUEENCODING  DS   F     Encoding of VALUE
MQSMPO_VALUECCSID     DS   F     Character set identifier of VALUE
MQSMPO_LENGTH         EQU  *-MQSMPO
MQSMPO_AREA           DS   CL(MQSMPO_LENGTH)
```

MQSRO-訂閱要求選項

本節說明訂閱要求選項、它包含哪些欄位，以及那些欄位的起始值。

欄位	說明	主題
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
Options	選項	選項
NumPubs	發佈數目	NumPubs

MQSRO 概觀

可用性: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows、 z/OS 以及連接至這些系統的 WebSphere MQ MQI 用戶端。

目的:MQSRO 結構可讓應用程式指定選項來控制如何提出訂閱要求。 此結構是 MQSUBRQ 呼叫上的輸入/輸出參數。

版本:MQSRO 的現行版本是 MQSRO_VERSION_1。

字集及編碼:MQSRO 中的資料必須採用 *CodedCharSetId* 佇列管理程式屬性所提供的字集，以及 MQENC_NATIVE 所提供本端佇列管理程式的編碼。 不過，如果應用程式以 MQ MQI 用戶端身分執行，則該結構必須採用用戶端的字集及編碼。

MQSRO 的欄位

MQSRO 結構包含下列欄位; 這些欄位按字母順序說明:

NumPubs (MQLONG)

這是一個輸出欄位，傳回給應用程式以指出由於此呼叫而傳送至訂閱佇列的發佈數。 雖然此呼叫已傳送此數目的發佈資訊，但不保證應用程式將可取得此數目的訊息，尤其是當它們是非持續訊息時。

如果訂閱的主題包含萬用字元，則可能有多個發佈。 當建立 *Hsub* 所代表的訂閱時，如果主題字串中沒有萬用字元，則此呼叫最多只會傳送一個發佈。

選項 *(MQLONG)*

必須指定下列其中一個選項。 只能指定一個選項。

MQSRO_FAIL_IF_QUIESCING

如果佇列管理程式處於靜止狀態，則 MQSUBRQ 呼叫會失敗。 在 z/OS 上，對於 CICS 或 IMS 應用程式，如果連線處於靜止狀態，此選項也會強制 MQSUBRQ 呼叫失敗。

預設選項: 如果不需要上述選項，則必須使用下列選項:

MQSRO_NONE

使用這個值來指出未指定其他選項；所有選項都採用其預設值。

MQSRO_NONE 可協助程式說明文件。 雖然此選項不會與任何其他選項一起使用，因為其值為零，但無法偵測此使用。

StrucId (MQCHAR4)

這是結構 ID; 值必須是:

MQSRO_STRUC_ID
「訂閱要求選項」結構的 ID。

對於 C 程式設計語言，也會定義常數 MQSRO_STRUC_ID_ARRAY; 此值與 MQSRO_STRUC_ID 相同，但卻是字元陣列而非字串。

這一律是輸入欄位。 此欄位的起始值是 MQSRO_STRUC_ID。

版本 *(MQLONG)*

這是結構版本號碼; 值必須是:

MQSRO_VERSION_1
Version-1 訂閱要求選項結構。

下列常數指定現行版本的版本號碼:

MqsRO_CURRENT_VERSION
「訂閱要求選項」結構的現行版本。

這一律是輸入欄位。 此欄位的起始值為 MQSRO_VERSION_1。

MQSRO 的起始值和語言宣告

欄位名稱	常數名稱	常數值
StrucId	MQSRO_STRUC_ID	'SRO¬'
Version	MQSRO_VERSION_1	1
Options	MQSRO_NONE	0
NumPubs	無	0

附註:

1. 符號 ¬ 代表單一空白字元。
2. 在 C 程式設計語言中，巨集變數 MQSRO_DEFAULT 包含上述值。 您可以用下列方式來提供結構中欄位的起始值:

```
MQSRO MySRO = {MQSRO_DEFAULT};
```

C 宣告

```
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 */
};
```

COBOL 宣告

```
** 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.
```

PL/I 宣告

```
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 */
```

High Level Assembler 宣告

```
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
                        ORG   MQSRO
MQSRO_AREA              DS    CL(MQSRO_LENGTH)
```

MQSTS-狀態報告結構

下表彙總結構中的欄位。

表 *551: MQSTS* 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
CompCode	第一個錯誤的完成碼	CompCode
Reason	第一個錯誤的原因碼	原因
PutSuccessCount	順利完成的非同步放置呼叫數	SuccessCount
PutWarningCount	發生警告的非同步放置呼叫數	WarningCount
PutFailureCount	失敗的非同步放置呼叫數	FailureCount
ObjectType	失敗物件的類型	ObjectType
ObjectName	失敗物件的名稱	ObjectName
ObjectQMgrName	擁有失敗物件的佇列管理程式名稱	ObjectQMgrName
ResolvedObjectName	解析的目的地佇列名稱	ResolvedObject 名稱
ResolvedQMgrName	解析的目的地佇列管理程式名稱	ResolvedQMgr 名稱
註: 如果版本小於 MQSTS_VERSION_2，則會忽略其餘欄位。		
ObjectString	失敗物件的長物件名稱	ObjectString
SubName	失敗訂閱的訂閱名稱	SubName
OpenOptions	與失敗相關聯的開啟選項	OpenOptions
SubOptions	與失敗相關聯的訂閱選項	SubOptions

MQSTS 概觀

目的:MQSTS 結構是 MQSTAT 指令的輸出參數。

字集及編碼:MQSTS 中的字元資料位於本端佇列管理程式的字集; 這是由 *CodedCharSetId* 佇列管理程式屬性所提供。 MQSTS 中的數值資料採用原生機器編碼; 這是由 編碼提供。

用法:MQSTAT 指令用來擷取狀態資訊。 此資訊以 MQSTS 結構傳回。 如需 MQSTAT 的相關資訊，請參閱 第 688 頁的『MQSTAT-擷取狀態資訊』。

MQSTS 的欄位

MQSTS 結構包含下列欄位; 這些欄位按 **字母順序**進行說明:

CompCode (MQLONG)
所報告作業的完成碼。

CompCode 的解譯取決於 MQSTAT Type 參數的值。

MQSTAT_TYPE_ASYNC_ERROR

這是在 ObjectName 中指定的物件上先前非同步放置作業所產生的完成碼。

MQSTAT_TYPE_RECONNECTION

如果連線正在重新連接或無法重新連接，則這是導致連線開始重新連接的完成碼。

如果目前已連接連線，則值為 MQCC_OK。

MQSTAT_TYPE_RECONNECTION_ERROR

如果連線無法重新連接，則這是導致重新連線失敗的完成碼。

如果連線目前已連接或重新連接，則值為 MQCC_OK。

CompCode 一律是輸出欄位。 其起始值為 MQCC_OK。

ObjectName (MQCHAR48)
所報告物件的名稱。

ObjectName 的解譯取決於 MQSTAT Type 參數的值。

MQSTAT_TYPE_ASYNC_ERROR

這是在 put 作業中使用的佇列或主題名稱，其失敗會在 MQSTS 結構的 *CompCode* 及 *Reason* 欄位中報告。

MQSTAT_TYPE_RECONNECTION

如果連線正在重新連接，則這是與連線相關聯的佇列管理程式名稱。

MQSTAT_TYPE_RECONNECTION_ERROR

如果連線無法重新連接，則這是導致重新連線失敗的物件名稱。 MQSTS 結構中的 *CompCode* 和 *Reason* 欄位會報告失敗的原因。

ObjectName 是輸出欄位。 它的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

ObjectQMgr 名稱 *(MQCHAR48)*
所報告的佇列管理程式名稱。

ObjectQMgrName 的解譯取決於 MQSTAT Type 參數的值。

MQSTAT_TYPE_ASYNC_ERROR

這是在其中定義 *ObjectName* 物件的佇列管理程式名稱。 直到第一個空值字元或欄位結尾都是完全空白的名稱，表示應用程式所連接的佇列管理程式 (本端佇列管理程式)。

MQSTAT_TYPE_RECONNECTION

空白。

MQSTAT_TYPE_RECONNECTION_ERROR

如果連線無法重新連接，則這是導致重新連線失敗的物件名稱。 MQSTS 結構中的 *CompCode* 和 *Reason* 欄位會報告失敗的原因。

ObjectQMgrName 是輸出欄位。 它的值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

ObjectString (MQCHARV)
報告失敗物件的長物件名稱。 僅在 MQSTS 第 2 版或更高版本中存在。

ObjectString 的解譯取決於 MQSTAT Type 參數的值。

MQSTAT_TYPE_ASYNC_ERROR

這是 MQPUT 作業中使用的佇列或主題的長物件名稱，但失敗。

MQSTAT_TYPE_RECONNECTION

零長度字串

MQSTAT_TYPE_RECONNECTION_ERROR

這是導致重新連線失敗之物件的長物件名稱。

ObjectString 是輸出欄位。 其起始值是長度為零的字串。

ObjectType (MQLONG)
正在報告的 *ObjectName* 中所指名之物件的類型。

ObjectType 的可能值列在 第 144 頁的『MQOT_* (物件類型和延伸物件類型)』中。

ObjectType 是輸出欄位。 其起始值為 MQOT_Q。

OpenOptions (MQLONG)
OpenOptions 用來開啟所報告的物件。 僅在 MQSTS 第 2 版或更高版本中存在。

OpenOptions 的值取決於 MQSTAT Type 參數的值。

MQSTAT_TYPE_ASYNC_ERROR

零

MQSTAT_TYPE_RECONNECTION

零

MQSTAT_TYPE_RECONNECTION_ERROR

發生失敗時使用的 OpenOptions 。 MQSTS 結構中的 *CompCode* 和 *Reason* 欄位會報告失敗的原因。

OpenOptions 是輸出欄位。 其起始值為零。

PutFailure 計數 *(MQLONG)*
失敗的非同步放置作業數。

PutFailureCount 的值取決於 MQSTAT Type 參數的值。

MQSTAT_TYPE_ASYNC_ERROR

對 MQSTS 結構中指定且使用 MQCC_FAILED 完成之物件的非同步放置作業數。

MQSTAT_TYPE_RECONNECTION

零

MQSTAT_TYPE_RECONNECTION_ERROR

零

PutFailureCount 是輸出欄位。 其起始值為零。

PutSuccess 計數 *(MQLONG)*
成功的非同步放置作業數。

PutSuccessCount 的值取決於 MQSTAT Type 參數的值。

MQSTAT_TYPE_ASYNC_ERROR

對 MQSTS 結構中指定且使用 MQCC_OK 完成之物件的非同步放置作業數。

MQSTAT_TYPE_RECONNECTION

零

MQSTAT_TYPE_RECONNECTION_ERROR

零

PutSuccessCount 是輸出欄位。 其起始值為零。

PutWarning 計數 (MQLONG)
已結束但有警告的非同步放置作業數。

PutWarningCount 的值取決於 MQSTAT Type 參數的值。

MQSTAT_TYPE_ASYNC_ERROR

對 MQSTS 結構中指定且使用 MQCC_WARNING 完成之物件的非同步放置作業數。

MQSTAT_TYPE_RECONNECTION

零

MQSTAT_TYPE_RECONNECTION_ERROR

零

PutWarningCount 是輸出欄位。 其起始值為零。

SubName (MQCHARV)
失敗訂閱的名稱。 僅在 MQSTS 第 2 版或更高版本中存在。

SubName 的解譯取決於 MQSTAT Type 參數的值。

MQSTAT_TYPE_ASYNC_ERROR

零長度字串。

MQSTAT_TYPE_RECONNECTION

零長度字串。

MQSTAT_TYPE_RECONNECTION_ERROR

導致重新連線失敗的訂閱名稱。 如果沒有可用的訂閱名稱，或失敗與訂閱無關，則這是長度為零的字串。

SubName 是輸出欄位。 其起始值是長度為零的字串。

SubOptions (MQLONG)
用來開啟失敗訂閱的 SubOptions 。 僅在 MQSTS 第 2 版或更高版本中存在。

SubOptions 的解譯取決於 MQSTAT Type 參數的值。

MQSTAT_TYPE_ASYNC_ERROR

零

MQSTAT_TYPE_RECONNECTION

零

MQSTAT_TYPE_RECONNECTION_ERROR

發生失敗時使用的 SubOptions 。 如果失敗與訂閱主題無關，則傳回的值為零。

SubOptions 是輸出欄位。 其起始值為零。

原因 (MQLONG)
所報告作業的原因碼。

Reason 的解譯取決於 MQSTAT Type 參數的值。

MQSTAT_TYPE_ASYNC_ERROR

這是在 ObjectName 中指定的物件上先前非同步放置作業所產生的原因碼。

MQSTAT_TYPE_RECONNECTION

如果連線正在重新連接或無法重新連接，這是導致重新連線開始重新連接的原因碼。

如果目前已連接連線，則值為 MQRC_NONE。

MQSTAT_TYPE_RECONNECTION_ERROR

如果連線無法重新連接，這是導致重新連線失敗的原因碼。

如果連線目前已連接或重新連接，則值為 MQRC_NONE。

Reason 是輸出欄位。 其起始值為 MQRC_NONE。

ResolvedObject 名稱 (MQCHAR48)
在本端佇列管理程式解析名稱之後，在 *ObjectName* 中所指名的物件名稱。

ResolvedObjectName 的解譯取決於 MQSTAT Type 參數的值。

MQSTAT_TYPE_ASYNC_ERROR

ResolvedObjectName 是在本端佇列管理程式解析名稱之後，在 *ObjectName* 中命名的物件名稱。 傳回的名稱是存在於 *ResolvedQMgrName* 所識別佇列管理程式上的物件名稱。

MQSTAT_TYPE_RECONNECTION

空白。

MQSTAT_TYPE_RECONNECTION_ERROR

空白。

ResolvedObjectName 是輸出欄位。 它的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

ResolvedQMgr 名稱 (MQCHAR48)
本端佇列管理程式解析名稱之後的目的地佇列管理程式名稱。

ResolvedQMgrName 的解譯取決於 MQSTAT Type 參數的值。

MQSTAT_TYPE_ASYNC_ERROR

ResolvedQMgrName 是本端佇列管理程式解析名稱之後的目的地佇列管理程式名稱。 傳回的名稱是擁有 *ResolvedObjectName* 所識別物件的佇列管理程式名稱。 *ResolvedQMgrName* 可能是本端佇列管理程式的名稱。

MQSTAT_TYPE_RECONNECTION

空白。

MQSTAT_TYPE_RECONNECTION_ERROR

空白。

ResolvedQMgrName 一律是輸出欄位。 它的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

StrucId (MQCHAR4)
狀態報告結構 MQSTS 的 ID。

StrucId 是結構 ID。 值必須為:

MQSTS_STRUC_ID
狀態報告結構的 ID。

對於 C 程式設計語言，也會定義常數 MQSTS_STRUC_ID_ARRAY；其值與 MQSTS_STRUC_ID 相同，但卻是字元陣列而非字串。

StrucId 一律是輸入欄位。 其起始值為 MQSTS_STRUC_ID。

版本 *(MQLONG)*
結構版本號碼。

值必須是:

MQSTS_VERSION_1
第 1 版狀態報告結構。

MQSTS_VERSION_2
第 2 版狀態報告結構。

下列常數指定現行版本的版本號碼:

MQSTS_CURRENT_VERSION
狀態報告結構的現行版本。 現行版本是 MQSTS_VERSION_2。

Version 一律是輸入欄位。 其起始值為 MQSTS_VERSION_1。

MQSTS 的起始值和語言宣告

表 *552: MQSTS* 中欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQSTS_STRUC_ID	'STAT¬'
Version	MQSTS_VERSION_1	1
CompCode	MQCC_OK	0
Reason	MQRC_NONE	0
PutSuccessCount	無	0
PutWarningCount	無	0
PutFailureCount	無	0
ObjectType	MQOT_Q	1
ObjectName	無	空字串或空白
ObjectQMgrName	無	空字串或空白
ResolvedObjectName	無	空字串或空白
ResolvedQMgrName	無	空字串或空白
ObjectString	MQCHARV_DEFAULT	{NULL,0,0,0,-3}
SubName	MQCHARV_DEFAULT	{NULL,0,0,0,-3}
OpenOptions	無	0
SubOptions	無	0

表 *552: MQSTS* 中欄位的起始值 *(*繼續*)*

欄位名稱	常數名稱	常數值

附註:

1. 符號 ¬ 代表單一空白字元。
2. 空值字串或空白值表示 C 中的空值字串，以及其他程式設計語言中的空白字元。
3. 在 C 程式設計語言中，巨集變數 MQSTS_DEFAULT 包含上述值。 您可以用下列方式來提供結構中欄位的起始值:

```
MQSTS MySTS = {MQSTS_DEFAULT};
```

C 宣告

```
typedef struct tagMQSTS MQSTS;
struct tagMQSTS {
 MQCHAR4   StrucId;            /* 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 */
};
```

COBOL 宣告

```
      ** 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 **
```

PL/I 宣告

```
dcl
   1 MQSTS based,
     3 StrucId             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 */
```

High Level Assembler 宣告

```
     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
                                    ORG   MQSTS_OBJECTSTRING
MQSTS_OBJECTSTRING_AREA       DS    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)
```

MQTM-觸發訊息

下表彙總結構中的欄位。

表 *553: MQTM* 中的欄位

欄位	說明	主題
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
QName	觸發佇列的名稱	完整名稱
ProcessName	處理程序物件的名稱	ProcessName
TriggerData	觸發資料	TriggerData
ApplType	應用程式類型	ApplType
ApplId	應用程式 ID	ApplId
EnvData	環境資料	EnvData
UserData	使用者資料	UserData

MQTM 概觀

目的:MQTM 結構說明當佇列發生觸發事件時，佇列管理程式傳送至觸發監視器應用程式之觸發訊息中的資料。

此結構是 WebSphere MQ 觸發監視器介面 (TMI) 的一部分，這是其中一個 WebSphere MQ 架構介面。

格式名稱:MQFMT_TRIGGER。

字集及編碼:MQTM 中的字元資料是在產生 MQTM 之佇列管理程式的字集中。 MQTM 中的數值資料採用產生 MQTM 之佇列管理程式的機器編碼。

MQTM 的字集及編碼是由下列項目中的 *CodedCharSetId* 及 *Encoding* 欄位所提供:

- MQMD (如果 MQTM 結構是在訊息資料的開頭) ，或
- MQTM 結構之前的標頭結構 (所有其他觀察值)。

用法: 觸發監視器應用程式可能需要將觸發訊息中的部分或所有資訊傳遞至觸發監視器應用程式啟動的應用程式。 已啟動的應用程式可能需要的資訊包括 *QName*、 *TriggerData* 及 *UserData*。 觸發監視器應用程式可以將 MQTM 結構直接傳遞至已啟動的應用程式，或改為傳遞 MQTMC2 結構，視環境允許的內容及已啟動應用程式的方便程度而定。 如需 MQTMC2 的相關資訊，請參閱 第 527 頁的『MQTMC2 -觸發訊息 2 (字元格式)』。

- 在 z/OS 上，對於使用 CKTI 交易啟動的 MQAT_CICS 應用程式，整個觸發訊息結構 MQTM 可供已啟動的交易使用; 可以使用 EXEC CICS RETRIEVE 指令來擷取資訊。

- 在 IBM i 上，隨 WebSphere MQ 提供的觸發監視器應用程式會將 MQTMC2 結構傳遞至已啟動的應用程式。

如需使用觸發程式的相關資訊，請參閱 使用觸發程式來啟動 WebSphere MQ 應用程式 。

觸發訊息的 MQMD: 佇列管理程式所產生觸發訊息的 MQMD 中的欄位設定如下:

MQMD 中的欄位	**使用的值**
StrucId	MQMD_STRUC_ID
Version	MQMD_VERSION_1
Report	MQRO_NONE
MsgType	MQMT_DATAGRAM
Expiry	MQEI_UNLIMITED
Feedback	MQFB_NONE
Encoding	MQENC_NATIVE
CodedCharSetId	佇列管理程式的 *CodedCharSetId* 屬性
Format	MQFMT_TRIGGER
Priority	起始佇列的 *DefPriority* 屬性
Persistence	MQPER_NOT_PERSISTENT
MsgId	唯一值
CorrelId	MQCI_NONE
BackoutCount	0
ReplyToQ	空白
ReplyToQMgr	佇列管理程式的名稱
UserIdentifier	空白
AccountingToken	MQACT_NONE
ApplIdentityData	空白
PutApplType	MQAT_QMGR，或適用於訊息通道代理程式
PutApplName	佇列管理程式名稱的前 28 個位元組
PutDate	傳送觸發訊息的日期
PutTime	傳送觸發訊息的時間
ApplOriginData	空白

建議使用產生觸發訊息的應用程式來設定類似的值，但下列項目除外:

- *Priority* 欄位可以設為 MQPRI_PRIORITY_AS_Q_DEF (放置訊息時，佇列管理程式會將此變更為起始佇列的預設優先順序)。
- *ReplyToQMgr* 欄位可以設為空白 (當放置訊息時，佇列管理程式會將此變更為本端佇列管理程式的名稱)。
- 將環境定義欄位設為適當的應用程式。

MQTM 的欄位

MQTM 結構包含下列欄位; 這些欄位按 **字母順序**說明:

ApplId (MQCHAR256)

這是識別要啟動之應用程式的字串，由接收觸發訊息的觸發監視器應用程式使用。 佇列管理程式會使用 *ProcessName* 欄位所識別之處理程序物件的 *ApplId* 屬性值來起始設定此欄位; 如需此屬性的詳細資料，請參閱 第 760 頁的『程序定義的屬性』 。 此資料的內容對佇列管理程式不重要。

ApplId 的意義由觸發監視器應用程式決定。 WebSphere MQ 所提供的觸發監視器需要 *ApplId* 是可執行程式的名稱。 下列注意事項適用於指出的環境:

- 在 z/OS 上， *ApplId* 是:
  - CICS 交易 ID ，適用於使用 CICS 觸發監視器交易 CKTI 啟動的應用程式
  - IMS 交易 ID ，適用於使用 IMS 觸發監視器 CSQQTRMN 啟動的應用程式
- 在 Windows 系統上，程式名稱可以字首為磁碟機及目錄路徑。
- 在 IBM i 上，程式名稱可以使用檔案庫名稱及/字元作為字首。
- 在 UNIX 系統上，程式名稱的字首可以是目錄路徑。

此欄位的長度由 MQ_PROCESS_APPL_ID_LENGTH 提供。 此欄位的起始值是 C 中的空字串，在其他程式設計語言中則是 256 個空白字元。

ApplType (MQLONG)

這會識別要啟動的程式本質，並由接收觸發訊息的觸發監視器應用程式使用。 佇列管理程式會使用 *ProcessName* 欄位所識別之處理程序物件的 *ApplType* 屬性值來起始設定此欄位; 如需此屬性的詳細資料，請參閱 第 760 頁的『程序定義的屬性』 。 此資料的內容對佇列管理程式不重要。

ApplType 可以具有下列其中一個標準值。 也可以使用使用者定義類型，但應該限制為 MQAT_USER_FIRST 到 MQAT_USER_LAST 範圍內的值:

MQAT_AIX
AIX 應用程式 (與 MQAT_UNIX 的值相同)。

MQ 批次
批次應用程式 (batch application)

MQ 屬性分配管理系統
分配管理系統應用程式

MQAT_CICS
CICS 交易。

MQAT_CICS_BRIDGE
CICS 橋接器應用程式。

MQAT_CICS_VSE
CICS/VSE 交易。

MQAT_DOS
PC DOS 上的 WebSphere MQ MQI 用戶端應用程式。

MQAT_IMS
IMS 應用程式。

MQAT_IMS_BRIDGE
IMS 橋接器應用程式。

MQAT_JAVA
Java 應用程式。

MQAT_MVS
MVS 或 TSO 應用程式 (與 MQAT_ZOS 的值相同)。

MQAT_NOTES_AGENT
Lotus Notes 代理程式應用程式。

MQAT_NSK
HP Integrity NonStop Server 應用程式。

MQAT_OS390
OS/390 應用程式 (與 MQAT_ZOS 相同值)。

MQAT_OS400
IBM i 應用程式。

MQ 屬性 _ rrS_BATCH
RRS 批次應用程式。

MQAT_UNIX
UNIX 應用程式。

MQAT_UNKNOWN
應用程式類型不明。

MQAT_USER
使用者定義應用程式類型。

MQAT_VOS
Stratus VOS 應用程式。

MQ 視窗
16 位元 Windows 應用程式。

MQAT_WINDOWS_NT
32 位元 Windows 應用程式。

MQAT_WLM
z/OS 工作量管理程式應用程式。

MQAT_XCF
XCF。

MQAT_ZOS
z/OS 應用程式。

MQAT_USER_FIRST
使用者定義應用程式類型的最低值。

MQAT_USER_LAST
使用者定義應用程式類型的最高值。

此欄位的起始值為 0。

EnvData (MQCHAR128)

這是一個字串，包含要啟動之應用程式的環境相關資訊，並由接收觸發訊息的 trigger-monitor 應用程式使用。 佇列管理程式會使用 *ProcessName* 欄位所識別之處理程序物件的 *EnvData* 屬性值來起始設定此欄位; 如需此屬性的詳細資料，請參閱 第 760 頁的『程序定義的屬性』。 此資料的內容對佇列管理程式不重要。

在 z/OS 上，若為使用 CKTI 交易啟動的 CICS 應用程式，或要使用 CSQQTRMN 交易啟動的 IMS 應用程式，則不會使用此資訊。

此欄位的長度由 MQ_PROCESS_ENV_DATA_LENGTH 提供。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 128 個空白字元。

ProcessName (MQCHAR48)

這是指定給觸發佇列的佇列管理程式處理程序物件名稱，可由接收觸發訊息的觸發監視器應用程式使用。 佇列管理程式會使用 *QName* 欄位所識別佇列的 *ProcessName* 屬性值來起始設定此欄位; 如需此屬性的詳細資料，請參閱 第 731 頁的『佇列的屬性』。

短於欄位定義長度的名稱一律以空白填補在右側; 它們不會過早以空值字元結束。

此欄位的長度由 MQ_PROCESS_NAME_LENGTH 提供。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

完整名稱 *(MQCHAR48)*

這是發生觸發事件的佇列名稱，由觸發監視器應用程式啟動的應用程式使用。 佇列管理程式會使用所觸發佇列的 *QName* 屬性值來起始設定此欄位; 如需此屬性的詳細資料，請參閱 第 731 頁的『佇列的屬性』 。

短於欄位定義長度的名稱會以空白填補在右側; 它們不會過早以空值字元結束。

此欄位的長度由 MQ_Q_NAME_LENGTH 指定。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

StrucId (MQCHAR4)

這是結構 ID。 值必須為:

MQTM_STRUC_ID
觸發訊息結構的 ID。

若為 C 程式設計語言，也會定義常數 MQTM_STRUCT_ID_ARRAY; 此值與 MQTM_STRUC_ID 相同，但是字元陣列而非字串。

此欄位的起始值是 MQTM_STRUC_ID。

TriggerData (MQCHAR64)

這是免費格式資料，供接收觸發訊息的觸發監視器應用程式使用。 佇列管理程式會使用 *QName* 欄位所識別佇列的 *TriggerData* 屬性值來起始設定此欄位; 如需此屬性的詳細資料，請參閱 第 731 頁的『佇列的屬性』 。 此資料的內容對佇列管理程式不重要。

在 z/OS 上，對於使用 CKTI 交易啟動的 CICS 應用程式，不會使用此資訊。

此欄位的長度由 MQ_TRIGGER_DATA_LENGTH 提供。 此欄位的起始值是 C 中的空字串，而在其他程式設計語言中則是 64 個空白字元。

UserData (MQCHAR128)

這是一個字串，包含與要啟動之應用程式相關的使用者資訊，並由接收觸發訊息的觸發監視器應用程式使用。 佇列管理程式會使用 *ProcessName* 欄位所識別之處理程序物件的 *UserData* 屬性值來起始設定此欄位; 如需此屬性的詳細資料，請參閱 第 760 頁的『程序定義的屬性』 。 此資料的內容對佇列管理程式不重要。

對於 Microsoft Windows，如果要將程序定義傳遞至 **runmqtrm**，則字串不得包含雙引號。

此欄位的長度由 MQ_PROCESS_USER_DATA_LENGTH 指定。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 128 個空白字元。

版本 *(MQLONG)*

這是結構版本號碼。 值必須為:

MQTM_VERSION_1
觸發訊息結構的版本號碼。

下列常數指定現行版本的版本號碼:

MQTM_CURRENT_VERSION
觸發訊息結構的現行版本。

此欄位的起始值為 MQTM_VERSION_1。

MQTM 的起始值和語言宣告

表 *554: MQTM for MQTM* 中欄位的起始值

欄位名稱	常數名稱	常數值
StrucId	MQTM_STRUC_ID	'TM¬¬'
Version	MQTM_VERSION_1	1
QName	無	空字串或空白

表 554: *MQTM for MQTM* 中欄位的起始值 *(繼續)*		
欄位名稱	**常數名稱**	**常數值**
ProcessName	無	空字串或空白
TriggerData	無	空字串或空白
ApplType	無	0
ApplId	無	空字串或空白
EnvData	無	空字串或空白
UserData	無	空字串或空白

附註:

1. 符號 ¬ 代表單一空白字元。
2. 空值字串或空白值表示 C 中的空值字串，以及其他程式設計語言中的空白字元。
3. 在 C 程式設計語言中，巨集變數 MQTM_DEFAULT 包含上述值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQTM MyTM = {MQTM_DEFAULT};
```

C 宣告

```
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  ApplId;       /* Application identifier */
  MQCHAR128  EnvData;      /* Environment data */
  MQCHAR128  UserData;     /* User data */
};
```

COBOL 宣告

```
**   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).
```

PL/I 宣告

```
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 */
```

High Level Assembler 宣告

```
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)
```

Visual Basic 宣告

```
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
```

MQTMC2 -觸發訊息 2 (字元格式)

下表彙總結構中的欄位。

表 *555: MQTMC2* 中的欄位

欄位	說明	主題
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
QName	觸發佇列的名稱	完整名稱
ProcessName	處理程序物件的名稱	ProcessName
TriggerData	觸發資料	TriggerData
ApplType	應用程式類型	ApplType
ApplId	應用程式 ID	ApplId
EnvData	環境資料	EnvData
UserData	使用者資料	UserData
QMgrName	佇列管理程式名稱	QMgrName

MQTMC2 概觀

用途: 當觸發監視器應用程式從起始佇列擷取觸發訊息 (MQTM) 時，觸發監視器可能需要將觸發訊息中的部分或所有資訊傳遞至觸發監視器啟動的應用程式。

已啟動應用程式可能需要的資訊包括 *QName*、 *TriggerData* 及 *UserData*。 觸發監視器應用程式可以直接將 MQTM 結構傳遞至已啟動的應用程式，或改為傳遞 MQTMC2 結構，視環境允許的內容及已啟動應用程式的方便程度而定。

此結構是 WebSphere MQ 觸發監視器介面 (TMI) 的一部分，這是其中一個 WebSphere MQ 架構介面。

字集及編碼: MQTMC2 中的字元資料位於本端佇列管理程式的字集; 這是由 *CodedCharSetId* 佇列管理程式屬性所提供。

用法: MQTMC2 結構非常類似於 MQTM 結構的格式。 差異是 MQTM 中的非字元欄位在 MQTMC2 中變更為相同長度的字元欄位，並在結構結尾新增佇列管理程式名稱。

- 在 z/OS 上，對於使用 CSQQTRMN 應用程式啟動的 MQAT_IMS 應用程式， MQTMC2 結構可供已啟動的應用程式使用。
- 在 IBM i 上，隨 WebSphere MQ 提供的觸發監視器應用程式會將 MQTMC2 結構傳遞至已啟動的應用程式。

MQTMC2 的欄位

MQTMC2 結構包含下列欄位; 這些欄位按 **字母順序**說明:

ApplId (MQCHAR256)

應用程式 ID。

請參閱 MQTM 結構中的 *ApplId* 欄位。

ApplType (MQCHAR4)
應用程式類型。

無論原始觸發訊息的 MQTM 結構中的 *ApplType* 欄位值為何，此欄位一律包含空白。

EnvData (MQCHAR128)
環境資料。

請參閱 MQTM 結構中的 *EnvData* 欄位。

ProcessName (MQCHAR48)
處理程序物件的名稱。

請參閱 MQTM 結構中的 *ProcessName* 欄位。

QMgrName (MQCHAR48)
佇列管理程式名稱。

這是發生觸發事件的佇列管理程式名稱。

完整名稱 *(MQCHAR48)*
觸發佇列的名稱。

請參閱 MQTM 結構中的 *QName* 欄位。

StrucId (MQCHAR4)
結構 ID。

值必須為:

MQTMC_STRUC_ID
觸發訊息 (字元格式) 結構的 ID。

對於 C 程式設計語言，也會定義常數 MQTMC_STRUC_ID_ARRAY; 此值與 MQTMC_STRUC_ID 相同，但卻是字元陣列而非字串。

TriggerData (MQCHAR64)
觸發資料。

請參閱 MQTM 結構中的 *TriggerData* 欄位。

UserData (MQCHAR128)
使用者資料。

請參閱 MQTM 結構中的 *UserData* 欄位。

版本 *(MQCHAR4)*
結構版本號碼。

值必須為:

MQTMC_VERSION_2
第 2 版觸發訊息 (字元格式) 結構。

對於 C 程式設計語言，也會定義常數 MQTMC_VERSION_2_ARRAY ; 其值與 MQTMC_VERSION_2 相同，但卻是字元陣列而非字串。

下列常數指定現行版本的版本號碼:

MQTMC_CURRENT_VERSION
觸發訊息 (字元格式) 結構的現行版本。

MQTMC2 的起始值和語言宣告

表 *556: MQTMC2* 中 *MQTMC2* 的欄位起始值

欄位名稱	常數名稱	常數值
StrucId	MQTMC_STRUC_ID	'TMC¬'
Version	MQTMC_VERSION_2	'¬¬¬2'
QName	無	空字串或空白
ProcessName	無	空字串或空白
TriggerData	無	空字串或空白
ApplType	無	空白
ApplId	無	空字串或空白
EnvData	無	空字串或空白
UserData	無	空字串或空白
QMgrName	無	空字串或空白

附註:

1. 符號 ¬ 代表單一空白字元。
2. 空值字串或空白值表示 C 中的空值字串，以及其他程式設計語言中的空白字元。
3. 在 C 程式設計語言中，巨集變數 MQTMC2_DEFAULT 包含上述值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQTMC2 MyTMC = {MQTMC2_DEFAULT};
```

C 宣告

```
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 */
};
```

COBOL 宣告

```
**   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).
```

PL/I 宣告

```
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 */
```

High Level Assembler 宣告

```
MQTMC              DSECT
MQTMC_STRUCID      DS   CL4    Structure identifier
MQTMC_VERSION      DS   CL4    Structure version number
MQTMC_QNAME        DS   CL48   Name of triggered queue
MQTMC_PROCESSNAME  DS   CL48   Name of process object
MQTMC_TRIGGERDATA  DS   CL64   Trigger data
MQTMC_APPLTYPE     DS   CL4    Application type
MQTMC_APPLID       DS   CL256  Application identifier
MQTMC_ENVDATA      DS   CL128  Environment data
MQTMC_USERDATA     DS   CL128  User data
MQTMC_QMGRNAME     DS   CL48   Queue manager name
*
MQTMC_LENGTH       EQU  *-MQTMC
```

```
                  ORG  MQTMC
MQTMC_AREA        DS   CL(MQTMC_LENGTH)
```

Visual Basic 宣告

```
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
```

MQWIH-工作資訊標頭

下表彙總結構中的欄位。

表 *557: MQWIH* 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
StrucLength	MQWIH 結構的長度	StrucLength
Encoding	MQWIH 之後的資料的數字編碼	編碼
CodedCharSetId	MQWIH 之後的資料的字集 ID	CodedCharSetId
Format	MQWIH 之後的資料的格式名稱	格式
Flags	旗標	旗標
ServiceName	服務名稱	ServiceName
ServiceStep	服務步驟名稱	ServiceStep
MsgToken	訊息記號	MsgToken
Reserved	已保留	已保留

MQWIH 概觀

可用性: 所有 WebSphere MQ 系統，以及連接至這些系統的 WebSphere MQ 用戶端。

目的:MQWIH 結構說明要由 z/OS 工作量管理程式處理之訊息開頭必須呈現的資訊。

格式名稱:MQFMT_WORK_INFO_HEADER。

字集及編碼:MQWIH 結構中的欄位是由 MQWIH 之前的標頭結構中的 *CodedCharSetId* 及 *Encoding* 欄位所提供的字集及編碼，如果 MQWIH 是在應用程式訊息資料的開頭，則是由 MQMD 結構中的那些欄位所提供。

對於佇列名稱中有效的字元，字集必須是具有單位元組字元的字集。

用法: 如果訊息要由 z/OS 工作量管理程式處理，則訊息必須以 MQWIH 結構開頭。

MQWIH 的欄位

MQWIH 結構包含下列欄位; 這些欄位按 **字母順序**進行說明:

CodedCharSetId (MQLONG)

這會指定遵循 MQWIH 結構之資料的字集 ID; 它不適用於 MQWIH 結構本身中的字元資料。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。 您可以使用下列特殊值:

MQCCSI_INHERIT
此結構 後面 的資料中的字元資料與此結構的字集相同。

佇列管理程式會將訊息中所傳送結構的這個值變更為結構的實際字集 ID。 如果沒有發生錯誤， MQGET 呼叫不會傳回值 MQCCSI_INHERIT。

如果 MQMD 中的 *PutApplType* 欄位值為 MQAT_BROKER ，則無法使用 MQCCSI_INHERIT。

此欄位的起始值為 MQCCSI_UNDEFINED。

編碼 *(MQLONG)*

這會指定遵循 MQWIH 結構之資料的數值編碼; 它不適用於 MQWIH 結構本身中的數值資料。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。

此欄位的起始值為 0。

旗標 *(MQLONG)*

值必須為:

MQWIH_NONE
沒有旗標。

此欄位的起始值是 MQWIH_NONE。

格式 *(MQCHAR8)*

這會指定遵循 MQWIH 結構的資料格式名稱。

在 MQPUT 或 MQPUT1 呼叫上，應用程式必須將此欄位設為適合資料的值。 此欄位的編碼規則與 MQMD 中 *Format* 欄位的編碼規則相同。

此欄位的長度由 MQ_FORMAT_LENGTH 提供。 此欄位的起始值為 MQFMT_NONE。

MsgToken (MQBYTE16)

這是唯一識別訊息的訊息記號。

對於 MQPUT 和 MQPUT1 呼叫，會忽略此欄位。 此欄位的長度由 MQ_MSG_TOKEN_LENGTH 給定。 此欄位的起始值為 MQMTOK_NONE。

保留 *(MQCHAR32)*

這是保留欄位; 它必須是空白。

ServiceName (MQCHAR32)

這是要處理訊息的服務名稱。

此欄位的長度由 MQ_SERVICE_NAME_LENGTH 提供。 此欄位的起始值為 32 個空白字元。

ServiceStep (MQCHAR8)

這是與訊息相關的 *ServiceName* 步驟名稱。

此欄位的長度由 MQ_SERVICE_STEP_LENGTH 提供。 此欄位的起始值為 8 個空白字元。

StrucId (MQCHAR4)

這是結構 ID。 值必須為:

MQWIH_STRUC_ID
工作資訊標頭結構的 ID。

對於 C 程式設計語言，也會定義常數 MQWIH_STRUC_ARRAY; 此值與 MQWIH_STRUC_ID 相同，但卻是字元陣列而非字串。

此欄位的起始值是 MQWIH_STRUC_ID。

StrucLength (MQLONG)

這是 MQWIH 結構的長度。 值必須為:

MQWIH_LENGTH_1
version-1 工作資訊標頭結構的長度。

下列常數指定現行版本的長度:

MQWIH_CURRENT_LENGTH
工作資訊標頭結構現行版本的長度。

此欄位的起始值為 MQWIH_LENGTH_1。

版本 *(MQLONG)*

這是結構版本號碼。 值必須為:

MQWIH_VERSION_1
Version-1 工作資訊標頭結構。

下列常數指定現行版本的版本號碼:

MQWIH_CURRENT_VERSION
工作資訊標頭結構的現行版本。

此欄位的起始值是 MQWIH_VERSION_1。

MQWIH 的起始值及語言宣告

表 *558: MQWIH* 中 *MQWIH* 的欄位起始值

欄位名稱	常數名稱	常數值
StrucId	MQWIH_STRUC_ID	'WIH¬'
Version	MQWIH_VERSION_1	1
StrucLength	MQWIH_LENGTH_1	120
Encoding	無	0
CodedCharSetId	未定義 MQCCSI_UNDEFINED	0
Format	MQFMT_NONE	空白
Flags	MQWIH_NONE	0
ServiceName	無	空白
ServiceStep	無	空白
MsgToken	MQMTOK_NONE	空值
Reserved	無	空白

附註:

1. 符號 ¬ 代表單一空白字元。
2. 在 C 程式設計語言中，巨集變數 MQWIH_DEFAULT 包含上述值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQWIH MyWIH = {MQWIH_DEFAULT};
```

C 宣告

```
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 */
};
```

COBOL 宣告

```
**   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-STRUCLENGTH    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).
```

PL/I 宣告

```
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 */
```

High Level Assembler 宣告

```
MQWIH                DSECT
MQWIH_STRUCID        DS   CL4   Structure identifier
MQWIH_VERSION        DS   F     Structure version number
MQWIH_STRUCLENGTH    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)
```

Visual Basic 宣告

```
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
```

MQXP-結束參數區塊

下表彙總結構中的欄位。

表 *559: MQXP* 中的欄位

欄位	說明	主題
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
ExitId	結束程式 ID	ExitId
ExitReason	呼叫結束的原因	ExitReason
ExitResponse	結束的回應	ExitResponse
ExitCommand	API 呼叫碼	ExitCommand
ExitParmCount	參數計數	ExitParm 計數
ExitUserArea	使用者區域	ExitUser 區域

MQXP 概觀

可用性: z/OS。

目的:MQXP 結構用來作為 API 交互結束程式的輸入/輸出參數。 如需此結束程式的相關資訊，請參閱 API 交互結束程式。

字集及編碼:MQXP 中的字元資料是本端佇列管理程式的字集; 這是由 *CodedCharSetId* 佇列管理程式屬性所提供。 MQXP 中的數值資料採用原生機器編碼; 這是由 MQENC_NATIVE 所提供。

MQXP 的欄位

MQXP 結構包含下列欄位; 這些欄位會按 **字母順序**來說明:

ExitCommand (MQLONG)

此欄位在進入結束常式時設定。 它會識別導致呼叫結束程式的 API 呼叫:

MQXC_CALLBACK
回呼呼叫。

MQXC_MQBACK
MQBACK 呼叫。

MQXC_MQCB
MQCB 呼叫。

MQXC_MQCLOSE
MQCLOSE 呼叫。

MQXC_MQCMIT
MQCMIT 呼叫。

MQXC_MQCTL
MQCTL 呼叫。

MQXC_MQGET
MQGET 呼叫。

MQXC_MQINQ
MQINQ 呼叫。

MQXC_MQOPEN
MQOPEN 呼叫。

MQXC_MQPUT
MQPUT 呼叫。

MQXC_MQPUT1
MQPUT1 呼叫。

MQXC_MQSET
MQSET 呼叫。

MQXC_MQSTAT
MQSTAT 呼叫。

MQXC_MQSUB
MQSUB 呼叫。

MQXC_MQSUBRQ
MQSUBRQ 呼叫。

這是結束程式的輸入欄位。

ExitId (MQLONG)

這是在進入結束常式時設定，並指出結束類型:

MQXT_API_CROSSING_EXIT
CICS 的 API 交互結束程式。

這是結束程式的輸入欄位。

ExitParm 計數 *(MQLONG)*

此欄位在進入結束常式時設定。 它包含 MQ 呼叫所採用的參數數目。 它們是:

呼叫名稱	參數數目
MQBACK	3
MQCLOSE	5
MQCMIT	3
MQGET	9

呼叫名稱	參數數目
MQINQ	10
MQOPEN	6
MQPUT	8
MQPUT1	8
MQSET	10

這是結束程式的輸入欄位。

ExitReason (MQLONG)

這是在進入結束常式時設定。 對於 API 交互結束程式，它會指出在執行 API 呼叫之前或之後呼叫常式:

MQXR_before
在 API 執行之前。

MQXR_AFTER
執行 API 之後。

這是結束程式的輸入欄位。

ExitResponse (MQLONG)

此值由結束程式設定，以與呼叫程式通訊。 已定義下列值:

MQXCC_OK
已順利完成結束。

MQXCC_SUPPRESS_FUNCTION
抑制函數。

當 API 呼叫 之前 呼叫的 API 交互結束程式設定此值時，不會執行 API 呼叫。 呼叫的 *CompCode* 設定為 MQCC_FAILED， *Reason* 設定為 MQRC_SUPPRESSED_BY_EXIT，且所有其他參數保留在結束程式離開它們時。

當 API 呼叫 之後 呼叫的 API 交互結束程式設定此值時，佇列管理程式會忽略它。

MQXCC_SKIP_FUNCTION
跳過函數。

當 API 交互結束程式在 API 呼叫 之前 設定此值時，不會執行 API 呼叫; 當結束程式離開它們時，*CompCode* 和 *Reason* 及所有其他參數都會保留。

當 API 呼叫 之後 呼叫的 API 交互結束程式設定此值時，佇列管理程式會忽略它。

這是結束程式的輸出欄位。

ExitUser 區域 *(MQBYTE16)*

這是可供結束程式使用的欄位。 在第一次呼叫作業的結束程式之前，它會針對欄位長度起始設定為二進位零，然後在呼叫結束程式時保留結束程式對此欄位所做的任何變更。 下列是已定義的值:

MQXUA_NONE
沒有使用者資訊。

欄位長度的值為二進位零。

對於 C 程式設計語言，也會定義常數 MQXUA_NONE_ARRAY; 其值與 MQXUA_NONE 相同，但卻是字元陣列而非字串。

此欄位的長度由 MQ_EXIT_USER_AREA_LENGTH 指定。 這是結束程式的輸入/輸出欄位。

保留 *(MQLONG)*

這是保留欄位。 其值對結束程式不重要。

StrucId (MQCHAR4)

這是結構 ID。 值必須為:

MQXP_STRUC_ID
結束程式參數結構的 ID。

對於 C 程式設計語言，也會定義常數 MQXP_STRUC_ARRAY; 此值與 MQXP_STRUC_ID 相同，但它是字元陣列而非字串。

這是結束程式的輸入欄位。

版本 *(MQLONG)*

這是結構版本號碼。 值必須為:

MQXP_VERSION_1
結束參數區塊結構的版本號碼。

註: 引進此結構的新版本時，現有組件的佈置不會變更。 因此，結束程式必須檢查版本號碼是否等於或大於包含結束程式需要使用之欄位的最低版本。

這是結束程式的輸入欄位。

語言宣告

下列程式設計語言支援此結構。

C 宣告

```
typedef struct tagMQXP MQXP;
struct tagMQXP {
  MQCHAR4   StrucId;        /* 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 */
};
```

COBOL 宣告

```
**   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).
```

PL/I 宣告

```
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 */
```

High Level Assembler 宣告

```
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)
```

MQXQH-傳輸佇列標頭

下表彙總結構中的欄位。

表 *560: MQXQH* 中的欄位		
欄位	**說明**	**主題**
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
RemoteQName	目的地佇列的名稱	RemoteQName
RemoteQMgrName	目的地佇列管理程式的名稱	RemoteQmgrName
MsgDesc	原始訊息描述子	MsgDesc

MQXQH 概觀

可用性: 所有 WebSphere MQ 系統及 WebSphere MQ 用戶端。

目的:MQXQH 結構說明當訊息位於傳輸佇列時，在應用程式訊息資料前面加上字首的資訊。 傳輸佇列是一種特殊類型的本端佇列，可暫時保留送往遠端佇列的訊息 (亦即，送往不屬於本端佇列管理程式的佇列)。 傳輸佇列由值為 MQUS_TRANSMISSION 的 *Usage* 佇列屬性表示。

格式名稱:MQFMT_XMIT_Q_HEADER。

字集及編碼:MQXOH 中的資料必須採用 *CodedCharSetId* 佇列管理程式屬性所提供的字集，以及 MQENC_NATIVE 所提供本端佇列管理程式的編碼。

在下列欄位中，將 MQXQH 的字集及編碼設為 *CodedCharSetId* 及 *Encoding* 欄位:

- 個別 MQMD (如果 MQXQH 結構是在訊息資料的開頭) ，或
- MQXQH 結構之前的標頭結構 (所有其他觀察值)。

用法: 傳輸佇列上的訊息有 兩個 訊息描述子:

- 其中一個訊息描述子與訊息資料分開儲存; 這稱為 個別訊息描述子，當訊息置於傳輸佇列時由佇列管理程式產生。 個別訊息描述子中的部分欄位會從應用程式在 MQPUT 或 MQPUT1 呼叫上提供的訊息描述子中複製。

個別訊息描述子是從傳輸佇列中移除訊息時，在 MQGET 呼叫的 *MsgDesc* 參數中傳回給應用程式的描述子。

- 第二個訊息描述子儲存在 MQXQH 結構中作為訊息資料的一部分; 這稱為 內嵌訊息描述子，是應用程式在 MQPUT 或 MQPUT1 呼叫上提供的訊息描述子副本 (具有次要變異)。

  內嵌訊息描述子一律是 version-1 MQMD。 如果應用程式放置的訊息對 MQMD 中的一個以上 version-2 欄位具有非預設值，則 MQMDE 結構會遵循 MQXQH ，然後接著應用程式訊息資料 (如果有的話)。 MQMDE 為:

  – 由佇列管理程式產生 (如果應用程式使用 version-2 MQMD 來放置訊息) ，或
  – 在應用程式訊息資料開始時已存在 (如果應用程式使用 version-1 MQMD 來放置訊息)。

  內嵌訊息描述子是從最終目的地佇列中移除訊息時，在 MQGET 呼叫的 *MsgDesc* 參數中傳回給應用程式的描述子。

個別訊息描述子中的欄位: 個別訊息描述子中的欄位由佇列管理程式設定，如下所示。 如果佇列管理程式不支援 version-2 MQMD ，則會使用 version-1 MQMD ，而不會失去功能。

個別 MQMD 中的欄位	使用的值
StrucId	MQMD_STRUC_ID
Version	MQMD_VERSION_2
Report	從內嵌訊息描述子複製，但 MQRO_ACCEPT_UNSUPP_IF_XMIT_MASK 將位元設為零。 (這可防止在傳輸佇列中放置或移除訊息時產生 COA 或 COD 報告訊息。)
MsgType	從內嵌訊息描述子複製。
Expiry	從內嵌訊息描述子複製。
Feedback	從內嵌訊息描述子複製。
Encoding	MQENC_NATIVE (請參閱附註)
CodedCharSetId	佇列管理程式的 *CodedCharSetId* 屬性。
Format	MQFMT_XMIT_Q_HEADER
Priority	從內嵌訊息描述子複製。
Persistence	從內嵌訊息描述子複製。
MsgId	佇列管理程式會產生新值。 此訊息 ID 不同於佇列管理程式可能針對內嵌訊息描述子所產生的 *MsgId* (請參閱上述)。
CorrelId	內嵌訊息描述子中的 *MsgId* 。 針對要放置到 SYSTEM.CLUSTER.TRANSMIT.QUEUE， *CorrelId* 保留供內部使用。
BackoutCount	0
ReplyToQ	從內嵌訊息描述子複製。
ReplyToQMgr	從內嵌訊息描述子複製。
UserIdentifier	從內嵌訊息描述子複製。
AccountingToken	從內嵌訊息描述子複製。 針對要放置到 SYSTEM.CLUSTER.TRANSMIT.QUEUE， *AccountingToken* 保留供內部使用。
ApplIdentityData	從內嵌訊息描述子複製。
PutApplType	MQAT_QMGR
PutApplName	佇列管理程式名稱的前 28 個位元組。
PutDate	將訊息放入傳輸佇列的日期。

個別 MQMD 中的欄位	使用的值
PutTime	將訊息放入傳輸佇列的時間。
ApplOriginData	空白
GroupId	MQGI_NONE
MsgSeqNumber	1
Offset	0
MsgFlags	無 MQMF_NONE
OriginalLength	未定義 MQOL_UNDEFINED

- 在 Windows 上， Micro Focus COBOL 的 MQENC_NATIVE 值不同於 C 的值。 在這些環境中，個別訊息描述子中 *Encoding* 欄位的值一律為 C 的值; 此值為 546 (十進位)。 此外， MQXQH 結構中的整數欄位採用對應於此值的編碼 (原生 Intel 編碼)。

內嵌訊息描述子中的欄位: 內嵌訊息描述子中的欄位值與 MQPUT 或 MQPUT1 呼叫之 *MsgDesc* 參數中的值相同，但下列項目除外:

- *Version* 欄位一律具有值 MQMD_VERSION_1。
- 如果 *Priority* 欄位具有值 MQPRI_PRIORITY_AS_Q_DEF，則會由佇列的 *DefPriority* 屬性值取代。
- 如果 *Persistence* 欄位具有值 MQPER_PERSISTENCE_AS_Q_DEF，它會取代為佇列的 *DefPersistence* 屬性值。
- 如果 *MsgId* 欄位具有值 MQMI_NONE，或者已指定 MQPMO_NEW_MSG_ID 選項，或者訊息是配送清單訊息，則佇列管理程式所產生的新訊息 ID 會取代 *MsgId* 。

  當配送清單訊息分割成放置在不同傳輸佇列上的較小配送清單訊息時，每一個新的內嵌訊息描述子中的 *MsgId* 欄位與原始配送清單訊息中的欄位相同。
- 如果指定了 MQPMO_NEW_CORREL_ID 選項，則 *CorrelId* 會取代為佇列管理程式所產生的新相關性 ID。
- 環境定義欄位會依照 *PutMsgOpts* 參數中指定的 MQPMO_ * _CONTEXT 選項所指示來設定; 環境定義欄位如下:
  - *AccountingToken*
  - *ApplIdentityData*
  - *ApplOriginData*
  - *PutApplName*
  - *PutApplType*
  - *PutDate*
  - *PutTime*
  - *UserIdentifier*
- 如果一或多個 version-2 欄位具有非預設值，則會從 MQMD 移除 version-2 欄位 (如果它們存在的話)，並將它們移至 MQMDE 結構。

將訊息放置在遠端佇列上: 當應用程式將訊息放置在遠端佇列上 (直接指定遠端佇列的名稱，或使用遠端佇列的本端定義) 時，本端佇列管理程式:

- 建立包含內嵌訊息描述子的 MQXQH 結構
- 附加 MQMDE (如果需要且尚未呈現)
- 附加應用程式訊息資料
- 將訊息放置在適當的傳輸佇列上

將訊息直接放置在傳輸佇列上: 應用程式也可以將訊息直接放置在傳輸佇列上。 在此情況下，應用程式必須以 MQXQH 結構作為應用程式訊息資料的字首，並以適當的值起始設定欄位。 此外， MQPUT 或 MQPUT1 呼叫的 *MsgDesc* 參數中的 *Format* 欄位值必須為 MQFMT_XMIT_Q_HEADER。

應用程式所建立 MQXQH 結構中的字元資料必須在本端佇列管理程式的字集 (由 *CodedCharSetId* 佇列管理程式屬性定義) 中，而整數資料必須在原生機器編碼中。 此外， MQXQH 結構中的字元資料必須以空白填補至欄位定義的長度; 資料不得使用空值字元過早結束，因為佇列管理程式不會將 MQXQH 結構中的空值及後續字元轉換成空白。

不過，佇列管理程式不會檢查 MQXQH 結構是否存在，或是否已指定欄位的有效值。

應用程式不應將其訊息直接放置到 SYSTEM.CLUSTER.TRANSMIT.QUEUE。

從傳輸佇列取得訊息: 從傳輸佇列取得訊息的應用程式必須以適當的方式處理 MQXQH 結構中的資訊。 在應用程式訊息資料開頭的 MQXQH 結構是否存在，由 MQGET 呼叫的 *MsgDesc* 參數的 *Format* 欄位中所傳回的 MQFMT_XMIT_Q_HEADER 值指出。 在 *MsgDesc* 參數中的 *CodedCharSetId* 及 *Encoding* 欄位中傳回的值指出 MQXQH 結構中字元及整數資料的字集及編碼。 應用程式訊息資料的字集及編碼是由內嵌訊息描述子中的 *CodedCharSetId* 及 *Encoding* 欄位所定義。

MQXQH 的欄位

MQXQH 結構包含下列欄位; 這些欄位按 **字母順序**進行說明:

MsgDesc (MQMD1)

這是內嵌的訊息描述子，是最初將訊息放置到遠端佇列時指定為 MQPUT 或 MQPUT1 呼叫上 *MsgDesc* 參數的訊息描述子 MQMD 的關閉副本。

註: 這是 version-1 MQMD。

此結構中的欄位起始值與 MQMD 結構中的欄位起始值相同。

RemoteQMgr 名稱 *(MQCHAR48)*

這是佇列管理程式或佇列共用群組的名稱，它擁有的佇列是訊息明顯的最終目的地。

如果訊息是配送清單訊息，則 *RemoteQMgrName* 為空白。

此欄位的長度由 MQ_Q_MGR_NAME_LENGTH 提供。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

RemoteQName (MQCHAR48)

這是訊息明顯最終目的地的訊息佇列名稱 (例如，如果此佇列在 *RemoteQMgrName* 定義為另一個遠端佇列的本端定義，則這可能證明不是最終目的地)。

如果訊息是配送清單訊息 (亦即，內嵌訊息描述子中的 *Format* 欄位是 MQFMT_DIST_HEADER) ，則 *RemoteQName* 為空白。

此欄位的長度由 MQ_Q_NAME_LENGTH 指定。 此欄位的起始值在 C 中是空字串，在其他程式設計語言中是 48 個空白字元。

StrucId (MQCHAR4)

這是結構 ID。 值必須為:

MQXqh_STRUC_ID
傳輸佇列標頭結構的 ID。

對於 C 程式設計語言，也會定義常數 MQXQH_STRUC_ID_ARRAY; 此值與 MQXQH_STRUC_ID 相同，但它是字元陣列而非字串。

此欄位的起始值為 MQXQH_STRUC_ID。

版本 *(MQLONG)*

這是結構版本號碼。 值必須為:

MQXQH_VERSION_1
傳輸佇列標頭結構的版本號碼。

下列常數指定現行版本的版本號碼:

MQXqh_CURRENT_VERSION
傳輸佇列標頭結構的現行版本。

此欄位的起始值為 MQXQH_VERSION_1。

MQXQH 的起始值和語言宣告

表 *561: MQXQH* 中 *MQXQH* 的欄位起始值

欄位名稱	常數名稱	常數值
StrucId	MQXqh_STRUC_ID	'XQH¬'
Version	MQXQH_VERSION_1	1
RemoteQName	無	空字串或空白
RemoteQMgrName	無	空字串或空白
MsgDesc	與 MQMD 相同的名稱和值; 請參閱 第 401 頁的表 515	-

附註:

1. 符號 ¬ 代表單一空白字元。
2. 空值字串或空白值表示 C 中的空值字串，以及其他程式設計語言中的空白字元。
3. 在 C 程式設計語言中，巨集變數 MQXQH_DEFAULT 包含上述值。 請透過下列方式來使用它，以提供結構中欄位的起始值:

```
MQXQH MyXQH = {MQXQH_DEFAULT};
```

C 宣告

```
typedef struct tagMQXQH MQXQH;
struct tagMQXQH {
  MQCHAR4   StrucId;         /* 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 */
};
```

COBOL 宣告

```
**   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).
```

PL/I 宣告

```
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,                         /* Original message descriptor */
   5 StrucId          char(4),       /* Structure identifier */
   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 */
```

High Level Assembler 宣告

```
MQXQH                           DSECT
MQXQH_STRUCID                   DS   CL4   Structure identifier
MQXQH_VERSION                   DS   F     Structure version number
MQXQH_REMOTEQNAME               DS   CL48  Name of destination queue
MQXQH_REMOTEQMGRNAME            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
*                                          data
MQXQH_MSGDESC_CODEDCHARSETID    DS   F     Character set identifier of
*                                          message data
MQXQH_MSGDESC_FORMAT            DS   CL8   Format name of message data
MQXQH_MSGDESC_PRIORITY          DS   F     Message priority
MQXQH_MSGDESC_PERSISTENCE       DS   F     Message persistence
MQXQH_MSGDESC_MSGID             DS   XL24  Message identifier
MQXQH_MSGDESC_CORRELID          DS   XL24  Correlation identifier
MQXQH_MSGDESC_BACKOUTCOUNT      DS   F     Backout counter
MQXQH_MSGDESC_REPLYTOQ          DS   CL48  Name of reply-to queue
MQXQH_MSGDESC_REPLYTOQMGR       DS   CL48  Name of reply queue manager
MQXQH_MSGDESC_USERIDENTIFIER    DS   CL12  User identifier
MQXQH_MSGDESC_ACCOUNTINGTOKEN   DS   XL32  Accounting token
MQXQH_MSGDESC_APPLIDENTITYDATA  DS   CL32  Application data relating to
*                                          identity
MQXQH_MSGDESC_PUTAPPLTYPE       DS   F     Type of application that put
*                                          the message
MQXQH_MSGDESC_PUTAPPLNAME       DS   CL28  Name of application that put
*                                          the message
MQXQH_MSGDESC_PUTDATE           DS   CL8   Date when message was put
MQXQH_MSGDESC_PUTTIME           DS   CL8   Time when message was put
MQXQH_MSGDESC_APPLORIGINDATA    DS   CL4   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)
```

Visual Basic 宣告

```
Type MQXQH
  StrucId        As String*4  'Structure identifier'
  Version        As Long      'Structure version number'
  RemoteQName    As String*48 'Name of destination queue'
  RemoteQMgrName As String*48 'Name of destination queue manager'
  MsgDesc        As MQMD1     'Original message descriptor'
End Type
```

函數呼叫

本節提供所有可能 MQI 呼叫的相關資訊。 針對每一個不同的呼叫，提供每一種可能語言的說明、語法、參數資訊、使用注意事項及語言呼叫。

通話說明

本節說明 MQI 呼叫。

- 第 547 頁的『MQBACK-回復變更』
- 第 551 頁的『MQBEGIN-開始工作單元』

- 第 554 頁的『MQBUFMH-將緩衝區轉換成訊息控點』
- 第 558 頁的『MQCB-管理回呼』
- 第 566 頁的『MQCB_XX_ENCODE_CASE_ONE function-回呼函數』
- 第 567 頁的『MQCLOSE-關閉物件』
- 第 574 頁的『MQCMIT-確定變更』
- 第 578 頁的『MQCONN-連接佇列管理程式』
- 第 585 頁的『MQCONNX-連接佇列管理程式 (延伸)』
- 第 590 頁的『MQCRTMH-建立訊息控點』
- 第 593 頁的『MQCTL-控制回呼』
- 第 599 頁的『MQDISC-切斷佇列管理程式連線』
- 第 602 頁的『MQDLTMH-刪除訊息控點』
- 第 604 頁的『MQDLTMP-刪除訊息內容』
- 第 607 頁的『MQGET-取得訊息』
- 第 618 頁的『MQINQ-查詢物件屬性』
- 第 633 頁的『MQINQMP-查詢訊息內容』
- 第 638 頁的『MQMHBUF-將訊息控點轉換為緩衝區』
- 第 641 頁的『MQOPEN-開啟物件』
- 第 657 頁的『MQPUT-放置訊息』
- 第 669 頁的『MQPUT1 -放置一則訊息』
- 第 678 頁的『MQSET-設定物件屬性』
- 第 684 頁的『MQSETMP-設定訊息內容』
- 第 688 頁的『MQSTAT-擷取狀態資訊』
- 第 638 頁的『MQMHBUF-將訊息控點轉換為緩衝區』
- 第 692 頁的『MQSUB-登錄訂閱』
- 第 698 頁的『MQSUBRQ-訂閱要求』

UNIX 平台上的線上說明 (採用 *man* 頁面形式) 可用於這些呼叫。

註: 與資料轉換 (MQXCNVC 和 MQ_DATA_CONV_EXIT) 相關聯的呼叫位於 第 791 頁的『資料轉換結束程式』中。

呼叫說明中使用的慣例

對於每一個呼叫，此主題集合以獨立於程式設計語言的格式提供呼叫參數及使用情形的說明。 在每一種支援的程式設計語言中，這後面接著呼叫的一般呼叫，以及其參數的一般宣告。

重要: 在撰寫 WebSphere MQ API 呼叫的程式碼時，您必須確定已提供所有相關參數 (如下列各節所述)。 否則會產生無法預期的結果。

每一個呼叫的說明包含下列區段:

呼叫名稱
呼叫名稱，後面接著呼叫目的的簡要說明。

參數
對於每一個參數，名稱後面接著用括弧 () 括住的資料類型 及下列其中一項:

輸入(input)
當您進行通話時，請在參數中提供資訊。

輸出
當呼叫完成或失敗時，佇列管理程式會在參數中傳回資訊。

輸入/輸出
當您進行呼叫時，您會在參數中提供資訊，當呼叫完成或失敗時，佇列管理程式會變更資訊。

例如:

Compcode (MQLONG)-輸出

在某些情況下，資料類型是結構。 在所有情況下，都有 第 214 頁的『基本資料類型』中資料類型或結構的相關資訊。

每個呼叫中的最後兩個參數是完成碼和原因碼。 完成碼指出呼叫是否順利完成、局部完成或完全未完成。 有關呼叫局部成功或失敗的進一步資訊，請參閱原因碼。 如需每一個完成碼和原因碼的相關資訊，請參閱 第 764 頁的『回覆碼』。

使用注意事項
通話的其他相關資訊，說明如何使用它，以及使用它的任何限制。

組譯語言呼叫
以組譯語言一般呼叫及其參數的宣告。

C 呼叫
呼叫的一般呼叫，以及其參數的宣告 (在 C 中)。

COBOL 呼叫
COBOL 中呼叫的一般呼叫及其參數的宣告。

PL/I 呼叫
PL/I 中呼叫的一般呼叫，以及其參數的宣告。

所有參數都依參照傳遞。

Visual Basic 呼叫
Visual Basic 中呼叫的一般呼叫及其參數的宣告。

其他表示法慣例如下:

常數
常數名稱以大寫顯示; 例如 MQOO_OUTPUT。 具有相同字首的一組常數如下所示 :MQIA_ *。 如需常數的值，請參閱 第 49 頁的『常數』。

陣列
在某些呼叫中，參數是沒有固定大小的字串陣列。 在這些參數的說明中，小寫 n 代表數值常數。 當您撰寫該參數的宣告時，請將 n 取代為您需要的數值。

使用 *C* 語言的呼叫

僅輸入 且類型為 MQHCONN、MQHOBJ、MQHMSG 或 MQLONG 的參數會依值傳遞。 對於所有其他參數，參數的 *address* 會依值傳遞。

您不需要在每次呼叫函數時指定位址所傳遞的所有參數。 如果您不需要特定參數，請指定空值指標作為函數呼叫上的參數，以取代參數資料的位址。 可以在呼叫說明中識別的參數。

未傳回任何參數作為呼叫的值; 在 C 術語中，這表示所有呼叫都會傳回 `void`。

宣告緩衝區參數

MQGET、 MQPUT 及 MQPUT1 呼叫各有一個具有未定義資料類型的參數: *Buffer* 參數。 使用此參數來傳送及接收應用程式的訊息資料。

此排序的參數在 C 範例中顯示為 MQBYTE 的陣列。 您可以用這種方式來宣告參數，但通常更方便將它們宣告為說明訊息中資料佈置的特定結構。 函數原型將參數宣告為 void 的指標，因此您可以將任何資料類型的位址指定為呼叫呼叫上的參數。

Pointer-to-void 是未定義格式之資料的指標。 它定義為:

```
typedef void *PMQVOID;
```

MQBACK-回復變更

MQBACK 呼叫會向佇列管理程式指出要取消自前次同步點以來所發生的所有訊息取得及放置。

作為工作單元的一部分放置的訊息會被刪除; 作為工作單元的一部分擷取的訊息會在佇列上恢復。

- 在 z/OS 上，此呼叫僅由批次程式 (包括 IMS 批次 DL/I 程式) 使用。
- 在 IBM i 上，以相容模式執行的應用程式不支援此呼叫。

語法

MQBACK (*Hconn*、 *Compcode*、 *Reason*)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 前一個 MQCONN 或 MQCONNX 呼叫已傳回 *Hconn* 的值。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130 ， X'852 ') 無法載入配接卡服務模組。

MQRC_API_EXIT_ERROR
(2374 ， X' 946 ') API 結束程式失敗。

MQRC_ASDID_MISMATCH
(2157 ， X'86D') 主要和起始 ASID 不同。

MQRC_CALL_IN_PROGRESS
(2219 ， X'8AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CF_STRUC_IN_USE
(2346 ， X'92A') 連結機能結構使用中。

MQRC_CONNECTION_BROKEN
(2009 ， X'7D9') 與佇列管理程式的連線遺失。

MQRC_ENVIRONMENT_ERROR
(2012 ， X'7DC') 在環境中呼叫無效。

MQRC_HCONN_ERROR
(2018 ， X'7E2') 連線控點無效。

MQRC_OBJECT_DAMAGED
(2101 ， X'835 ') 物件已損壞。

MQRC_OUTCOME_MIXED
(2123 ， X'84B') 確定或取消作業的結果混合。

MQRC_Q_MGR_STOPPING
(2162 ， X'872 ') 佇列管理程式關閉。

MQRC_RESOURCE_PROBLEM
(2102 , X'836 ') 可用的系統資源不足。

MQRC_STORAGE_MEDIUM_FULL
(2192 , X'890 ') 外部儲存媒體已滿。

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'817 ') 可用的儲存體不足。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

如需這些代碼的詳細資訊，請參閱 原因碼

使用注意事項

1. 只有在佇列管理程式本身協調工作單元時，才能使用此呼叫。 這可以是:
   - 本端工作單元，其中變更只會影響 MQ 資源。
   - 廣域工作單元，其中變更可能會影響屬於其他資源管理程式的資源，以及影響 MQ 資源。

   如需本端及廣域工作單元的進一步詳細資料，請參閱 第 551 頁的『MQBEGIN-開始工作單元』。
2. 在佇列管理程式未協調工作單元的環境中，請使用適當的回呼而非 MQBACK。 環境也可能支援應用程式異常終止所造成的隱含取消。
   - 在 z/OS 上，請使用下列呼叫:
     - 如果工作單元僅影響 MQ 資源，則批次程式 (包括 IMS 批次 DL/I 程式) 可以使用 MQBACK 呼叫。 不過，如果工作單元同時影響 MQ 資源和屬於其他資源管理程式的資源 (例如， DB2) ，請使用 z/OS 可回復資源服務 (RRS) 提供的 SRRBACK 呼叫。 SRRBACK 呼叫會取消對屬於已啟用 RRS 協調之資源管理程式的資源所做的變更。
     - CICS 應用程式必須使用 EXEC CICS SYNCPOINT ROLLBACK 指令來退出工作單元。 請勿對 CICS 應用程式使用 MQBACK 呼叫。
     - IMS 應用程式 (非批次 DL/I 程式) 必須使用 IMS 呼叫 (例如 ROLB) 來退出工作單元。 請勿對 IMS 應用程式 (非批次 DL/I 程式) 使用 MQBACK 呼叫。
   - 在 IBM i 上，針對佇列管理程式所協調的本端工作單元使用此呼叫。 這表示確定定義不得存在於工作層次，亦即，不得對工作發出具有 CMTSCOPE(*JOB) 參數的 STRCMTCTL 指令。
3. 如果應用程式以工作單元中未確定的變更結束，則那些變更的處置取決於應用程式是正常結束還是異常結束。 如需進一步詳細資料，請參閱 第 599 頁的『MQDISC-切斷佇列管理程式連線』 中的使用注意事項。
4. 當應用程式在邏輯訊息的群組或區段中放置或取得訊息時，佇列管理程式會保留與前次成功 MQPUT 及 MQGET 呼叫的訊息群組及邏輯訊息相關的資訊。 此資訊與佇列控點相關聯，並包括如下內容:
   - MQMD 中 *GroupId*、 *MsgSeqNumber*、 *Offset* 及 *MsgFlags* 欄位的值。
   - 訊息是否為工作單元的一部分。
   - 對於 MQPUT 呼叫: 訊息是持續還是非持續。

   佇列管理程式會保留 三組 群組及區段資訊，下列每一組各一組:
   - 前次成功的 MQPUT 呼叫 (這可以是工作單元的一部分)。
   - 前次從佇列中移除訊息的成功 MQGET 呼叫 (這可以是工作單元的一部分)。
   - 前次瀏覽佇列上訊息的成功 MQGET 呼叫 (這個 不能 是工作單元的一部分)。
5. 與 MQGET 呼叫相關聯的資訊會還原為它在現行工作單元中該佇列控點的第一次成功 MQGET 呼叫之前所擁有的值。

   在工作單元啟動之後，但在工作單元範圍之外，由應用程式更新的佇列，如果工作單元取消，則不會還原其群組及區段資訊。

當工作單元取消時，將群組及區段資訊還原至其先前的值，可讓應用程式將大型訊息群組或大型邏輯訊息 (由許多區段組成) 分散在數個工作單元中，並在訊息群組或邏輯訊息中的正確點重新啟動 (如果其中一個工作單元失敗)。

如果本端佇列管理程式只有有限的佇列儲存體，則使用數個工作單元可能有利。 不過，如果發生系統故障，應用程式必須維護足夠的資訊，才能在正確的點重新啟動放置或取得訊息。

如需如何在系統失敗之後正確點重新啟動的詳細資料，請參閱 第 432 頁的『MQPMO-放置訊息選項』中說明的 MQPMO_LOGICAL_ORDER 選項，以及 第 323 頁的『MQGMO-取得訊息選項』中說明的 MQGMO_LOGICAL_ORDER 選項。

只有在佇列管理程式協調工作單元時，其餘使用注意事項才適用。

6. 工作單元與連線控點具有相同的範圍。 所有影響特定工作單元的 MQ 呼叫都必須使用相同的連線控點來執行。 使用不同連線控點發出的呼叫 (例如，由另一個應用程式發出的呼叫) 會影響不同的工作單元。 如需連線控點範圍的相關資訊，請參閱 第 578 頁的『MQCONN-連接佇列管理程式』中說明的 *Hconn* 參數。
7. 只有作為現行工作單元一部分放置或擷取的訊息才會受到此呼叫的影響。
8. 在工作單元內發出 MQGET、MQPUT 或 MQPUT1 呼叫，但永不發出確定或取消呼叫的長時間執行應用程式，可以在佇列中填入其他應用程式無法使用的訊息。 為了防止這種可能性，管理者必須將 *MaxUncommittedMsgs* queue-manager 屬性設為足夠低，以防止失控的應用程式填滿佇列，但足夠高，以容許預期的傳訊應用程式正確運作。

C 呼叫

```
MQBACK (Hconn, &CompCode, &Reason);
```

宣告參數如下:

```
MQHCONN  Hconn;     /* Connection handle */
MQLONG   CompCode;  /* Completion code */
MQLONG   Reason;    /* Reason code qualifying CompCode */
```

COBOL 呼叫

```
CALL 'MQBACK' USING HCONN, COMPCODE, REASON.
```

宣告參數如下:

```
**   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 呼叫

```
call MQBACK (Hconn, CompCode, Reason);
```

宣告參數如下:

```
dcl Hconn     fixed bin(31);  /* Connection handle */
dcl CompCode  fixed bin(31);  /* Completion code */
dcl Reason    fixed bin(31);  /* Reason code qualifying CompCode */
```

High Level Assembler 呼叫

```
CALL MQBACK,(HCONN,COMPCODE,REASON)
```

宣告參數如下:

```
HCONN     DS  F  Connection handle
COMPCODE  DS  F  Completion code
REASON    DS  F  Reason code qualifying COMPCODE
```

Visual Basic 呼叫

```
MQBACK Hconn, CompCode, Reason
```

宣告參數如下:

```
Dim Hconn    As Long 'Connection handle'
Dim CompCode As Long 'Completion code'
Dim Reason   As Long 'Reason code qualifying CompCode'
```

MQBEGIN-開始工作單元

MQBEGIN 呼叫會開始由佇列管理程式協調且可能涉及外部資源管理程式的工作單元。

語法

MQBEGIN (*Hconn*、 *BeginOptions*、 *Compcode*、 *Reason*)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 前一個 MQCONN 或 MQCONNX 呼叫已傳回 *Hconn* 的值。

Hconn 必須是非共用連線控點。 如果指定共用連線控點，則呼叫會失敗，原因碼為 MQRC_HCONN_ERROR。 如需共用及非共用控點的相關資訊，請參閱 第 282 頁的『MQCNO-連接選項』 中 MQCNO_HANDLE_SHARE_ * 選項的說明。

BeginOptions
類型 :MQBO-輸入/輸出

這些選項可控制 MQBEGIN 的動作，如 第 248 頁的『MQBO-開始選項』中所述。

如果不需要任何選項，則以 C 或 S/390 組合器撰寫的程式可以指定空值參數位址，而不是指定 MQBO 結構的位址。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_WARNING
警告 (局部完成)。

MQCC_FAILED
呼叫失敗。

原因

類型 :MQLONG-輸出

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_WARNING:

MQ RC_NO_EXTERNAL_PARATENTS
(2121 , X'849 ') 未登錄參與的資源管理程式。

MQRC_PARTICIPANT_NOT_AVAILABLE
(2122、X'84A') 無法使用參與資源管理程式。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_API_EXIT_ERROR
(2374 , X' 946 ') API 結束程式失敗。

MQRC_BO_ERROR
(2134 , X'856 ') 開始選項結構無效。

MQRC_CALL_IN_PROGRESS
(2219 , X'8AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CONNECTION_BROKEN
(2009 , X'7D9') 與佇列管理程式的連線遺失。

MQRC_ENVIRONMENT_ERROR
(2012 , X'7DC') 在環境中呼叫無效。

MQRC_HCONN_ERROR
(2018 , X'7E2') 連線控點無效。

MQRC_OPTIONS_ERROR
(2046 , X'7FE') 選項無效或不一致。

MQRC_Q_MGR_STOPPING
(2162 , X'872 ') 佇列管理程式關閉。

MQRC_RESOURCE_PROBLEM
(2102 , X'836 ') 可用的系統資源不足。

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'817 ') 可用的儲存體不足。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

MQ RC_UOW_IN_PROGRESS
(2128、X'850 ') 工作單元已啟動。

如需這些原因碼的相關資訊，請參閱 原因碼。

使用注意事項

1. 使用 MQBEGIN 呼叫來啟動工作單元，該工作單元由佇列管理程式協調，且可能涉及其他資源管理程式所擁有資源的變更。 佇列管理程式支援三種工作單元類型:
   - **佇列管理程式協調本端工作單元**: 一種工作單元，其中佇列管理程式是唯一參與的資源管理程式，因此佇列管理程式會作為工作單元協調程式。
     - 若要啟動這種工作單元，請在工作單元中的第一個 MQPUT、 MQPUT1 或 MQGET 呼叫上指定 MQPMO_SYNCPOINT 或 MQGMO_SYNCPOINT 選項。
     - 若要確定或取消此類型的工作單元，請使用 MQCMIT 或 MQBACK 呼叫。

- **佇列管理程式協調的廣域工作單元**: 針對 MQ 資源 和 屬於其他資源管理程式的資源，佇列管理程式在其中充當工作單元協調程式的工作單元。 這些資源管理程式會與佇列管理程式合作，以確保一起確定或取消對工作單元中資源的所有變更。
  - 若要啟動此類型的工作單元，請使用 MQBEGIN 呼叫。
  - 若要確定或取消此類型的工作單元，請使用 MQCMIT 和 MQBACK 呼叫。
- **外部協調的廣域工作單元**: 一種工作單元，其中佇列管理程式是參與者，但佇列管理程式不會作為工作單元協調程式。 相反地，佇列管理程式會與外部工作單元協調程式合作。
  - 若要啟動這種類型的工作單元，請使用外部工作單元協調程式所提供的相關呼叫。

    如果使用 MQBEGIN 呼叫來嘗試啟動工作單元，則呼叫會失敗，原因碼為 MQRC_ENVIRONMENT_ERROR。
  - 若要確定或取消此類型的工作單元，請使用外部工作單元協調程式所提供的確定及取消呼叫。

    如果您使用 MQCMIT 或 MQBACK 呼叫來確定或取消工作單元，則呼叫會失敗，原因碼為 MQRC_ENVIRONMENT_ERROR。

2. 如果應用程式在工作單元中以未確定的變更結束，則這些變更的處置取決於應用程式是正常結束還是異常結束。 如需進一步詳細資料，請參閱 第 599 頁的『MQDISC-切斷佇列管理程式連線』 中的使用注意事項。
3. 應用程式一次只能參與一個工作單元。 如果應用程式已有工作單元存在，則不論工作單元類型為何，MQBEGIN 呼叫都會失敗，原因碼為 MQRC_UOW_IN_PROGRESS。
4. MQBEGIN 呼叫在 MQ MQI 用戶端環境中無效。 嘗試使用呼叫失敗，原因碼為 MQRC_ENVIRONMENT_ERROR。
5. 當佇列管理程式作為廣域工作單元的工作單元協調程式時，可以參與工作單元的資源管理程式會定義在佇列管理程式配置檔中。
6. 在 IBM i 上，支援三種工作單元類型，如下所示:
   - 只有在工作層次上不存在確定定義時，才可以使用 **佇列管理程式協調本端工作單元** ，亦即，不得針對工作發出具有 CMTSCOPE(*JOB) 參數的 STRCMTCTL 指令。
   - 不支援 **佇列管理程式協調廣域工作單元** 。
   - 只有在工作層次存在確定定義時，才能使用 **外部協調的廣域工作單元** ，亦即，必須已針對工作發出具有 CMTSCOPE(*JOB) 參數的 STRCMTCTL 指令。 如果已這樣做， IBM i COMMIT 和 ROLLBACK 作業會套用至 MQ 資源，以及屬於其他參與資源管理程式的資源。

C 呼叫

```
MQBEGIN (Hconn, &BeginOptions, &CompCode, &Reason);
```

宣告參數如下:

```
MQHCONN  Hconn;         /* Connection handle */
MQBO     BeginOptions;  /* Options that control the action of MQBEGIN */
MQLONG   CompCode;      /* Completion code */
MQLONG   Reason;        /* Reason code qualifying CompCode */
```

COBOL 呼叫

```
     CALL 'MQBEGIN' USING HCONN, BEGINOPTIONS, COMPCODE, REASON.
```

宣告參數如下:

```
**   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 呼叫

```
call MQBEGIN (Hconn, BeginOptions, CompCode, Reason);
```

宣告參數如下:

```
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 呼叫

```
MQBEGIN Hconn, BeginOptions, CompCode, Reason
```

宣告參數如下:

```
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-將緩衝區轉換成訊息控點

MQBUFMH 函數呼叫會將緩衝區轉換為訊息控點，並與 MQMHBUF 呼叫反向。

此呼叫會取得緩衝區中的訊息描述子及 MQRFH2 內容，並讓它們可透過訊息控點使用。 訊息資料中的 MQRFH2 內容會選擇性地移除。 必要的話，會更新訊息描述子的 *Encoding*、 *CodedCharSetId* 及 *Format* 欄位，以在移除內容之後正確地說明緩衝區的內容。

語法

MQBUFMH (*Hconn*、 *Hmsg*、 *BufMsgHOpts* 、 *MsgDesc*、 緩衝區、 *BufferLength*、 *DataLength*、 *Compcode* 、 *Reason*)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 *Hconn* 的值必須符合用來建立 *Hmsg* 參數中所指定訊息控點的連線控點。

如果訊息控點是使用 MQHC_UNASSOCIED_HCONN 建立的，則必須在將緩衝區轉換為訊息控點的執行緒上建立有效連線。 如果未建立有效連線，則呼叫會失敗並產生 MQRC_CONNECTION_BROKEN。

Hmsg
類型 :MQHMQSG-輸入

這是需要緩衝區的訊息控點。 此值是由前一個 MQCRTMH 呼叫所傳回。

BufMsgHOpts
類型 :MQBMHO-輸入

MQBMHO 結構可讓應用程式指定選項，以控制如何從緩衝區產生訊息處理。

請參閱第 246 頁的『MQBMHO-緩衝區至訊息控點選項』，以取得詳細資料。

MsgDesc
類型 :MQMD-輸入/輸出

MsgDesc 結構包含訊息描述子內容，並說明緩衝區的內容。

在呼叫的輸出上，會選擇性地從緩衝區中移除內容，在此情況下，會更新訊息描述子以正確說明緩衝區。

此結構中的資料必須在應用程式的字集及編碼中。

BufferLength
類型 :MQLONG-輸入

BufferLength 是「緩衝區」區域的長度 (以位元組為單位)。

零位元組的 *BufferLength* 是有效的，指出緩衝區不包含任何資料。

緩衝區
類型: MQBYTExBuffer 長度-輸入/輸出

這些選項會控制 MQBEGIN 的動作，如 第 551 頁的『MQBEGIN-開始工作單元』中所述。

Buffer 定義包含訊息緩衝區的區域。 對於大部分資料，您應該在 4 位元組界限上對齊緩衝區。

如果 *Buffer* 包含字元或數值資料，請將 *MsgDesc* 參數中的 *CodedCharSetId* 和 *Encoding* 欄位設為適合資料的值; 必要的話，這可讓資料進行轉換。

如果在訊息緩衝區中找到內容，則會選擇性地移除它們; 稍後從呼叫返回時，它們會從訊息控點中變成可用。

在 C 程式設計語言中，參數宣告為 void 的指標，這表示任何資料類型的位址都可以指定為參數。

如果 *BufferLength* 參數為零，則不會參照 *Buffer* ; 在此情況下，以 C 或 System/390 組合器撰寫的程式所傳遞的參數位址可以是空值。

DataLength
類型 :MQLONG-輸出

可能已移除內容的緩衝區長度 (以位元組為單位)。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204 , X'089C') 配接卡無法使用。

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130 , X'852 ') 無法載入配接卡服務模組。

MQRC_ASDID_MISMATCH
(2157 , X'86D') 主要和起始 ASID 不同。

MQRC_BMHO_ERROR
(2489 , X'09B9') 緩衝區至訊息處理選項結構無效。

MQRC_BUFFER_ERROR
(2004 , X'07D4') 緩衝區參數無效。

MQRC_BUFFER_LENGTH_ERROR
(2005 , X'07D5') 緩衝區長度參數無效。

MQRC_CALL_IN_PROGRESS
(2219 , X'08AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CONNECTION_BROKEN
(2009 , X'07D9') 與佇列管理程式的連線遺失。

MQRC_HMSG_ERROR
(2460 , X'099C') 訊息控點無效。

MQRC_MD_ERROR
(2026 , X'07EA') 訊息描述子無效。

MQRC_MSG_HANDLE_IN_USE
(2499 , X'09C3') 訊息控點已在使用中。

MQRC_OPTIONS_ERROR
(2046 , X'07FE') 選項無效或不一致。

MQRC_RFH_ERROR
(2334 , X'091E') MQRFH2 結構無效。

MQRC_RFH_FORMAT_ERROR
(2421 , X'0975 ') 無法剖析包含內容的 MQRFH2 資料夾。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

如需這些代碼的詳細資訊，請參閱 原因碼。

使用注意事項

API 結束程式無法截取 MQBUFMH 呼叫-在應用程式空間中，緩衝區會轉換成訊息控點; 呼叫不會到達佇列管理程式。

C 呼叫

```
MQBUFMH (Hconn, Hmsg, &BufMsgHOpts, &MsgDesc, BufferLength, Buffer,
         &DataLength, &CompCode, &Reason);
```

宣告參數如下:

```
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 呼叫

```
CALL 'MQBUFMH' USING HCONN, HMSG, BUFMSGHOPTS, MSGDESC, BUFFERLENGTH,
```

```
                          BUFFER, DATALENGTH, COMPCODE, REASON.
```

宣告參數如下:

```
**   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 呼叫

```
call MQBUFMH (Hconn, Hmsg, BufMsgHOpts, MsgDesc, BufferLength, Buffer,
              DataLength, CompCode, Reason);
```

宣告參數如下:

```
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 */
```

High Level Assembler 呼叫

```
CALL MQBUFMH,(HCONN,HMSG,BUFMSGHOPTS,MSGDESC,BUFFERLENGTH,BUFFER,
                DATALENGTH,COMPCODE,REASON)
```

宣告參數如下:

```
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-管理回呼

MQCB 呼叫會登錄指定物件控點的回呼，並控制回呼的啟動和變更。

回呼是 IBM WebSphere MQ 在發生特定事件時所呼叫的程式碼片段 (指定為可動態鏈結的函數名稱或函數指標)。

若要在 V7 用戶端上使用 MQCB 及 MQCTL，您必須連接至 V7 伺服器，且通道的 **SHARECNV** 參數必須具有非零值。

可定義的回呼類型如下:

訊息消費者
當物件控點有符合指定選取準則的訊息可用時，會呼叫訊息消費者回呼函數。

每一個物件控點只能登錄一個回呼函數。 如果要以多重選取準則來讀取單一佇列，則必須多次開啟佇列，並在每一個控點上登錄一個消費者函數。

事件處理程式
會針對會影響整個回呼環境的條件呼叫事件處理程式。

當發生事件條件 (例如，佇列管理程式或連線停止或靜止) 時，會呼叫此函數。

不會針對單一訊息消費者特定的條件 (例如 MQRC_GET_INHIBITED) 呼叫此函數; 不過，如果回呼函數未正常結束，則會呼叫此函數。

語法

MQCB (*Hconn*、 作業、 *CallbackDesc*、 *Hobj*、 *MsgDesc* 、 *GetMsgOpts*、 *CompCode*、 *Reason*)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 前一個 MQCONN 或 MQCONNX 呼叫已傳回 *Hconn* 的值。

在 z/OS for CICS 應用程式上，以及在 IBM i for 以相容模式執行的應用程式上，您可以為 *MQHC_DEF_HCONN* 指定下列特殊值，以使用與此執行單元相關聯的連線控點。

operation
類型 :MQLONG-輸入

在針對指定物件控點定義的回呼上正在處理的作業。 您必須指定下列其中一個選項; 如果需要多個選項，則值可以是:

- 相加（請不要將同一常數相加多次），或者
- 利用執行位元 OR 運算來組合（如果程式設計語言支援位元運算）。

MQOP_REGISTER
定義指定物件控點的回呼函數。 此作業定義要呼叫的函數及要使用的選取準則。

如果已定義物件控點的回呼函數，則會取代定義。 如果在取代回呼時偵測到錯誤，則會取消登錄函數。

如果回呼登錄在先前已取消登錄的相同回呼函數中，則會將此視為取代作業; 不會呼叫任何起始或最終呼叫。

您可以將 MQOP_REGISTER 與 MQOP_SUSPEND 或 MQOP_RESUME 搭配使用。

MQ 作業 _ 取消登錄
停止耗用物件控點的訊息，並從適合回呼的控點中移除控點。

如果關閉相關聯的控點，則會自動取消登錄回呼。

如果從消費者內呼叫 MQOP_DEREGISTER，且回呼已定義停止呼叫，則會在消費者傳回時呼叫它。

如果針對沒有已登錄消費者的 *Hobj* 發出此作業，則呼叫會傳回 MQRC_CALLBACK_NOT_REGISTERED。

MQOP_SUSPEND
暫停耗用物件控點的訊息。

如果此作業套用至事件處理程式，則事件處理程式在暫停時不會取得事件，且在回復作業時，不會提供任何在處於暫停狀態時遺失的事件給作業。

暫停時，消費者函數會繼續取得控制項類型回呼。

MQ 作業 _ 回復
回復耗用物件控點的訊息。

如果此作業套用至事件處理程式，則事件處理程式在暫停時不會取得事件，且在回復作業時，不會提供任何在處於暫停狀態時遺失的事件給作業。

CallbackDesc
類型 :MQCBD-輸入

這是一種結構，可識別應用程式所登錄的回呼函數，以及登錄它時所使用的選項。

如需結構的詳細資料，請參閱 MQCBD 。

只有 MQOP_REGISTER 選項才需要回呼描述子; 如果不需要描述子，則傳遞的參數位址可以是空值。

HOBJ
類型 :MQHOBJ-輸入

此控點代表已建立對要從中耗用訊息之物件的存取權。 這是從前一個 MQOPEN 或 MQSUB 呼叫 (在 *Hobj* 參數中) 傳回的控點。

定義事件處理常式 (MQCBT_EVENT_HANDLER) 時不需要 *Hobj* ，應該指定為 MQHO_NONE。

如果已從 MQOPEN 呼叫傳回 *Hobj* ，則必須已使用下列一個以上選項開啟佇列:

- MQOO_INPUT_SHARED
- MQOO_INPUT_EXCLUSIVE
- MQOO_INPUT_AS_Q_DEF
- MQ 瀏覽

MsgDesc
類型 :MQMD-input

此結構說明所需訊息的屬性，以及所擷取訊息的屬性。

MsgDesc 參數定義消費者所需的訊息屬性，以及要傳遞至訊息消費者的 MQMD 版本。

MQMD 中的 *MsgId*、 *CorrelId*、 *GroupId* 、 *MsgSeqNumber* 及 *Offset* 用於選取訊息，視 *GetMsgOpts* 參數中指定的選項而定。

如果您指定 MQGMO_CONVERT 選項，則會使用 *Encoding* 和 *CodedCharSetId* 來進行訊息轉換。

如需詳細資料，請參閱 MQMD 。

MsgDesc 用於 MQOP_REGISTER ，並且如果您需要任何欄位的預設值以外的值。 *MsgDesc* 不用於事件處理程式。

如果不需要描述子，則傳遞的參數位址可以是空值。

請注意，如果針對具有重疊選取元的相同佇列登錄多個消費者，則未定義每一個訊息所選擇的消費者。

***GetMsg* 選項**
類型 :MQGMO-輸入

GetMsgOpts 參數控制訊息消費者取得訊息的方式。 在 MQGET 呼叫中使用此參數的所有選項都具有第 323 頁的『MQGMO-取得訊息選項』中所說明的意義，但下列情況除外:

MQGMO_SET_SIGNAL
不允許此選項。

MQGMO_BROWSE_FIRST、MQGMO_BROWSE_NEXT、MQGMO_MARK_ * 遞送至瀏覽消費者的訊息順序是由這些選項的組合所指定。 顯著組合如下:

MQGMO_BROWSE_FIRST
佇列上的第一個訊息會反覆地遞送給消費者。 當消費者破壞性使用回呼中的訊息時，這很有用。 請小心使用此選項。

MQGMO_BROWSE_NEXT
消費者會從現行游標位置取得佇列上的每一則訊息，直到到達佇列結尾為止。

MQGMO_BROWSE_FIRST + MQGMO_BROWSE_NEXT
游標會重設為佇列的開頭。 然後會提供每一則訊息給消費者，直到游標到達佇列結尾為止。

MQGMO_BROWSE_FIRST + MQGMO_MARK_ *
從佇列開頭開始，會為消費者提供佇列上第一個未標示的訊息，然後會針對此消費者標示此訊息。 此組合可確保消費者可以接收新增在現行游標點後面的新訊息。

MQGMO_BROWSE_NEXT + MQGMO_MARK_ *
從游標位置開始，會為消費者提供佇列上下一個未標示的訊息，然後會針對此消費者標示此訊息。 請小心使用此組合，因為訊息可以新增至現行游標位置後面的佇列。

MQGMO_BROWSE_FIRST + MQGMO_BROWSE_NEXT + MQGMO_MARK_ *
不允許此組合。 如果使用的話，呼叫會傳回 MQRC_OPTIONS_ERROR。

MQGMO_NO_WAIT、MQGMO_WAIT 及 WaitInterval
這些選項控制如何呼叫消費者。

MQGMO_NO_WAIT
絕不會使用 MQRC_NO_MSG_AVAILABLE 來呼叫消費者。 只會針對訊息和事件來呼叫消費者。

MQGMO_WAIT 具有零 WaitInterval
當沒有可用的訊息且消費者已啟動或自最後一個「無訊息」原因碼以來至少已遞送一則訊息時，MQRC_NO_MSG_AVAILABLE 代碼會傳遞給消費者。

當指定零等待間隔時，這可防止消費者在忙碌迴圈中輪詢。

MQGMO_WAIT 及正數 WaitInterval
在指定的等待間隔之後呼叫消費者，原因碼為 MQRC_NO_MSG_AVAILABLE。 不論是否有任何訊息遞送至消費者，都會進行此呼叫。 這可讓使用者執行活動訊號或批次類型處理。

MQWI_UNLIMITED 的 MQGMO_WAIT 及 WaitInterval
這指定在傳回 MQRC_NO_MSG_AVAILABLE 之前的無限等待。 絕不會使用 MQRC_NO_MSG_AVAILABLE 來呼叫消費者。

GetMsgOpts 僅用於 MQOP_REGISTER，且如果您需要任何欄位的預設值以外的值。 *GetMsgOpts* 不用於事件處理程式。

如果不需要 *GetMsgOpts*，則傳遞的參數位址可以是空值。 使用此參數與將 MQGMO_DEFAULT 與 MQGMO_FAIL_IF_QUIESCING 一起指定相同。

如果在 MQGMO 結構中提供訊息內容控點，則會在傳遞至消費者回呼的 MQGMO 結構中提供副本。 從 MQCB 呼叫返回時，應用程式可以刪除訊息內容控點。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_WARNING
警告 (局部完成)。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

下列清單中的原因碼是佇列管理程式可以針對 *Reason* 參數傳回的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204 , X'89C') 配接卡無法使用。

MQRC_ADAPTER_CONV_LOAD_ERROR
(2133 , X'855 ') 無法載入資料轉換服務模組。

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130 , X'852 ') 無法載入配接卡服務模組。

MQRC_API_EXIT_ERROR
(2374 , X' 946 ') API 結束程式失敗。

MQRC_API_EXIT_LOAD_ERROR
(2183 , X'887 ') 無法載入 API 結束程式。

MQRC_ASDID_MISMATCH
(2157 , X'86D') 主要和起始 ASID 不同。

MQRC_BUFFER_LENGTH_ERROR
(2005 , X'7D5') 緩衝區長度參數無效。

MQRC_CALL_IN_PROGRESS
(2219 , X'8AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CALLBACK_LINK_ERROR
(2487 , X'9B7') 回呼類型欄位不正確。

MQRC_CALLBACK_NOT_REGISTERED
(2448 , X' 990 ') 無法取消登錄、暫停或回復，因為沒有已登錄的回呼。

MQRC_CALLBACK_ROUTINE_ERROR
(2486、X'9B6') 必須指定 *CallbackFunction* 或 *CallbackName* ，但不能同時指定兩者。

MQRC_CALLBACK_TYPE_ERROR
(2483 , X'9B3') 回呼類型欄位不正確。

MQRC_CBD_OPTIONS_ERROR
(2484 , X'9B4') MQCBD 選項欄位不正確。

MQRC_CICS_WAIT_FAILED
(2140 , X'85C') 等待要求被 CICS 拒絕。

MQRC_CONNECTION_BROKEN
(2009 , X'7D9') 與佇列管理程式的連線遺失。

MQRC_CONNECTION_NOT_AUTHORIZED
(2217 , X'8A9') 未獲連線授權。

MQRC_CONNECTION_QUIESCING
(2202 , X'89A') 連線靜止。

MQRC_CONNECTION_STOPPING
(2203 , X'89B') 連線關閉。

MQRC_CORREL_ID_ERROR
(2207 , X'89F') 相關性 ID 錯誤。

MQRC_DATA_LENGTH_ERROR
(2010 , X'7DA') 資料長度參數無效。

MQRC_FUNCTION_NOT_SUPPORTED
(2298 , X'8FA') 在現行環境中無法使用所要求的功能。

MQRC_GET_INHIBITED
(2016 , X'7E0') 禁止佇列取得。

MQRC_GLOBAL_UOW_CONFLICT
(2351 , X'92F') 廣域工作單元衝突。

MQRC_GMO_ERROR
(2186 , X'88A') 取得訊息選項結構無效。

MQRC_HANDLE_IN_USE_FOR_UOW
(2353 , X' 931 ') 用於廣域工作單元的控點。

MQRC_HCONN_ERROR
(2018 , X'7E2') 連線控點無效。

MQRC_HOBJ_ERROR
(2019 , X'7E3') 物件控點無效。

MQRC_INCONSISTENT_BROWSE
(2259 , X'8D3') 不一致的瀏覽規格。

MQRC_INCONSISTENT_UOW
(2245 , X'8C5') 工作單元規格不一致。

MQRC_INVALID_MSG_UNDER_CURSOR
(2246 , X'8C6') 游標下的訊息不適用於擷取。

MQRC_LOCAL_UOW_CONFLICT
(2352 , X' 930 ') 廣域工作單元與本端工作單元衝突。

MQRC_MATCH_OPTIONS_ERROR
(2247 , X'8C7') 比對選項無效。

MQRC_MAX_MSG_LENGTH_ERROR
(2485 , X'9B4') 不正確 *MaxMsgLength* 欄位。

MQRC_MD_ERROR
(2026 , X'7EA') 訊息描述子無效。

MQRC_MODULE_ENTRY_NOT_FOUND
(2497 , X'9C1') 在模組中找不到指定的函數進入點。

MQRC_MODULE_INVALID
(2496 , X'9C0') 找到模組，但其類型錯誤; 不是 32 位元、64 位元或有效的動態鏈結程式庫。

找不到 MQRC_MODULE_NOT_FOUND
(2495 , X'9BF') 在搜尋路徑中找不到模組或未獲授權載入。

MQRC_MSG_SEQ_NUMBER_ERROR
(2250 , X'8CA') 訊息序號無效。

MQRC_MSG_TOKEN_ERROR
(2331 , X'91B') 使用訊息記號無效。

MQRC_NO_MSG_AVAILABLE
(2033 , X'7F1') 沒有可用的訊息。

MQRC_NO_MSG_UNDER_CURSOR
(2034 , X'7F2') 瀏覽游標未定位在訊息上。

MQRC_NOT_OPEN_FOR_BROWSE
(2036 , X'7F4') 佇列未開啟以供瀏覽。

MQRC_NOT_OPEN_FOR_INPUT
(2037 , X'7F5') 佇列未開啟以供輸入。

已變更 MQRC_OBJECT_CHANGED
(2041 , X'7F9') 物件定義自開啟以來已變更。

MQRC_OBJECT_DAMAGED
(2101 , X'835 ') 物件已損壞。

MQRC_OPERATION_ERROR
(2206 , X'89E') API 呼叫上的作業碼不正確。

MQRC_OPTIONS_ERROR
(2046 , X'7FE') 選項無效或不一致。

MQRC_PAGESET_ERROR
(2193 , X'891 ') 存取頁集資料集時發生錯誤。

MQRC_Q_DELETED
(2052 , X'804 ') 已刪除佇列。

MQRC_Q_INDEX_TYPE_ERROR
(2394 , X'95A') 佇列具有錯誤索引類型。

MQRC_Q_MGR_NAME_ERROR
(2058 , X'80A') 佇列管理程式名稱無效或不明。

MQRC_Q_MGR_NOT_AVAILABLE
(2059 , X'80B') 佇列管理程式無法用於連線。

MQRC_Q_MGR_QUIESCING
(2161 , X'871 ') 佇列管理程式靜止中。

MQRC_Q_MGR_STOPPING
(2162 , X'872 ') 佇列管理程式關閉。

MQRC_RESOURCE_PROBLEM
(2102 , X'836 ') 可用的系統資源不足。

MQRC_SIGNAL_OUTSTANDING
(2069 , X'815 ') 此握把未發出信號。

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'817 ') 可用的儲存體不足。

MQRC_SUPPRESSED_BY_EXIT
(2109 , X'83D') 跳出程式暫停呼叫。

MQRC_SYNCPOINT_LIMIT_REACHED
(2024 , X'7E8') 在現行工作單元內無法處理更多訊息。

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072 , X'818 ') 同步點支援無法使用。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

MQRC_UOW_ENLISTMENT_ERROR
(2354 , X' 932 ') 在廣域工作單元中列入失敗。

不支援 **MQRC_UOW_MIX_NOT_SUPPORTED**
(2355 , X' 933 ') 不支援混合工作單元呼叫。

MQRC_UOW_NOT_AVAILABLE
(2255 , X'8CF') 佇列管理程式無法使用的工作單元。

MQRC_WAIT_INTERVAL_ERROR
(2090 , X'82A') MQGMO 中的等待間隔無效。

MQRC_WRONG_GMO_VERSION
(2256 , X'8D0') 提供的 MQGMO 版本錯誤。

MQRC_WRONG_MD_VERSION
(2257 , X'8D1') 提供的 MQMD 版本錯誤。

如需這些代碼的詳細資訊，請參閱 原因碼。

使用注意事項

1. MQCB 用來定義要針對每一個訊息呼叫的動作，符合佇列上可用的指定準則。 當處理動作時，會從佇列中移除訊息並傳遞至已定義的訊息消費者，或提供用來擷取訊息的訊息記號。
2. 在使用 MQCTL 開始使用之前，可以使用 MQCB 來定義回呼常式，也可以從回呼常式內使用 MQCB。
3. 若要從回呼常式外部使用 MQCB ，您必須先使用 MQCTL 暫停訊息耗用，然後再回復耗用。
4. IMS 配接器不支援 MQCB。

訊息消費者回呼順序

您可以配置消費者在消費者生命週期的關鍵點呼叫回呼。 例如:

- 當消費者第一次註冊時,
- 當連線啟動時,
- 當連線停止且
- 當 MQCLOSE 明確或隱含地取消登錄消費者時。

表 *562: MQCTL* 動詞定義	
動詞	**意義**
MQCTL (START)	使用 MQOP_START 作業的 MQCTL 呼叫
MQCTL (STOP)	使用 MQOP_STOP 作業的 MQCTL 呼叫
MQCTL (WAIT)	使用 MQOP_START_WAIT 作業的 MQCTL 呼叫

這可讓消費者維護與消費者相關聯的狀態。 當應用程式要求回呼時，消費者呼叫的規則如下:

登錄
一律是回呼呼叫的第一種類型。

一律在與 MQCB (REGISTER) 呼叫相同的執行緒上呼叫。

開始
一律與 MQCTL (START) 動詞同步呼叫。

- 在 MQCTL (START) 動詞傳回之前，已完成所有 START 回呼。

與訊息遞送位於相同執行緒上 (如果要求 THREAD_AFFINITY)。

例如，如果前一個回呼在 MQCTL (START) 期間發出 MQCTL (STOP) ，則不保證具有 start 的呼叫。

停止
在此呼叫之後，除非重新啟動連線，否則不會遞送進一步的訊息或事件。

如果先前已針對 START、訊息或事件呼叫應用程式，則可保證 STOP。

取消登錄
一律是回呼的最後一種呼叫類型。

請確定您的應用程式在 START 和 STOP 回呼中執行執行緒型起始設定和清除。 您可以使用 REGISTER 和 DEREGISTER 回呼來執行非執行緒型起始設定和清除。

除了說明之外，請勿對執行緒的生命期限和可用性做出任何假設。 例如，在最後一次呼叫 DEREGISTER 之後，不要依賴保持作用中的執行緒。 同樣地，當您選擇不使用 THREAD_AFFINITY 時，請勿在每次啟動連線時都假設該執行緒存在。

如果您的應用程式具有執行緒性質的特定需求，則一律可以相應地建立執行緒，然後使用 MQCTL (WAIT)。 這具有「捐贈」執行緒給 IBM WebSphere MQ 以進行非同步訊息遞送的效果。

訊息消費者連線使用情形

您可以配置消費者在消費者生命週期的關鍵點呼叫回呼。 例如:

- 當消費者第一次註冊時,
- 當連線啟動時,
- 當連線停止且
- 當 MQCLOSE 明確或隱含地取消登錄消費者時。

表 563: MQCTL 動詞定義	
動詞	**意義**
MQCTL (START)	使用 MQOP_START 作業的 MQCTL 呼叫
MQCTL (STOP)	使用 MQOP_STOP 作業的 MQCTL 呼叫
MQCTL (WAIT)	使用 MQOP_START_WAIT 作業的 MQCTL 呼叫

這可讓消費者維護與消費者相關聯的狀態。 當應用程式要求回呼時，消費者呼叫的規則如下:

登錄

一律是回呼呼叫的第一種類型。

一律在與 MQCB (REGISTER) 呼叫相同的執行緒上呼叫。

開始

一律與 MQCTL (START) 動詞同步呼叫。

- 在 MQCTL (START) 動詞傳回之前，已完成所有 START 回呼。

與訊息遞送位於相同執行緒上 (如果要求 THREAD_AFFINITY)。

例如，如果前一個回呼在 MQCTL (START) 期間發出 MQCTL (STOP) ，則不保證具有 start 的呼叫。

停止

在此呼叫之後，除非重新啟動連線，否則不會遞送進一步的訊息或事件。

如果先前已針對 START、訊息或事件呼叫應用程式，則可保證 STOP。

取消登錄

一律是回呼的最後一種呼叫類型。

請確定您的應用程式在 START 和 STOP 回呼中執行執行緒型起始設定和清除。 您可以使用 REGISTER 和 DEREGISTER 回呼來執行非執行緒型起始設定和清除。

除了說明之外，請勿對執行緒的生命期限和可用性做出任何假設。 例如，在最後一次呼叫 DEREGISTER 之後，不要依賴保持作用中的執行緒。 同樣地，當您選擇不使用 THREAD_AFFINITY 時，請勿在每次啟動連線時都假設該執行緒存在。

如果您的應用程式具有執行緒性質的特定需求，則一律可以相應地建立執行緒，然後使用 MQCTL (WAIT)。 這具有「捐贈」執行緒給 IBM WebSphere MQ 以進行非同步訊息遞送的效果。

C 呼叫

```
MQCB (Hconn, Operation, CallbackDesc, Hobj, MsgDesc,
GetMsgOpts, &CompCode, &Reason);
```

宣告參數如下:

```
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 呼叫

```
CALL 'MQCB' USING HCONN, OPERATION, CBDESC, HOBJ, MSGDESC,
            GETMSGOPTS, COMPCODE, REASON.
```

宣告參數如下:

```
**   Connection handle
 01  HCONN    PIC S9(9) BINARY.
**   Operation
 01  OPERATION PIC S9(9) BINARY.
**   Callback Descriptior
 01  CBDESC.
     COPY CMQCBDV.
01  HOBJ     PIC S9(9) BINARY.
**   Message Descriptior
 01  MSGDESC.
     COPY CMQMDV.
**   Get Message Options
 01  GETMSGOPTS.
     COPY CMQGMOV.
**   Completion code
 01  COMPCODE PIC S9(9) BINARY.
**   Reason code qualifying COMPCODE
 01  REASON   PIC S9(9) BINARY.
```

PL/I 呼叫

```
call MQCB(Hconn, Operation, CallbackDesc, Hobj, MsgDesc, GetMsgOpts,
          CompCode, Reason)
```

宣告參數如下:

```
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_XX_ENCODE_CASE_ONE function-回呼函數

MQCB_XX_ENCODE_CASE_ONE function 函數呼叫是用於事件處理及非同步訊息耗用的回呼函數。

提供 MQCB_XX_ENCODE_CASE_ONE function 呼叫定義只是為了說明傳遞給回呼函數的參數。 佇列管理程式未提供稱為 MQCB_XX_ENCODE_CASE_ONE function 的進入點。

要呼叫的實際函數規格是 MQCB 呼叫的輸入，並透過 MQCBD 結構傳入。

語法

MQCB_FUNCTION (*Hconn*、 *MsgDesc*、 *GetMsgOpts*、 緩衝區、 環境定義)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 前一個 MQCONN 或 MQCONNX 呼叫已傳回 *Hconn* 的值。 在 z/OS (適用於 CICS 應用程式) 及 IBM i (適用於以相容模式執行的應用程式) 上，可以省略 MQCONN 呼叫，並為 Hconn 指定下列值:

MQHC_DEF_CONN

預設連線控點。

MsgDesc
類型 :MQMD-input

此結構說明所擷取訊息的屬性。

請參閱第 363 頁的『MQMD-訊息描述子』，以取得詳細資料。

所傳遞的 MQMD 版本與定義消費者函數的 MQCB 呼叫所傳遞的版本相同。

如果使用第 4 版 MQGMO 來要求傳回「訊息控點」而非 MQMD，則 MQMD 的位址會以空值字元傳遞。

這是訊息消費者函數的輸入欄位; 它與事件處理程式函數無關。

***GetMsg* 選項**
類型 :MQGMO-輸入

用來控制訊息消費者動作的選項。 此參數也包含所傳回訊息的相關資訊。

如需詳細資料，請參閱 MQGMO 。

所傳遞的 MQGMO 版本是受支援的最新版本。

這是訊息消費者函數的輸入欄位; 它與事件處理程式函數無關。

緩衝區
類型: MQBYTExBuffer 長度-輸入

這是包含訊息資料的區域。

如果此呼叫沒有可用的訊息，或訊息未包含任何訊息資料，則會以空值傳遞 *Buffer* 的位址。

這是訊息消費者函數的輸入欄位; 它與事件處理程式函數無關。

環境定義
類型 :MQCBC-輸入/輸出

此結構提供環境定義資訊給回呼函數。 請參閱第 250 頁的『MQCBC-回呼環境定義』，以取得詳細資料。

使用注意事項

1. 請注意，如果回呼常式使用可能延遲或封鎖執行緒的服務 (例如 MQGET 與等待)，則可能會延遲其他回呼的分派。
2. 不會針對每次呼叫回呼常式自動建立個別工作單元，因此常式可以發出確定呼叫，或延遲確定，直到已處理邏輯工作批次為止。 當確定工作批次時，它會確定自前次同步點以來所呼叫的所有回呼函數的訊息。
3. CICS LINK 或 CICS START 呼叫的程式會透過稱為通道儲存器的具名物件，使用 CICS 服務來擷取參數。 儲存器名稱與參數名稱相同。 如需相關資訊，請參閱 CICS 說明文件。
4. 回呼常式可以發出 MQDISC 呼叫，但不能用於自己的連線。 例如，如果回呼常式已建立連線，則它也可以中斷連線。
5. 一般而言，回呼常式不應依賴每次都從相同的執行緒呼叫。 必要的話，請在啟動連線時使用 MQCTLO_THREAD_AFFINITY。
6. 當回呼常式收到非零原因碼時，它必須採取適當的動作。
7. IMS 配接器內不支援 MQCB_XX_ENCODE_CASE_ONE function。

MQCLOSE-關閉物件

MQCLOSE 呼叫放棄對物件的存取權，與 MQOPEN 和 MQSUB 呼叫相反。

語法

MQCLOSE (*Hconn*、 *Hobj*、 *Options*、 *CompCode*、 *Reason*)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 前一個 MQCONN 或 MQCONNX 呼叫已傳回 *Hconn* 的值。

在 z/OS for CICS 應用程式上，以及在 IBM i for 以相容模式執行的應用程式上，您可以省略 MQCONN 呼叫，並為 *Hconn* 指定下列值:

MQHC_DEF_HCONN
預設連線控點。

HOBJ
類型 :MQHOBJ-輸入/輸出

此控點代表正在關閉的物件。 物件可以是任何類型。 前一個 MQOPEN 呼叫傳回 *Hobj* 的值。

順利完成呼叫時，佇列管理程式會將此參數設為對環境無效的控點值。 此值為:

MQHO_UNUSABLE_HOBJ
無法使用物件控點。

在 z/OS 上， *Hobj* 設為未定義的值。

選項
類型 :MQLONG-輸入

此參數控制如何關閉物件。

只能以多種方式關閉永久動態佇列和訂閱，因為它們必須保留或刪除; 這些佇列具有值為 MQQDT_PERMANENT_DYNAMIC 的 *DefinitionType* 屬性 (請參閱 第 731 頁的『佇列的屬性』中說明的 *DefinitionType* 屬性)。 關閉選項在本主題中彙總。

可延續訂閱可以保留或移除; 這些可延續訂閱是使用 MQSUB 呼叫搭配 MQSO_DURABLE 選項來建立。

關閉受管理目的地的控點 (即在使用 MQSO_MANAGED 選項的 MQSUB 呼叫上傳回的 *Hobj* 參數) 時，佇列管理程式會清除任何在同時移除相關聯訂閱時尚未擷取的發佈。 在 MQSUB 呼叫所傳回的 *Hsub* 參數上使用 MQCO_REMOVE_SUB 選項來移除訂閱。 請注意 MQCO_REMOVE_SUB 是 MQCLOSE 上不可延續訂閱的預設行為。

關閉非受管理目的地的控點時，您負責清除傳送發佈的佇列。 請先使用 MQCO_REMOVE_SUB 來關閉訂閱，然後處理佇列外的訊息，直到沒有訊息為止。

您只能從下列選項中指定一個選項:

動態佇列選項: 這些選項控制如何關閉永久動態佇列。

MQCO_DELETE
如果下列一項為真，則會刪除佇列:

- 它是永久動態佇列，由先前的 MQOPEN 呼叫所建立，且佇列上沒有訊息，且佇列沒有未完成的未確定取得或放置要求 (針對現行作業或任何其他作業)。
- 它是由傳回 *Hobj* 的 MQOPEN 呼叫所建立的暫時動態佇列。 在此情況下，會清除佇列上的所有訊息。

在所有其他情況下，包括在 MQSUB 呼叫中傳回 *Hobj* 的情況下，呼叫會失敗，原因碼為 MQRC_OPTION_NOT_VALID_FOR_TYPE ，且不會刪除物件。

在 z/OS 上，如果佇列是已邏輯刪除的動態佇列，且這是它的最後一個控點，則會實際刪除佇列。 有關更多詳細資料，請參閱第 572 頁的『使用注意事項』。

MQCO_DELETE_PURGE
如果下列一項為真，則會刪除佇列，並清除其中的任何訊息:

- 它是由前一個 MQOPEN 呼叫所建立的永久動態佇列，且佇列沒有未完成的未確定取得或放置要求 (針對現行作業或任何其他作業)。
- 它是由傳回 *Hobj* 的 MQOPEN 呼叫所建立的暫時動態佇列。

在所有其他情況下，包括在 MQSUB 呼叫中傳回 *Hobj* 的情況下，呼叫會失敗，原因碼為 MQRC_OPTION_NOT_VALID_FOR_TYPE ，且不會刪除物件。

此表格顯示哪些關閉選項有效，以及物件是保留還是刪除。			
物件或佇列的類型	**MQCO_NONE**	**MQCO_DELETE**	**MQCO_DELETE_PURGE**
佇列以外的物件	已保留	無效	無效
預先定義的佇列	已保留	無效	無效
永久動態佇列 (permanent dynamic queue)	已保留	如果是空的且沒有擱置中的更新項目，則已刪除	已刪除訊息; 如果沒有擱置中的更新項目，則已刪除佇列
暫時動態佇列 (由佇列建立者發出呼叫)	已刪除	已刪除	已刪除
暫時動態佇列 (佇列建立者未發出呼叫)	已保留	無效	無效
配送清單	已保留	無效	無效
受管理訂閱目的地	已保留	無效	無效
發佈清單 (已移除訂閱)	已刪除訊息; 已刪除佇列	無效	無效

訂閱關閉選項: 這些選項控制是否在關閉控點時移除可延續訂閱，以及是否清除仍在等待應用程式讀取的發佈。 這些選項僅適用於 MQSUB 呼叫的 *Hsub* 參數中傳回的物件控點。

MQCO_KEEP_SUB
已關閉訂閱的控點，但會保留所進行的訂閱。 發佈會繼續傳送至訂閱中指定的目的地。 只有在使用選項 MQSO_DURABLE 進行訂閱時，此選項才有效。

如果可延續訂閱，則 MQCO_KEEP_SUB 是預設值

MQCO_REMOVE_SUB
即會移除訂閱，並關閉訂閱的控點。

MQSUB 呼叫的 *Hobj* 參數不會因關閉 *Hsub* 參數而失效，而且可能會繼續用於 MQGET 或 MQCB 來接收其餘發佈。 當 MQSUB 呼叫的 *Hobj* 參數也關閉時，如果它是受管理目的地，則會移除任何未擷取的發佈。

如果訂閱不可延續，則 MQCO_REMOVE_SUB 是預設值。

下表彙總這些訂閱關閉選項。

若要關閉可延續訂閱控點但保留訂閱，請使用下列訂閱關閉選項:

作業	**訂閱關閉選項**
保留 MQOPENed 控點上的發佈	MQCO_KEEP_SUB
移除 MQOPENed 控點上的發佈	不容許動作
保留 MQSO_MANAGED 控點上的發佈	MQCO_KEEP_SUB
移除 MQSO_MANAGED 控點上的發佈	不容許動作

若要取消訂閱，請關閉可延續訂閱控點並取消訂閱，或關閉不可延續訂閱控點，使用下列訂閱關閉選項:

作業	**訂閱關閉選項**
保留 MQOPENed 控點上的發佈	MQCO_REMOVE_SUB
移除 MQOPENed 控點上的發佈	不容許動作
保留 MQSO_MANAGED 控點上的發佈	MQCO_REMOVE_SUB

先讀選項: 下列選項控制在應用程式要求非持續訊息之前已傳送至用戶端，且應用程式尚未耗用這些非持續訊息的情況。 這些訊息儲存在等待應用程式要求的用戶端先讀緩衝區中，可以在 MQCLOSE 完成之前從佇列中捨棄或耗用。

MQCO_IMMEDIATE
物件會立即關閉，並捨棄在應用程式要求之前已傳送至用戶端的任何訊息，且任何應用程式都無法使用這些訊息。 這是預設值。

MQCO_QUIESCE
已提出關閉物件的要求，但如果在應用程式要求之前已傳送至用戶端的任何訊息仍位於用戶端先讀緩衝區中， MQCLOSE 呼叫會傳回警告 MQRC_READ_AHEAD_MSGS ，且物件控點仍然有效。

然後，應用程式可以繼續使用物件控點來擷取訊息，直到無法再使用為止，然後再次關閉物件。 在要求訊息的應用程式之前，不再將其他訊息傳送至用戶端，現在已關閉先讀。

建議應用程式使用 MQCO_QUIESCE ，而不是嘗試達到用戶端先讀緩衝區中沒有其他訊息的點，因為如果使用 MQCO_IMMEDIATE ，則會捨棄最後一個 MQGET 呼叫與下列 MQCLOSE 之間的訊息。

如果從非同步回呼函數內發出具有 MQCO_QUIESCE 的 MQCLOSE ，則會套用提前讀取訊息的相同行為。 如果傳回警告 MQRC_READ_AHEAD_MSGS ，則至少會再呼叫回呼函數一次。 當向前讀取的最後剩餘訊息已傳遞至回呼函數時， MQCBC ConsumerFlags 欄位會設為 MQCBCF_READA_BUFFER_EMPTY。

預設選項: 如果您不需要上述任何選項，則可以使用下列選項:

MQCO_NONE
不需要選用性關閉處理程序。

這個 必須 指定給:

- 佇列以外的物件
- 預先定義的佇列數
- 暫時動態佇列 (但僅在 *Hobj* 是 不 建立佇列之 MQOPEN 呼叫傳回的控點的情況下)。
- 分送清單

在上述所有情況下，物件會保留而不會刪除。

如果對暫時動態佇列指定此選項:

- 如果佇列是由傳回 *Hobj* 的 MQOPEN 呼叫所建立，則會刪除該佇列; 佇列上的任何訊息都會被清除。
- 在所有其他情況下，會保留佇列 (及其上的任何訊息)。

如果針對永久動態佇列指定此選項，則會保留且不會刪除佇列。

在 z/OS 上，如果佇列是已邏輯刪除的動態佇列，且這是它的最後一個控點，則會實際刪除佇列。 有關更多詳細資料，請參閱第 572 頁的『使用注意事項』。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_WARNING
警告 (局部完成)。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

列出的原因碼是佇列管理程式可以針對 *Reason* 參數傳回的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_WARNING:

MQRC_INCOMPLETE_GROUP
(2241 , X'8C1') 訊息群組不完整。

MQRC_INCOMPLETE_MSG
(2242 , X'8C2') 邏輯訊息不完整。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204 , X'89C') 配接卡無法使用。

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130 , X'852 ') 無法載入配接卡服務模組。

MQRC_API_EXIT_ERROR
(2374 , X' 946 ') API 結束程式失敗。

MQRC_API_EXIT_LOAD_ERROR
(2183 , X'887 ') 無法載入 API 結束程式。

MQRC_ASDID_MISMATCH
(2157 , X'86D') 主要和起始 ASID 不同。

MQRC_CALL_IN_PROGRESS
(2219 , X'8AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CF_STRUC_FAILED
(2373 , X' 945 ') 連結機能結構失敗。

MQRC_CF_STRUC_IN_USE
(2346 , X'92A') 連結機能結構使用中。

MQRC_CICS_WAIT_FAILED
(2140 , X'85C') 等待要求被 CICS 拒絕。

MQRC_CONNECTION_BROKEN
(2009 , X'7D9') 與佇列管理程式的連線遺失。

MQRC_CONNECTION_NOT_AUTHORIZED
(2217 , X'8A9') 未獲連線授權。

MQRC_CONNECTION_STOPPING
(2203 , X'89B') 連線關閉。

MQRC__DB2_NOT_AVAILABLE
(2342 , X' 926 ') Db2 子系統無法使用。

MQRC_HCONN_ERROR
(2018 , X'7E2') 連線控點無效。

MQRC_HOBJ_ERROR
(2019 , X'7E3') 物件控點無效。

MQRC_NOT_AUTHORIZED
(2035 , X'7F3') 未獲授權存取。

MQRC_OBJECT_DAMAGED
(2101 , X'835 ') 物件已損壞。

MQRC_OPTION_NOT_VALID_FOR_TYPE
(2045 , X'7FD') 在 MQOPEN 或 MQCLOSE 呼叫上: 選項對物件類型無效。

MQRC_OPTIONS_ERROR
(2046 , X'7FE') 選項無效或不一致。

MQRC_PAGESET_ERROR
(2193 , X'891 ') 存取頁集資料集時發生錯誤。

MQRC_Q_MGR_NAME_ERROR
(2058 , X'80A') 佇列管理程式名稱無效或不明。

MQRC_Q_MGR_NOT_AVAILABLE
(2059 , X'80B') 佇列管理程式無法用於連線。

MQRC_Q_MGR_STOPPING
(2162 , X'872 ') 佇列管理程式關閉。

MQRC_Q_NOT_EMPTY
(2055 , X'807 ') 佇列包含一則以上訊息或未確定的放置或取得要求。

MQRC_READ_AHEAD_MSGS
(nnnn , X'xxx ') 用戶端已先讀應用程式尚未耗用的訊息。

MQRC_RESOURCE_PROBLEM
(2102 , X'836 ') 可用的系統資源不足。

MQRC_SECURITY_ERROR
(2063 , X'80F') 發生安全錯誤。

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'817 ') 可用的儲存體不足。

MQRC_SUPPRESSED_BY_EXIT
(2109 , X'83D') 跳出程式暫停呼叫。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

如需這些代碼的詳細資訊，請參閱 原因碼。

使用注意事項

1. 當應用程式發出 MQDISC 呼叫，或正常或異常結束時，會使用 MQCO_NONE 選項自動關閉應用程式開啟且仍開啟的任何物件。
2. 如果要關閉的物件是 佇列，則下列要點適用:
   - 如果佇列上的作業作為工作單元的一部分執行，則可以在同步點發生之前或之後關閉佇列，而不會影響同步點的結果。 如果觸發佇列，在關閉佇列之前執行回復可能會導致發出觸發訊息。 如需觸發訊息的相關資訊，請參閱 觸發訊息的內容。
   - 如果使用 MQOO_BROWSE 選項開啟佇列，則瀏覽游標會毀損。 如果隨後使用 MQOO_BROWSE 選項重新開啟佇列，則會建立新的瀏覽游標 (請參閱 MQOO_BROWSE)。
   - 如果在 MQCLOSE 呼叫時目前已鎖定此控點的訊息，則會釋放鎖定 (請參閱 MQGMO_LOCK)。
   - 在 z/OS 上，如果針對正在關閉的佇列控點有 MQGMO_SET_信號選項未解決的 MQGET 要求，則會取消要求 (請參閱 MQGMO_SET_SIGNAL)。 相同佇列但針對不同控點 (*Hobj*) 提出的信號要求不受影響 (除非正在刪除動態佇列，在此情況下也會取消動態佇列)。
3. 如果要關閉的物件是 動態佇列 (永久或暫時) ，則下列要點適用:
   - 對於動態佇列，您可以指定 MQCO_DELETE 及 MQCO_DELETE_PURGE 選項，而不論對應 MQOPEN 呼叫上指定的選項為何。
   - 刪除動態佇列時，會取消對佇列具有 MQGMO_WAIT 選項且未完成的所有 MQGET 呼叫，並傳回原因碼 MQRC_Q_DELETED。 請參閱 MQGMO_WAIT。

     雖然應用程式無法存取已刪除的佇列，但在所有參照該佇列的控點都已關閉，且影響該佇列的所有工作單元都已確定或取消之前，不會從系統中移除該佇列，且不會釋放相關資源。

     在 z/OS 上，已邏輯刪除但尚未從系統中移除的佇列會阻止建立與已刪除佇列同名的新佇列; 在此情況下， MQOPEN 呼叫會失敗，原因碼為 MQRC_NAME_IN_USE。 此外，即使應用程式無法存取此類佇列，仍可以使用 MQSC 指令來顯示此類佇列。
   - 刪除永久動態佇列時，如果 MQCLOSE 呼叫上指定的 *Hobj* 控點 不是 建立佇列之 MQOPEN 呼叫所傳回的控點，則會檢查用來驗證 MQOPEN 呼叫的使用者 ID 是否有權刪除佇列。 如果在 MQOPEN 呼叫上指定 MQOO_ALTERNATE_USER_AUTHORITY 選項，則檢查的使用者 ID 是 *AlternateUserId*。

在下列情況下，不會執行此檢查:

– 指定的控點是建立佇列的 MQOPEN 呼叫所傳回的控點。

– 要刪除的佇列是暫時動態佇列。

- 關閉暫時動態佇列時，如果 MQCLOSE 呼叫上指定的 *Hobj* 控點是建立佇列的 MQOPEN 呼叫所傳回的控點，則會刪除佇列。 不論 MQCLOSE 呼叫上指定的關閉選項為何，都會發生這種情況。 如果佇列中有訊息，則會捨棄這些訊息; 不會產生任何報告訊息。

  如果有未確定的工作單元會影響佇列，則仍會刪除佇列及其訊息，但工作單元不會失敗。 不過，如上所述，在確定或取消每一個工作單元之前，不會釋放與工作單元相關聯的資源。

4. 如果要關閉的物件是 配送清單，則下列要點適用:

   - 發佈清單唯一有效的關閉選項是 MQCO_NONE; 如果指定任何其他選項，則呼叫會失敗，原因碼為 MQRC_OPTIONS_ERROR 或 MQRC_OPTION_NOT_VALID_FOR_TYPE。
   - 關閉配送清單時，不會針對清單中的佇列傳回個別完成碼及原因碼; 只有呼叫的 *CompCode* 及 *Reason* 參數可用於診斷。

     如果關閉其中一個佇列失敗，佇列管理程式會繼續處理，並嘗試關閉配送清單中的其餘佇列。 呼叫的 *CompCode* 及 *Reason* 參數設定為傳回說明失敗的資訊。 即使大部分佇列已順利關閉，完成碼也可能是 MQCC_FAILED。 無法識別發生錯誤的佇列。

     如果多個佇列上發生失敗，則不會定義在 *CompCode* 及 *Reason* 參數中報告的失敗。

5. 在 IBM i 上，如果在發出第一個 MQOPEN 呼叫時隱含地連接應用程式，則在發出最後一個 MQCLOSE 時會發生隱含 MQDISC。

   只能隱含地連接以相容模式執行的應用程式; 其他應用程式必須發出 MQCONN 或 MQCONNX 呼叫，才能明確地連接至佇列管理程式。

C 呼叫

```
MQCLOSE (Hconn, &Hobj, Options, &CompCode, &Reason);
```

宣告參數如下:

```
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 呼叫

```
     CALL 'MQCLOSE' USING HCONN, HOBJ, OPTIONS, COMPCODE, REASON.
```

宣告參數如下:

```
**   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 呼叫

```
call MQCLOSE (Hconn, Hobj, Options, CompCode, Reason);
```

宣告參數如下:

```
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 */
```

High Level Assembler 呼叫

```
CALL MQCLOSE,(HCONN,HOBJ,OPTIONS,COMPCODE,REASON)
```

宣告參數如下:

```
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 呼叫

```
MQCLOSE Hconn, Hobj, Options, CompCode, Reason
```

宣告參數如下:

```
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-確定變更

MQCMIT 呼叫會向佇列管理程式指出應用程式已達到同步點，且自前次同步點以來發生的所有訊息取得及放置都會變成永久。

作為工作單元一部分放置的訊息可供其他應用程式使用; 作為工作單元一部分擷取的訊息會被刪除。

- 在 z/OS 上，呼叫僅由批次程式 (包括 IMS 批次 DL/I 程式) 使用。
- 在 IBM i 上，以相容模式執行的應用程式不支援此呼叫。

語法

MQCMIT (*Hconn*， *CompCode*， *Reason*)

參數

Hconn

類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 前一個 MQCONN 或 MQCONNX 呼叫已傳回 *Hconn* 的值。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_WARNING
警告 (局部完成)。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

列出的原因碼是佇列管理程式可以針對 *Reason* 參數傳回的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_WARNING:

MQRC_BACKED_OUT
(2003 , X'7D3') 工作單元已取消。

MQRC_OUTCOME_PENDING
(2124 , X'84C') 確定作業的結果擱置中。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130 , X'852 ') 無法載入配接卡服務模組。

MQRC_API_EXIT_ERROR
(2374 , X' 946 ') API 結束程式失敗。

MQRC_ASDID_MISMATCH
(2157 , X'86D') 主要和起始 ASID 不同。

MQRC_CALL_IN_PROGRESS
(2219 , X'8AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CALL_INTERRUPTED
(2549 , X'9F5') MQPUT 或 MQCMIT 已岔斷，且重新連線處理程序無法重新建立明確的結果。

MQRC_CF_STRUC_IN_USE
(2346 , X'92A') 連結機能結構使用中。

MQRC_CONNECTION_BROKEN
(2009 , X'7D9') 與佇列管理程式的連線遺失。

MQRC_ENVIRONMENT_ERROR
(2012 , X'7DC') 在環境中呼叫無效。

MQRC_HCONN_ERROR
(2018 , X'7E2') 連線控點無效。

MQRC_OBJECT_DAMAGED
(2101 , X'835 ') 物件已損壞。

MQRC_OUTCOME_MIXED
(2123 , X'84B') 確定或取消作業的結果混合。

MQRC_Q_MGR_STOPPING
(2162 , X'872 ') 佇列管理程式關閉。

MQRC_RECONNECT_FAILED
(2548 , X'9F4') 重新連接之後，發生錯誤，恢復可重新連接連線的控點。

MQRC_RESOURCE_PROBLEM
(2102 , X'836 ') 可用的系統資源不足。

MQRC_STORAGE_MEDIUM_FULL
(2192 , X'890 ') 外部儲存媒體已滿。

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'817 ') 可用的儲存體不足。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

如需這些代碼的詳細資訊，請參閱 原因碼。

使用注意事項

1. 只有在佇列管理程式本身協調工作單元時，才使用此呼叫。 這可以是:
   - 本端工作單元，其中變更只會影響 WebSphere MQ 資源。
   - 廣域工作單元，其中變更可能會影響屬於其他資源管理程式的資源，以及影響 WebSphere MQ 資源。

   如需本端及廣域工作單元的進一步詳細資料，請參閱 第 551 頁的『MQBEGIN-開始工作單元』。
2. 在佇列管理程式未協調工作單元的環境中，必須使用適當的確定呼叫，而非 MQCMIT。 環境也可能支援應用程式正常終止所造成的隱含確定。
   - 在 z/OS 上，使用下列呼叫:
     - 如果工作單元僅影響 WebSphere MQ 資源，則批次程式 (包括 IMS 批次 DL/I 程式) 可以使用 MQCMIT 呼叫。 不過，如果工作單元同時影響 WebSphere MQ 資源和屬於其他資源管理程式的資源 (例如， DB2) ，請使用 z/OS 可回復資源服務 (RRS) 提供的 SRRCMIT 呼叫。 SRRCMIT 呼叫會對屬於已啟用 RRS 協調之資源管理程式的資源確定變更。
     - CICS 應用程式必須使用 EXEC CICS SYNCPOINT 指令來明確地確定工作單元。 或者，結束交易會導致隱含地確定工作單元。 MQCMIT 呼叫無法用於 CICS 應用程式。
     - IMS 應用程式 (非批次 DL/I 程式) 必須使用 IMS 呼叫 (例如 GU 和 CHKP) 來確定工作單元。 MQCMIT 呼叫無法用於 IMS 應用程式 (批次 DL/I 程式除外)。
   - 在 IBM i 上，針對佇列管理程式所協調的本端工作單元使用此呼叫。 這表示確定定義不得存在於工作層次，亦即，不得對工作發出具有 CMTSCOPE(*JOB) 參數的 STRCMTCTL 指令。
3. 如果應用程式以工作單元中未確定的變更結束，則那些變更的處置取決於應用程式是正常結束還是異常結束。 如需進一步詳細資料，請參閱 MQDISC 使用注意事項 。
4. 當應用程式在邏輯訊息的群組或區段中放置或取得訊息時，佇列管理程式會保留與前次成功 MQPUT 及 MQGET 呼叫的訊息群組及邏輯訊息相關的資訊。 此資訊與佇列控點相關聯，並包括如下內容:
   - MQMD 中 *GroupId*、 *MsgSeqNumber*、 *Offset* 及 *MsgFlags* 欄位的值。
   - 訊息是否為工作單元的一部分。
   - 對於 MQPUT 呼叫: 訊息是持續還是非持續。

   確定工作單元時，佇列管理程式會保留群組及區段資訊，且應用程式可以繼續在現行訊息群組或邏輯訊息中放置或取得訊息。

   在確定工作單元時保留群組和區段資訊，可讓應用程式將大型訊息群組或大型邏輯訊息散佈在數個工作單元之間，由多個區段組成。 如果本端佇列管理程式只有有限的佇列儲存體，則使用數個工作單元是有利的。 不過，如果發生系統故障，應用程式必須維護足夠的資訊，以在正確的點重新啟動放置或取得訊息。 如需如何在系統失敗之後正確點重新啟動的詳細資料，請參閱 MQPMO_LOGICAL_ORDER 和 MQGMO_LOGICAL_ORDER。

   只有在佇列管理程式協調工作單元時，其餘使用注意事項才適用:
5. 工作單元的範圍與連線控點相同; 所有影響特定工作單元的 WebSphere MQ 呼叫必須使用相同的連線控點來執行。 使用不同連線控點發出的呼叫 (例如，由另一個應用程式發出的呼叫) 會影響不同的工作單元。 如需連線控點範圍的相關資訊，請參閱 MQCONN 中說明的 *Hconn* 參數。
6. 只有作為現行工作單元一部分放置或擷取的訊息才會受到此呼叫的影響。

7. 在工作單元內發出 MQGET、MQPUT 或 MQPUT1 呼叫，但永不發出確定或取消呼叫的長時間執行應用程式，可以在佇列中填入其他應用程式無法使用的訊息。 為了防止這種情況，管理者必須將 *MaxUncommittedMsgs* 佇列管理程式屬性設為足夠低，以防止失控的應用程式填滿佇列，但足夠高，以容許預期的傳訊應用程式正確運作。
8. 在 UNIX 和 Windows 系統上，如果 *Reason* 參數是 MQRC_CONNECTION_BROKEN (*CompCode* 為 MQCC_FAILED) 或 MQRC_UNEXPECTED_ERROR ，則可能已順利確定工作單元。

C 呼叫

```
MQCMIT (Hconn, &CompCode, &Reason);
```

宣告參數如下:

```
MQHCONN  Hconn;     /* Connection handle */
MQLONG   CompCode;  /* Completion code */
MQLONG   Reason;    /* Reason code qualifying CompCode */
```

COBOL 呼叫

```
CALL 'MQCMIT' USING HCONN, COMPCODE, REASON.
```

宣告參數如下:

```
**   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 呼叫

```
call MQCMIT (Hconn, CompCode, Reason);
```

宣告參數如下:

```
dcl Hconn     fixed bin(31);  /* Connection handle */
dcl CompCode  fixed bin(31);  /* Completion code */
dcl Reason    fixed bin(31);  /* Reason code qualifying CompCode */
```

High Level Assembler 呼叫

```
CALL MQCMIT,(HCONN,COMPCODE,REASON)
```

宣告參數如下:

```
HCONN     DS  F  Connection handle
COMPCODE  DS  F  Completion code
REASON    DS  F  Reason code qualifying COMPCODE
```

Visual Basic 呼叫

```
MQCMIT Hconn, CompCode, Reason
```

宣告參數如下:

```
Dim Hconn    As Long 'Connection handle'
Dim CompCode As Long 'Completion code'
Dim Reason   As Long 'Reason code qualifying CompCode'
```

MQCONN-連接佇列管理程式

MQCONN 呼叫會將應用程式連接至佇列管理程式。

它提供佇列管理程式連線控點，供應用程式在後續的訊息佇列作業呼叫中使用。

- 在 z/OS 上， CICS 應用程式不需要發出此呼叫。 這些應用程式會自動連接至 CICS 系統所連接的佇列管理程式。 不過，仍然接受來自 CICS 應用程式的 MQCONN 及 MQDISC 呼叫。
- 在 IBM i 上，以相容模式執行的應用程式不需要發出此呼叫。 當這些應用程式發出第一個 MQOPEN 呼叫時，它們會自動連接至佇列管理程式。 不過，仍然接受來自 IBM i 應用程式的 MQCONN 和 MQDISC 呼叫。

  其他應用程式 (亦即，未以相容模式執行的應用程式) 必須使用 MQCONN 或 MQCONNX 呼叫來連接至佇列管理程式，以及使用 MQDISC 呼叫來切斷與佇列管理程式的連線。 這是建議的程式設計樣式。

無法在僅限伺服器安裝上建立用戶端連線，也無法在僅限用戶端安裝上建立本端連線。

語法

MQCONN (*QMgrName*、 *Hconn*、 *CompCode*、 *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

這是應用程式要連接的佇列管理程式名稱。 名稱可以包含下列字元:

- 大寫英文字母 (A 到 Z)
- 小寫英文字母 (a 到 z)
- 數字 (0 到 9)
- 句點 (.)、正斜線 (/)、底線 (_)、百分比 (%)

名稱不能包含前導或內含空白，但可以包含尾端空白。 空值字元可用來指出名稱中有效資料的結尾; 空值及其後面的任何字元會被視為空白。 下列限制適用於指出的環境:

- 在使用 EBCDIC Katakana 的系統上，無法使用小寫字元。
- 在 z/OS 上，作業及控制台無法處理以底線開頭或結尾的名稱。 基於此原因，請避免使用此類名稱。
- 在 IBM i 上，當在指令上指定時，請用引號括住包含小寫字元、正斜線或百分比的名稱。 請勿在 *QMgrName* 參數中指定這些引號。

如果名稱完全由空白組成，則會使用 預設 佇列管理程式的名稱。

指定給 *QMgrName* 的名稱必須是 可連接 佇列管理程式的名稱。

在 z/OS 上，可以連接的佇列管理程式由環境決定:

- 若為 CICS，您只能使用 CICS 系統所連接的佇列管理程式。 仍必須指定 *QMgrName* 參數，但會忽略其值; 建議使用空白。
- 若為 IMS，只有子系統定義表 (CSQQDEFV) 中列出的佇列管理程式，以及 IMS 中的 SSM 表格中列出的 和 佇列管理程式才可連接 (請參閱用法附註 6)。

- 若為 z/OS 批次和 TSO，只有位於與應用程式相同的系統上的佇列管理程式可連接 (請參閱用法附註 6)。

佇列共用群組:在數個佇列管理程式存在且配置成形成佇列共用群組的系統上，可以針對 *QMgrName* 指定佇列共用群組的名稱，以取代佇列管理程式的名稱。 這可讓應用程式連接至 佇列共用群組中可用的任何佇列管理程式，且與應用程式位於相同 z/OS 映像檔。 系統也可以配置成使用空白 *QMgrName* 來連接至佇列共用群組，而不是預設佇列管理程式。

如果 *QMgrName* 指定佇列共用群組的名稱，但系統上也有一個具有該名稱的佇列管理程式，則會優先於前者與後者建立連線。 只有在連線失敗時，才會嘗試連接佇列共用群組中的其中一個佇列管理程式。

如果連線成功，您可以使用 MQCONN 或 MQCONNX 呼叫傳回的控點來存取屬於已建立連線之佇列管理程式的 所有 資源 (共用及非共用)。 對這些資源的存取受一般授權控制。

如果應用程式發出兩個 MQCONN 或 MQCONNX 呼叫來建立並行連線，且其中一個或兩個呼叫指定佇列共用群組的名稱，則當第二個呼叫連接至與第一個呼叫相同的佇列管理程式時，會傳回完成碼 MQCC_WARNING 及原因碼 MQRC_ALREADY_CONNECTED。

佇列共用群組僅在 z/OS 上受支援。 只有在批次、RRS 批次和 TSO 環境中，才支援佇列共用群組的連線。

WebSphere MQ MQI 用戶端應用程式: 對於 WebSphere MQ MQI 用戶端應用程式，會針對每一個具有指定佇列管理程式名稱的用戶端連線通道定義，嘗試建立連線，直到成功為止。 不過，佇列管理程式必須具有與指定名稱相同的名稱。 如果指定 all-blank 名稱，則會嘗試每一個具有 all-blank 佇列管理程式名稱的用戶端連線通道，直到成功為止; 在此情況下，不會對佇列管理程式的實際名稱進行檢查。

WebSphere MQ 用戶端應用程式在 z/OS 中不受支援，但 z/OS 可以作為 WebSphere MQ 用戶端應用程式可以連接的 WebSphere MQ 伺服器。

WebSphere MQ MQI 用戶端佇列管理程式群組: 如果指定的名稱以星號 (*) 開頭，則與之建立連線的佇列管理程式名稱可能與應用程式指定的名稱不同。 指定的名稱 (不含星號) 會定義適合連線的佇列管理程式 群組 。 實作會從群組中選取一個，方法是依序嘗試每一個群組，直到找到一個可建立連線的群組為止。 嘗試連線的順序受到候選通道的用戶端通道加權及連線親緣性值的影響。 如果群組中沒有任何佇列管理程式可用於連線，則呼叫會失敗。 每一個佇列管理程式只會嘗試一次。 如果只指定星號作為名稱，則會使用實作定義的預設佇列管理程式群組。

只有在 MQ 用戶端環境中執行的應用程式才支援佇列管理程式群組; 如果非用戶端應用程式指定以星號開頭的佇列管理程式名稱，則呼叫會失敗。 透過提供數個具有相同佇列管理程式名稱 (不含星號的指定名稱) 的用戶端連線通道定義來定義群組，以與群組中的每一個佇列管理程式進行通訊。 預設群組的定義方式是提供一或多個用戶端連線通道定義，每一個定義都具有空白佇列管理程式名稱 (因此指定全空白名稱的效果與指定用戶端應用程式名稱的單一星號相同)。

連接至群組的一個佇列管理程式之後，應用程式可以在訊息及物件描述子的佇列管理程式名稱欄位中以一般方式指定空白，以表示應用程式已連接的佇列管理程式名稱 (本端佇列管理程式)。 如果應用程式需要知道此名稱，請使用 MQINQ 呼叫來查詢 *QMgrName* 佇列管理程式屬性。

在連線名稱前面加上星號表示應用程式不依賴連接至群組中的特定佇列管理程式。 適合的應用程式如下:

- 放置訊息但不取得訊息的應用程式。
- 放置要求訊息，然後從 暫時動態 佇列取得回覆訊息的應用程式。

不適用的應用程式是需要從特定佇列管理程式的特定佇列中取得訊息的應用程式; 此類應用程式不得在名稱前面加上星號。

如果您指定星號，則名稱其餘部分的長度上限為 47 個字元。

z/OS 不支援佇列管理程式群組。

此參數的長度由 MQ_Q_MGR_NAME_LENGTH 提供。

Hconn

類型 :MQHCONN-輸出

此控點代表佇列管理程式的連線。 在應用程式發出的所有後續訊息佇列作業呼叫中指定它。 當發出 MQDISC 呼叫時，或當定義控點範圍的處理單元終止時，它就不再有效。

WebSphere MQ 現在為 mqm 程式庫提供用戶端套件及伺服器套件。 這表示當進行在 mqm 程式庫中找到的 MQI 呼叫時，會檢查連線類型以查看它是否為用戶端或伺服器連線，然後進行正確的基礎呼叫。 因此，現在可以針對 mqm 程式庫鏈結傳遞 *Hconn* 的結束程式，但在用戶端安裝上使用。

處理範圍:傳回的控點範圍視用來連接佇列管理程式 (MQCONN 或 MQCONNX) 的呼叫而定。 如果使用的呼叫是 MQCONNX，則控點的範圍也取決於 MQCNO 結構的 *Options* 欄位中指定的 MQCNO_HANDLE_SHARE_ * 選項。

- 如果呼叫是 MQCONN，或指定 MQCNO_HANDLE_SHARE_NONE 選項，則傳回的控點是 非共用 控點。

  非共用控點的範圍是執行應用程式的平台所支援的最小平行處理單元 (如需詳細資料，請參閱 第 580 頁的表 564); 控點在發出呼叫的平行處理單元之外無效。
- 如果您指定 MQCNO_HANDLE_SHARE_BLOCK 或 MQCNO_HANDLE_SHARE_NO_BLOCK 選項，則傳回的控點是 共用 控點。

  共用控點的範圍是擁有從中發出呼叫之執行緒的處理程序; 控點可以從屬於該處理程序的任何執行緒使用。 並非所有平台都支援執行緒。
- 如果 MQCONN 或 MQCONNX 呼叫失敗，且完成碼等於 MQCC_FAILED，則未定義 Hconn 值。

表 *564:* 各種平台上的非共用控點範圍

平台	**非共用控點的範圍**
z/OS	• CICS: CICS 作業 • IMS: 作業，直到下一個同步點 (排除作業的子作業) • z/OS 批次和 TSO: 作業 (排除作業的子作業)
IBM i	工作
UNIX 系統	執行緒
16 位元 Windows 應用程式	處理程序
32 位元 Windows 應用程式	執行緒

在 z/OS (適用於 CICS 應用程式) 及 IBM i (適用於以相容模式執行的應用程式) 上，傳回的值如下:

MQHC_DEF_HCONN
預設連線控點。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_WARNING
警告 (局部完成)。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_WARNING:

MQRC_ALREADY_CONNECTED
(2002 , X'7D2') 應用程式已連接。

MQRC_CLUSTER_EXIT_LOAD_ERROR
(2267 , X'8DB') 無法載入叢集工作量結束程式。

MQRC_SSL_ALREADY_INITIALIZED
(2391 , X' 957 ') SSL 已起始設定。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_CONN_LOAD_ERROR
(2129 , X'851 ') 無法載入配接卡連線模組。

MQRC_ADAPTER_DEFS_ERROR
(2131 , X'853 ') 配接卡子系統定義模組無效。

MQRC_ADAPTER_DEFS_LOAD_ERROR
(2132 , X'854 ') 無法載入配接卡子系統定義模組。

MQRC_ADAPTER_NOT_AVAILABLE
(2204 , X'89C') 配接卡無法使用。

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130 , X'852 ') 無法載入配接卡服務模組。

MQRC_ADAPTER_STORAGE_短缺
(2127 , X'84F') 配接卡的儲存體不足。

MQRC_ANOTHER_Q_MGR_CONNECTED
(2103 , X'837 ') 已連接另一個佇列管理程式。

MQRC_API_EXIT_ERROR
(2374 , X' 946 ') API 結束程式失敗。

MQRC_API_EXIT_INIT_ERROR
(2375 , X' 947 ') API 結束程式起始設定失敗。

MQRC_API_EXIT_TERM_ERROR
(2376 , X' 948 ') API 結束程式終止失敗。

MQRC_ASDID_MISMATCH
(2157 , X'86D') 主要和起始 ASID 不同。

MQRC_BUFFER_LENGTH_ERROR
(2005 , X'7D5') 緩衝區長度參數無效。

MQRC_CALL_IN_PROGRESS
(2219 , X'8AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CONN_ID_IN_USE
(2160 , X'870 ') 連線 ID 已在使用中。

MQRC_CONNECTION_BROKEN
(2009 , X'7D9') 與佇列管理程式的連線遺失。

MQRC_CONNECTION_ERROR
(2273 , X'8E1') 處理 MQCONN 呼叫時發生錯誤。

MQRC_CONNECTION_NOT_AVAILABLE
(2568 , X'A08') 當佇列管理程式無法在現行安裝上提供所要求連線類型的連線時，在 MQCONN 或 MQCONNX 呼叫中發生。 無法僅在伺服器安裝上建立用戶端連線。 無法僅在用戶端安裝上建立本端連線。

MQRC_CONNECTION_QUIESCING
(2202 , X'89A') 連線靜止。

MQRC_CONNECTION_STOPPING
(2203 , X'89B') 連線關閉。

MQRC_CRYPTO_HARDWARE_ERROR
(2382 , X'94E') 加密硬體配置錯誤。

MQRC_DUPLICATE_RECOV_COORD
(2163 , X'873 ') 回復協調程式存在。

MQRC_ENVIRONMENT_ERROR
(2012 , X'7DC') 在環境中呼叫無效。

MQRC_HCONN_ERROR
(2018 , X'7E2') 連線控點無效。

MQRC_HOST_NOT_AVAILABLE
(2538 , X'9EA') 已從用戶端發出 MQCONN 呼叫來連接至佇列管理程式，但嘗試將交談配置給遠端系統失敗。

MQRC_INSTALLER_MISMATCH
(2583 , X'A17') 佇列管理程式安裝與選取的程式庫之間不符。

MQRC_KEY_REPOSITORY_ERROR
(2381 , X'94D') 金鑰儲存庫無效。

MQRC_MAX_CONNS_LIMIT_REACHED
(2025 , X'7E9') 已達到連線數目上限。

MQRC_NOT_AUTHORIZED
(2035 , X'7F3') 未獲授權存取。

MQRC_OPEN_FAILED
(2137 , X'859 ') 物件未順利開啟。

MQRC_Q_MGR_NAME_ERROR
(2058 , X'80A') 佇列管理程式名稱無效或不明。

MQRC_Q_MGR_NOT_AVAILABLE
(2059 , X'80B') 佇列管理程式無法用於連線。

MQRC_Q_MGR_QUIESCING
(2161 , X'871 ') 佇列管理程式靜止中。

MQRC_Q_MGR_STOPPING
(2162 , X'872 ') 佇列管理程式關閉。

MQRC_RESOURCE_PROBLEM
(2102 , X'836 ') 可用的系統資源不足。

MQRC_SECURITY_ERROR
(2063 , X'80F') 發生安全錯誤。

MQRC_SSL_INITIALIZATION_ERROR
(2393 , X' 959 ') SSL 起始設定錯誤。

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'817 ') 可用的儲存體不足。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

如需這些代碼的詳細資訊，請參閱 原因碼。

使用注意事項

1. 使用 MQCONN 呼叫建立連線的佇列管理程式稱為 本端佇列管理程式。
2. 在應用程式中，本端佇列管理程式所擁有的佇列會顯示為本端佇列。 可以將訊息放置在這些佇列上，並從這些佇列取得訊息。

   本端佇列管理程式所屬的佇列共用群組所擁有的共用佇列，在應用程式中會顯示為本端佇列。 可以將訊息放置在這些佇列上，並從這些佇列取得訊息。

   遠端佇列管理程式所擁有的佇列會顯示為遠端佇列。 可以將訊息放置在這些佇列上，但無法從這些佇列取得訊息。

3. 如果佇列管理程式在應用程式執行時失敗，則應用程式必須重新發出 MQCONN 呼叫，以取得要在後續 WebSphere MQ 呼叫中使用的新連線控點。 應用程式可以定期發出 MQCONN 呼叫，直到呼叫成功為止。

   如果應用程式不確定是否已連接至佇列管理程式，則應用程式可以安全地發出 MQCONN 呼叫來取得連線控點。 如果已連接應用程式，則傳回的控點與前一個 MQCONN 呼叫所傳回的控點相同，但具有完成碼 MQCC_WARNING 及原因碼 MQRC_ALREADY_CONNECTED。
4. 當應用程式完成使用 WebSphere MQ 呼叫時，應用程式必須使用 MQDISC 呼叫來切斷與佇列管理程式的連線。
5. 如果 MQCONN 呼叫失敗，且完成碼等於 MQCC_FAILED ，則未定義 Hconn 值。
6. 在 z/OS 上:

   - 批次、TSO 和 IMS 應用程式必須發出 MQCONN 呼叫來使用其他 WebSphere MQ 呼叫。 這些應用程式可以同時連接至多個佇列管理程式。

     如果佇列管理程式失敗，應用程式必須在佇列管理程式重新啟動之後重新發出呼叫，以取得新的連線控點。

     雖然 IMS 應用程式可以反覆地發出 MQCONN 呼叫，即使已連接也不建議用於線上訊息處理程式 (MPP)。
   - CICS 應用程式不需要發出 MQCONN 呼叫即可使用其他 WebSphere MQ 呼叫，但如果他們想要的話，也可以這樣做; MQCONN 呼叫和 MQDISC 呼叫都會被接受。 不過，無法同時連接多個佇列管理程式。

     如果佇列管理程式失敗，當佇列管理程式重新啟動時，這些應用程式會自動重新連接，因此不需要發出 MQCONN 呼叫。
7. 在 z/OS 上，若要定義可用的佇列管理程式:

   - 對於批次應用程式，系統程式設計師可以使用 CSQBDEF 巨集來建立模組 (CSQBDEFV) ，以定義預設佇列管理程式名稱或佇列共用群組名稱。
   - 對於 IMS 應用程式，系統程式設計師可以使用 CSQQDEFFX 巨集來建立模組 (CSQQDEFV) ，以定義可用佇列管理程式的名稱並指定預設佇列管理程式。

     此外，每一個佇列管理程式都必須定義給 IMS 控制區域，以及定義給存取該佇列管理程式的每一個相依區域。 若要這麼做，您必須在 IMS 中建立子系統成員。 PROCLIB 程式庫及識別適用 IMS 區域的子系統成員。 如果應用程式嘗試連接至未定義在其 IMS 區域之子系統成員中的佇列管理程式，則應用程式會異常終止。
8. 在 IBM i 上，針對舊版佇列管理程式所撰寫的應用程式可以在不重新編譯的情況下執行。 這叫做 相容模式。 此作業模式為應用程式提供相容的執行時期環境。 它包括:

   - 位於檔案庫 QMQM 中的服務程式 AMQZSTUB。

     AMQZSTUB 提供與舊版相同的公用介面，且具有相同的簽章。 使用此服務程式可透過連結程序呼叫來存取 MQI。
   - 位於檔案庫 QMQM 中的程式 QMQM。

     QMQM 提供透過動態程式呼叫來存取 MQI 的方法。
   - 程式 MQCLOSE、MQCONN、MQDISC、MQGET、MQINQ、MQOPEN、MQPUT、 MQPUT1 及 MQSET 位於檔案庫 QMQM 中。

     這些程式也提供透過動態程式呼叫存取 MQI 的方法，但具有對應於 WebSphere MQ 呼叫標準說明的參數清單。

   這三個介面不包括 WebSphere MQ 5.1 中引進的功能。 例如，不支援 MQBACK、MQCMIT 及 MQCONNX 呼叫。 這些介面提供的支援僅適用於單一執行緒應用程式。

   支援單一執行緒應用程式中的新 WebSphere MQ 呼叫，以及多執行緒應用程式中的所有 WebSphere MQ 呼叫，可透過服務程式 LIBMQM 和 LIBMQM_R 提供。
9. 在 IBM i 上，異常結束的程式不會自動與佇列管理程式斷線。 撰寫應用程式以容許 MQCONN 或 MQCONNX 呼叫傳回完成碼 MQCC_WARNING 及原因碼 MQRC_ALREADY_CONNECTED 的可能性。 正常使用在此狀況中傳回的連線控點。

C 呼叫

```
MQCONN (QMgrName, &Hconn, &CompCode, &Reason);
```

宣告參數如下:

```
MQCHAR48  QMgrName;  /* Name of queue manager */
MQHCONN   Hconn;     /* Connection handle */
MQLONG    CompCode;  /* Completion code */
MQLONG    Reason;    /* Reason code qualifying CompCode */
```

COBOL 呼叫

```
CALL 'MQCONN' USING QMGRNAME, HCONN, COMPCODE, REASON.
```

宣告參數如下:

```
**   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 呼叫

```
call MQCONN (QMgrName, Hconn, CompCode, Reason);
```

宣告參數如下:

```
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 */
```

High Level Assembler 呼叫

```
CALL MQCONN,(QMGRNAME,HCONN,COMPCODE,REASON)
```

宣告參數如下:

```
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 呼叫

```
MQCONN QMgrName, Hconn, CompCode, Reason
```

宣告參數如下:

```
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-連接佇列管理程式 (延伸)

MQCONNX 呼叫會將應用程式連接至佇列管理程式。 它提供佇列管理程式連線控點，供應用程式在後續的 WebSphere MQ 呼叫中使用。

MQCONNX 呼叫與 MQCONN 呼叫類似，但 MQCONNX 容許指定選項來控制呼叫的運作方式。

- 所有 WebSphere MQ 系統及連接至這些系統的 WebSphere MQ 用戶端都支援此呼叫。
- 在 IBM i 上，以相容模式執行的應用程式不支援此呼叫。

無法在僅限伺服器安裝上建立用戶端連線，也無法在僅限用戶端安裝上建立本端連線。

語法

MQCONNX (*QMgrName*、 *ConnectOpts*、 *Hconn*、 *CompCode*、 *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

如需詳細資料，請參閱 第 578 頁的『MQCONN-連接佇列管理程式』 中說明的 *QMgrName* 參數。

ConnectOpts
類型 :MQCNO-輸入/輸出

請參閱第 282 頁的『MQCNO-連接選項』，以取得詳細資料。

Hconn
類型 :MQHCONN-輸出

此控點代表佇列管理程式的連線。 在應用程式發出的所有後續訊息佇列作業呼叫中指定它。 當發出 MQDISC 呼叫時，或當定義控點範圍的處理單元終止時，它就不再有效。

WebSphere MQ 現在為 mqm 程式庫提供用戶端套件及伺服器套件。 這表示當進行在 mqm 程式庫中找到的 MQI 呼叫時，會檢查連線類型以查看它是否為用戶端或伺服器連線，然後進行正確的基礎呼叫。 因此，現在可以針對 mqm 程式庫鏈結傳遞 *Hconn* 的結束程式，但在用戶端安裝上使用。

處理範圍:傳回的控點範圍視用來連接佇列管理程式 (MQCONN 或 MQCONNX) 的呼叫而定。 如果使用的呼叫是 MQCONNX ，則控點的範圍也取決於 MQCNO 結構的 *Options* 欄位中指定的 MQCNO_HANDLE_SHARE_ * 選項。

- 如果呼叫是 MQCONN ，或指定 MQCNO_HANDLE_SHARE_NONE 選項，則傳回的控點是 非共用 控點。

  非共用控點的範圍是執行應用程式的平台所支援的最小平行處理單元 (如需詳細資料，請參閱 第 586 頁的表 565); 控點在發出呼叫的平行處理單元之外無效。
- 如果您指定 MQCNO_HANDLE_SHARE_BLOCK 或 MQCNO_HANDLE_SHARE_NO_BLOCK 選項，則傳回的控點是 共用 控點。

  共用控點的範圍是擁有從中發出呼叫之執行緒的處理程序; 控點可以從屬於該處理程序的任何執行緒使用。 並非所有平台都支援執行緒。
- 如果 MQCONN 或 MQCONNX 呼叫失敗，且完成碼等於 MQCC_FAILED ，則未定義 Hconn 值。

表 *565:* 各種平台上的非共用控點範圍	
平台	**非共用控點的範圍**
z/OS	• CICS: CICS 作業 • IMS: 作業，直到下一個同步點 (排除作業的子作業) • z/OS 批次和 TSO: 作業 (排除作業的子作業)
IBM i	工作
UNIX 系統	執行緒
16 位元 Windows 應用程式	處理程序
32 位元 Windows 應用程式	執行緒

在 z/OS (適用於 CICS 應用程式) 及 IBM i (適用於以相容模式執行的應用程式) 上，傳回的值如下:

MQHC_DEF_HCONN
預設連線控點。

CompCode
類型 :MQLONG-輸出

如需詳細資料，請參閱 第 578 頁的『MQCONN-連接佇列管理程式』 中說明的 *CompCode* 參數。

原因
類型 :MQLONG-輸出

下列代碼可以由 MQCONN 及 MQCONNX 呼叫傳回。 如需 MQCONNX 呼叫可傳回的其他代碼清單，請參閱下列代碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_WARNING:

MQRC_ALREADY_CONNECTED
(2002 ， X'7D2') 應用程式已連接。

MQRC_CLUSTER_EXIT_LOAD_ERROR
(2267 ， X'8DB') 無法載入叢集工作量結束程式。

MQRC_SSL_ALREADY_INITIALIZED
(2391 ， X' 957 ') SSL 已起始設定。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_CONN_LOAD_ERROR
(2129 ， X'851 ') 無法載入配接卡連線模組。

MQRC_ADAPTER_DEFS_ERROR
(2131 ， X'853 ') 配接卡子系統定義模組無效。

MQRC_ADAPTER_DEFS_LOAD_ERROR
(2132 ， X'854 ') 無法載入配接卡子系統定義模組。

MQRC_ADAPTER_NOT_AVAILABLE
(2204 ， X'89C') 配接卡無法使用。

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130 ， X'852 ') 無法載入配接卡服務模組。

MQRC_ADAPTER_STORAGE_短缺
(2127 ， X'84F') 配接卡的儲存體不足。

MQRC_ANOTHER_Q_MGR_CONNECTED
(2103 , X'837 ') 已連接另一個佇列管理程式。

MQRC_API_EXIT_ERROR
(2374 , X' 946 ') API 結束程式失敗。

MQRC_API_EXIT_INIT_ERROR
(2375 , X' 947 ') API 結束程式起始設定失敗。

MQRC_API_EXIT_TERM_ERROR
(2376 , X' 948 ') API 結束程式終止失敗。

MQRC_ASDID_MISMATCH
(2157 , X'86D') 主要和起始 ASID 不同。

MQRC_BUFFER_LENGTH_ERROR
(2005 , X'7D5') 緩衝區長度參數無效。

MQRC_CALL_IN_PROGRESS
(2219 , X'8AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CONN_ID_IN_USE
(2160 , X'870 ') 連線 ID 已在使用中。

MQRC_CONNECTION_BROKEN
(2009 , X'7D9') 與佇列管理程式的連線遺失。

MQRC_CONNECTION_ERROR
(2273 , X'8E1') 處理 MQCONN 呼叫時發生錯誤。

MQRC_CONNECTION_NOT_AVAILABLE
(2568 , X'A08') 當佇列管理程式無法在現行安裝上提供所要求連線類型的連線時，在 MQCONN 或 MQCONNX 呼叫中發生。 無法僅在伺服器安裝上建立用戶端連線。 無法僅在用戶端安裝上建立本端連線。

MQRC_CONNECTION_QUIESCING
(2202 , X'89A') 連線靜止。

MQRC_CONNECTION_STOPPING
(2203 , X'89B') 連線關閉。

MQRC_CRYPTO_HARDWARE_ERROR
(2382 , X'94E') 加密硬體配置錯誤。

MQRC_DUPLICATE_RECOV_COORD
(2163 , X'873 ') 回復協調程式存在。

MQRC_ENVIRONMENT_ERROR
(2012 , X'7DC') 在環境中呼叫無效。

MQRC_HCONN_ERROR
(2018 , X'7E2') 連線控點無效。

MQRC_HOST_NOT_AVAILABLE
(2538 , X'9EA') 已從用戶端發出 MQCONN 呼叫來連接至佇列管理程式，但嘗試將交談配置給遠端系統失敗。

MQRC_INSTALLER_MISMATCH
(2583 , X'A17') 佇列管理程式安裝與選取的程式庫之間不符。

MQRC_KEY_REPOSITORY_ERROR
(2381 , X'94D') 金鑰儲存庫無效。

MQRC_MAX_CONNS_LIMIT_REACHED
(2025 , X'7E9') 已達到連線數目上限。

MQRC_NOT_AUTHORIZED
(2035 , X'7F3') 未獲授權存取。

MQRC_OPEN_FAILED
(2137 , X'859 ') 物件未順利開啟。

MQRC_Q_MGR_NAME_ERROR
(2058 , X'80A') 佇列管理程式名稱無效或不明。

MQRC_Q_MGR_NOT_AVAILABLE
(2059 , X'80B') 佇列管理程式無法用於連線。

MQRC_Q_MGR_QUIESCING
(2161 , X'871 ') 佇列管理程式靜止中。

MQRC_Q_MGR_STOPPING
(2162 , X'872 ') 佇列管理程式關閉。

MQRC_RESOURCE_PROBLEM
(2102 , X'836 ') 可用的系統資源不足。

MQRC_SECURITY_ERROR
(2063 , X'80F') 發生安全錯誤。

MQRC_SSL_INITIALIZATION_ERROR
(2393 , X' 959 ') SSL 起始設定錯誤。

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'817 ') 可用的儲存體不足。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

MQCONNX 呼叫可以傳回下列其他原因碼:

如果 *CompCode* 是 MQCC_FAILED:

MQRC_AIR_ERROR
(2385 , X' 951 ') 鑑別資訊記錄無效。

MQRC_AUTH_INFO_CONN_NAME_ERROR
(2387 , X' 953 ') 鑑別資訊連線名稱無效。

MQRC_AUTH_INFO_REC_COUNT_ERROR
(2383 , X'94F') 鑑別資訊記錄計數無效。

MQRC_AUTH_INFO_REC_ERROR
(2384 , X' 950 ') 鑑別資訊記錄欄位無效。

MQRC_AUTH_INFO_TYPE_ERROR
(2386 , X' 952 ') 鑑別資訊類型無效。

MQRC_CD_ERROR
(2277 , X'8E5') 通道定義無效。

MQRC_CLIENT_CONN_ERROR
(2278 , X'8E6') 用戶端連線欄位無效。

MQRC_CNO_ERROR
(2139 , X'85B') 連接選項結構無效。

MQRC_CONN_TAG_IN_USE
(2271 , X'8DF') Connection 標籤使用中。

MQRC_CONN_TAG_NOT_USABLE
(2350 , X'92E') 連線標籤無法使用。

MQRC_LDAP_PASSWORD_ERROR
(2390 , X' 956 ') LDAP 密碼無效。

MQRC_LDAP_USER_NAME_ERROR
(2388 , X' 954 ') LDAP 使用者名稱欄位無效。

MQRC_LDAP_USER_NAME_LENGTH_ERR
(2389 , X' 955 ') LDAP 使用者名稱長度無效。

MQRC_OPTIONS_ERROR
(2046 , X'7FE') 選項無效或不一致。

MQRC_SCO_ERROR
(2380 , X'94C') SSL 配置選項結構無效。

MQRC_SSL_CONFIG_ERROR
(2392 , X' 958 ') SSL 配置錯誤。

如需這些代碼的詳細資訊，請參閱 原因碼。

使用注意事項

對於 Visual Basic 程式設計語言，下列要點適用:

- *ConnectOpts* 參數宣告為 MQCNO 類型。 如果應用程式以 WebSphere MQ MQI 用戶端身分執行，且您想要指定用戶端連線通道的參數，請將 *ConnectOpts* 參數宣告為 Any 類型，讓應用程式可以在呼叫時指定 MQCNOCD 結構取代 MQCNO 結構。 不過，這表示無法檢查 *ConnectOpts* 參數，以確定它是正確的資料類型。

C 呼叫

```
MQCONNX (QMgrName, &ConnectOpts, &Hconn, &CompCode, &Reason);
```

宣告參數如下:

```
MQCHAR48  QMgrName;     /* Name of queue manager */
MQCNO     ConnectOpts;  /* Options that control the action of MQCONNX */
MQHCONN   Hconn;        /* Connection handle */
MQLONG    CompCode;     /* Completion code */
MQLONG    Reason;       /* Reason code qualifying CompCode */
```

COBOL 呼叫

```
CALL 'MQCONNX' USING QMGRNAME, CONNECTOPTS, HCONN, COMPCODE,
REASON.
```

宣告參數如下:

```
**   Name of queue manager
 01  QMGRNAME     PIC X(48).
**   Options that control the action of MQCONNX
 01  CONNECTOPTS.
     COPY CMQCNOV.
**   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 呼叫

```
call MQCONNX (QMgrName, ConnectOpts, Hconn, CompCode, Reason);
```

宣告參數如下:

```
dcl QMgrName     char(48);       /* Name of queue manager */
dcl ConnectOpts  like MQCNO;     /* Options that control the action of
                                    MQCONNX */
dcl Hconn        fixed bin(31);  /* Connection handle */
dcl CompCode     fixed bin(31);  /* Completion code */
dcl Reason       fixed bin(31);  /* Reason code qualifying CompCode */
```

High Level Assembler 呼叫

```
CALL MQCONNX,(QMGRNAME,CONNECTOPTS,HCONN,COMPCODE,REASON)
```

宣告參數如下:

```
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 呼叫

```
MQCONNX QMgrName, ConnectOpts, Hconn, CompCode, Reason
```

宣告參數如下:

```
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-建立訊息控點

MQCRTMH 呼叫會傳回訊息控點。

應用程式可以對後續的訊息佇列作業呼叫使用 MQCRTMH 呼叫:

- 使用 MQSETMP 呼叫來設定訊息控點的內容。
- 使用 MQINQMP 呼叫來查詢訊息控點的內容值。
- 使用 MQDLTMP 呼叫來刪除訊息控點的內容。

訊息控點可以在 MQPUT 及 MQPUT1 呼叫上使用，以將訊息控點的內容與所放置訊息的內容相關聯。 同樣地，透過在 MQGET 呼叫上指定訊息控點，可以在 MQGET 呼叫完成時使用訊息控點來存取所擷取訊息的內容。

使用 MQDLTMH 來刪除訊息控點。

語法

MQCRTMH (*Hconn*、 *CrtMsgHOpts*、 *Hmsg*、 *CompCode*、 *Reason*)

參數

Hconn

類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 前一個 MQCONN 或 MQCONNX 呼叫已傳回 *Hconn* 的值。 如果佇列管理程式的連線不再有效，且沒有任何 WebSphere MQ 呼叫在訊息控點上運作，則會隱含地呼叫 MQDLTMH 來刪除訊息。

或者，您可以指定下列值:

MQHC_UNASSOCITED_HCONN

連線控點不代表與任何特定佇列管理程式的連線。

使用此值時，必須以明確呼叫 MQDLTMH 來刪除訊息控點，才能釋放配置給它的任何儲存體; WebSphere MQ 絕不會隱含地刪除訊息控點。

在建立訊息控點的執行緒上建立的佇列管理程式必須至少有一個有效連線，否則呼叫會失敗，並產生 MQRC_HCONN_ERROR。

在單一系統上具有多個安裝的環境中， MQHC_UNASSOCITED_HCONN 值限制為與載入處理程序中的第一個安裝搭配使用。 如果將訊息控點提供給不同的安裝，則會傳回原因碼 MQRC_HMSG_NOT_AVAILABLE。

在 z/OS (適用於 CICS 應用程式) 及 IBM i (適用於以相容模式執行的應用程式) 上，可以省略 MQCONN 呼叫，且您可以對 *Hconn* 指定下列值:

MQHC_DEF_CONN
預設連線控點

CrtMsgHOpts
類型 :MQCMHO-輸入

控制 MQCRTMH 動作的選項。 如需詳細資料，請參閱 MQCMHO 。

Hmsg
類型 :MQHMSG-輸出

輸出時會傳回訊息控點，可用來設定、查詢及刪除訊息控點的內容。 最初訊息控點不包含任何內容。

訊息控點也有相關聯的訊息描述子。 一開始，這會包含預設值。 可以使用 MQSETMP 及 MQINQMP 呼叫來設定及查詢相關聯訊息描述子欄位的值。 MQDLTMP 呼叫會將訊息描述子的欄位重設回其預設值。

如果 *Hconn* 參數指定為值 MQHC_UNASSOCIED_HCONN ，則在 MQGET、MQPUT 或 MQPUT1 呼叫中可以使用所傳回的訊息控點，且在處理單元內具有任何連線，但一次只能由一個 WebSphere MQ 呼叫使用。 當第二個 WebSphere MQ 呼叫嘗試使用相同的訊息控點時，如果控點在使用中，第二個 WebSphere MQ 呼叫會失敗，原因碼為 MQRC_MSG_HANDLE_IN_USE。

如果 *Hconn* 參數不是 MQHC_UNASSOCIATED_HCONN ，則只能在指定的連線上使用傳回的訊息控點。

在使用此訊息控點的後續 MQI 呼叫中，必須使用相同的 *Hconn* 參數值:

- MQDLTMH
- MQSETMP
- MQINQMP
- MQDLTMP
- MQMHBUF
- MQBUFMH

對訊息控點發出 MQDLTMH 呼叫時，或定義控點範圍的處理單元終止時，傳回的訊息控點不再有效。 如果在建立訊息控點時提供特定連線，且佇列管理程式的連線不再有效 (例如，如果呼叫 MQDBC) ，則會隱含地呼叫 MQDLTMH。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204 , X'089C') 配接卡無法使用。

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130 , X'852 ') 無法載入配接卡服務模組。

MQRC_ASDID_MISMATCH
(2157 , X'86D') 主要和起始 ASID 不同。

MQRC_CALL_IN_PROGRESS
(2219 , X'08AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CMHO_ERROR
(2461 , X'099D') 建立訊息控點選項結構無效。

MQRC_CONNECTION_BROKEN
(2273 , X'7D9') 與佇列管理程式的連線遺失。

MQRC_HANDLE_NOT_AVAILABLE
(2017 , X'07E1') 沒有可用的控點。

MQRC_HCONN_ERROR
(2018 , X'7E2') 連線控點無效。

MQRC_HMSG_ERROR
(2460 , X'099C') 訊息控點指標無效。

MQRC_OPTIONS_ERROR
(2046 , X'07FE') 選項無效或不一致。

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'817 ') 可用的儲存體不足。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

如需這些代碼的詳細資訊，請參閱 原因碼。

C

```
MQCRTMH (Hconn, &CrtMsgHOpts, &Hmsg, &CompCode, &Reason);
```

宣告參數如下:

```
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, CRTMSGOPTS, HMSG, COMPCODE, REASON.
```

宣告參數如下:

```
**   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);
```

宣告參數如下:

```
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)
```

宣告參數如下:

```
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-控制回呼

MQCTL 呼叫會對回呼執行控制動作，並開啟物件控點進行連線。

語法

MQCTL (*Hconn*、 作業、 *ControlOpts*、 *CompCode*、 *Reason*)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 前一個 MQCONN 或 MQCONNX 呼叫已傳回 *Hconn* 的值。

在 z/OS (適用於 CICS 應用程式) 及 IBM i (適用於以相容模式執行的應用程式) 上，可以省略 MQCONN 呼叫，且您可以為 *Hconn* 指定下列特殊值:

MQHC_DEF_HCONN
預設連線控點。

operation
類型 :MQLONG-輸入

在針對指定物件控點定義的回呼上正在處理的作業。 您必須只指定下列其中一個選項:

MQOP_START

針對指定連線控點的所有已定義訊息消費者功能，開始耗用訊息。

在系統所啟動的執行緒上執行回呼，這與任何應用程式執行緒都不同。

這項作業可讓您控制提供給系統的連線控點。 除了消費者執行緒之外，可由其他執行緒發出的 MQI 呼叫只有下列:

- 具有作業 MQOP_STOP 的 MQCTL
- 具有作業 MQOP_SUSPEND 的 MQCTL
- MQDISC-在中斷 HConn 連線之前，執行 MQOP_STOP 作業的 MQCTL。

如果在啟動連線控點時發出 WebSphere MQ API 呼叫，且該呼叫並非源自訊息消費者函數，則會傳回 MQRC_HCONN_ASYNC_ACTIVE。

如果訊息消費者在 MQCBCT_START_CALL 期間停止連線，則 MQCTL 呼叫會傳回失敗原因碼 MQRC_CONNECTION_STOPPED。

這可以在消費者功能中發出。 對於與回呼常式相同的連線，其唯一目的是取消先前發出的 MQOP_STOP 作業。

在下列環境中不支援此選項: CICS on z/OS ，或者如果應用程式與無執行緒 WebSphere MQ 程式庫連結。

MQOP_START_WAIT
針對指定連線控點的所有已定義訊息消費者功能，開始耗用訊息。

在相同執行緒上執行的訊息消費者，在下列之前，不會將控制權傳回給 MQCTL 的呼叫端:

- 由使用 MQCTL MQOP_STOP 或 MQOP_SUSPEND 作業所釋放，或
- 已取消登錄或暫停所有消費者常式。

如果取消登錄或暫停所有消費者，則會發出隱含的 MQOP_STOP 作業。

這個選項無法從回呼常式內使用，可能是針對現行連線控點或任何其他連線控點。 如果嘗試呼叫，則會傳回 MQRC_ENVIRONMENT_ERROR。

如果在 MQOP_START_WAIT 作業期間隨時沒有已登錄、未暫停的消費者，則呼叫會失敗，原因碼為 MQRC_NO_CALLBACKS_ACTIVE。

如果在 MQOP_START_WAIT 作業期間暫停連線， MQCTL 呼叫會傳回警告原因碼 MQRC_CONNECTION_SUSPENDED; 連線會維持「已啟動」。

應用程式可以選擇發出 MQOP_STOP 或 MQOP_RESUME。 在此實例中， MQOP_RESUME 作業會封鎖。

單一執行緒用戶端不支援此選項。

MQ 停止
停止耗用訊息，並在此選項完成之前等待所有消費者完成其作業。 這項作業會釋放連線控點。

如果從回呼常式內發出，則在常式結束之前，此選項不會生效。 在已讀取訊息的消費者常式完成之後，以及對回呼常式發出停止呼叫 (如果要求的話) 之後，不再呼叫其他訊息消費者常式。

如果在回呼常式之外發出，則在訊息已讀取的消費者常式完成之前，以及在對回呼發出停止呼叫 (如果要求的話) 之後，控制不會回到呼叫程式。 不過，回呼本身仍會保持已登錄。

此功能對先讀訊息沒有影響。 您必須確定消費者從回呼函數內執行 MQCLOSE (MQCO_QUIESCE) ，以判斷是否有任何進一步的訊息可供遞送。

MQOP_SUSPEND
暫停耗用訊息。 這項作業會釋放連線控點。

這對讀取應用程式的訊息之前沒有任何影響。 如果您想要長時間停止耗用訊息，請考慮關閉佇列，並在繼續使用時重新開啟它。

如果從回呼常式內發出，則在常式結束之前不會生效。 在現行常式結束之後，將不再呼叫其他訊息消費者常式。

如果在回呼外部發出，則在現行消費者常式完成且不再呼叫之前，控制不會回到呼叫端。

MQ 作業 _ 回復
回復耗用訊息。

此選項通常是從主要應用程式執行緒發出，但也可以從回呼常式內使用它來取消在相同常式中發出的較早暫停要求。

如果使用 MQOP_RESUME 來回復 MQOP_START_WAIT，則作業會封鎖。

ControlOpts
類型 :MQCTLO-輸入

控制 MQCTL 動作的選項

如需結構的詳細資料，請參閱 第 298 頁的『MQCTLO-控制回呼選項結構』。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_WARNING
警告 (局部完成)。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_CONV_LOAD_ERROR
(2133， X'855 ') 無法載入資料轉換服務模組。

MQRC_ADAPTER_NOT_AVAILABLE
(2204， X'89C') 配接卡無法使用。

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130， X'852 ') 無法載入配接卡服務模組。

MQRC_API_EXIT_ERROR
(2374， X' 946 ') API 結束程式失敗。

MQRC_API_EXIT_LOAD_ERROR
(2183， X'887 ') 無法載入 API 結束程式。

MQRC_ASDID_MISMATCH
(2157， X'86D') 主要和起始 ASID 不同。

MQRC_BUFFER_LENGTH_ERROR
(2005， X'7D5') 緩衝區長度參數無效。

MQRC_CALLBACK_LINK_ERROR
(2487， X'9B7') 無法呼叫回呼常式

MQRC_CALLBACK_NOT_ 已登錄
(2448， X' 990 ') 無法取消登錄、暫停或回復，因為沒有已登錄的回呼

MQRC_CALLBACK_ROUTINE_ERROR
(2486， X'9B6') 已在 MQOP_REGISTER 呼叫上同時指定 CallbackFunction 和 CallbackName。

或者已指定 CallbackFunction 或 CallbackName，但不符合目前已登錄的回呼函數。

MQRC_CALLBACK_TYPE_ERROR
(2483， X'9B3') CallBack 類型欄位不正確。

MQRC_CALL_IN_PROGRESS
(2219， X'8AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CBD_ERROR
(2444， X'98C') 選項區塊不正確。

MQRC_CBD_OPTIONS_ERROR
(2484 , X'9B4') MQCBD 選項欄位不正確。

MQRC_CICS_WAIT_FAILED
(2140 , X'85C') 等待要求被 CICS 拒絕。

MQRC_CONNECTION_BROKEN
(2009 , X'7D9') 與佇列管理程式的連線遺失。

MQRC_CONNECTION_NOT_AUTHORIZED
(2217 , X'8A9') 未獲連線授權。

MQRC_CONNECTION_QUIESCING
(2202 , X'89A') 連線靜止。

MQRC_CONNECTION_STOPPING
(2203 , X'89B') 連線關閉。

MQRC_CORREL_ID_ERROR
(2207 , X'89F') 相關性 ID 錯誤。

MQRC_FUNCTION_NOT_SUPPORTED
(2298 , X'8FA') 在現行環境中無法使用所要求的功能。

MQRC_GET_INHIBITED
(2016 , X'7E0') 禁止佇列取得。

MQRC_GLOBAL_UOW_CONFLICT
(2351 , X'92F') 廣域工作單元衝突。

MQRC_GMO_ERROR
(2186 , X'88A') 取得訊息選項結構無效。

MQRC_HANDLE_IN_USE_FOR_UOW
(2353 , X' 931 ') 用於廣域工作單元的控點。

MQRC_HCONN_ERROR
(2018 , X'7E2') 連線控點無效。

MQRC_HOBJ_ERROR
(2019 , X'7E3') 物件控點無效。

MQRC_INCONSISTENT_BROWSE
(2259 , X'8D3') 不一致的瀏覽規格。

MQRC_INCONSISTENT_UOW
(2245 , X'8C5') 工作單元規格不一致。

MQRC_INVALID_MSG_UNDER_CURSOR
(2246 , X'8C6') 游標下的訊息不適用於擷取。

MQRC_LOCAL_UOW_CONFLICT
(2352 , X' 930 ') 廣域工作單元與本端工作單元衝突。

MQRC_MATCH_OPTIONS_ERROR
(2247 , X'8C7') 比對選項無效。

MQRC_MAX_MSG_LENGTH_ERROR
(2485 , X'9B5') 不正確 MaxMsg 長度欄位

MQRC_MD_ERROR
(2026 , X'7EA') 訊息描述子無效。

MQRC_MODULE_ENTRY_NOT_FOUND
(2497 , X'9C1') 在模組中找不到指定的函數進入點。

MQRC_MODULE_INVALID
(2496 , X'9C0') 找到模組，但其類型錯誤 (32 bit/64 位元) 或不是有效的 dll。

找不到 MQRC_MODULE_NOT_FOUND
(2495 , X'9BF') 在搜尋路徑中找不到模組或未獲授權載入。

MQRC_MSG_ID_ERROR
(2206 , X'89E') 訊息 ID 錯誤。

MQRC_MSG_SEQ_NUMBER_ERROR
(2250 , X'8CA') 訊息序號無效。

MQRC_MSG_TOKEN_ERROR
(2331 , X'91B') 使用訊息記號無效。

MQRC_NOT_OPEN_FOR_BROWSE
(2036 , X'7F4') 佇列未開啟以供瀏覽。

MQRC_NOT_OPEN_FOR_INPUT
(2037 , X'7F5') 佇列未開啟以供輸入。

已變更 MQRC_OBJECT_CHANGED
(2041 , X'7F9') 物件定義自開啟以來已變更。

MQRC_OBJECT_DAMAGED
(2101 , X'835 ') 物件已損壞。

MQRC_OPERATION_ERROR
(2488 , X'9B8') API 呼叫上的作業碼不正確

MQRC_OPTIONS_ERROR
(2046 , X'7FE') 選項無效或不一致。

MQRC_PAGESET_ERROR
(2193 , X'891 ') 存取頁集資料集時發生錯誤。

MQRC_Q_DELETED
(2052 , X'804 ') 已刪除佇列。

MQRC_Q_INDEX_TYPE_ERROR
(2394 , X'95A') 佇列具有錯誤索引類型。

MQRC_Q_MGR_NAME_ERROR
(2058 , X'80A') 佇列管理程式名稱無效或不明。

MQRC_Q_MGR_NOT_AVAILABLE
(2059 , X'80B') 佇列管理程式無法用於連線。

MQRC_Q_MGR_QUIESCING
(2161 , X'871 ') 佇列管理程式靜止中。

MQRC_Q_MGR_STOPPING
(2162 , X'872 ') 佇列管理程式關閉。

MQRC_RESOURCE_PROBLEM
(2102 , X'836 ') 可用的系統資源不足。

MQRC_SIGNAL_OUTSTANDING
(2069 , X'815 ') 此握把未發出信號。

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'817 ') 可用的儲存體不足。

MQRC_SUPPRESSED_BY_EXIT
(2109 , X'83D') 跳出程式暫停呼叫。

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072 , X'818 ') 無法使用同步點支援。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

MQRC_UOW_ENLISTMENT_ERROR
(2354 , X' 932 ') 在廣域工作單元中列入失敗。

不支援 MQRC_UOW_MIX_NOT_SUPPORTED
(2355 , X' 933 ') 不支援混合工作單元呼叫。

MQRC_UOW_NOT_AVAILABLE
(2255 , X'8CF') 佇列管理程式無法使用的工作單元。

MQRC_WAIT_INTERVAL_ERROR
(2090 , X'82A') MQGMO 中的等待間隔無效。

MQRC_WRONG_GMO_VERSION
(2256 , X'8D0') 提供的 MQGMO 版本錯誤。

MQRC_WRONG_MD_VERSION
(2257 , X'8D1') 提供的 MQMD 版本錯誤。

如需這些代碼的詳細資訊，請參閱 原因碼。

使用注意事項

1. 回呼常式必須檢查它們所呼叫之所有服務的回應，如果常式偵測到無法解決的狀況，則必須發出 MQCB MQOP_DEREGISTER 指令，以防止重複呼叫回呼常式。
2. 在 z/OS 上，當作業是 MQOP_START 時:
   - 必須授權使用非同步回呼常式的程式使用 z/OS UNIX 系統服務 (USS)。
   - 使用非同步回呼常式的 Language Environment (LE) 程式必須使用 LE 執行時期選項 POSIX(ON)。
   - 使用非同步回呼常式的非 LE 程式不得使用 USS pthread_create 介面 (可呼叫服務 BPX1PTC)。
3. IMS 配接器內不支援 MQCTL。

註: 在 CICS 中，不支援 MQOP_START。 請改用 MQOP_START_WAIT 函數呼叫。

C 呼叫

```
MQCTL (Hconn, Operation, &ControlOpts, &CompCode, &Reason)
```

宣告參數如下:

```
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 呼叫

```
CALL 'MQCTL' USING HCONN, OPERATION, CTLOPTS, COMPCODE, REASON.
```

宣告參數如下:

```
**   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 呼叫

```
call MQCTL(Hconn, Operation, CtlOpts, CompCode, Reason)
```

宣告參數如下:

```
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-切斷佇列管理程式連線

MQDISC 呼叫會岔斷佇列管理程式與應用程式之間的連線，且與 MQCONN 或 MQCONNX 呼叫相反。

- 在 z/OS 上，所有使用非同步訊息耗用、事件處理或回呼的應用程式，主要控制執行緒必須在結束之前發出 MQDISC 呼叫。 如需詳細資料，請參閱 WebSphere MQ 訊息的非同步使用 。
- 在 z/OS 上， CICS 應用程式不需要發出此呼叫來中斷與佇列管理程式的連線，但可能需要發出它來結束使用連線標籤。
- 在 IBM i 上，以相容模式執行的應用程式不需要發出此呼叫。 如需相關資訊，請參閱第 578 頁的『MQCONN-連接佇列管理程式』。

語法

MQDISC (*Hconn*， *CompCode*， *Reason*)

參數

Hconn
類型 :MQHCONN-輸入/輸出

此控點代表佇列管理程式的連線。 前一個 MQCONN 或 MQCONNX 呼叫已傳回 *Hconn* 的值。

在 z/OS for CICS 應用程式上，以及在 IBM i for 以相容模式執行的應用程式上，您可以省略 MQCONN 呼叫，並為 *Hconn* 指定下列值:

MQHC_DEF_HCONN
預設連線控點。

順利完成呼叫時，佇列管理程式會將 *Hconn* 設為對環境無效的控點值。 此值為:

MQHC_UNUSABLE_HCONN
無法使用連線控點。

在 z/OS 上， *Hconn* 設為未定義的值。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一個代碼:

MQCC_OK
順利完成。

MQCC_WARNING
警告 (局部完成)。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_WARNING:

MQRC_BACKED_OUT
(2003， X'7D3') 工作單元已取消。

未釋放 MQRC_CONN_TAG_NOT_RELEARED
(2344 , X' 928 ') 未釋放連線標籤。

MQRC_OUTCOME_PENDING
(2124 , X'84C') 確定作業的結果擱置中。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_DISC_LOAD_ERROR
(2138 , X'85A') 無法載入配接卡斷線模組。

MQRC_ADAPTER_NOT_AVAILABLE
(2204 , X'89C') 配接卡無法使用。

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130 , X'852 ') 無法載入配接卡服務模組。

MQRC_API_EXIT_ERROR
(2374 , X' 946 ') API 結束程式失敗。

MQRC_API_EXIT_INIT_ERROR
(2375 , X' 947 ') API 結束程式起始設定失敗。

MQRC_API_EXIT_TERM_ERROR
(2376 , X' 948 ') API 結束程式終止失敗。

MQRC_ASDID_MISMATCH
(2157 , X'86D') 主要和起始 ASID 不同。

MQRC_CALL_IN_PROGRESS
(2219 , X'8AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CONNECTION_BROKEN
(2009 , X'7D9') 與佇列管理程式的連線遺失。

MQRC_CONNECTION_STOPPING
(2203 , X'89B') 連線關閉。

MQRC_HCONN_ERROR
(2018 , X'7E2') 連線控點無效。

MQRC_OUTCOME_MIXED
(2123 , X'84B') 確定或取消作業的結果混合。

MQRC_PAGESET_ERROR
(2193 , X'891 ') 存取頁集資料集時發生錯誤。

MQRC_Q_MGR_NAME_ERROR
(2058 , X'80A') 佇列管理程式名稱無效或不明。

MQRC_Q_MGR_NOT_AVAILABLE
(2059 , X'80B') 佇列管理程式無法用於連線。

MQRC_Q_MGR_STOPPING
(2162 , X'872 ') 佇列管理程式關閉。

MQRC_RESOURCE_PROBLEM
(2102 , X'836 ') 可用的系統資源不足。

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'817 ') 可用的儲存體不足。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

如需這些代碼的詳細資訊，請參閱 原因碼。

使用注意事項

1. 當連線仍有在該連線下開啟的物件時，如果發出 MQDISC 呼叫，佇列管理程式會關閉那些物件，並將關閉選項設為 MQCO_NONE。

2. 如果應用程式以工作單元中未確定的變更結束，則那些變更的處置取決於應用程式結束的方式:
   a. 如果應用程式在結束之前發出 MQDISC 呼叫:
      - 對於佇列管理程式協調的工作單元，佇列管理程式會代表應用程式發出 MQCMIT 呼叫。 如果可能的話，會確定工作單元，如果沒有，則會取消。
      - 對於外部協調的工作單元，工作單元的狀態沒有變更; 不過，佇列管理程式通常指出當工作單元協調程式要求時，必須確定工作單元。

        在 z/OS 上， CICS、 IMS (批次 DL/1 程式除外) 及 RRS 應用程式都是如此。
   b. 如果應用程式正常結束，但未發出 MQDISC 呼叫，則所採取的動作視環境而定:
      - 在 z/OS 上，除了 MQ Java 或 MQ JMS 應用程式之外，還會發生附註 2a 中說明的動作。
      - 在所有其他情況下，會發生附註 2c 中說明的動作。

      由於環境之間的差異，請確定您要移轉的應用程式在結束之前確定或取消工作單元。
   c. 如果應用程式 異常 結束而未發出 MQDISC 呼叫，則會取消工作單元。
3. 在 z/OS 上， 下列要點適用:
   - CICS 應用程式不需要發出 MQDISC 呼叫來切斷與佇列管理程式的連線，因為 CICS 系統本身會連接至佇列管理程式，且 MQDISC 呼叫不會影響此連線。
   - CICS、 IMS (非批次 DL/1 程式) 及 RRS 應用程式使用由外部工作單元協調程式所協調的工作單元。 因此， MQDISC 呼叫不會影響發出呼叫時存在的工作單元 (如果有的話) 狀態。

     不過， MQDISC 呼叫 確實 指出應用程式發出的較早 MQCONNX 呼叫已結束使用與連線相關聯的連線標籤 *ConnTag* 。 當發出 MQDISC 呼叫時，如果有作用中的工作單元參照連線標籤，則呼叫會完成，完成碼為 MQCC_WARNING ，原因碼為 MQRC_CONN_TAG_NOT_RELEARED。 在外部工作單元協調程式解決工作單元之前，連線標籤無法重複使用。
4. 在 IBM i 上，以相容模式執行的應用程式不需要發出此呼叫; 如需詳細資料，請參閱 MQCONN 呼叫。

註: 在 CICS 中，不支援 MQOP_START。 請改用 MQOP_START_WAIT 函數呼叫。

C 呼叫

```
MQDISC (&Hconn, &CompCode, &Reason);
```

宣告參數如下:

```
MQHCONN  Hconn;     /* Connection handle */
MQLONG   CompCode;  /* Completion code */
MQLONG   Reason;    /* Reason code qualifying CompCode */
```

COBOL 呼叫

```
CALL 'MQDISC' USING HCONN, COMPCODE, REASON.
```

宣告參數如下:

```
**   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 呼叫

```
call MQDISC (Hconn, CompCode, Reason);
```

宣告參數如下:

```
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 組譯器呼叫

```
CALL MQDISC,(HCONN,COMPCODE,REASON)
```

宣告參數如下:

```
HCONN     DS  F  Connection handle
COMPCODE  DS  F  Completion code
REASON    DS  F  Reason code qualifying COMPCODE
```

Visual Basic 呼叫

```
MQDISC Hconn, CompCode, Reason
```

宣告參數如下:

```
Dim Hconn    As Long 'Connection handle'
Dim CompCode As Long 'Completion code'
Dim Reason   As Long 'Reason code qualifying CompCode'
```

MQDLTMH-刪除訊息控點

MQDLTMH 呼叫會刪除訊息控點，且與 MQCRTMH 呼叫相反。

語法

MQDLTMH (*Hconn*、 *Hmsg*、 *DltMsgHOpts*、 *CompCode*、 *Reason*)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。

此值必須符合用來建立 *Hmsg* 參數中所指定訊息控點的連線控點。

如果使用 MQHC_UNASSOCIATED_HCONN 建立訊息控點，則必須在刪除訊息控點的執行緒上建立有效連線，否則呼叫會失敗並產生 MQRC_CONNECTION_BROKEN。

Hmsg
類型 :MQHMSG-輸入/輸出

這是要刪除的訊息控點。 此值是由前一個 MQCRTMH 呼叫所傳回。

順利完成呼叫時，控點會設為環境的無效值。 此值為:

MQHM_UNUSABLE_HMSG
無法使用訊息控點。

如果另一個 WebSphere MQ 呼叫正在進行中，且已傳遞相同的訊息控點，則無法刪除訊息控點。

DltMsgHOpts
類型 :MQDMHO-輸入

請參閱第 314 頁的『MQDMHO-刪除訊息控點選項』，以取得詳細資料。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204 , X'089C') 配接卡無法使用。

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130 , X'852 ') 無法載入配接卡服務模組。

MQRC_ASDID_MISMATCH
(2157 , X'86D') 主要和起始 ASID 不同。

MQRC_CALL_IN_PROGRESS
(2219 , X'08AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CONNECTION_BROKEN
(2009 , X'07D9') 與佇列管理程式的連線遺失。

MQRC_DMHO_ERROR
(2462 , X'099E') 刪除訊息控點選項結構無效。

MQRC_HMSG_ERROR
(2460 , X'099C') 訊息控點指標無效。

MQRC_MSG_HANDLE_IN_USE
(2499 , X'09C3') 訊息控點已在使用中。

MQRC_OPTIONS_ERROR
(2046 , X'07FE') 選項無效或不一致。

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'817 ') 可用的儲存體不足。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

如需這些代碼的詳細資訊，請參閱 原因碼。

C 呼叫

```
MQDLTMH (Hconn, &Hmsg, &DltMsgHOpts, &CompCode, &Reason);
```

宣告參數如下:

```
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 呼叫

```
CALL 'MQDLTMH' USING HCONN, HMSG, DLTMSGOPTS, COMPCODE, REASON.
```

宣告參數如下:

```
** Connection handle
01  HCONN  PIC S9(9) BINARY.

** Options that control the action of MQDLTMH
01  DLTMSGHOPTS.
   COPY CMQDLMHOV.

** Completion code
01 COMPCODE  PIC S9(9) BINARY.

** Reason code qualifying COMPCODE
 01  REASON   PIC S9(9) BINARY.
```

PL/I 呼叫

```
call MQDLTMH (Hconn, Hmsg, DltMsgHOpts, CompCode, Reason);
```

宣告參數如下:

```
dcl Hconn       fixed bin(31);  /* Connection handle */
dcl Hmsg        fixed bin(63);  /* Message handle */
dcl DltMsgHOpts like MQDMHO;    /* Options that control the action of MQDLTMH */
dcl CompCode    fixed bin(31);  /* Completion code */
dcl Reason      fixed bin(31);  /* Reason code qualifying CompCode */
```

High Level Assembler 呼叫

```
CALL MQDLTMH,(HCONN,HMSG,DLTMSGHOPTS,COMPCODE,REASON)
```

宣告參數如下:

```
HCONN          DS        F  Connection handle
HMSG           DS        D  Message handle
DLTMSGHOPTS    CMQDMHOA  ,  Options that control the action of MQDLTMH
COMPCODE       DS        F  Completion code
REASON         DS        F  Reason code qualifying COMPCODE
```

MQDLTMP-刪除訊息內容

MQDLTMP 呼叫會從訊息控點中刪除內容，它是 MQSETMP 呼叫的反向。

語法

MQDLTMP (*Hconn*、 *Hmsg*、 *DltPropOpts*、 *Name*、 *CompCode*、 *Reason*)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 此值必須符合用來建立 *Hmsg* 參數中所指定訊息控點的連線控點。

如果訊息控點是使用 MQHC_UNASSOCIED_HCONN 建立的，則必須在刪除訊息控點的執行緒上建立有效連線，否則呼叫會因 MQRC_CONNECTION_BROKEN 而失敗。

Hmsg
類型 :MQHMSG-輸入

這是包含要刪除之內容的訊息控點。 此值是由前一個 MQCRTMH 呼叫所傳回。

***DltProp* 選項**
類型 :MQDMPO-輸入

如需詳細資料，請參閱 MQDMPO 資料類型。

name
類型 :MQCHARV-輸入

要刪除的內容名稱。 如需內容名稱的進一步相關資訊，請參閱 內容名稱 。

內容名稱中不接受萬用字元。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_WARNING
警告 (局部完成)。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_WARNING:

MQRC_PROPERTY_NOT_AVAILABLE
(2471 , X'09A7') 內容無法使用。

MQRC_RFH_FORMAT_ERROR
(2421 , X'0975 ') 無法剖析包含內容的 MQRFH2 資料夾。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204 , X'089C') 配接卡無法使用。

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130 , X'0852 ') 無法載入配接卡服務模組。

MQRC_ASDID_MISMATCH
(2157 , X'086D') 主要和起始 ASID 不同。

MQRC_CALL_IN_PROGRESS
(2219 , X'08AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CONNECTION_BROKEN
(2009 , X'07D9') 與佇列管理程式的連線遺失。

MQRC_DMPO_ERROR
(2481 , X'09B1') 刪除訊息內容選項結構無效。

MQRC_HMSG_ERROR
(2460 , X'099C') 訊息控點無效。

MQRC_MSG_HANDLE_IN_USE
(2499 , X'09C3') 訊息控點已在使用中。

MQRC_OPTIONS_ERROR
(2046 , X'07FE') 選項無效或不一致。

MQRC_PROPERTY_NAME_ERROR
(2442 , X'098A') 內容名稱無效。

MQRC_SOURCE_CCSID_ERROR
(2111 , X'083F') 內容名稱編碼字集 ID 無效。

MQRC_UNEXPECTED_ERROR
(2195 , X'0893 ') 發生非預期的錯誤。

如需這些代碼的詳細資訊，請參閱:

- WebSphere MQ for z/OS 的 原因碼
- 其他 WebSphere MQ 平台的 API 原因碼

.

C 呼叫

```
MQDLTMP (Hconn, Hmsg, &DltPropOpts, &Name, &CompCode, &Reason)
```

宣告參數如下:

```
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 呼叫

```
CALL 'MQDLTMP' USING HCONN, HMSG, DLTPROPOPTS, NAME, COMPCODE, REASON.
```

宣告參數如下:

```
**   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 呼叫

```
call MQDLTMP (Hconn, Hmsg, DltPropOpts, Name, CompCode, Reason);
```

宣告參數如下:

```
dcl Hconn       fixed bin(31); /* Connection handle */
dcl Hmsg        fixed bin(63); /* Message handle */
dcl DltPropOpts like MQDMPO;   /* Options that control the action of MQDLTMP */
dcl Name        like MQCHARV;  /* Property name */
dcl CompCode    fixed bin(31); /* Completion code */
dcl Reason      fixed bin(31); /* Reason code qualifying CompCode */
```

High Level Assembler 呼叫

```
CALL MQDLTMP,(HCONN,HMSG,DLTPROPOPTS,NAME,COMPCODE,REASON)
```

宣告參數如下:

```
HCONN        DS         F     Connection handle
HMSG         DS         D     Message handle
DLTPROPOPTS  CMQDMPOA   ,     Options that control the action of MQDLTMP
NAME         CMQCHRVA   ,     Property name
COMPCODE     DS         F     Completion code
REASON       DS         F     Reason code qualifying COMPCODE
```

MQGET-取得訊息

MQGET 呼叫會從已使用 MQOPEN 呼叫開啟的本端佇列中擷取訊息。

語法

MQGET (*Hconn*、 *Hobj*、 *MsgDesc*、 *GetMsgOpts*、 *BufferLength*、 *Buffer*、 *DataLength*、 *CompCode*、 *Reason*)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 前一個 MQCONN 或 MQCONNX 呼叫已傳回 *Hconn* 的值。

在 z/OS (適用於 CICS 應用程式) 及 IBM i (適用於以相容模式執行的應用程式) 上，可以省略 MQCONN 呼叫，並為 *Hconn* 指定下列值:

MQHC_DEF_HCONN
預設連線控點。

HOBJ
類型 :MQHOBJ-輸入

此控點代表要從中擷取訊息的佇列。 前一個 MQOPEN 呼叫傳回 *Hobj* 的值。 必須已使用下列一或多個選項開啟佇列 (如需詳細資料，請參閱 第 641 頁的『MQOPEN-開啟物件』):

- MQOO_INPUT_SHARED
- MQOO_INPUT_EXCLUSIVE
- MQOO_INPUT_AS_Q_DEF
- MQ 瀏覽

MsgDesc
類型 :MQMD-輸入/輸出

此結構說明所需訊息的屬性，以及所擷取訊息的屬性。 請參閱第 363 頁的『MQMD-訊息描述子』，以取得詳細資料。

如果 *BufferLength* 小於訊息長度，則佇列管理程式會填入 *MsgDesc* ，以決定是否在 *GetMsgOpts* 參數上指定 MQGMO_ACCEPT_TRUNCATED_MSG (請參閱 MQGMO-Options 欄位)。

如果應用程式提供 version-1 MQMD ，則傳回的訊息會在應用程式訊息資料之前加上 MQMDE ，但 只有在 MQMDE 中有一或多個欄位具有非預設值時。 如果 MQMDE 中的所有欄位都具有預設值，則會省略 MQMDE。 MQMD 中 格式 欄位的 MQFMT_MD_EXTENSION 格式名稱指出存在 MQMDE。

如果在 *MsgHandle* 欄位中提供有效的訊息控點，則應用程式不需要提供 MQMD 結構。 如果此欄位中未提供任何內容，則會從與訊息控點相關聯的描述子取得訊息的描述子。

如果應用程式提供訊息控點而非 MQMD 結構，並指定 MQGMO_PROPERTIES_FORCE_MQRFH2，則呼叫會失敗，原因碼為 MQRC_MD_ERROR。 如果應用程式未提供 MQMD 結構並指定 MQGMO_PROPERTIES_AS_Q_DEF ，且 *PropertyControl* 佇列屬性為 MQPROP_FORCE_MQRFH2，則呼叫也會失敗，原因碼為 MQRC_MD_ERROR。

如果已指定符合選項，且正在使用與訊息控點相關聯的訊息描述子，則用於比對的輸入欄位來自訊息控點。

***GetMsg* 選項**
類型 :MQGMO-輸入/輸出

請參閱第 323 頁的『MQGMO-取得訊息選項』，以取得詳細資料。

BufferLength
類型 :MQLONG-輸入

這是 *Buffer* 區域的長度 (以位元組為單位)。 對於沒有資料的訊息，或如果要從佇列中移除訊息並捨棄資料，請指定零 (在此情況下，您必須指定 MQGMO_ACCEPT_TRUNCATED_MSG)。

註: 可從佇列讀取的最長訊息長度由 *MaxMsgLength* 佇列屬性提供; 請參閱 第 731 頁的『佇列的屬性』。

緩衝區
類型: MQBYTExBuffer 長度-輸出

這是包含訊息資料的區域。 在適合訊息中資料本質的界限上對齊緩衝區。 4 位元組對齊方式適用於大部分訊息 (包括包含 IBM WebSphere MQ 標頭結構的訊息) ，但部分訊息可能需要更嚴格的對齊方式。 例如，包含 64 位元二進位整數的訊息可能需要 8 位元組對齊。

如果 *BufferLength* 小於訊息長度，則會將儘可能多的訊息移至 *Buffer*; 不論是否在 *GetMsgOpts* 參數上指定 MQGMO_ACCEPT_TRUNCATED_MSG ，都會發生此情況 (如需相關資訊，請參閱 MQGMO-選項欄位)。

Buffer 中資料的字集及編碼由 *MsgDesc* 參數中傳回的 *CodedCharSetId* 及 *Encoding* 欄位提供。 如果這些值與接收端所需的值不同，則接收端必須將應用程式訊息資料轉換為所需的字集及編碼。 可以使用 MQGMO_CONVERT 選項 (必要的話，搭配使用者撰寫的結束程式) 來轉換訊息資料; 如需此選項的詳細資料，請參閱 第 323 頁的『MQGMO-取得訊息選項』 。

註: MQGET 呼叫中的所有其他參數都採用本端佇列管理程式的字集及編碼 (由 *CodedCharSetId* 佇列管理程式屬性及 MQENC_NATIVE 提供)。

如果呼叫失敗，緩衝區內容可能仍會變更。

在 C 程式設計語言中，參數宣告為 void 的指標: 任何資料類型的位址都可以指定為參數。

如果 *BufferLength* 參數為零，則不會參照 *Buffer* ; 在此情況下，以 C 或 System/390 組合器撰寫的程式所傳遞的參數位址可以是空值。

DataLength
類型 :MQLONG-輸出

這是 訊息中應用程式資料的長度 (以位元組為單位)。 如果此值大於 *BufferLength*，則 *Buffer* 參數中只會傳回 *BufferLength* 個位元組 (即截斷訊息)。 如果值為零，則訊息不包含應用程式資料。

如果 *BufferLength* 小於訊息長度，則佇列管理程式仍會完成 *DataLength* ，是否在 *GetMsgOpts* 參數上指定 MQGMO_ACCEPT_TRUNCATED_MSG (如需相關資訊，請參閱 MQGMO-選項欄位)。 這可讓應用程式判斷容納訊息資料所需的緩衝區大小，然後以適當大小的緩衝區重新發出呼叫。

不過，如果指定 MQGMO_CONVERT 選項，且已轉換的訊息資料太長而無法放入 *Buffer* 中，則針對 *DataLength* 傳回的值為:

- 佇列管理程式定義格式的 未轉換 資料長度。

  在此情況下，如果資料本質導致在轉換期間擴充，則應用程式必須配置大於佇列管理程式針對 *DataLength* 所傳回值的緩衝區。
- 資料轉換結束程式針對應用程式定義的格式所傳回的值。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_WARNING
警告 (局部完成)。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

列出的原因碼是佇列管理程式可以針對 *Reason* 參數傳回的原因碼。 如果應用程式指定 MQGMO_CONVERT 選項，並且呼叫使用者撰寫的結束程式來轉換部分或所有訊息資料，則結束程式會決定針對 *Reason* 參數傳回的值。 因此，所記載的那些值以外的值是可能的。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_WARNING:

MQ RC_CONVERTED_MSG_TOO_Big
(2120 ， X'848 ') 轉換的資料對緩衝區而言太大。

MQRC_CONVERTED_STRING_TOO_BIG
(2190 ， X'88E') 轉換的字串對欄位而言太大。

MQRC_DBCS_ERROR
(2150 ， X'866 ') DBCS 字串無效。

MQRC_FORMAT_ERROR
(2110 ， X'83E') 訊息格式無效。

MQRC_INCOMPLETE_GROUP
(2241 ， X'8C1') 訊息群組不完整。

MQRC_INCOMPLETE_MSG
(2242 ， X'8C2') 邏輯訊息不完整。

MQRC_INCONSISTENT_CCSIDS
(2243 ， X'8C3') 訊息區段具有不同的 CCSID。

MQRC_INCONSISTENT_ENCODINGS
(2244 ， X'8C4') 訊息區段具有不同的編碼。

MQRC_INCONSISTENT_UOW
(2245 ， X'8C5') 工作單元規格不一致。

MQRC_MSG_TOKEN_ERROR
(2331 ， X'91B') 訊息記號的使用無效。

MQRC_NO_MSG_LOCKED
(2209 , X'8A1') 未鎖定訊息。

MQRC_NOT_CONVERTED
(2119 , X'847 ') 訊息資料未轉換。

MQRC_OPTIONS_CHANGED
(nnnn , X'xxx ') 已變更需要一致的選項。

MQRC_PARTIALLY_CONVERTED
(2272 , X'8E0') 訊息資料已部分轉換。

已接受 MQRC_SIGNAL_REQUEST_ACCEPTED
(2070 , X'816 ') 未傳回任何訊息 (但接受信號要求)。

MQRC_SOURCE_BUFFER_ERROR
(2145 , X'861 ') 來源緩衝區參數無效。

MQRC_SOURCE_CCSID_ERROR
(2111 , X'83F') 來源編碼字集 ID 無效。

MQRC_SOURCE_DECIMAL_ENC_ERROR
(2113 , X'841 ') 無法辨識訊息中的壓縮十進位編碼。

MQRC_SOURCE_FLOAT_ENC_ERROR
(2114 , X'842 ') 無法辨識訊息中的浮點數編碼。

MQRC_SOURCE_INTEGER_ENC_ERROR
(2112 , X'840 ') 無法辨識來源整數編碼。

MQRC_SOURCE_LENGTH_ERROR
(2143 , X'85F') 來源長度參數無效。

MQRC_TARGET_BUFFER_ERROR
(2146 , X'862 ') 目標緩衝區參數無效。

MQRC_TARGET_CCSID_ERROR
(2115 , X'843 ') 目標編碼字集 ID 無效。

MQRC_TARGET_DECIMAL_ENC_ERROR
(2117 , X'845 ') 接收器指定的壓縮十進位編碼無法辨識。

MQRC_TARGET_FLOAT_ENC_ERROR
(2118 , X'846 ') 接收器指定的浮點數編碼無法辨識。

MQRC_TARGET_INTEGER_ENC_ERROR
(2116 , X'844 ') 無法辨識目標整數編碼。

MQRC_TRUNCATED_MSG_ACCEPTED
(2079 , X'81F') 傳回截斷訊息 (處理完成)。

MQRC_TRUNCATED_MSG_FAILED
(2080 , X'820 ') 已傳回截斷訊息 (處理未完成)。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204 , X'89C') 配接卡無法使用。

MQRC_ADAPTER_CONV_LOAD_ERROR
(2133 , X'855 ') 無法載入資料轉換服務模組。

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130 , X'852 ') 無法載入配接卡服務模組。

MQRC_API_EXIT_ERROR
(2374 , X' 946 ') API 結束程式失敗。

MQRC_API_EXIT_LOAD_ERROR
(2183 , X'887 ') 無法載入 API 結束程式。

MQRC_ASDID_MISMATCH
(2157 , X'86D') 主要和起始 ASID 不同。

MQRC_BACKED_OUT
(2003 , X'7D3') 工作單元已取消。

MQRC_BUFFER_ERROR
(2004 , X'7D4') 緩衝區參數無效。

MQRC_BUFFER_LENGTH_ERROR
(2005 , X'7D5') 緩衝區長度參數無效。

MQRC_CALL_IN_PROGRESS
(2219 , X'8AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CF_STRUC_FAILED
(2373 , X' 945 ') 連結機能結構失敗。

MQRC_CF_STRUC_IN_USE
(2346 , X'92A') 連結機能結構使用中。

MQRC_CF_STRUC_LIST_HDR_IN_USE
(2347 , X'92B') 連結機能結構清單標頭使用中。

MQRC_CICS_WAIT_FAILED
(2140 , X'85C') 等待要求被 CICS 拒絕。

MQRC_CONNECTION_BROKEN
(2009 , X'7D9') 與佇列管理程式的連線遺失。

MQRC_CONNECTION_NOT_AUTHORIZED
(2217 , X'8A9') 未獲連線授權。

MQRC_CONNECTION_QUIESCING
(2202 , X'89A') 連線靜止。

MQRC_CONNECTION_STOPPING
(2203 , X'89B') 連線關閉。

MQRC_CORREL_ID_ERROR
(2207 , X'89F') 相關性 ID 錯誤。

MQRC_DATA_LENGTH_ERROR
(2010 , X'7DA') 資料長度參數無效。

MQRC_DB2_NOT_AVAILABLE
(2342 , X' 926 ') Db2 子系統無法使用。

MQRC_GET_INHIBITED
(2016 , X'7E0') 禁止佇列取得。

MQRC_GLOBAL_UOW_CONFLICT
(2351 , X'92F') 廣域工作單元衝突。

MQRC_GMO_ERROR
(2186 , X'88A') 取得訊息選項結構無效。

MQRC_HANDLE_IN_USE_FOR_UOW
(2353 , X' 931 ') 用於廣域工作單元的控點。

MQRC_HCONN_ERROR
(2018 , X'7E2') 連線控點無效。

MQRC_HOBJ_ERROR
(2019 , X'7E3') 物件控點無效。

MQRC_INCONSISTENT_BROWSE
(2259 , X'8D3') 不一致的瀏覽規格。

MQRC_INCONSISTENT_UOW
(2245 , X'8C5') 工作單元規格不一致。

MQRC_INVALID_MSG_UNDER_CURSOR
(2246 , X'8C6') 游標下的訊息不適用於擷取。

MQRC_LOCAL_UOW_CONFLICT
(2352 , X' 930 ') 廣域工作單元與本端工作單元衝突。

MQRC_MATCH_OPTIONS_ERROR
(2247 , X'8C7') 比對選項無效。

MQRC_MD_ERROR
(2026 , X'7EA') 訊息描述子無效。

MQRC_MSG_ID_ERROR
(2206 , X'89E') 訊息 ID 錯誤。

MQRC_MSG_SEQ_NUMBER_ERROR
(2250 , X'8CA') 訊息序號無效。

MQRC_MSG_TOKEN_ERROR
(2331 , X'91B') 使用訊息記號無效。

MQRC_NO_MSG_AVAILABLE
(2033 , X'7F1') 沒有可用的訊息。

MQRC_NO_MSG_UNDER_CURSOR
(2034 , X'7F2') 瀏覽游標未定位在訊息上。

MQRC_NOT_OPEN_FOR_BROWSE
(2036 , X'7F4') 佇列未開啟以供瀏覽。

MQRC_NOT_OPEN_FOR_INPUT
(2037 , X'7F5') 佇列未開啟以供輸入。

已變更 MQRC_OBJECT_CHANGED
(2041 , X'7F9') 物件定義自開啟以來已變更。

MQRC_OBJECT_DAMAGED
(2101 , X'835 ') 物件已損壞。

MQRC_OPTIONS_ERROR
(2046 , X'7FE') 選項無效或不一致。

MQRC_PAGESET_ERROR
(2193 , X'891 ') 存取頁集資料集時發生錯誤。

MQRC_Q_DELETED
(2052 , X'804 ') 已刪除佇列。

MQRC_Q_INDEX_TYPE_ERROR
(2394 , X'95A') 佇列具有錯誤索引類型。

MQRC_Q_MGR_NAME_ERROR
(2058 , X'80A') 佇列管理程式名稱無效或不明。

MQRC_Q_MGR_NOT_AVAILABLE
(2059 , X'80B') 佇列管理程式無法用於連線。

MQRC_Q_MGR_QUIESCING
(2161 , X'871 ') 佇列管理程式靜止中。

MQRC_Q_MGR_STOPPING
(2162 , X'872 ') 佇列管理程式關閉。

MQRC_RESOURCE_PROBLEM
(2102 , X'836 ') 可用的系統資源不足。

不接受 MQRC_SECOND_MARK_NOT_ALLOWED
(2062 , X'80E') 已標示訊息。

MQRC_SIGNAL_OUTSTANDING
(2069 , X'815 ') 此握把未發出信號。

MQRC_SIGNAL1_ERROR
(2099 , X'833 ') 信號欄位無效。

MQRC_STORAGE_MEDIUM_FULL
(2192 , X'890 ') 外部儲存媒體已滿。

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'817 ') 可用的儲存體不足。

MQRC_SUPPRESSED_BY_EXIT
(2109 , X'83D') 跳出程式暫停呼叫。

MQRC_SYNCPOINT_LIMIT_REACHED
(2024 , X'7E8') 在現行工作單元內無法處理更多訊息。

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072 , X'818 ') 同步點支援無法使用。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

MQRC_UOW_ENLISTMENT_ERROR
(2354 , X' 932 ') 在廣域工作單元中列入失敗。

不支援 MQRC_UOW_MIX_NOT_SUPPORTED
(2355 , X' 933 ') 不支援混合工作單元呼叫。

MQRC_UOW_NOT_AVAILABLE
(2255 , X'8CF') 佇列管理程式無法使用的工作單元。

MQRC_WAIT_INTERVAL_ERROR
(2090 , X'82A') MQGMO 中的等待間隔無效。

MQRC_WRONG_GMO_VERSION
(2256 , X'8D0') 提供的 MQGMO 版本錯誤。

MQRC_WRONG_MD_VERSION
(2257 , X'8D1') 提供的 MQMD 版本錯誤。

如需這些代碼的詳細資訊，請參閱 原因碼。

使用注意事項

1. 通常會從佇列中刪除擷取的訊息。 這項刪除可能是 MQGET 呼叫本身的一部分，或同步點的一部分。

   瀏覽選項為 :MQGMO_BROWSE_FIRST、MQGMO_BROWSE_NEXT 及 MQGMO_BROWSE_MSG_UNDER_CURSOR。

2. 如果 MQGMO_LOCK 選項指定了其中一個瀏覽選項，則會鎖定已瀏覽的訊息，以便只有此控點可以看見它。

   如果指定 MQGMO_UNLOCK 選項，則會解除鎖定先前鎖定的訊息。 在此情況下不會擷取任何訊息，且不會檢查或變更 *MsgDesc*、 *BufferLength*、 *Buffer* 及 *DataLength* 參數。

3. 對於發出 MQGET 呼叫的應用程式，如果應用程式異常終止或在處理呼叫時中斷連線，則擷取的訊息可能會遺失。 發生此問題的原因是代理程式與代表應用程式發出 MQGET 呼叫的佇列管理程式在相同平台上執行，無法偵測到應用程式遺失，直到代理程式將訊息從佇列中移除 之後 ，才會將訊息傳回給應用程式。 持續訊息和非持續訊息都可能發生此問題。

   為了避免以這種方式遺失訊息的風險，請一律擷取工作單元內的訊息。 亦即，在 MQGET 呼叫上指定 MQGMO_SYNCPOINT 選項，並在訊息處理完成時使用 MQCMIT 或 MQBACK 呼叫來確定或取消工作單元。 如果指定 MQGMO_SYNCPOINT ，且用戶端異常終止或連線中斷，代理程式會取消佇列管理程式上的工作單元，並在佇列上恢復訊息。 如需同步點的相關資訊，請參閱 WebSphere MQ 應用程式中的同步點考量。

   IBM WebSphere MQ 用戶端以及與佇列管理程式在相同平台上執行的應用程式可能會發生這種狀況。

4. 如果應用程式將一連串訊息放置在特定單一工作單元內的佇列，然後順利確定該工作單元，訊息會變成可供擷取，如下所示:

   - 如果佇列是 非共用 佇列 (亦即，本端佇列) ，則工作單元內的所有訊息會同時變成可用。
   - 如果佇列是 共用 佇列，則工作單元內的訊息會依其放置順序變成可用，但並非全部同時。 當系統負載繁重時，可以順利擷取工作單元中的第一個訊息，但工作單元中的第二個或後續訊息的 MQGET 呼叫可能會失敗，因為 MQRC_NO_MSG_AVAILABLE。 如果發生此問題，應用程式必須稍待片刻，然後重試作業。

5. 如果應用程式將一連串訊息放置在相同佇列上，而不使用訊息群組，如果滿足特定條件，則會保留這些訊息的順序。 如需詳細資料，請參閱 MQPUT 使用注意事項 。 如果滿足條件，在下列情況下，訊息會以傳送順序呈現給接收端應用程式:

   - 只有一個接收端從佇列取得訊息。

     如果有兩個以上應用程式從佇列取得訊息，則它們必須同意傳送端用來識別屬於某個序列之訊息的機制。 例如，寄件者可能會將序列中訊息的所有 *CorrelId* 欄位設定為該訊息序列所特有的值。
   - 接收端不會故意變更擷取順序，例如指定特定的 *MsgId* 或 *CorrelId*。

   如果傳送端應用程式將訊息當作訊息群組來放置，則當接收端應用程式在 MQGET 呼叫上指定 MQGMO_LOGICAL_ORDER 選項時，訊息會以正確的順序呈現給接收端應用程式。 如需訊息群組的相關資訊，請參閱:

   - MQMD- MsgFlags 欄位
   - MQPMO_LOGICAL_ORDER
   - MQGMO_LOGICAL_ORDER

   如果使用者在同步點下取得群組中的訊息，則必須先確保處理完整群組，然後再嘗試完成交易。
6. 應用程式必須在 *MsgDesc* 參數的 *Feedback* 欄位中測試回饋碼 MQFB_QUIT ，並在找到此值時結束。 如需相關資訊，請參閱 MQMD-Feedback 欄位 。
7. 如果以 MQOO_SAVE_ALL_CONTEXT 選項開啟了 *Hobj* 所識別的佇列，且來自 MQGET 呼叫的完成碼是 MQCC_OK 或 MQCC_WARNING ，則與佇列控點 *Hobj* 相關聯的環境定義會設為已擷取之訊息的環境定義 (除非設定 MQGMO_BROWSE_FIRST、MQGMO_BROWSE_NEXT 或 MQGMO_BROWSE_MSG_UNDER_CURSOR 選項，在此情況下會將環境定義標示為無法使用)。

   您可以透過指定 MQPMO_PASS_IDENTITY_CONTEXT 或 MQPMO_PASS_ALL_CONTEXT 選項，在後續 MQPUT 或 MQPUT1 呼叫上使用已儲存的環境定義。 這可讓所接收訊息的環境定義完整或部分傳送至另一個訊息 (例如，當訊息轉遞至另一個佇列時)。 如需訊息環境定義的相關資訊，請參閱 訊息環境定義。
8. 如果您在 *GetMsgOpts* 參數中包括 MQGMO_CONVERT 選項，則在 *Buffer* 參數中放置資料之前，會將應用程式訊息資料轉換為接收端應用程式所要求的表示法:

   - 訊息中控制資訊的 *Format* 欄位會識別應用程式資料結構，訊息中控制資訊的 *CodedCharSetId* 及 *Encoding* 欄位會指定其字集 ID 及編碼。
   - 發出 MQGET 呼叫的應用程式在 *MsgDesc* 參數的 *CodedCharSetId* 及 *Encoding* 欄位中指定要將應用程式訊息資料轉換成的字集 ID 及編碼。

   當需要轉換訊息資料時，根據訊息中控制資訊的 *Format* 欄位值，由佇列管理程式本身或使用者撰寫的結束程式執行轉換:

   - 下列格式名稱是佇列管理程式所轉換的格式; 這些格式稱為「內建」格式:
     - MQFMT_ADMIN
     - MQFMT_CICS (僅限 z/OS)
     - MQFMT_COMMAND_1
     - MQFMT_COMMAND_2
     - MQFMT_DEAD_LETTER_HEADER
     - MQFMT_DIST_HEADER
     - MQFMT_EVENT 第 1 版
     - MQFMT_EVENT 第 2 版 (僅限 z/OS)
     - 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 (僅限 z/OS)
- MQFMT_XMIT_Q_HEADER

- 格式名稱 MQFMT_NONE 是特殊值，指出訊息中資料的本質未定義。 因此，當從佇列擷取訊息時，佇列管理程式不會嘗試轉換。

  註: 如果 MQGET 呼叫中指定 MQGMO_CONVERT 給格式名稱為 MQFMT_NONE 的訊息，且訊息的字集或編碼不同於 *MsgDesc* 參數中指定的，則會在 *Buffer* 參數中傳回訊息 (假設沒有其他錯誤) ，但呼叫會完成，完成碼為 MQCC_WARNING ，原因碼為 MQRC_FORMAT_ERROR。

  當訊息資料的本質表示它不需要轉換時，或當傳送及接收應用程式已彼此同意傳送訊息資料的表單時，您可以使用 MQFMT_NONE。
- 所有其他格式名稱都會將訊息傳遞至使用者撰寫的結束程式，以進行轉換。 除了環境特定的新增項目之外，該結束程式還具有與格式相同的名稱。 使用者指定的格式名稱不得以字母 WebSphere MQ 開頭。

  如需資料轉換結束程式的詳細資料，請參閱 第 791 頁的『資料轉換結束程式』 。

訊息中的使用者資料可以在任何支援的字集與編碼之間轉換。 不過，請注意，如果訊息包含一個以上 WebSphere MQ 標頭結構，則對於佇列名稱中有效的任何字元，無法將訊息轉換為具有雙位元組或多位元組字元的字集，也無法將訊息轉換為具有雙位元組或多位元組字元的字集。 如果嘗試這樣做，則會產生原因碼 MQRC_SOURCE_CCSID_ERROR 或 MQRC_TARGET_CCSID_ERROR ，且會傳回未轉換的訊息。 Unicode 字集 UCS-2 是這類字集的範例。

從 MQGET 傳回時， 下列原因碼指出已順利轉換訊息:

- MQRC_NONE

下列原因碼指出 可能 已順利轉換訊息; 應用程式必須檢查 *MsgDesc* 參數中的 *CodedCharSetId* 及 *Encoding* 欄位，才能找出:

- MQRC_TRUNCATED_MSG_ACCEPTED

所有其他原因碼都指出未轉換訊息。

註: 對於使用者撰寫的結束程式所執行的轉換， 只有 在結束程式符合 第 791 頁的『資料轉換結束程式』 中說明的處理準則時，才會解譯此原因碼。

9. 使用物件導向介面來取得訊息時，您可以選擇不指定緩衝區來保留 MQGET 呼叫的訊息資料。 不過，在舊版 WebSphere MQ 中， MQGET 可能會失敗，原因碼為 MQRC_CONVERTED_MSG_TO_BIG ，即使未指定緩衝區也一樣。 在 WebSphere MQ 第 7 版中，當您使用物件導向應用程式取得訊息而未限制接收訊息緩衝區的大小時，應用程式不會因 MQRC_CONVERTED_MSG_TOO_BIG 而失敗，並接收已轉換的訊息。 這適用於下列環境:

   - .NET ，包括完全受管理的應用程式
   - C++
   - Java (WebSphere MQ for Java 類別)

   註: 對於所有用戶端，如果 *sharingConversations* 的值為零，則通道會像 WebSphere MQ 7.0 之前一樣運作，且訊息處理會回復為第 6 版行為。 在此狀況下，如果緩衝區太小而無法接收已轉換的訊息，則會傳回未轉換的訊息，原因碼為 MQRC_CONVERTED_MSG_TOO_BIG。 如需 *sharingConversations* 的相關資訊，請參閱 在用戶端應用程式中使用共用交談。
10. 對於內建格式，當指定 MQGMO_CONVERT 選項時，佇列管理程式可以在訊息中執行字串的 預設轉換 。 預設轉換可讓佇列管理程式在轉換字串資料時使用安裝指定的預設字集，其近似實際字集。 因此， MQGET 呼叫可以順利完成，完成碼為 MQCC_OK ，而不是完成 MQCC_WARNING 及原因碼 MQRC_SOURCE_CCSID_ERROR 或 MQRC_TARGET_CCSID_ERROR。

    註: 使用近似字集來轉換字串資料的結果是部分字元可能轉換不正確。 若要避免此情況，請在字串中使用實際字集及預設字集共用的字元。

預設轉換同時適用於應用程式訊息資料及 MQMD 和 MQMDE 結構中的字元欄位:

- 只有在 全部 符合下列條件時，才會進行應用程式訊息資料的預設轉換:
  - 應用程式指定 MQGMO_CONVERT。
  - 訊息包含必須從不支援的字集轉換或轉換成不支援的字集的資料。
  - 安裝或重新啟動佇列管理程式時已啟用預設轉換。
- 如果佇列管理程式已啟用預設轉換，則 MQMD 及 MQMDE 結構中字元欄位的預設轉換會依需要進行。即使 MQGET 呼叫上的應用程式未指定 MQGMO_CONVERT 選項，也會執行轉換。

11. 對於 Visual Basic 程式設計語言，下列要點適用:
    - 如果 *Buffer* 參數的大小小於 *BufferLength* 參數指定的長度，則呼叫會失敗，原因碼為 MQRC_STORAGE_NOT_AVAILABLE。
    - *Buffer* 參數宣告為 String 類型。 如果要從佇列擷取的資料不是 String 類型，請使用 MQGETAny 呼叫取代 MQGET。

      MQGETAny 呼叫具有與 MQGET 呼叫相同的參數，但 *Buffer* 參數宣告為類型 Any，容許擷取任何類型的資料。 不過，這表示無法檢查 *Buffer* ，以確定其大小至少為 *BufferLength* 個位元組。
12. 當啟用先讀時，並非所有 MQGET 選項都受支援。 下表指出容許哪些選項，以及在 MQGET 呼叫之間是否可以變更這些選項。

表 566: 啟用先讀時允許 *MQGET* 選項

	已啟用先讀且可以在 MQGET 呼叫之間變更時允許	已啟用先讀但無法在 MQGET 呼叫之間變更 [a] 時允許	啟用先讀 [b] 時不允許的 MQGET 選項
MQGET MD 值	MsgId[c] CorrelId[c]	編碼 CodedCharSetId	
MQGET MQGMO 選項	MQGMO_WAIT MQGMO_NO_WAIT MQGMO_FAIL_IF_QUIESCING MQGMO_BROWSE_FIRST[d] MQGMO_BROWSE_NEXT[d] MQGMO_BROWSE_MESSAGE _UNDER_CURSOR[d]	MQGMO_SYNCPOINT_IF_PERSISTENT MQGMO_NO_SYNCPOINT MQGMO_ACCEPT_TRUNCATED_MSG MQ GMO_CONVERT MQ GMO_LOGICAL_ORDER MQ GMO_COMPLETE_MSG MQGMO_ALL_MSGS_AVAILABLE MQGMO_ALL_SEGMENTS_AVAILABLE MQ GMO_MARK_BROWSE_HANDLE MQGMO_MARK_BROWSE_CO_OP MQGMO_UNMARK_BROWSE_CO_OP MQ GMO_UNMARK_BROWSE_HANDLE MQGMO_UNMARKED_BROWSE_MSG MQGMO_PROPERTIES_FORCE_MQRFH2 MQGMO_NO_PROPERTIES MQGMO_PROPERTIES_IN_HANDLE MQGMO_PROPERTIES_COMPATIBILITY	MQGMO_SET_SIGNAL MQ GMO_同步點 MQGMO_MARK_SKIP _BACKOUT MQ GMO_MSG_UNDER _CURSOR[d] MQGMO_LOCK MQGMO_UNLOCK
MQGMO 值		MsgHandle	

    a. 如果在 MQGET 呼叫之間變更了這些選項，則會傳回 MQRC_OPTIONS_CHANGED 原因碼。

    b. 如果在第一個 MQGET 呼叫上指定這些選項，則會停用先讀功能。 如果在後續的 MQGET 呼叫上指定了這些選項，則會傳回原因碼 MQRC_OPTIONS_ERROR。

    c. 用戶端應用程式需要意識到，如果 MsgId 和 CorrelId 值在 MQGET 呼叫之間變更，則具有先前值的訊息可能已經傳送至用戶端，並保留在用戶端先讀緩衝區中，直到被取用（或自動清除）為止。

    d. 第一個 MQGET 呼叫決定在啟用先讀功能時，是否要從佇列中瀏覽或取得訊息。 如果應用程式嘗試同時執行瀏覽與取得動作，將傳回原因碼 MQRC_OPTIONS_CHANGED。

    e. MQGMO_MSG_UNDER_CURSOR 不能與先讀功能一起使用。 在啟用先讀功能之後，可瀏覽或取得訊息，但不能同時執行這兩項動作。

13. 只有在那些訊息與 get 位於相同的本端工作單元時，應用程式才能破壞性地取得未確定的訊息。 應用程式無法以非破壞性方式取得未確定的訊息。
14. 瀏覽游標下的訊息可以在工作單元中擷取。 無法以這種方式擷取未確定的訊息。

C 呼叫

```
MQGET (Hconn, Hobj, &MsgDesc, &GetMsgOpts, BufferLength, Buffer,
       &DataLength, &CompCode, &Reason);
```

宣告參數如下:

```
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 呼叫

```
CALL 'MQGET' USING HCONN, HOBJ, MSGDESC, GETMSGOPTS, BUFFERLENGTH,
BUFFER, DATALENGTH, COMPCODE, REASON.
```

宣告參數如下:

```
**   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 CMQGMOV.
**   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 呼叫

```
call MQGET (Hconn, Hobj, MsgDesc, GetMsgOpts, BufferLength, Buffer,
           DataLength, CompCode, Reason);
```

宣告參數如下:

```
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 */
```

High Level Assembler 呼叫

```
         CALL MQGET,(HCONN,HOBJ,MSGDESC,GETMSGOPTS,BUFFERLENGTH,
               BUFFER,DATALENGTH,COMPCODE,REASON)
```

宣告參數如下:

```
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 呼叫

```
MQGET Hconn, Hobj, MsgDesc, GetMsgOpts, BufferLength, Buffer,
DataLength, CompCode, Reason
```

宣告參數如下:

```
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-查詢物件屬性

MQINQ 呼叫會傳回整數陣列及一組包含物件屬性的字串。

有效的物件類型如下:

- 佇列管理程式
- 佇列
- 名稱清單
- 程序定義

語法

MQINQ (*Hconn*、 *Hobj*、 *SelectorCount*、 選取元、 *IntAttrCount*、 *IntAttrs*、 *CharAttr* 長度、 *CharAttrs*、 *CompCode*、 *Reason*)

參數

Hconn
類型: MQHCONN -輸入

此控點代表佇列管理程式的連線。 前一個 MQCONN 或 MQCONNX 呼叫已傳回 *Hconn* 的值。

在 z/OS for CICS 應用程式上，以及在 IBM i for 以相容模式執行的應用程式上，可以省略 MQCONN 呼叫，並為 *Hconn* 指定下列值:

MQHC_DEF_HCONN
預設連線控點。

HOBJ
類型: MQHOBJ -輸入

此控點代表具有必要屬性的物件 (任何類型)。 控點必須由前一個指定 MQOO_INQUIRE 選項的 MQOPEN 呼叫傳回。

SelectorCount
類型: MQLONG -輸入

這是 *Selectors* 陣列中提供的選取元計數。 它是要傳回的屬性數目。 零是有效值。 容許的數目上限為 256。

選取器
類型: MQLONG × *SelectorCount* -輸入

這是 *SelectorCount* 屬性選取元的陣列; 每一個選取元都會識別具有必要值的屬性 (整數或字元)。

每一個選取器都必須適用於 *Hobj* 所代表的物件類型，否則呼叫會失敗，並產生完成碼 MQCC_FAILED 和原因碼 MQRC_SELECTOR_ERROR。

在佇列的特殊情況下:

- 如果選取元對任何類型的佇列無效，則呼叫會失敗，並顯示完成碼 MQCC_FAILED 及原因碼 MQRC_SELECTOR_ERROR。
- 如果選取元僅適用於類型不是物件類型的佇列，則呼叫會成功，並具有完成碼 MQCC_WARNING 及原因碼 MQRC_SELECTOR_NOT_FOR_TYPE。
- 如果要查詢的佇列是叢集佇列，則有效的選取元視佇列解析方式而定; 如需進一步詳細資料，請參閱 第 630 頁的『使用注意事項』 。

您可以按任何順序指定選取元。 對應於整數屬性選取元 (MQIA_* 選取元) 的屬性值會以這些選取元在 *Selectors* 中出現的相同順序在 *IntAttrs* 中傳回。 對應於字元屬性選取元 (MQCA_* 選取元) 的屬性值，會以這些選取元出現的相同順序在 *CharAttrs* 中傳回。 MQIA_* 選取元可以與 MQCA_* 選取元交錯; 只有每一種類型內的相對順序才重要。

註:

1. 整數和字元屬性選取元配置在兩個不同的範圍內; MQIA_* 選取元位於 MQIA_FIRST 至 MQIA_LAST 範圍內， MQCA_* 選取元位於 MQCA_FIRST 至 MQCA_LAST 範圍內。

   對於每一個範圍，常數 MQIA_LAST_USED 及 MQCA_LAST_USED 會定義佇列管理程式可接受的最高值。
2. 如果所有 MQIA_* 選取元都先出現，則可以使用相同的元素號碼來處理 *Selectors* 和 *IntAttrs* 陣列中對應的元素。
3. 如果 *SelectorCount* 參數為零，則不會參照 *Selectors* 。 在此情況下， 以 C 或 S/390 組譯器撰寫的程式所傳遞的參數位址可能是空值。

下表列出可查詢的屬性。 對於 MQCA_* 選取元，括弧中提供了常數，用於定義 *CharAttrs* 中所產生字串的長度 (以位元組為單位)。

後面的表格按物件的字母順序列出選取元，如下所示:

- 佇列的 第 619 頁的表 567 MQINQ 屬性選取器
- 名稱清單的 第 622 頁的表 568 MQINQ 屬性選取器
- 程序定義的 第 622 頁的表 569 MQINQ 屬性選取器
- 佇列管理程式的 第 623 頁的表 570 MQINQ 屬性選取器

所有 IBM WebSphere MQ 平台都支援所有選取元，但 **附註** 直欄中指出的除外，如下所示:

不 z/OS
在 **除** z/OS 以外的所有平台上受支援

z/OS
僅 在 z/OS 上受支援

表 *567:* 佇列的 *MQINQ* 屬性選取器

選取元	欄位長度	說明	附註
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	最近變更的日期	
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	最近變更的時間	

表 567: 佇列的 *MQINQ* 屬性選取器 (繼續)

選取元	欄位長度	說明	附註
MQCA_BACKOUT_REQ_Q_NAME	MQ_Q_NAME_LENGTH	取消重新排入佇列的名稱過多	
MQCA_BASE_Q_NAME	MQ_Q_NAME_LENGTH	別名解析成的佇列名稱	
MQCA_CF_STRUC_NAME	MQ_CF_STRUC_NAME_LENGTH	連結機能結構名稱	z/OS
MQCA_CLUS_CHL_NAME	MQ_CHANNEL_NAME_LENGTH	使用此佇列作為傳輸佇列的叢集傳送端通道名稱。	不 z/OS
MQCA_CLUSTER_NAME	MQ_CLUSTER_NAME_LENGTH	叢集名稱	
MQCA_CLUSTER_NAMELIST	MQ_NAMELIST_NAME_LENGTH	叢集名單	
MQCA_CREATION_DATE	MQ_CREATION_DATE_LENGTH	佇列建立日期	
MQCA_CREATION_TIME	MQ_CREATION_TIME_LENGTH	佇列建立時間	
MQCA_INITIATION_Q_NAME	MQ_Q_NAME_LENGTH	起始佇列名稱	
MQCA_PROCESS_NAME	MQ_PROCESS_NAME_LENGTH	程序定義的名稱	
MQCA_Q_DESC	MQ_Q_DESC_LENGTH	佇列說明	
MQCA_Q_NAME	MQ_Q_NAME_LENGTH	佇列名稱	
MQCA_REMOTE_Q_MGR_NAME	MQ_Q_MGR_NAME_LENGTH	遠端佇列管理程式的名稱	
MQCA_REMOTE_Q_NAME	MQ_Q_NAME_LENGTH	遠端佇列管理程式上已知的遠端佇列名稱	
MQCA_STORAGE_CLASS	MQ_STORAGE_CLASS_LENGTH	儲存類別的名稱	z/OS
MQCA_TRIGGER_DATA	MQ_TRIGGER_DATA_LENGTH	觸發資料	
MQCA_XMIT_Q_NAME	MQ_Q_NAME_LENGTH	傳輸佇列名稱	
MQIA_ACCOUNTING_Q	MQLONG	控制佇列的帳戶資料收集	不 z/OS
MQIA_BACKOUT_THRESHOLD	MQLONG	取消臨界值	
MQIA_CLWL_Q_PRIORITY	MQLONG	佇列的優先順序	
MQIA_CLWL_Q_RANK	MQLONG	佇列等級	
MQIA_CLWL_USEQ	MQLONG	使用遠端佇列	
MQIA_CURRENT_Q_DEPTH	MQLONG	佇列上的訊息數	
MQIA_DEF_BIND	MQLONG	預設連結	
MQIA_DEF_INPUT_OPEN_OPTION	MQLONG	預設 open-for-input 選項	
MQIA_DEF_PERSISTENCE	MQLONG	預設訊息持續性	
MQIA_DEF_PRIORITY	MQLONG	預設訊息優先順序	

表 567: 佇列的 *MQINQ* 屬性選取器 (繼續)

選取元	欄位長度	說明	附註
MQIA_DEFINITION_TYPE	MQLONG	佇列定義類型	
MQIA_DIST_LISTS	MQLONG	配送清單支援	不 z/OS
MQIA_HARDEN_GET_BACKOUT	MQLONG	是否強化取消計數	
MQIA_INDEX_TYPE	MQLONG	為佇列維護的索引類型	z/OS
MQIA_INHIBIT_GET	MQLONG	是否容許取得作業	
MQIA_INHIBIT_PUT	MQLONG	是否容許放置作業	
MQIA_MAX_MSG_LENGTH	MQLONG	訊息長度上限	
MQIA_MAX_Q_DEPTH	MQLONG	佇列上容許的訊息數上限	
MQIA_MSG_DELIVERY_SEQUENCE	MQLONG	訊息優先順序是否相關	
MQIA_NPM_CLASS	MQLONG	非持續訊息的可靠性層次	
MQIA_OPEN_INPUT_COUNT	MQLONG	開啟佇列以供輸入的 MQOPEN 呼叫數	
MQIA_OPEN_OUTPUT_COUNT	MQLONG	開啟佇列以供輸出的 MQOPEN 呼叫數	
MQIA_PROPERTY_CONTROL	MQLONG	內容控制項屬性	
MQIA_Q_DEPTH_HIGH_EVENT	MQLONG	佇列深度高事件的控制屬性	不 z/OS
MQIA_Q_DEPTH_HIGH_LIMIT	MQLONG	佇列深度的高限制	不 z/OS
MQIA_Q_DEPTH_LOW_EVENT	MQLONG	佇列深度低事件的控制屬性	不 z/OS
MQIA_Q_DEPTH_LOW_LIMIT	MQLONG	佇列深度的下限	不 z/OS
MQIA_Q_DEPTH_MAX_EVENT	MQLONG	佇列深度事件數上限的控制屬性	不 z/OS
MQIA_Q_SERVICE_INTERVAL	MQLONG	佇列服務間隔的限制	不 z/OS
MQIA_Q_SERVICE_INTERVAL_EVENT	MQLONG	佇列服務間隔事件的控制屬性	不 z/OS
MQIA_Q_TYPE	MQLONG	佇列類型	
MQIA_QSG_DISP	MQLONG	佇列共用群組處置	z/OS
MQIA_RETENTION_INTERVAL	MQLONG	佇列保留間隔	
MQIA_SCOPE	MQLONG	佇列定義範圍	不 z/OS
MQIA_SHAREABILITY	MQLONG	是否可以共用佇列以供輸入	
MQIA_STATISTICS_Q	MQLONG	控制佇列統計資料的收集	不 z/OS
MQIA_TRIGGER_CONTROL	MQLONG	觸發控制	

表 *567:* 佇列的 *MQINQ* 屬性選取器 *(*繼續*)*

選取元	欄位長度	說明	附註
MQIA_TRIGGER_DEPTH	MQLONG	觸發深度	
MQIA_TRIGGER_MSG_PRIORITY	MQLONG	觸發程式的臨界值訊息優先順序	
MQIA_TRIGGER_TYPE	MQLONG	觸發類型	
MQIA_USAGE	MQLONG	使用情形	

表 *568:* 名稱清單的 *MQINQ* 屬性選取器

選取元	欄位長度	說明	附註
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	最近變更的日期	
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	最近變更的時間	
MQCA_NAMELIST_DESC	MQ_NAMELIST_DESC_LENGTH	名單說明	
MQCA_NAMELIST_NAME	MQ_NAMELIST_NAME_LENGTH	名單物件的名稱	
MQIA_NAMELIST_TYPE	MQLONG	名稱清單類型	z/OS
MQCA_NAMES	MQ_Q_NAME_LENGTH × 清單中的名稱數目	名單中的名稱	
MQIA_NAME_COUNT	MQLONG	名單中的名稱數	
MQIA_QSG_DISP	MQLONG	佇列共用群組處置	z/OS

表 *569:* 程序定義的 *MQINQ* 屬性選取器

選取元	欄位長度	說明	附註
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	最近變更的日期	
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	最近變更的時間	
MQCA_APPL_ID	MQ_PROCESS_APPL_ID_LENGTH	應用程式 ID	
MQCA_ENV_DATA	MQ_PROCESS_ENV_DATA_LENGTH	環境資料	
MQCA_PROCESS_DESC	MQ_PROCESS_DESC_LENGTH	程序定義的說明	
MQCA_PROCESS_NAME	MQ_PROCESS_NAME_LENGTH	程序定義的名稱	
MQCA_USER_DATA	MQ_PROCESS_USER_DATA_LENGTH	使用者資料	
MQIA_APPL_TYPE	MQLONG	應用程式類型	
MQIA_QSG_DISP	MQLONG	佇列共用群組處置	z/OS

表 570: 佇列管理程式的 *MQINQ* 屬性選取器

選取元	欄位長度	說明	附註
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	最近變更的日期	
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	最近變更的時間	
MQCA_CHANNEL_AUTO_DEF_EXIT	MQ_EXIT_NAME_LENGTH	自動通道定義結束程式名稱	
MQCA_CHINIT_SERVICE_PARM		保留供 IBM 使用	
MQCA_CLUSTER_WORKLOAD_DATA	MQ_EXIT_DATA_LENGTH	傳遞至叢集工作量結束程式的資料	
MQCA_CLUSTER_WORKLOAD_EXIT	MQ_EXIT_NAME_LENGTH	叢集工作量結束程式的名稱	
MQCA_COMMAND_INPUT_Q_NAME	MQ_Q_NAME_LENGTH	系統指令輸入佇列名稱	
MQCA_DEAD_LETTER_Q_NAME	MQ_Q_NAME_LENGTH	無法傳送郵件的佇列名稱	
MQCA_DEF_XMIT_Q_NAME	MQ_Q_NAME_LENGTH	預設傳輸佇列名稱	
MQCA_DNS_GROUP	MQ_DNS_GROUP_NAME_LENGTH	TCP 接聽器的群組名稱，用於處理要結合之佇列共用群組的入埠傳輸。 使用「工作量管理程式動態網域名稱服務」時，會套用此名稱。	z/OS
MQCA_IGQ_USER_ID	MQ_USER_ID_LENGTH	內部群組佇列作業使用者 ID	z/OS
MQCA_INSTALLATION_DESC	MQ_INSTALLATION_DESC_LENGTH	相關聯安裝的說明	非 z/OS。不 IBM i
MQCA_INSTALLATION_NAME	MQ_INSTALLATION_NAME_LENGTH	與佇列管理程式相關聯的安裝名稱	非 z/OS。不 IBM i
MQCA_INSTALLATION_PATH	MQ_INSTALLATION_PATH_LENGTH	相關聯 IBM WebSphere MQ 的安裝路徑	非 z/OS。不 IBM i
MQCA_LU_GROUP_NAME	MQ_LU_NAME_LENGTH	LU 6.2 接聽器的一般 LU 名稱，用於處理佇列共用群組要使用的入埠傳輸	z/OS
MQCA_LU_NAME	MQ_LU_NAME_LENGTH	用於出埠 LU 6.2 傳輸的 LU 名稱。 將此名稱設為接聽器用於入埠傳輸的相同 LU	z/OS
MQCA_LU62_ARM_SUFFIX	MQ_ARM_SUFFIX_LENGTH	SYS1.PARMLIB 成員 APPCPM*xx* 的字尾，指定此通道起始程式的 LUADD	z/OS
MQCA_PARENT	MQ_Q_MGR_NAME_LENGTH	指定為這個佇列管理程式母項的階層式連接佇列管理程式名稱	
MQCA_Q_MGR_DESC	MQ_Q_MGR_DESC_LENGTH	佇列管理程式說明	
MQCA_Q_MGR_IDENTIFIER	MQ_Q_MGR_IDENTIFIER_LENGT	佇列管理程式 ID (H)	

表 570: 佇列管理程式的 *MQINQ* 屬性選取器 (繼續)

選取元	欄位長度	說明	附註
MQCA_Q_MGR_NAME	MQ_Q_MGR_NAME_LENGTH	本端佇列管理程式的名稱	
MQCA_QSG_NAME	MQ_QSG_NAME_LENGTH	佇列共用群組名稱	z/OS
MQCA_REPOSITORY_NAME	MQ_CLUSTER_NAME_LENGTH	佇列管理程式為其提供儲存庫服務的叢集名稱	
MQCA_REPOSITORY_NAMELIST	MQ_NAMELIST_NAME_LENGTH	包含佇列管理程式為其提供儲存庫服務之叢集名稱的名單物件名稱	
MQCA_TCP_NAME	MQ_TCP_NAME_LENGTH	您正在使用的 TCP/IP 系統名稱	z/OS
MQIA_ACCOUNTING_CONN_OVERRIDE	MQLONG	置換帳戶設定	不 z/OS
MQIA_ACCOUNTING_INTERVAL	MQLONG	寫入中間帳戶記錄的頻率	不 z/OS
MQIA_ACCOUNTING_MQI	MQLONG	控制 MQI 資料的帳戶資訊收集	不 z/OS
MQIA_ACCOUNTING_Q	MQLONG	控制收集佇列的帳戶資訊	不 z/OS
MQIA_ACTIVE_CHANNELS	MQLONG	隨時可處於作用中的通道數上限	z/OS
MQIA_ADOPTNEWMCA_CHECK	MQLONG	檢查以決定是否採用 MCA 的元素。 當偵測到新的入埠通道與已在作用中的 MCA 同名時，即會執行檢查。	z/OS
MQIA_ADOPTNEWMCA_INTERVAL	MQLONG	新通道等待孤立通道結束的時間量 (秒)	不 z/OS
MQIA_ADOPTNEWMCA_TYPE	MQLONG	當偵測到符合 AdoptNewMCACheck 參數的新入埠通道要求時，是否自動重新啟動特定通道類型 MCA 的孤立實例	z/OS
MQIA_AUTHORITY_EVENT	MQLONG	權限事件的控制屬性	不 z/OS
MQIA_BRIDGE_EVENT	MQLONG	IMS 橋接器事件的控制屬性	z/OS
MQIA_CHANNEL_AUTO_DEF	MQLONG	自動通道定義的控制屬性	不 z/OS
MQIA_CHANNEL_AUTO_DEF_EVENT	MQLONG	自動通道定義事件的控制屬性	不 z/OS
MQIA_CHANNEL_EVENT	MQLONG	通道事件的控制屬性	
MQIA_CHINIT_ADAPTERS	MQLONG	用於處理 IBM WebSphere MQ 呼叫的配接器子作業數目	z/OS
MQIA_CHINIT_DISPATCHERS	MQLONG	用於通道起始程式的分派器數目	z/OS
MQIA_CHINIT_TRACE_AUTO_START	MQLONG	是否自動啟動通道起始程式追蹤	z/OS
MQIA_CHINIT_TRACE_TABLE_SIZE	MQLONG	通道起始程式的追蹤資料空間大小 (MB)	z/OS
MQIA_CLUSTER_WORKLOAD_LENGTH	MQLONG	叢集工作量長度。	

表 570: 佇列管理程式的 *MQINQ* 屬性選取器 (繼續)

選取元	欄位長度	說明	附註
MQIA_CLWL_MRU_CHANNELS	MQLONG	叢集工作量平衡最近使用的通道數	
MQIA_CLWL_USEQ	MQLONG	使用遠端佇列	
MQIA_CODED_CHAR_SET_ID	MQLONG	編碼字集 ID	
MQIA_COMMAND_EVENT	MQLONG	指令事件的控制屬性	
MQIA_COMMAND_LEVEL	MQLONG	佇列管理程式支援的指令層次	
MQIA_CONFIGURATION_EVENT	MQLONG	配置事件的控制屬性	不 z/OS
MQIA_DEF_CLUSTER_XMIT_Q_TYPE	MQLONG	要用於叢集傳送端通道的預設傳輸佇列類型。	不 z/OS
MQIA_DIST_LISTS	MQLONG	配送清單支援	不 z/OS
MQIA_DNS_WLM	MQLONG	處理佇列共用群組之入埠傳輸的 TCP 接聽器是否向「動態網域名稱服務」的「工作量管理程式」登錄	z/OS
MQIA_EXPIRY_INTERVAL	MQLONG	掃描過期訊息的間隔	z/OS
MQIA_GROUP_UR	MQLONG	此佇列管理程式是否啟用 GROUP 回復單元的控制屬性。 只有在佇列管理程式是佇列共用群組的成員時，才能使用 GROUP 回復單元處置	z/OS
MQIA_IGQ_PUT_AUTHORITY	MQLONG	內部群組佇列作業放置權限	z/OS
MQIA_INHIBIT_EVENT	MQLONG	禁止事件的控制屬性	不 z/OS
MQIA_INTRA_GROUP_QUEUING	MQLONG	內部群組佇列作業支援	z/OS
MQIA_LISTENER_TIMER	MQLONG	如果 APPC 或 TCP/IP 失敗， IBM WebSphere MQ 嘗試重新啟動接聽器的時間間隔 (以秒為單位)。	z/OS
MQIA_LOCAL_EVENT	MQLONG	本端事件的控制屬性	不 z/OS
MQIA_LOGGER_EVENT	MQLONG	禁止事件的控制屬性	不 z/OS
MQIA_LU62_CHANNELS	MQLONG	可使用 LU 6.2 傳輸通訊協定的現行通道數或可連接的用戶端數上限	z/OS
MQIA_MSG_MARK_BROWSE_INTERVAL	MQLONG	時間間隔 (毫秒)，在此之後佇列管理程式可以自動從瀏覽訊息中移除標記。 **小心:** 您不應將此值設為低於預設值 5000。	
MQIA_MAX_CHANNELS	MQLONG	可以現行的通道數上限 (包括具有已連接用戶端的伺服器連線通道)	z/OS
MQIA_MAX_HANDLES	MQLONG	控點數目上限	
MQIA_MAX_MSG_LENGTH	MQLONG	訊息長度上限	

表 570: 佇列管理程式的 *MQINQ* 屬性選取器 (繼續)

選取元	欄位長度	說明	附註
MQIA_MAX_PRIORITY	MQLONG	最大優先順序	
MQIA_MAX_UNCOMMITTED_MSGS	MQLONG	工作單元內未確定的訊息數上限	
MQIA_OUTBOUND_PORT_MAX	MQLONG	使用 MQIA_OUTBOUND_PORT_MIN, 定義連結送出通道時要使用的埠號範圍	z/OS
MQIA_OUTBOUND_PORT_MIN	MQLONG	使用 MQIA_OUTBOUND_PORT_MAX, 定義連結送出通道時要使用的埠號範圍	z/OS
MQIA_PERFORMANCE_EVENT	MQLONG	效能事件的控制屬性	不 z/OS
MQIA_PLATFORM	MQLONG	佇列管理程式所在的平台	
MQIA_PROT_POLICY_CAPABILITY	MQLONG	指出 WebSphere MQ Advanced Message Security 的安全功能是否可用於佇列管理程式。	
MQIA_PUBSUB_MAXMSG_RETRY_COUNT	MQLONG	在同步點下重新處理失敗指令訊息的嘗試次數	
MQIA_PUBSUB_MODE	MQLONG	發佈/訂閱引擎及已排入佇列的發佈/訂閱介面是否在執行中。 使用應用程式設計介面來發佈或訂閱的應用程式需要發佈/訂閱引擎。 佇列發佈/訂閱介面所監視的佇列需要佇列發佈/訂閱介面在執行中。	
MQIA_PUBSUB_NP_MSG	MQLONG	是否捨棄 (或保留) 未遞送的輸入訊息	
MQIA_PUBSUB_NP_RESP	MQLONG	控制未遞送回應訊息的行為	
MQIA_PUBSUB_SYNC_PT	MQLONG	是否僅在同步點下處理持續 (或所有) 訊息	
MQIA_QMGR_CFCONLOS	MQLONG	指定在 CFCONLOS 設為 ASQMGR 的情況下，當佇列管理程式失去與管理結構或任何 CF 結構的連線時要採取的動作	z/OS
MQIA_RECEIVE_TIMEOUT	MQLONG	在回到非作用中狀態之前，大約 TCP/IP 通道等待從其友機接收資料 (包括活動訊號) 的時間長度。 該值為數值，由 MQIA_RECEIVE_TIMEOUT_TYPE 限定。	z/OS
MQIA_RECEIVE_TIMEOUT_MIN	MQLONG	在回到非作用中狀態之前， TCP/IP 通道等待從其友機接收資料 (包括活動訊號) 的時間下限	z/OS
MQIA_RECEIVE_TIMEOUT_TYPE	MQLONG	在回到非作用中狀態之前，大約 TCP/IP 通道等待從其友機接收資料 (包括活動訊號) 的時間長度。 MQIA_RECEIVE_TIMEOUT_TYPE 是套用至 MQIA_RECEIVE_TIMEOUT 的限定元。	z/OS

表 570: 佇列管理程式的 *MQINQ* 屬性選取器 (繼續)

選取元	欄位長度	說明	附註
MQIA_REMOTE_EVENT	MQLONG	遠端事件的控制屬性	不 z/OS
MQIA_SECURITY_CASE	MQLONG	安全設定檔的案例	z/OS
MQIA_SSL_EVENT	MQLONG	通道事件的控制屬性	
MQIA_SSL_FIPS_REQUIRED	MQLONG	僅將 FIPS 認證的演算法用於加密法	
MQIA_SSL_RESET_COUNT	MQLONG	SSL 金鑰重設計數	
MQIA_START_STOP_EVENT	MQLONG	啟動停止事件的控制屬性	不 z/OS
MQIA_STATISTICS_AUTO_CLUSSDR	MQLONG	控制收集叢集傳送端通道的統計資料監視資訊	不 z/OS
MQIA_STATISTICS_CHANNEL	MQLONG	控制通道統計資料的收集	不 z/OS
MQIA_STATISTICS_INTERVAL	MQLONG	寫入統計資料監視資料的頻率	不 z/OS
MQIA_STATISTICS_MQI	MQLONG	控制收集佇列管理程式的統計資料監視資訊	不 z/OS
MQIA_STATISTICS_Q	MQLONG	控制收集佇列的統計資料	不 z/OS
MQIA_SYNCPOINT	MQLONG	同步點可用性	
MQIA_TCP_CHANNELS	MQLONG	可使用 TCP/IP 傳輸通訊協定的現行通道或可連接的用戶端數目上限	z/OS
MQIA_TCP_KEEP_ALIVE	MQLONG	是否使用 TCP KEEPALIVE 機能來檢查連線的另一端是否仍然可用	z/OS
MQIA_TCP_STACK_TYPE	MQLONG	通道起始程式是否只能使用 TCPNAME 中指定的 TCP/IP 位址空間，或可以選擇性地連結至任何選取的 TCP/IP 位址	z/OS
MQIA_TRACE_ROUTE_RECORDING	MQLONG	控制追蹤路徑資訊的記錄	z/OS
MQIA_TREE_LIFE_TIME	MQLONG	未用非管理主題的生命期限	
MQIA_TRIGGER_INTERVAL	MQLONG	觸發間隔	

IntAttrCount
類型: MQLONG -輸入

這是 *IntAttrs* 陣列中的元素數目。 零是有效值。

如果 IntAttr **計數** 至少是 *Selectors* 參數中 MQIA_* 選取元的數目，則會傳回所要求的所有整數屬性。

IntAttrs
類型: MQLONG ×*IntAttrCount* -輸出

這是 *IntAttrCount* 整數屬性值的陣列。

整數屬性值的傳回順序與 *Selectors* 參數中 MQIA_* 選取元的傳回順序相同。 如果陣列包含的元素數目超過 MQIA_* 選取元數目，則多餘的元素保持不變。

如果 *Hobj* 代表佇列，但屬性選取器不適用於該類型的佇列，則會傳回特定值 MQIAV_NOT_APPLICABLE。 *IntAttrs* 陣列中的對應元素會傳回它。

如果 *IntAttrCount* 或 *SelectorCount* 參數為零，則不會參照 *IntAttrs*。 在此情況下，以 C 或 S/390 組譯器撰寫的程式所傳遞的參數位址可能是空值。

***CharAttr* 長度**
類型: MQLONG -輸入

這是 *CharAttrs* 參數的長度 (以位元組為單位)。

CharAttr **長度** 必須至少為所要求字元屬性的長度總和 (請參閱 *Selectors*)。 零是有效值。

CharAttrs
類型: MQCHAR × *CharAttr* 長度 -輸出

這是在其中傳回字元屬性並連結在一起的緩衝區。 緩衝區的長度由 *CharAttrLength* 參數提供。

字元屬性會以 *Selectors* 參數中 MQCA_* 選取元的相同順序傳回。 每一個屬性字串的長度都是固定的 (請參閱 *Selectors*)，必要的話，會在其中的值右側填補空白。 您可以提供大於所需的緩衝區，以包含所有要求的字元屬性及填補。 超出所傳回最後一個屬性值的位元組保持不變。

如果 *Hobj* 代表佇列，但屬性選取器不適用於該類型的佇列，則會傳回完全由星號 (*) 組成的字串。 在 *CharAttrs* 中，會傳回星號作為該屬性的值。

如果 *CharAttrLength* 或 *SelectorCount* 參數為零，則不會參照 *CharAttrs*。 在此情況下，以 C 或 S/390 組譯器撰寫的程式所傳遞的參數位址可能是空值。

CompCode
類型: MQLONG -輸出

完成碼:

MQCC_OK
順利完成。

MQCC_WARNING
警告 (局部完成)。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0, X'000') 沒有理由報告。

如果 *CompCode* 是 MQCC_WARNING:

MQRC_CHAR_ATTRS_TOO_SHORT
(2008, X'7D8') 字元屬性的空間不足。

MQRC_INT_ATTR_COUNT_TOO_SMALL
(2022, X'7E6') 整數屬性不容許有足夠的空間。

MQRC_SELECTOR_NOT_FOR_TYPE
(2068, X'814') 選取器不適用於佇列類型。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') 配接器無法使用。

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852') 無法載入配接器服務模組。

MQRC_API_EXIT_ERROR
(2374, X'946') API 結束程式失敗。

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887') 無法載入 API 結束程式。

MQRC_ASID_MISMATCH
(2157, X'86D') 主要和起始 ASID 不同。

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CF_STRUC_FAILED
(2373, X'945') 連結機能結構失敗。

MQRC_CF_STRUC_IN_USE
(2346, X'92A') 連結機能結構使用中。

MQRC_CHAR_ATTR_LENGTH_ERROR
(2006, X'7D6') 字元屬性的長度無效。

MQRC_CHAR_ATTRS_ERROR
(2007, X'7D7') 字元屬性字串無效。

MQRC_CICS_WAIT_FAILED
(2140, X'85C') CICS 已拒絕等待要求。

MQRC_CONNECTION_BROKEN
(2009, X'7D9') 失去佇列管理程式的連線。

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') 未獲連線授權。

MQRC_CONNECTION_STOPPING
(2203, X'89B') 連線關閉。

MQRC_HCONN_ERROR
(2018, X'7E2') 連線控點無效。

MQRC_HOBJ_ERROR
(2019, X'7E3') 物件控點無效。

MQRC_INT_ATTR_COUNT_ERROR
(2021, X'7E5') 整數屬性計數無效。

MQRC_INT_ATTRS_ARRAY_ERROR
(2023, X'7E7') 整數屬性陣列無效。

MQRC_NOT_OPEN_FOR_INQUIRE
(2038, X'7F6') 未開啟佇列進行查詢。

MQRC_OBJECT_CHANGED
(2041, X'7F9') 物件定義自開啟以來已變更。

MQRC_OBJECT_DAMAGED
(2101, X'835') 物件已損壞。

MQRC_PAGESET_ERROR
(2193, X'891') 存取頁集資料集時發生錯誤。

MQRC_Q_DELETED
(2052, X'804') 已刪除佇列。

MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') 佇列管理程式名稱無效或不明。

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') 佇列管理程式無法用於連線。

MQRC_Q_MGR_STOPPING
(2162, X'872') 佇列管理程式關閉。

MQRC_RESOURCE_PROBLEM
(2102, X'836') 可用的系統資源不足。

MQRC_SELECTOR_COUNT_ERROR
(2065, X'811') 選取元計數無效。

MQRC_SELECTOR_ERROR
(2067, X'813') 屬性選取元無效。

MQRC_SELECTOR_LIMIT_EXCEEDED
(2066, X'812') 選取元計數太大。

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') 可用的儲存體不足。

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') 結束程式已暫停呼叫。

MQRC_UNEXPECTED_ERROR
(2195, X'893') 發生非預期的錯誤。

如需這些代碼的詳細資訊; 請參閱 原因碼

使用注意事項

1. 傳回的值是所選取屬性的 Snapshot。 在應用程式可以處理所傳回的值之前，不保證屬性會維持相同。
2. 當您開啟模型佇列時，會建立動態本端佇列。 即使您開啟模型佇列來查詢其屬性，也會建立動態本端佇列。

   動態佇列的屬性基本上與建立動態佇列時模型佇列的屬性相同。 如果您隨後在此佇列上使用 MQINQ 呼叫，則佇列管理程式會傳回動態佇列的屬性，而不是模型佇列的屬性。 如需動態佇列繼承模型佇列哪些屬性的詳細資料，請參閱 第 732 頁的表 573 。
3. 如果要查詢的物件是別名佇列，則 MQINQ 呼叫所傳回的屬性值是別名佇列的屬性。 不是別名所解析成的基本佇列或主題的屬性。
4. 如果要查詢的物件是叢集佇列，則可以查詢的屬性取決於開啟佇列的方式:
   - 您可以開啟叢集佇列以進行查詢，以及一或多個輸入、瀏覽或設定作業。 若要這樣做，必須有叢集佇列的本端實例，才能順利開啟。 在此情況下，可以查詢的屬性是適用於本端佇列的屬性。

     如果在未指定輸入、瀏覽或設定的情況下開啟叢集佇列以進行查詢，則如果您嘗試查詢僅適用於本端佇列而非叢集佇列的屬性，則呼叫會傳回完成碼 MQCC_WARNING 及原因碼 MQRC_SELECTOR_NOT_FOR_TYPE (2068)。
   - 在傳遞所連接佇列管理程式的基本佇列管理程式名稱時，您可以開啟叢集佇列進行查詢。

     若要這樣做，必須有叢集佇列的本端實例，才能順利開啟。 如果未傳遞基本佇列管理程式，如果您嘗試查詢僅對本端佇列有效且不適用於叢集佇列的屬性，則呼叫會傳回完成碼 MQCC_WARNING 及原因碼 MQRC_SELECTOR_NOT_FOR_TYPE (2068)。
   - 如果只開啟叢集佇列進行查詢或查詢及輸出，則只能查詢列出的屬性。 在此情況下， **QType** 屬性具有值 MQQT_CLUSTER :
     - MQCA_Q_DESC
     - MQCA_Q_NAME
     - MQIA_DEF_BIND
     - MQIA_DEF_PERSISTENCE
     - MQIA_DEF_PRIORITY
     - MQIA_INHIBIT_PUT
     - MQIA_Q_TYPE

     您可以開啟沒有固定連結的叢集佇列。 您可以在 MQOPEN 呼叫上指定 MQOO_BIND_NOT_FIXED 來開啟它。 或者，指定 MQOO_BIND_AS_Q_DEF，並將佇列的 **DefBind** 屬性設為 MQBND_BIND_NOT_FIXED。 如果您開啟沒有固定連結的叢集佇列，則佇列的連續 MQINQ 呼叫可能會查詢叢集佇列的不同實例。 不過，對於具有相同屬性值的所有實例，這是典型的情況。
   - 可以為叢集定義別名佇列物件。 因為 TARGTYPE 及 TARGET 不是叢集屬性，所以在別名佇列上執行 MQOPEN 處理程序的處理程序不知道別名所解析成的物件。

在起始 MQOPEN 期間，別名佇列會解析為佇列管理程式及叢集中的佇列。 名稱解析會在遠端佇列管理程式上再次進行，並在這裡解析別名佇列的 TARGTPYE 。

如果別名佇列解析為主題別名，則會在此遠端佇列管理程式上發佈放入別名佇列的訊息。

請參閱 叢集佇列

5. 您可能想要查詢一些屬性，然後使用 MQSET 呼叫來設定部分屬性。 若要有效率地進行程式查詢及設定，請將要設定的屬性定位在選取元陣列的開頭。 如果這樣做，則可以將計數減少的相同陣列用於 MQSET。
6. 如果發生多個警告狀況 (請參閱 *CompCode* 參數) ，則傳回的原因碼是下列清單中第一個適用的原因碼:
   a. MQRC_SELECTOR_NOT_FOR_TYPE
   b. MQRC_INT_ATTR_COUNT_TOO_SMALL
   c. MQRC_CHAR_ATTRS_TOO_SHORT
7. 下列主題提供物件屬性的相關資訊:
   - 第 731 頁的『佇列的屬性』
   - 第 758 頁的『名稱清單的屬性』
   - 第 760 頁的『程序定義的屬性』
   - 第 701 頁的『佇列管理程式的屬性』

C 呼叫

```
MQINQ (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs,
      CharAttrLength, CharAttrs, &CompCode, &Reason);
```

宣告參數如下:

```
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 呼叫

```
     CALL 'MQINQ' USING HCONN, HOBJ, SELECTORCOUNT, SELECTORS-TABLE,
                        INTATTRCOUNT, INTATTRS-TABLE, CHARATTRLENGTH,
                        CHARATTRS, COMPCODE, REASON.
```

宣告參數如下:

```
**   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 呼叫

```
call MQINQ (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount,
           IntAttrs, CharAttrLength, CharAttrs, CompCode, Reason);
```

宣告參數如下:

```
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 */
```

High Level Assembler 呼叫

```
         CALL MQINQ,(HCONN,HOBJ,SELECTORCOUNT,SELECTORS,INTATTRCOUNT,  X
               INTATTRS,CHARATTRLENGTH,CHARATTRS,COMPCODE,REASON)
```

宣告參數如下:

```
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 呼叫

```
MQINQ Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs,
      CharAttrLength, CharAttrs, CompCode, Reason
```

宣告參數如下:

```
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-查詢訊息內容

MQINQMP 呼叫會傳回訊息內容的值。

語法

MQINQMP (*Hconn*、 *Hmsg*、 *InqPropOpts*、 名稱、 *PropDesc*、 類型、 *ValueLength*、 值、 *DataLength*、 *CompCode*、 *Reason*)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 *Hconn* 的值必須符合用來建立 *Hmsg* 參數中所指定訊息控點的連線控點。

如果訊息控點是使用 MQHC_UNASSOCATED_HCONN 建立的，則必須在查詢訊息控點內容的執行緒上建立有效連線，否則呼叫會因 MQRC_CONNECTION_BROKEN 而失敗。

Hmsg
類型 :MQHMSG-輸入

這是要查詢的訊息控點。 前一個 **MQCRTMH** 呼叫已傳回值。

***InqProp* 選項**
類型 :MQIMPO-輸入/輸出

如需詳細資料，請參閱 MQIMPO 資料類型。

name
類型 :MQCHARV-輸入/輸出

要查詢的內容名稱。

如果找不到具有此名稱的內容，則呼叫會失敗，原因為 MQRC_PROPERTY_NOT_AVAILABLE。

您可以在內容名稱結尾使用萬用字元百分比符號 (%)。 萬用字元符合零個以上字元，包括句點 (.) 字元。 這可讓應用程式查詢許多內容的值。 使用選項 MQIMPO_INQ_FIRST 來呼叫 MQINQMP ， 以取得第一個相符內容，並再次使用選項 MQIMPO_INQ_NEXT 來取得下一個相符內容。 當沒有其他相符內容可用時，呼叫會失敗，且 MQRC_PROPERTY_NOT_AVAILABLE。 如果使用所傳回內容名稱的位址或偏移來起始設定 InqPropOpts 結構的 *ReturnedName* 欄位，則會在從 MQINQMP 傳回具有相符內容名稱的內容時完成此作業。 如果 InqPropOpts 結構中 *ReturnedName* 的 *VSBufSize* 欄位小於所傳回內容名稱的長度，則完成碼會設定 MQCC_FAILED ， 原因為 MQRC_PROPERTY_NAME_TOO_BIG。

會傳回具有已知同義字的內容，如下所示:

1. 字首為 "mqps" 的內容。 會以 WebSphere MQ 內容名稱傳回。 例如， "MQTopicString" 是傳回的名稱，而不是 "mqps.Top"
2. 字首為 "jms" 的內容。 或 "mcd"。 會以 JMS 標頭欄位名稱傳回，例如， "JMSExpiration" 是傳回的名稱，而不是 "jms.Exp"。
3. 字首為 "usr." 的內容 會傳回不含該字首的，例如，傳回 "Color" 而不是 "usr.Color"。

具有同義字的內容只會傳回一次。

在 C 程式設計語言中，下列巨集變數定義為查詢所有內容，然後查詢開頭為 "usr." 的所有內容:

MQPROP_INQUIRE_ALL
查詢訊息的所有內容。

MQPROP_INQUIRE_ALL 可透過下列方式使用:

```
MQCHARV Name = {MQPROP_INQUIRE_ALL};
```

MQPROP_INQUIRE_ALL_USR
查詢啟動 "usr." 之訊息的所有內容。 傳回的名稱不含 "usr."。 字首。

如果指定了 MQIMP_INQ_NEXT，但自前一次呼叫後名稱已變更，或這是第一次呼叫，則會隱含 MQIMPO_INQ_FIRST。

如需使用內容名稱的進一步相關資訊，請參閱 內容名稱 及 內容名稱限制 。

PropDesc
類型 :MQPD-輸出

此結構用來定義內容的屬性，包括內容不受支援時所發生的情況、內容所屬的訊息環境定義，以及內容應該複製到的訊息。 如需此結構的詳細資料，請參閱 MQPD 。

type
類型 :MQLONG-輸入/輸出

從 MQINQMP 呼叫返回時，此參數會設為資料類型 值。 資料類型可以是下列任何一項:

MQ 類型 _ 布林
布林。

MQTYPE_BYTE_STRING
位元組字串。

MQTYPE_INT8
8 位元帶正負號的整數。

MQTYPE_INT16
16 位元帶正負號的整數。

MQTYPE_INT32
32 位元，帶正負號的整數。

MQTYPE_INT64
64 位元帶正負號的整數。

MQTYPE_FLOAT32
32 位元浮點數字。

MQTYPE_FLOAT64
64 位元浮點數字。

MQTYPE_STRING
字串。

MQTYPE_NULL
內容存在，但具有空值。

如果無法辨識內容值的資料類型，則會傳回 MQTYPE_STRING，並將值的字串表示法置於 值 區域中。 資料類型的字串表示法可在 *InqPropOpts* 參數的 *TypeString* 欄位中找到。 傳回警告完成碼，原因為 MQRC_PROP_TYPE_NOT_SUPPORTED。

此外，如果指定選項 MQIMPO_CONVERT_TYPE，則會要求轉換內容值。 使用 類型 作為輸入，以指定您要傳回內容的資料類型。 如需資料類型轉換的詳細資料，請參閱 MQIMPO 結構之 MQIMPO_CONVERT_TYPE 選項的說明。

如果您未要求類型轉換，則可以在輸入上使用下列值:

MQTYPE_AS_SET
會傳回內容的值，但不轉換其資料類型。

ValueLength
類型 :MQLONG-輸入

「值」區域的長度 (以位元組為單位)。 對於您不需要所傳回值的內容，請指定零。 這些內容可以是由應用程式設計成具有空值或空字串的內容。 如果已指定 MQIMPO_QUERY_LENGTH 選項，也請指定零; 在此情況下，不會傳回任何值。

value
類型 :MQBYTEx*ValueLength* -輸出

這是要包含所查詢內容值的區域。 緩衝區應該在適合所傳回值的界限上對齊。 如果無法這樣做，則在稍後存取值時可能會導致錯誤。

如果 *ValueLength* 小於內容值的長度，則會將儘可能多的內容值移入 值 中，且呼叫會失敗，並出現完成碼 MQCC_FAILED 及原因 MQRC_PROPERTY_VALUE_TOO_BIG。

值 中資料的字集是由 InqPropOpts 參數中的 ReturnedCCSID 欄位所提供。 值 中的資料編碼由 InqPropOpts 參數中的 ReturnedEncoding 欄位提供。

在 C 程式設計語言中，參數宣告為 void 的指標; 任何資料類型的位址都可以指定為參數。

如果 *ValueLength* 參數為零，則不會參照 值 ，且以 C 或 System/390 組合器撰寫的程式所傳遞的值可以是空值。

DataLength
類型 :MQLONG-輸出

這是在 值 區域中傳回的實際內容值的長度 (以位元組為單位)。

如果 *DataLength* 小於內容值長度，則 MQINQMP 呼叫傳回時仍會填入 *DataLength* 。 這可讓應用程式判斷容納內容值所需的緩衝區大小，然後以適當大小的緩衝區重新發出呼叫。

也可以傳回下列值。

如果 *Type* 參數設為 MQTYPE_STRING 或 MQTYPE_BYTE_STRING:

MQVL_EMPTY_STRING
內容存在，但未包含任何字元或位元組。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_WARNING
警告 (局部完成)。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_WARNING:

MQRC_PROP_NAME_NOT_CONVERTED
(2492 , X'09BC') 未轉換傳回的內容名稱。

MQRC_PROP_VALUE_NOT_CONVERTED
(2466 , X'09A2') 未轉換內容值。

MQRC_PROP_TYPE_NOT_SUPPORTED
(2467 , X'09A3') 不支援內容資料類型。

MQRC_RFH_FORMAT_ERROR
(2421 , X'0975 ') 無法剖析包含內容的 MQRFH2 資料夾。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204 , X'089C') 配接卡無法使用。

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130 , X'0852 ') 無法載入配接卡服務模組。

MQRC_ASDID_MISMATCH
(2157 , X'086D') 主要和起始 ASID 不同。

MQRC_BUFFER_ERROR
(2004 , X'07D4') 值參數無效。

MQRC_BUFFER_LENGTH_ERROR
(2005 , X'07D5') 值長度參數無效。

MQRC_CALL_IN_PROGRESS
(2219 , X'08AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CONNECTION_BROKEN
(2009 , X'07D9') 與佇列管理程式的連線遺失。

MQRC_DATA_LENGTH_ERROR
(2010 , X'07DA') 資料長度參數無效。

MQRC_IMPO_ERROR
(2464 , X'09A0') 查詢訊息內容選項結構無效。

MQRC_HMSG_ERROR
(2460 , X'099C') 訊息控點無效。

MQRC_MSG_HANDLE_IN_USE
(2499 , X'09C3') 訊息控點已在使用中。

MQRC_OPTIONS_ERROR
(2046 , X'07F8') 選項無效或不一致。

MQRC_PD_ERROR
(2482 , X'09B2') 內容描述子結構無效。

MQRC_PROP_CONV_NOT_SUPPORTED
(2470 , X'09A6') 不支援從實際轉換為所要求的資料類型。

MQRC_PROPERTY_NAME_ERROR
(2442 , X'098A') 內容名稱無效。

MQRC_PROPERTY_NAME_TOO_BIG
(2465 , X'09A1') 傳回名稱緩衝區的內容名稱太大。

MQRC_PROPERTY_NOT_AVAILABLE
(2471 , X'09A7) 內容無法使用。

MQRC_PROPERTY_VALUE_TOO_BIG
(2469 , X'09A5') 內容值對「值」區域而言太大。

MQRC_PROP_NUMBER_FORMAT_ERROR
(2472 , X'09A8') 在值資料中發現數字格式錯誤。

MQRC_PROPERTY_TYPE_ERROR
(2473 , X'09A9') 所要求的內容類型無效。

MQRC_SOURCE_CCSID_ERROR
(2111 , X'083F') 內容名稱編碼字集 ID 無效。

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'0871 ') 可用的儲存體不足。

MQRC_UNEXPECTED_ERROR
(2195 , X'0893 ') 發生非預期的錯誤。

如需這些代碼的詳細資訊，請參閱 原因碼。

C 呼叫

```
MQINQMP (Hconn, Hmsg, &InqPropOpts, &Name, &PropDesc, &Type,
ValueLength, Value, &DataLength, &CompCode, &Reason);
```

宣告參數如下:

```
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 呼叫

```
CALL 'MQINQMP' USING HCONN, HMSG, INQMSGOPTS, NAME, PROPDESC, TYPE,
VALUELENGTH, VALUE, DATALENGTH, COMPCODE, REASON.
```

宣告參數如下:

```
**   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 呼叫

```
call MQINQMP (Hconn, Hmsg, InqPropOpts, Name, PropDesc, Type,
ValueLength, Value, DataLength, CompCode, Reason);
```

宣告參數如下:

```
dcl Hconn       fixed bin(31);  /* Connection handle */
dcl Hmsg        fixed bin(63);  /* Message handle */
dcl InqPropOpts like MQIMPO;    /* Options that control the action of MQINQMP */
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 in bytes of the Value area */
dcl Value       char (n);       /* Area to contain the property value */
dcl DataLength  fixed bin (31); /* Length of the property value */
dcl CompCode    fixed bin (31); /* Completion code */
dcl Reason      fixed bin (31); /* Reason code qualifying CompCode */
```

High Level Assembler 呼叫

```
CALL MQINQMP,(HCONN,HMSG,INQMSGOPTS,NAME,PROPDESC,TYPE,
VALUELENGTH,VALUE,DATALENGTH,COMPCODE,REASON)
```

宣告參數如下:

```

HCONN        DS         F     Connection handle
```

```
HMSG          DS         D     Message handle
INQMSGOPTS    CMQIMPOA   ,     Options that control the action of MQINQMP
NAME          CMQCHRVA   ,     Property name
PROPDESC      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-將訊息控點轉換為緩衝區

MQMHBUF 呼叫會將訊息控點轉換為緩衝區，並與 MQBUFMH 呼叫反向。

語法

MQMHBUF (*Hconn*、 *Hmsg*、 *MsgHBufOpts*、 *Name*、 *MsgDesc*、 *BufferLength*、 *Buffer*、 *DataLength*、 *CompCode*、 *Reason*)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 *Hconn* 的值必須符合用來建立 *Hmsg* 參數中所指定訊息控點的連線控點。

如果訊息控點是使用 MQHC_UNASSOCIED_HCONN 建立的，則必須在刪除訊息控點的執行緒上建立有效連線。 如果未建立有效連線，則呼叫會失敗並產生 MQRC_CONNECTION_BROKEN。

Hmsg
類型 :MQHMSG-輸入

這是需要緩衝區的訊息控點。 此值是由前一個 MQCRTMH 呼叫所傳回。

***MsgHBuf* 選項**
類型 :MQMHBO-輸入

MQMHBO 結構可讓應用程式指定選項來控制如何從訊息控點產生緩衝區。

請參閱第 411 頁的『MQMHBO-緩衝區選項的訊息控點』，以取得詳細資料。

名稱
類型 :MQCHARV-輸入

要放入緩衝區的一或多個內容名稱。

如果找不到符合名稱的內容，則呼叫會失敗，且 MQRC_PROPERTY_NOT_AVAILABLE。

您可以使用萬用字元，將多個內容放入緩衝區中。 若要這麼做，請在內容名稱結尾使用萬用字元 '%'。 此萬用字元符合零個以上字元，包括 '.' 來區隔。

在 C 程式設計語言中，定義下列巨集變數以查詢所有內容及開頭為 'usr' 的所有內容:

MQPROP_INQUIRE_ALL
將訊息的所有內容放入緩衝區

MQPROP_INQUIRE_ALL_USR
放置以字元 'usr.' 開頭之訊息的所有內容。 到緩衝區中。

如需使用內容名稱的進一步相關資訊，請參閱 內容名稱 及 內容名稱限制 。

MsgDesc
類型 :MQMD-輸入/輸出

MsgDesc 結構說明緩衝區的內容。

在輸出上，會設定 *Encoding*、 *CodedCharSetId* 及 *Format* 欄位，以正確說明呼叫所寫入緩衝區中資料的編碼、字集 ID 及格式。

此結構中的資料是在應用程式的字集及編碼中。

BufferLength
類型 :MQLONG-輸入

BufferLength 是「緩衝區」區域的長度 (以位元組為單位)。

緩衝區
類型: MQBYTExBuffer 長度-輸出

Buffer 定義要包含訊息內容的區域。 您必須在 4 位元組界限上對齊緩衝區。

如果 *BufferLength* 小於在 *Buffer* 中儲存內容所需的長度，則 MQMHBUF 會與 MQRC_PROPERTY_VALUE_TOO_BIG 一起失敗。

即使呼叫失敗，緩衝區的內容也可能會變更。

DataLength
類型 :MQLONG-輸出

DataLength 是緩衝區中所傳回內容的長度 (以位元組為單位)。 如果值為零，則沒有任何內容符合 *Name* 中提供的值，且呼叫會失敗，原因碼為 MQRC_PROPERTY_NOT_AVAILABLE。

如果 *BufferLength* 小於在緩衝區中儲存內容所需的長度，則 MQMHBUF 呼叫會失敗，並傳回 MQRC_PROPERTY_VALUE_TOO_BIG ，但仍在 *DataLength* 中輸入值。 這可讓應用程式判斷容納內容所需的緩衝區大小，然後以必要的 *BufferLength* 重新發出呼叫。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204 ， X'089C') 配接卡無法使用。

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130 ， X'852 ') 無法載入配接卡服務模組。

MQRC_ASDID_MISMATCH
(2157 ， X'86D') 主要和起始 ASID 不同。

MQRC_MHBO_ERROR
(2501 ， X'095C') 緩衝區選項結構的訊息控點無效。

MQRC_BUFFER_ERROR
(2004 ， X'07D4') 緩衝區參數無效。

MQRC_BUFFER_LENGTH_ERROR
(2005 ， X'07D5') 緩衝區長度參數無效。

MQRC_CALL_IN_PROGRESS
(2219 ， X'08AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CONNECTION_BROKEN
(2009 , X'07D9') 與佇列管理程式的連線遺失。

MQRC_DATA_LENGTH_ERROR
(2010 , X'07DA') 資料長度參數無效。

MQRC_HMSG_ERROR
(2460 , X'099C') 訊息控點無效。

MQRC_MD_ERROR
(2026 , X'07EA') 訊息描述子無效。

MQRC_MSG_HANDLE_IN_USE
(2499 , X'09C3') 訊息控點已在使用中。

MQRC_OPTIONS_ERROR
(2046 , X'07FE') 選項無效或不一致。

MQRC_PROPERTY_NAME_ERROR
(2442 , X'098A') 內容名稱無效。

MQRC_PROPERTY_NOT_AVAILABLE
(2471 , X'09A7') 內容無法使用。

MQRC_PROPERTY_VALUE_TOO_BIG
(2469 , X'09A5') BufferLength 值太小，無法包含指定的內容。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

如需這些代碼的詳細資訊，請參閱 原因碼。

C 呼叫

```
MQMHBUF (Hconn, Hmsg, &MsgHBufOpts, &Name, &MsgDesc, BufferLength, Buffer,
         &DataLength, &CompCode, &Reason);
```

宣告參數如下:

```
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 */
```

使用注意事項

MQMHBUF 將訊息控點轉換為緩衝區。

您可以搭配使用它與 MQGET API 結束程式，以使用訊息內容 API 來存取特定內容，然後將這些內容在緩衝區中傳遞回設計為使用 MQRFH2 標頭而非訊息控點的應用程式。

此呼叫是 MQBUFMH 呼叫的反向呼叫，可用來將訊息內容從緩衝區剖析至訊息控點。

COBOL 呼叫

```
     CALL 'MQMHBUF' USING HCONN, HMSG, MSGHBUFOPTS, NAME, MSGDESC,
                          BUFFERLENGTH, BUFFER, DATALENGTH, COMPCODE, REASON.
```

宣告參數如下:

```
**   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 CMQMHBOV.
**   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 呼叫

```
call MQMHBUF (Hconn, Hmsg, MsgHBufOpts, Name, MsgDesc, BufferLength, Buffer,
              DataLength, CompCode, Reason);
```

宣告參數如下:

```
dcl Hconn        fixed bin(31); /* Connection handle */
dcl Hmsg         fixed bin(63); /* Message handle */
dcl MsgHBufOpts  like MQMHBO;   /* Options that control the action of MQMHBUF */
dcl Name         like MQCHARV;  /* Property name */
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 properties */
dcl DataLength   fixed bin(31); /* Length of the properties */
dcl CompCode     fixed bin(31); /* Completion code */
dcl Reason       fixed bin(31); /* Reason code qualifying CompCode */
```

High Level Assembler 呼叫

```
CALL MQMHBUF,(HCONN,HMSG,MSGHBUFOPTS,NAME,MSGDESC,BUFFERLENGTH,
                BUFFER,DATALENGTH,COMPCODE,REASON)
```

宣告參數如下:

```
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-開啟物件

MQOPEN 呼叫會建立對物件的存取權。

有效的物件類型如下:

- 佇列 (包括配送清單)
- 名稱清單
- 程序定義
- 佇列管理程式
- 主題

語法

MQOPEN (*Hconn*、 *ObjDesc*、 *Options*、 *Hobj*、 *CompCode*、 *Reason*)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 前一個 MQCONN 或 MQCONNX 呼叫已傳回 *Hconn* 的值。

在 z/OS for CICS 應用程式上，以及在 IBM i for 以相容模式執行的應用程式上，可以省略 MQCONN 呼叫，並針對 *Hconn* 指定下列值:

MQHC_DEF_HCONN
預設連線控點。

ObjDesc
類型 :MQOD-輸入/輸出

這是識別要開啟之物件的結構; 如需詳細資料，請參閱 第 413 頁的『MQOD-物件描述子』 。

如果 *ObjDesc* 參數中的 *ObjectName* 欄位是模型佇列的名稱，則為動態本端佇列使用模型佇列的屬性建立; 無論您在 *Options* 參數上指定任何選項，都會發生此情況。 使用 MQOPEN 呼叫所傳回之 *Hobj* 的後續作業是在新的動態佇列上執行，而不是在模型佇列上執行。 即使 MQINQ 和 MQSET 呼叫也是如此。 *ObjDesc* 參數中的模型佇列名稱會取代為所建立動態佇列的名稱。 動態佇列的類型由模型佇列的 *DefinitionType* 屬性值決定 (請參閱 第 731 頁的『佇列的屬性』)。 如需適用於動態佇列之關閉選項的相關資訊，請參閱 MQCLOSE 呼叫的說明。

選項
類型 :MQLONG-輸入

您必須至少指定下列其中一個選項:

- MQ 瀏覽
- MQOO_INPUT_ * (僅其中一項)
- MQOO_INQUIRE
- MQOO_OUTPUT
- MQOO_SET
- MQOO_BIND_ * (僅其中一項)

如需這些選項的詳細資料，請參閱下表; 您可以視需要指定其他選項。 如果需要多個選項，則值可以是:

- 加在一起 (不要多次新增相同的常數) ，或
- 利用執行位元 OR 運算來組合（如果程式設計語言支援位元運算）。

會記下無效的組合; 所有其他組合都有效。 僅容許適用於 *ObjDesc* 所指定物件類型的選項。 下表顯示查詢及主題的有效 MQOPEN 選項。

選項	別名 [1]	本端及模型	遠端	非本端叢集	配送清單	主題
MQOO_INPUT_AS_Q_DEF	是	是	否	否	否	否
MQOO_INPUT_SHARED	是	是	否	否	否	否
MQOO_INPUT_EXCLUSIVE	是	是	否	否	否	否

選項	別名[1]	本端及模型	遠端	非本端叢集	配送清單	主題
MQOO_OUTPUT	是	是	是	是	是	是
MQOO_BROWSE	是	是	否	否	否	否
MQOO_CO_OP	是	是	否	否	否	否
MQOO_INQUIRE	是	是	[2]	是	否	否
MQOO_SET	是	是	[2]	否	否	否
MQOO_BIND_ON_OPEN[3]	是	是	是	是	是	否
MQOO_BIND_NOT_FIXED[3]	是	是	是	是	是	否
MQOO_BIND_ON_GROUP[3]	是	是	是	是	是	否
MQOO_BIND_AS_Q_DEF[3]	是	是	是	是	是	否
MQOO_SAVE_ALL_CONTEXT	是	是	否	否	否	否
MQOO_PASS_IDENTITY_CONTEXT	是	是	是	是	是	[4]
MQOO_PASS_ALL_CONTEXT	是	是	是	是	是	是
MQOO_SET_IDENTITY_CONTEXT	是	是	是	是	是	[4]
MQOO_SET_ALL_CONTEXT	是	是	是	是	是	是
MQOO_NO_READ_AHEAD	是	是	否	否	否	否
MQOO_READ_ahead	是	是	否	否	否	否
MQOO_READ_AHEAD_AS_Q_DEF	是	是	否	否	否	否
MQOO_ALTERNATE_USER_AUTHORITY	是	是	是	是	是	是
MQOO_FAIL_IF_QUIESCING	是	是	是	是	是	是
MQOO_RESOLVE_LOCAL_Q	是	是	是	是	否	否
MQOO_RESOLVE_LOCAL_TOPIC	否	否	否	否	否	是
MQOO_NO_多重播送	否	否	否	否	否	是

註:

1. 別名選項的有效性取決於別名解析成的佇列選項的有效性。
2. 此選項僅適用於遠端佇列的本端定義。
3. 此選項可以指定給任何佇列類型，但如果佇列不是叢集佇列，則會被忽略。 不過，即使別名佇列不在叢集中， *DefBind* 佇列屬性也會置換基本佇列。
4. 這些屬性可以與主題搭配使用，但只會影響保留訊息的環境定義集，不會影響傳送至任何訂閱者的環境定義欄位。

存取選項: 下列選項控制可對物件執行的作業類型:

MQOO_INPUT_AS_Q_DEF
開啟佇列以使用佇列定義的預設值來取得訊息。

開啟佇列以與後續 MQGET 呼叫搭配使用。 存取權類型是共用或專用，視 *DefInputOpenOption* 佇列屬性的值而定; 如需詳細資料，請參閱 第 731 頁的『佇列的屬性』 。

此選項僅適用於本端、別名及模型佇列; 它不適用於遠端佇列、配送清單及非佇列的物件。

MQOO_INPUT_SHARED
開啟佇列以取得具有共用存取權的訊息。

開啟佇列以與後續 MQGET 呼叫搭配使用。 如果佇列目前由這個或另一個具有 MQOO_INPUT_SHARED 的應用程式開啟，則呼叫會成功，但如果佇列目前以 MQOO_INPUT_EXCLUSIVE 開啟，則會失敗，原因碼為 MQRC_OBJECT_IN_USE。

此選項僅適用於本端、別名及模型佇列; 它不適用於遠端佇列、配送清單及非佇列的物件。

MQOO_INPUT_EXCLUSIVE
開啟佇列以取得具有專用存取權的訊息。

開啟佇列以與後續 MQGET 呼叫搭配使用。 如果佇列目前由這個或另一個應用程式開啟以進行任何類型的輸入 (MQOO_INPUT_SHARED 或 MQOO_INPUT_EXCLUSIVE)，則呼叫會失敗，原因碼為 MQRC_OBJECT_IN_USE。

此選項僅適用於本端、別名及模型佇列; 它不適用於遠端佇列、配送清單及非佇列的物件。

MQOO_OUTPUT

開啟佇列以放置訊息，或開啟主題或主題字串以發佈訊息。

開啟佇列或主題，以便與後續的 MQPUT 呼叫搭配使用。

即使 *InhibitPut* 佇列屬性設為 MQQA_PUT_INHIBITED，具有此選項的 MQOPEN 呼叫仍會成功 (雖然屬性設為此值時後續 MQPUT 呼叫會失敗)。

此選項適用於所有類型的佇列，包括配送清單及主題。

下列注意事項適用於這些選項:

- 只能指定其中一個選項。
- 即使 *InhibitGet* 佇列屬性設為 MQQA_GET_INHIBITED (雖然屬性設為此值時後續 MQGET 呼叫會失敗)，具有下列其中一個選項的 MQOPEN 呼叫仍會成功。
- 如果佇列定義為不可共用 (亦即， *Shareability* 佇列屬性具有值 MQQA_NOT_SHAREABLE)，則嘗試開啟共用存取權的佇列會被視為嘗試開啟具有專用存取權的佇列。
- 如果使用下列其中一個選項開啟別名佇列，則專用 (或另一個應用程式是否具有專用) 的測試會針對別名所解析成的基本佇列。
- 如果 *ObjectQMgrName* 是佇列管理程式別名，則這些選項無效; 即使用於佇列管理程式別名化之遠端佇列的本端定義中的 *RemoteQMgrName* 屬性值是本端佇列管理程式的名稱，也是如此。

MQ 瀏覽
開啟佇列以瀏覽訊息。

使用下列其中一個選項開啟佇列，以與後續 MQGET 呼叫搭配使用:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_NEXT
- MQGMO_BROWSE_MSG_UNDER_CURSOR

即使目前已針對 MQOO_INPUT_EXCLUSIVE 開啟佇列，也容許這樣做。 具有 MQOO_BROWSE 選項的 MQOPEN 呼叫會建立瀏覽游標，並將其邏輯定位在佇列上第一個訊息之前; 如需進一步資訊，請參閱 MQGMO-選項欄位 。

此選項僅適用於本端、別名及模型佇列; 它不適用於遠端佇列、配送清單及非佇列的物件。 如果 *ObjectQMgrName* 是佇列管理程式別名，則它也無效; 即使用於佇列管理程式別名化之遠端佇列的本端定義中的 *RemoteQMgrName* 屬性值是本端佇列管理程式的名稱，也是如此。

MQOO_CO_OP
開啟為一組控點的合作成員。

此選項僅適用於 MQOO_BROWSE 選項。 如果未指定 MQOO_BROWSE，則 MQOPEN 會傳回 MQRC_OPTIONS_ERROR。

傳回的控點被視為具有下列其中一個選項之後續 MQGET 呼叫的一組協同作業控點的成員:

- MQGMO_MARK_BROWSE_CO_OP
- MQGMO_UNMARKED_BROWSE_MSG
- MQGMO_UNMARK_BROWSE_CO_OP

此選項僅適用於本端、別名及模型佇列; 它不適用於遠端佇列、配送清單及非佇列的物件。

MQOO_INQUIRE
開啟物件以查詢屬性。

開啟佇列、名單、程序定義或佇列管理程式，以與後續 MQINQ 呼叫搭配使用。

此選項適用於配送清單以外的所有物件類型。 如果 *ObjectQMgrName* 是佇列管理程式別名，則無效; 即使用於佇列管理程式別名化之遠端佇列的本端定義中的 *RemoteQMgrName* 屬性值是本端佇列管理程式的名稱，也是如此。

MQOO_SET
開啟佇列以設定屬性。

開啟佇列以與後續的 MQSET 呼叫搭配使用。

此選項對配送清單以外的所有佇列類型有效。 如果 *ObjectQMgrName* 是遠端佇列的本端定義名稱，則無效; 即使用於佇列管理程式別名化之遠端佇列的本端定義中的 *RemoteQMgrName* 屬性值是本端佇列管理程式的名稱，也是如此。

連結選項: 當開啟的物件是叢集佇列時，下列選項適用; 這些選項控制佇列控點與叢集佇列實例的連結:

MQOO_BIND_ON_OPEN
當開啟佇列時，本端佇列管理程式會將佇列控點連結至目的地佇列的實例。 因此，使用此控點放置的所有訊息都會透過相同的路徑傳送至目的地佇列的相同實例。

此選項僅適用於佇列，且僅影響叢集佇列。 如果指定給非叢集佇列的佇列，則會忽略該選項。

MQOO_BIND_NOT_FIXED
這會停止將佇列控點連結至目的地佇列實例的本端佇列管理程式。 因此，使用此控點的連續 MQPUT 呼叫會將訊息傳送至目的地佇列的 不同 實例，或透過不同的路徑傳送至相同實例。 它還容許稍後由本端佇列管理程式、遠端佇列管理程式或訊息通道代理程式 (MCA) 根據網路狀況變更所選取的實例。

註: 需要交換 系列 訊息以完成交易的用戶端及伺服器應用程式不得使用 MQOO_BIND_NOT_FIXED (或 MQOO_BIND_AS_Q_DEF when *DefBind* 具有值 MQBND_BIND_NOT_FIXED) ，因為系列中的連續訊息可能會傳送至伺服器應用程式的不同實例。

如果為叢集佇列指定 MQOO_BROWSE 或其中一個 MQOO_Input_ * 選項，則會強制佇列管理程式選取叢集佇列的本端實例。 因此，即使指定 MQOO_BIND_NOT_FIXED ，也會修正佇列控點的連結。

如果 MQOO_INQUIRE 與 MQOO_BIND_NOT_FIXED 一起指定，則使用該控點的後續 MQINQ 呼叫可能會查詢叢集佇列的不同實例，雖然通常所有實例都具有相同的屬性值。

MQOO_BIND_NOT_FIXED 僅對佇列有效，僅影響叢集佇列。 如果指定給非叢集佇列的佇列，則會忽略該選項。

MQ OO_BIND_ON_GROUP
容許應用程式要求將訊息群組全部配置給相同的目的地實例。

此選項僅適用於佇列，且僅影響叢集佇列。 如果指定給非叢集佇列的佇列，則會忽略該選項。

MQOO_BIND_AS_Q_DEF
本端佇列管理程式會以 *DefBind* 佇列屬性所定義的方式來連結佇列控點。 此屬性的值為 MQBND_BIND_ON_OPEN、MQBND_BIND_NOT_FIXED 或 MQBND_BIND_ON_GROUP。

當未指定 MQOO_BIND_ON_OPEN、MQOO_BIND_NOT_FIXED 或 MQOO_BIND_ON_GROUP 時，MQOO_BIND_AS_Q_DEF 是預設值。

MQOO_BIND_AS_Q_DEF 輔助程式說明文件。 此選項不是要與其他兩個連結選項中的任何一個搭配使用，但因為其值為零，所以無法偵測到這類使用。

環境定義選項: 下列選項控制訊息環境定義的處理:

MQOO_SAVE_ALL_CONTEXT
環境定義資訊與此佇列控點相關聯。 此資訊是從使用此控點所擷取之任何訊息的環境定義中設定。 如需訊息環境定義的相關資訊，請參閱 訊息環境定義 及 控制環境定義資訊。

此環境定義資訊可以傳遞至訊息，然後使用 MQPUT 或 MQPUT1 呼叫將訊息放置在佇列上。 請參閱第 432 頁的『MQPMO-放置訊息選項』中說明的 MQPMO_PASS_IDENTITY_CONTEXT 及 MQPMO_PASS_ALL_CONTEXT 選項。

在順利擷取訊息之前，無法將環境定義傳遞至放置在佇列上的訊息。

使用其中一個 MQGMO_BROWSE_ * 瀏覽選項擷取的訊息未儲存其環境定義資訊 (雖然在瀏覽之後會設定 *MsgDesc* 參數中的環境定義欄位)。

此選項僅適用於本端、別名及模型佇列; 它不適用於遠端佇列、配送清單及非佇列的物件。 必須指定其中一個 MQOO_INPUT_ * 選項。

MQOO_PASS_IDENTITY_CONTEXT
當訊息放置在佇列上時，這可讓您在 *PutMsgOpts* 參數中指定 MQPMO_PASS_IDENTITY_CONTEXT 選項; 這會提供訊息來自以 MQOO_SAVE_ALL_CONTEXT 選項開啟之輸入佇列的身分環境定義資訊。 如需訊息環境定義的相關資訊，請參閱 訊息環境定義 及 控制環境定義資訊。

必須指定 MQOO_OUTPUT 選項。

此選項適用於所有類型的佇列，包括配送清單。

MQOO_PASS_ALL_CONTEXT
這容許在將訊息放置在佇列上時，在 *PutMsgOpts* 參數中指定 MQPMO_PASS_ALL_CONTEXT 選項; 這會為訊息提供使用 MQOO_SAVE_ALL_CONTEXT 選項開啟的輸入佇列中的身分及原始環境定義資訊。 如需訊息環境定義的相關資訊，請參閱 訊息環境定義 及 控制環境定義資訊 。

此選項暗示 MQOO_PASS_IDENTITY_CONTEXT ，因此不需要指定。 必須指定 MQOO_OUTPUT 選項。

此選項適用於所有類型的佇列，包括配送清單。

MQOO_SET_IDENTITY_CONTEXT
這容許在將訊息放入佇列時，在 *PutMsgOpts* 參數中指定 MQPMO_SET_IDENTITY_CONTEXT 選項; 這會為訊息提供在 MQPUT 或 MQPUT1 呼叫上指定的 *MsgDesc* 參數中包含的身分環境定義資訊。 如需訊息環境定義的相關資訊，請參閱 訊息環境定義 及 控制環境定義資訊 。

此選項暗示 MQOO_PASS_IDENTITY_CONTEXT ，因此不需要指定。 必須指定 MQOO_OUTPUT 選項。

此選項適用於所有類型的佇列，包括配送清單。

MQOO_SET_ALL_CONTEXT
當訊息放置在佇列上時，這容許在 *PutMsgOpts* 參數中指定 MQPMO_SET_ALL_CONTEXT 選項; 這會為訊息提供 MQPUT 或 MQPUT1 呼叫所指定 *MsgDesc* 參數中包含的身分及原始環境定義資訊。 如需訊息環境定義的相關資訊，請參閱 訊息環境定義 及 控制環境定義資訊 。

此選項表示下列選項，因此不需要指定這些選項:

- MQOO_PASS_IDENTITY_CONTEXT
- MQOO_PASS_ALL_CONTEXT
- MQOO_SET_IDENTITY_CONTEXT

必須指定 MQOO_OUTPUT 選項。

此選項適用於所有類型的佇列，包括配送清單。

先讀選項:

當您使用 MQOO_READ_ahead 呼叫 MQOPEN 時， WebSphere MQ 用戶端只會在符合特定條件時啟用先讀。 這些條件包括:

- 用戶端和遠端佇列管理程式都必須位於 WebSphere MQ 第 7 版或更新版本。
- 用戶端應用程式必須針對含執行緒作業的 WebSphere MQ MQI 用戶端程式庫進行編譯及鏈結。
- 用戶端通道必須使用 TCP/IP 通訊協定。
- 該通道必須在用戶端和伺服器通道定義中都具有非零 SharingConversations (SHARECNV) 設定。

下列選項控制在應用程式要求非持續訊息之前，是否將它們傳送至用戶端。 下列附註適用於先讀選項:

- 只能指定其中一個選項。
- 這些選項僅適用於本端、別名及模型佇列。 它們對遠端佇列、配送清單、主題或佇列管理程式無效。

- 僅當同時指定 MQOO_BROWSE、MQOO_INPUT_SHARED 及 MQOO_INPUT_EXCLUSIVE 其中之一時，這些選項才適用，雖然使用 MQOO_INQUIRE 或 MQOO_SET 指定這些選項不是錯誤。
- 如果應用程式不是作為 IBM WebSphere MQ 用戶端執行，則會忽略這些選項。

MQOO_NO_READ_AHEAD
在應用程式要求非持續訊息之前，不會將它們傳送給用戶端。

MQOO_READ_AHEAD
在應用程式要求非持續訊息之前，會先將它們傳送至用戶端。

MQOO_READ_AHEAD_AS_Q_DEF
先讀行為由所開啟佇列的預設先讀屬性決定。 這是預設值。

其他選項: 下列選項控制授權檢查、佇列管理程式靜止時發生的情況、是否解析本端佇列名稱及多重播送:

MQOO_ALTERNATE_USER_AUTHORITY
ObjDesc 參數中的 *AlternateUserId* 欄位包含用來驗證此 MQOPEN 呼叫的使用者 ID。 只有在此 *AlternateUserId* 獲授權以指定的存取選項開啟物件時，不論執行應用程式的使用者 ID 是否獲授權開啟物件，呼叫才會成功。 這不適用於任何指定的環境定義選項，不過，一律會根據執行應用程式的使用者 ID 來檢查這些選項。

此選項適用於所有類型的物件。

MQOO_FAIL_IF_QUIESCING
如果佇列管理程式處於靜止狀態，則 MQOPEN 呼叫會失敗。

在 z/OS 上，對於 CICS 或 IMS 應用程式，如果連線處於靜止狀態，此選項也會強制 MQOPEN 呼叫失敗。

此選項適用於所有類型的物件。

如需用戶端通道的相關資訊，請參閱 IBM WebSphere MQ MQI 用戶端概觀 。

MQOO_RESOLVE_LOCAL_Q
以已開啟的本端佇列名稱填入 MQOD 結構中的 ResolvedQName 。 同樣地， ResolvedQMgr 名稱會填入管理本端佇列的本端佇列管理程式名稱。 如果 MQOD 結構小於第 3 版，則會忽略 MQOO_RESOLVE_LOCAL_Q ，且不會傳回任何錯誤。

當開啟本端、別名或模型佇列時，一律會傳回本端佇列，但例如，當開啟遠端佇列或非本端叢集佇列而沒有使用 MQOO_RESOLVE_LOCAL_Q 選項時，不會傳回這種情況; ResolvedQName 和 ResolvedQMgr 名稱會填入在遠端佇列定義中找到的 RemoteQName 和 RemoteQMgr 名稱，或類似所選擇的遠端叢集佇列。

如果您在開啟時指定 MQOO_RESOLVE_LOCAL_Q (例如，遠端佇列) ，則 ResolvedQName 是放置訊息的傳輸佇列。 ResolvedQMgr 名稱會填入管理傳輸佇列的本端佇列管理程式名稱。

如果您已獲授權在佇列上瀏覽、輸入或輸出，則具有在 MQOPEN 呼叫上指定此旗標的必要權限。 不需要特殊權限。

此選項僅適用於佇列及佇列管理程式。

MQOO_RESOLVE_LOCAL_TOPIC
以開啟的管理主題名稱填入 MQOD 結構中的 ResolvedQName 。

MQOO_NO_MULTICAST
不使用多重播送來傳送發佈訊息。

此選項僅適用於 MQOO_OUTPUT 選項。 如果未指定 MQOO_OUTPUT ，則 MQOPEN 會傳回 MQRC_OPTIONS_ERROR。

此選項僅適用於主題。

HOBJ
類型 :MQHOBJ-輸出

此控點代表已建立對物件的存取權。 必須在對物件進行操作的後續 IBM WebSphere MQ 呼叫中指定它。 當發出 MQCLOSE 呼叫時，或當定義控點範圍的處理單元終止時，它就不再有效。

傳回的物件控點範圍與呼叫上指定的連線控點範圍相同。 如需控點範圍的相關資訊，請參閱 MQCONN-Hconn 參數 。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_WARNING
警告 (局部完成)。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_WARNING:

MQRC_MULTIPLE_REASONS
(2136 , X'858 ') 傳回多個原因碼。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204 , X'89C') 配接卡無法使用。

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130 , X'852 ') 無法載入配接卡服務模組。

MQRC_ALIAS_BASE_Q_TYPE_ERROR
(2001 , X'7D1') 別名基本佇列不是有效的類型。

MQRC_API_EXIT_ERROR
(2374 , X' 946 ') API 結束程式失敗。

MQRC_API_EXIT_LOAD_ERROR
(2183 , X'887 ') 無法載入 API 結束程式。

MQRC_ASDID_MISMATCH
(2157 , X'86D') 主要和起始 ASID 不同。

MQRC_CALL_IN_PROGRESS
(2219 , X'8AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

無法使用 MQRC_CF_NOT_AVAILABLE
(2345 , X' 929 ') 無法使用連結機能。

MQRC_CF_STRUC_AUTH_FAILED
(2348 , X'92C') 連結機能結構授權檢查失敗。

MQRC_CF_STRUC_ERROR
(2349 , X'92D') 連結機能結構無效。

MQRC_CF_STRUC_FAILED
(2373 , X' 945 ') 連結機能結構失敗。

MQRC_CF_STRUC_IN_USE
(2346 , X'92A') 連結機能結構使用中。

MQRC_CF_STRUC_LIST_HDR_IN_USE
(2347 , X'92B') 連結機能結構清單標頭使用中。

MQRC_CICS_WAIT_FAILED
(2140 , X'85C') 等待要求被 CICS 拒絕。

MQRC_CLUSTER_EXIT_ERROR
(2266 , X'8DA') 叢集工作量結束程式失敗。

MQRC_CLUSTER_PUT_INHIBITED
(2268 , X'8DC') 禁止叢集中所有佇列的 Put 呼叫。

MQRC_CLUSTER_RESOLUTION_ERROR
(2189 , X'88D') 叢集名稱解析失敗。

MQRC_CLUSTER_RESOURCE_ERROR
(2269 , X'8DD') 叢集資源錯誤。

MQRC_CONNECTION_BROKEN
(2009 , X'7D9') 與佇列管理程式的連線遺失。

MQRC_CONNECTION_NOT_AUTHORIZED
(2217 , X'8A9') 未獲連線授權。

MQRC_CONNECTION_QUIESCING
(2202 , X'89A') 連線靜止。

MQRC_CONNECTION_STOPPING
(2203 , X'89B') 連線關閉。

MQRC_DB2_NOT_AVAILABLE
(2342 , X' 926 ') Db2 子系統無法使用。

MQRC_DEF_XMIT_Q_TYPE_ERROR
(2198 , X'896 ') 預設傳輸佇列不是本端。

MQRC_DEF_XMIT_Q_USAGE_ERROR
(2199 , X'897 ') 預設傳輸佇列使用錯誤。

MQRC_DYNAMIC_Q_NAME_ERROR
(2011 , X'7DB') 動態佇列名稱無效。

MQRC_HANDLE_NOT_AVAILABLE
(2017 , X'7E1') 沒有可用的控點。

MQRC_HCONN_ERROR
(2018 , X'7E2') 連線控點無效。

MQRC_HOBJ_ERROR
(2019 , X'7E3') 物件控點無效。

MQRC_MULTIPLE_REASONS
(2136 , X'858 ') 傳回多個原因碼。

MQRC_NAME_IN_USE
(2201 , X'899 ') 名稱使用中。

MQRC_NAME_NOT_VALID_FOR_TYPE
(2194 , X'892 ') 物件名稱對物件類型無效。

MQRC_NOT_AUTHORIZED
(2035 , X'7F3') 未獲授權存取。

MQRC_OBJECT_ALREADY_EXISTS
(2100 , X'834 ') 物件存在。

MQRC_OBJECT_DAMAGED
(2101 , X'835 ') 物件已損壞。

MQRC_OBJECT_IN_USE
(2042 , X'7FA') 已使用衝突選項開啟物件。

MQRC_OBJECT_LEVEL_INCOMPATIBLE
(2360 , X' 938 ') 物件層次不相容。

MQRC_OBJECT_NAME_ERROR
(2152 , X'868 ') 物件名稱無效。

MQRC_OBJECT_NOT_UNIQUE
(2343 , X' 927 ') 物件不是唯一的。

MQRC_OBJECT_Q_MGR_NAME_ERROR
(2153 , X'869 ') 物件佇列管理程式名稱無效。

MQRC_OBJECT_RECORDS_ERROR
(2155 , X'86B') 物件記錄無效。

MQRC_OBJECT_STRING_ERROR
(2441 , X'0989 ') Objectstring 欄位無效

MQRC_OBJECT_TYPE_ERROR
(2043 , X'7FB') 物件類型無效。

MQRC_OD_ERROR
(2044 , X'7FC') 物件描述子結構無效。

MQRC_OPTION_NOT_VALID_FOR_TYPE
(2045 , X'7FD') 選項對物件類型無效。

MQRC_OPTIONS_ERROR
(2046 , X'7FE') 選項無效或不一致。

MQRC_PAGESET_ERROR
(2193 , X'891 ') 存取頁集資料集時發生錯誤。

MQRC_PAGESET_FULL
(2192 , X'890 ') 外部儲存媒體已滿。

MQRC_Q_DELETED
(2052 , X'804 ') 已刪除佇列。

MQRC_Q_MGR_NAME_ERROR
(2058 , X'80A') 佇列管理程式名稱無效或不明。

MQRC_Q_MGR_NOT_AVAILABLE
(2059 , X'80B') 佇列管理程式無法用於連線。

MQRC_Q_MGR_QUIESCING
(2161 , X'871 ') 佇列管理程式靜止中。

MQRC_Q_MGR_STOPPING
(2162 , X'872 ') 佇列管理程式關閉。

MQRC_Q_TYPE_ERROR
(2057 , X'809 ') 佇列類型無效。

MQRC_RECS_PRESENT_ERROR
(2154 , X'86A') 存在的記錄數無效。

MQRC_REMOTE_Q_NAME_ERROR
(2184 , X'888 ') 遠端佇列名稱無效。

MQRC_RESOURCE_PROBLEM
(2102 , X'836 ') 可用的系統資源不足。

MQRC_RESPONSE_RECORDS_ERROR
(2156 , X'86C') 回應記錄無效。

MQRC_SECURITY_ERROR
(2063 , X'80F') 發生安全錯誤。

MQRC_SELECTOR_SYNTAX_ERROR
2459 (X'099B') 已發出 MQOPEN、 MQPUT1 或 MQSUB 呼叫，但指定了包含語法錯誤的選取字串。

MQRC_STOPPED_BY_CLUSTER_EXIT
(2188 , X'88C') 叢集工作量結束程式拒絕呼叫。

MQRC_STORAGE_MEDIUM_FULL
(2192 , X'890 ') 外部儲存媒體已滿。

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'817 ') 可用的儲存體不足。

MQRC_SUPPRESSED_BY_EXIT
(2109 , X'83D') 跳出程式暫停呼叫。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

MQRC_UNKNOWN_ALIAS_BASE_Q
(2082 , X'822 ') 不明別名基本佇列。

MQRC_UNKNOWN_DEF_XMIT_Q
(2197 , X'895 ') 不明預設傳輸佇列。

MQRC_UNKNOWN_OBJECT_NAME
(2085 , X'825 ') 不明物件名稱。

MQRC_UNKNOWN_OBJECT_Q_MGR
(2086 , X'826 ') 不明物件佇列管理程式。

MQRC_UNKNOWN_REMOTE_Q_MGR
(2087、X'827 ') 不明遠端佇列管理程式。

MQRC_UNKNOWN_XMIT_Q
(2196、X'894 ') 不明傳輸佇列。

MQRC_WRONG_CF_LEVEL
(2366 , X'93E') 連結機能結構層次錯誤。

MQRC_XMIT_Q_TYPE_ERROR
(2091 , X'82B') 傳輸佇列不是本端。

MQRC_XMIT_Q_USAGE_ERROR
(2092 , X'82C') 使用錯誤的傳輸佇列。

如需這些代碼的詳細資訊，請參閱:

- 原因碼，適用於 z/OS 以外的所有其他 IBM WebSphere MQ 平台。

一般使用注意事項

1. 開啟的物件是下列其中一項:
   - 佇列至:
     - 取得或瀏覽訊息 (使用 MQGET 呼叫)
     - 放置訊息 (使用 MQPUT 呼叫)
     - 查詢佇列的屬性 (使用 MQINQ 呼叫)
     - 設定佇列的屬性 (使用 MQSET 呼叫)

     如果名為的佇列是模型佇列，則會建立動態本端佇列。 請參閱 第 641 頁的『MQOPEN-開啟物件』中說明的 *ObjDesc* 參數。

     配送清單是一種特殊類型的佇列物件，包含佇列清單。 它可以開啟以放置訊息，但不能取得或瀏覽訊息，或查詢或設定屬性。 如需進一步詳細資料，請參閱使用注意事項 8。

     具有 QSGDISP(GROUP) 的佇列是特殊類型的佇列定義，無法與 MQOPEN 或 MQPUT1 呼叫搭配使用。
   - 用於查詢清單中佇列名稱的名單 (使用 MQINQ 呼叫)。
   - 查詢處理程序屬性 (使用 MQINQ 呼叫) 的處理程序定義。
   - 要查詢本端佇列管理程式屬性的佇列管理程式 (使用 MQINQ 呼叫)。
   - 發佈訊息的主題 (使用 MQPUT 呼叫)
2. 應用程式可以多次開啟相同的物件。 每一個開啟都會傳回不同的物件控點。 所傳回的每一個控點都可以用於執行對應開啟的函數。

3. 如果要開啟的物件是叢集佇列以外的佇列，則會在 MQOPEN 呼叫時進行本端佇列管理程式內的所有名稱解析。 這包括：

   - 將遠端佇列的本端定義名稱解析為遠端佇列管理程式的名稱，以及在遠端佇列管理程式上用來識別佇列的名稱
   - 將遠端佇列管理程式名稱解析為本端傳輸佇列的名稱
   - (僅限 z/OS) 將遠端佇列管理程式名稱解析為 IGQ 代理程式所使用的共用傳輸佇列名稱 (僅當本端及遠端佇列管理程式屬於相同的佇列共用群組時才適用)
   - 基本佇列或主題物件名稱的別名解析。

   不過，請注意，控點的後續 MQINQ 或 MQSET 呼叫只會與已開啟的名稱相關，而不會與發生名稱解析之後所產生的物件相關。 例如，如果開啟的物件是別名，則 MQINQ 呼叫所傳回的屬性是別名的屬性，而不是別名所解析的基本佇列或主題物件的屬性。

   如果開啟的物件是叢集佇列，則名稱解析可以在 MQOPEN 呼叫時進行，或延遲到稍後。 發生解析的點是由 MQOPEN 呼叫上指定的 MQOO_BIND_ * 選項所控制：

   - MQOO_BIND_ON_OPEN
   - MQOO_BIND_NOT_FIXED
   - MQOO_BIND_AS_Q_DEF
   - MQ OO_BIND_ON_GROUP

   如需叢集佇列名稱解析的相關資訊，請參閱 名稱解析 。
4. 具有 MQOO_BROWSE 選項的 MQOPEN 呼叫會建立瀏覽游標，以與指定物件控點及其中一個瀏覽選項的 MQGET 呼叫搭配使用。 這容許掃描佇列而不變更其內容。 可以使用 MQGMO_MSG_UNDER_CURSOR 選項從佇列中移除透過瀏覽找到的訊息。

   透過對相同佇列發出數個 MQOPEN 要求，單一應用程式可以有多個作用中的瀏覽游標。
5. 當應用程式啟動時，觸發監視器所啟動的應用程式會傳遞與應用程式相關聯的佇列名稱。 此佇列名稱可以在 *ObjDesc* 參數中指定，以開啟佇列。 有關更多詳細資料，請參閱第 527 頁的『MQTMC2 -觸發訊息 2 (字元格式)』。
6. 在 IBM i 上，以相容模式執行的應用程式會由應用程式發出的第一個 MQOPEN 呼叫自動連接至佇列管理程式 (如果應用程式尚未使用 MQCONN 呼叫連接至佇列管理程式)。

   未在相容模式下執行的應用程式必須先發出 MQCONN 或 MQCONNX 呼叫以明確地連接至佇列管理程式，然後再使用 MQOPEN 呼叫來開啟物件。

先讀選項

當您使用 MQOO_READ_ahead 呼叫 MQOPEN 時， WebSphere MQ 用戶端只會在符合特定條件時啟用先讀。 這些條件包括：

- 用戶端和遠端佇列管理程式都必須位於 WebSphere MQ 第 7 版或更新版本。
- 用戶端應用程式必須針對含執行緒作業的 WebSphere MQ MQI 用戶端程式庫進行編譯及鏈結。
- 用戶端通道必須使用 TCP/IP 通訊協定。
- 該通道必須在用戶端和伺服器通道定義中都具有非零 SharingConversations (SHARECNV) 設定。

下列注意事項適用於使用先讀選項。

1. 僅當同時指定 MQOO_BROWSE、MQOO_INPUT_SHARED 及 MQOO_INPUT_EXCLUSIVE 其中一個選項時，才適用先讀選項。 如果使用 MQOO_ INQUIRE 或 MQOO_SET 選項指定先讀選項，則不會擲出錯誤。
2. 如果第一個 MQGET 呼叫中使用的選項不支援與先讀搭配使用，則在要求時不會啟用先讀。 此外，當用戶端連接不支援先讀的佇列管理程式時，也會停用先讀。
3. 如果應用程式不是以 IBM WebSphere MQ 用戶端身分執行，則會忽略先讀選項。

叢集佇列數

下列注意事項適用於叢集佇列的使用。

1. 第一次開啟叢集佇列時，如果本端佇列管理程式不是完整儲存庫佇列管理程式，則本端佇列管理程式會從完整儲存庫佇列管理程式取得叢集佇列的相關資訊。 當網路忙碌時，本端佇列管理程式可能需要幾秒鐘才能從儲存庫佇列管理程式接收所需的資訊。 因此，發出 MQOPEN 呼叫的應用程式可能必須等待最多 10 秒，然後控制才會從 MQOPEN 呼叫返回。 如果本端佇列管理程式在此時間內未收到叢集佇列的必要相關資訊，則呼叫會失敗，原因碼為 MQRC_CLUSTER_RESOLUTION_ERROR。
2. 當開啟叢集佇列且叢集中有多個佇列實例時，開啟的實例取決於 MQOPEN 呼叫上指定的選項:
   - 如果指定的選項包括下列任何一項:
     - MQ 瀏覽
     - MQOO_INPUT_AS_Q_DEF
     - MQOO_INPUT_EXCLUSIVE
     - MQOO_INPUT_SHARED
     - MQOO_SET

     開啟的叢集佇列實例必須是本端實例。 如果沒有佇列的本端實例， MQOPEN 呼叫會失敗。
   - 如果指定的選項未包含先前說明的任何選項，但包含下列其中一項或兩項:
     - MQOO_INQUIRE
     - MQOO_OUTPUT

     開啟的實例是本端實例 (如果有的話)，否則是遠端實例 (如果使用 CLWLUSEQ 預設值)。 不過，叢集工作量結束程式 (如果有的話) 可以變更佇列管理程式所選擇的實例。
3. 如果有佇列的訂閱，但完整儲存庫未確認它，則該物件不會出現在叢集中，且呼叫會失敗，原因碼為 MQRC_OBJECT_NAME。

如需叢集佇列的相關資訊，請參閱 叢集佇列。

分送清單

下列注意事項適用於配送清單的使用。

在下列環境中支援配送清單: AIX、 HP-UX、 IBM i、 Solaris、 Linux、 Windows，以及連接至這些系統的 IBM WebSphere MQ MQI 用戶端。

1. 開啟配送清單時， MQOD 結構中的欄位必須設定如下:
   - *Version* 必須是 MQOD_VERSION_2 或更高版本。
   - *ObjectType* 必須是 MQOT_Q。
   - *ObjectName* 必須是空白或空字串。
   - *ObjectQMgrName* 必須是空白或空字串。
   - *RecsPresent* 必須大於零。
   - *ObjectRecOffset* 和 *ObjectRecPtr* 其中一個必須是零，另一個必須是非零。
   - *ResponseRecOffset* 和 *ResponseRecPtr* 最多只能有一個是非零。
   - 必須有 *RecsPresent* 物件記錄，由 *ObjectRecOffset* 或 *ObjectRecPtr* 定址。 物件記錄必須設為要開啟的目的地佇列名稱。
   - 如果其中一個 *ResponseRecOffset* 和 *ResponseRecPtr* 不是零，則必須存在 *RecsPresent* 回應記錄。 如果呼叫完成，且原因碼為 MQRC_MULTIPLE_REASONS，則佇列管理程式會設定它們。

   透過確保 *RecsPresent* 為零，也可以使用 version-2 MQOD 來開啟不在配送清單中的單一佇列。
2. *Options* 參數中只有下列開啟選項有效:
   - MQOO_OUTPUT
   - MQOO_PASS_ * _CONTEXT

- MQOO_SET_ * _CONTEXT
- MQOO_ALTERNATE_USER_AUTHORITY
- MQOO_FAIL_IF_QUIESCING

3. 配送清單中的目的地佇列可以是本端、別名或遠端佇列，但不能是模型佇列。 如果指定模型佇列，則該佇列無法開啟，原因碼為 MQRC_Q_TYPE_ERROR。 不過，這並不會阻止順利開啟清單中的其他佇列。
4. 完成碼和原因碼參數設定如下:
   - 如果發佈清單中佇列的開啟作業都以相同方式成功或失敗，則會設定完成碼及原因碼參數來說明一般結果。 在此情況下，不會設定 MQRR 回應記錄 (如果應用程式提供的話)。

     例如，如果每個開啟成功，則完成碼會設為 MQCC_OK ，原因碼會設為 MQRC_NONE; 如果每個開啟都失敗，因為沒有佇列存在，則參數會設為 MQCC_FAILED 及 MQRC_UNKNOWN_OBJECT_NAME。
   - 如果配送清單中佇列的開啟作業並非全部以相同方式成功或失敗:
     - 如果至少一個開啟成功，則完成碼參數會設為 MQCC_WARNING ，如果全部失敗，則會設為 MQCC_FAILED。
     - 原因碼參數設為 MQRC_MULTIPLE_REASONS。
     - 回應記錄 (如果由應用程式提供) 會設為配送清單中佇列的個別完成碼及原因碼。
5. 順利開啟配送清單時，呼叫所傳回的控點 *Hobj* 可以在後續的 MQPUT 呼叫中將訊息放入配送清單中的佇列，以及在 MQCLOSE 呼叫中釋放對配送清單的存取權。 配送清單的唯一有效關閉選項是 MQCO_NONE。

   MQPUT1 呼叫也可以用來將訊息放入配送清單; 定義清單中佇列的 MQOD 結構會指定為該呼叫上的參數。
6. 當檢查應用程式是否超出允許的控點數目上限時 (請參閱 *MaxHandles* 佇列管理程式屬性) ，配送清單中每一個順利開啟的目的地都會視為個別控點。 即使配送清單中有兩個以上目的地解析為相同的實體佇列，也會如此。 如果配送清單的 MQOPEN 或 MQPUT1 呼叫會導致應用程式正在使用的控點數目超出 *MaxHandles*，則呼叫會失敗，原因碼為 MQRC_HANDLE_NOT_AVAILABLE。
7. 每一個順利開啟的目的地都有其 *OpenOutputCount* 屬性的值加 1。 如果配送清單中有兩個以上目的地解析為相同的實體佇列，則該佇列的 *OpenOutputCount* 屬性會隨著配送清單中解析為該佇列的目的地數目而增加。
8. 如果個別開啟佇列 (例如，解析路徑中的變更) ，則對佇列定義所做的任何變更會導致控點變成無效，不會導致配送清單控點變成無效。 不過，當在後續 MQPUT 呼叫中使用配送清單控點時，它會導致該特定佇列失敗。
9. 配送清單只能包含一個目的地。

遠端佇列

下列注意事項適用於遠端佇列的使用。

在此呼叫的 *ObjDesc* 參數中，可以使用兩種方式之一來指定遠端佇列。

- 透過為 *ObjectName* 指定遠端佇列的本端定義名稱。 在此情況下， *ObjectQMgrName* 會參照本端佇列管理程式，且可以指定為空白或 (在 C 程式設計語言中) 空字串。

  本端佇列管理程式所執行的安全驗證會驗證使用者是否已獲授權開啟遠端佇列的本端定義。
- 透過為 *ObjectName* 指定遠端佇列管理程式已知的遠端佇列名稱。 在此情況下， *ObjectQMgrName* 是遠端佇列管理程式的名稱。

  本端佇列管理程式所執行的安全驗證會驗證使用者是否已獲授權將訊息傳送至名稱解析處理程序所產生的傳輸佇列。

在任一情況下:

- 本端佇列管理程式不會將任何訊息傳送至遠端佇列管理程式，以檢查使用者是否已獲授權將訊息放置在佇列上。
- 當訊息到達遠端佇列管理程式時，遠端佇列管理程式可能會拒絕它，因為產生訊息的使用者未獲授權。

如需相關資訊，請參閱 第 413 頁的『MQOD-物件描述子』 中說明的 *ObjectName* 和 *ObjectQMgrName* 欄位。

物件

安全

下列注意事項與使用 MQOPEN 的安全層面相關。

當發出 MQOPEN 呼叫時，佇列管理程式會執行安全檢查，以驗證執行應用程式的使用者 ID 在允許存取之前具有適當的權限層次。 權限檢查是對正在開啟的物件名稱進行，而不是對名稱進行，在解析名稱之後所產生的名稱。

如果要開啟的物件是指向主題物件的別名佇列，則在執行主題的安全檢查之前，佇列管理程式會對別名佇列名稱執行安全檢查，如同直接使用主題物件一樣。

如果要開啟的物件是主題物件 (單獨使用 *ObjectName* 或使用 *ObjectString* (具有或沒有基礎 *ObjectName*)) ，則佇列管理程式會使用產生的主題字串 (取自 *ObjectName* 中指定的主題物件內) 來執行安全檢查，如果需要將它與 *ObjectString* 中提供的主題物件連結在一起，然後在主題樹狀結構中該點或該點以上尋找最接近的主題物件來執行安全檢查。 這可能不是 *ObjectName* 中指定的相同主題物件。

如果要開啟的物件是模型佇列，則佇列管理程式會對模型佇列的名稱及所建立動態佇列的名稱執行完整安全檢查。 如果隨後明確開啟產生的動態佇列，則會針對動態佇列名稱執行進一步的資源安全檢查。

屬性

下列附註與屬性相關。

當應用程式開啟物件時，物件的屬性可能會變更。 在許多情況下，應用程式不會注意到這一點，但對於某些屬性，佇列管理程式會將控點標示為不再有效。 這些屬性如下:

- 任何會影響物件名稱解析的屬性。 不論使用的開啟選項為何，這都適用，並包括下列各項:
  - 對開啟之別名佇列的 *BaseQName* 屬性所做的變更。
  - 對開啟之別名佇列的 *TargetType* 屬性所做的變更。
  - *RemoteQName* 或 *RemoteQMgrName* 佇列屬性的變更，針對此佇列開啟的任何控點，或針對透過此定義解析為佇列管理程式別名的佇列。
  - 導致遠端佇列目前開啟的控點解析為不同的傳輸佇列，或完全無法解析為傳輸佇列的任何變更。 例如，這可能包括:
    - 對遠端佇列之本端定義的 *XmitQName* 屬性所做的變更，不論該定義是用於佇列，還是用於佇列管理程式別名。
    - (僅限 z/OS) *IntraGroupQueuing* 佇列管理程式屬性值的變更，或共用傳輸佇列 (SYSTEM.QSG.TRANSMIT.QUEUE)。

    有一個例外: 建立新的傳輸佇列。 如果控點在開啟時已存在，但改為解析為預設傳輸佇列，則該控點不會變成無效。
  - *DefXmitQName* 佇列管理程式屬性的變更。 在此情況下，所有解析為先前指名佇列的開啟控點 (僅因為它是預設傳輸佇列而解析為該佇列) 都會標示為無效。 因其他原因而解析至此佇列的控點不受影響。
- *Shareability* 佇列屬性 (如果目前有兩個以上控點為此佇列或解析為此佇列的佇列提供 MQOO_INPUT_SHARED 存取權)。 如果是這樣，則不論開啟選項為何，針對此佇列或針對解析此佇列的佇列開啟的 所有 控點都會標示為無效。

  在 z/OS 上，如果一或多個控點目前提供佇列的 MQOO_INPUT_SHARED 或 MQOO_INPUT_EXCLUSIVE 存取權，則先前說明的控點會標示為無效。
- *Usage* 佇列屬性，適用於針對此佇列開啟的所有控點，或解析為此佇列的佇列，不論開啟選項為何。

當控點標示為無效時，使用此控點的所有後續呼叫 (非 MQCLOSE) 都會失敗，原因碼為 MQRC_OBJECT_CHANGED。 應用程式必須發出 MQCLOSE 呼叫 (使用原始控點) ，然後重新開啟佇列。 根據應用程式邏輯的要求，仍然可以確定或取消對先前成功呼叫的舊控點所做的任何未確定的更新。

如果變更屬性導致發生此情況，請使用特殊強制版本的呼叫。

C 呼叫

```
MQOPEN (Hconn, &ObjDesc, Options, &Hobj, &CompCode,
       &Reason);
```

宣告參數如下:

```
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 呼叫

```
CALL 'MQOPEN' USING HCONN, OBJDESC, OPTIONS, HOBJ, COMPCODE, REASON
```

宣告參數如下:

```
**   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 呼叫

```
call MQOPEN (Hconn, ObjDesc, Options, Hobj, CompCode, Reason);
```

宣告參數如下:

```
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 */
```

High Level Assembler 呼叫

```
CALL MQOPEN,(HCONN,OBJDESC,OPTIONS,HOBJ,COMPCODE,REASON)
```

宣告參數如下:

```
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 呼叫

```
MQOPEN Hconn, ObjDesc, Options, Hobj, CompCode, Reason
```

宣告參數如下:

```
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-放置訊息

MQPUT 呼叫會將訊息放置在佇列或配送清單中，或放置在主題中。 佇列、配送清單或主題必須已開啟。

語法

MQPUT (*Hconn*、 *Hobj*、 *MsgDesc*、 *PutMsgOpts*、 *BufferLength*、 *Buffer*、 *CompCode*、 *Reason*)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 前一個 MQCONN 或 MQCONNX 呼叫已傳回 *Hconn* 的值。

在 z/OS (適用於 CICS 應用程式) 及 IBM i (適用於以相容模式執行的應用程式) 上，可以省略 MQCONN 呼叫，並為 *Hconn* 指定下列值:

MQHC_DEF_HCONN
預設連線控點。

HOBJ
類型 :MQHOBJ-輸入

此控點代表訊息新增至其中的佇列，或訊息發佈至其中的主題。 *Hobj* 的值是由先前指定 MQOO_OUTPUT 選項的 MQOPEN 呼叫所傳回。

MsgDesc
類型 :MQMD-輸入/輸出

此結構說明所傳送訊息的屬性，並在放置要求完成之後接收訊息的相關資訊。 請參閱第 363 頁的『MQMD-訊息描述子』，以取得詳細資料。

如果應用程式提供 version-1 MQMD ，則訊息資料可以使用 MQMDE 結構作為字首，以指定存在於 version-2 MQMD 但不存在於 version-1 中的欄位值。 MQMD 中的 格式 欄位必須設為 MQFMT_MD_EXTENSION ，以指出 MQMDE 存在。 如需詳細資料，請參閱 第 405 頁的『MQMDE-訊息描述子延伸』。

如果在 MQPMO 結構的 *OriginalMsgHandle* 或 *NewMsgHandle* 欄位中提供有效的訊息控點，則應用程式不需要提供 MQMD 結構。 如果在其中一個欄位中未提供任何內容，則會從與訊息控點相關聯的描述子取得訊息的描述子。

如果您使用或計劃使用 API 結束程式，則建議您明確提供 MQMD 結構，且不要使用與訊息控點相關聯的訊息描述子。 這是因為與 MQPUT 或 MQPUT1 呼叫相關聯的「API 結束程式」無法確定佇列管理程式使用哪些 MQMD 值來完成 MQPUT 或 MQPUT1 要求。

***PutMsg* 選項**
類型 :MQPMO-輸入/輸出

請參閱第 432 頁的『MQPMO-放置訊息選項』，以取得詳細資料。

BufferLength
類型 :MQLONG-輸入

Buffer 中訊息的長度。 零是有效的，表示訊息不包含任何應用程式資料。 *BufferLength* 的上限取決於各種因素:

- 如果目的地是本端佇列或解析為本端佇列，則上限取決於是否:
  - 本端佇列管理程式支援分段。
  - 傳送端應用程式指定容許佇列管理程式將訊息分段的旗標。 此旗標是 MQMF_SEGMENTATION_ALLOWED ，可以在 version-2 MQMD 中指定，也可以在與 version-1 MQMD 一起使用的 MQMDE 中指定。

  如果同時滿足這兩個條件，則 *BufferLength* 不能超過 999 999 999 減去 MQMD 中 *Offset* 欄位的值。 因此，可放置的最長邏輯訊息是 999 999 999 位元組 (當 *Offset* 為零時)。 不過，應用程式執行所在的作業系統或環境所強制的資源限制可能會導致較低的限制。

  如果未滿足上述其中一個或兩個條件，則 *BufferLength* 不能超出佇列的 *MaxMsgLength* 屬性及佇列管理程式的 *MaxMsgLength* 屬性中較小的一個。
- 如果目的地是遠端佇列或解析為遠端佇列，則本端佇列的條件適用 ，但在每一個佇列管理程式上，訊息必須透過哪個佇列管理程式傳遞，才能到達目的地佇列; 特別是:
  1. 用來暫時將訊息儲存在本端傳輸佇列
  2. 在本端與目的地佇列管理程式之間的路徑上，用來將訊息儲存在佇列管理程式中的中間傳輸佇列 (如果有的話)
  3. 目的地佇列管理程式中的目的地佇列

  因此，可以放置的最長訊息是由這些佇列及佇列管理程式中最嚴格的限制所控管。

  當訊息位於傳輸佇列時，其他資訊會與訊息資料一起常駐，這會減少可攜帶的應用程式資料量。 在此狀況下，在決定 *BufferLength* 的限制時，請從傳輸佇列的 *MaxMsgLength* 值扣除 MQ_MSG_HEADER_LENGTH 位元組。

  註: 放置訊息時，只能同步診斷未符合條件 1 (原因碼為 MQRC_MSG_TOO_BIG_FOR_Q 或 MQRC_MSG_TOO_BIG_FOR_Q_MGR)。 如果未滿足條件 2 或 3 ，則會在中間佇列管理程式或目的地佇列管理程式中，將訊息重新導向至無法傳送的郵件 (無法遞送的訊息) 佇列。 如果發生此情況，如果傳送者要求報告訊息，則會產生報告訊息。

緩衝區
類型: MQBYTExBuffer 長度-輸入

這是包含要傳送之應用程式資料的緩衝區。 緩衝區必須在適合訊息中資料本質的界限上對齊。 4 位元組對齊方式適用於大部分訊息 (包括包含 WebSphere MQ 標頭結構的訊息) ， 但部分訊息可能需要更嚴格的對齊方式。 例如，包含 64 位元二進位整數的訊息可能需要 8 位元組對齊。

如果 *Buffer* 包含字元或數值資料，請將 *MsgDesc* 參數中的 *CodedCharSetId* 及 *Encoding* 欄位設為適合資料的值; 這可讓訊息接收端將資料 (必要的話) 轉換為接收端所使用的字集及編碼。

註: MQPUT 呼叫中的所有其他參數都必須採用本端佇列管理程式的字集及編碼 (由 *CodedCharSetId* 佇列管理程式屬性及 MQENC_NATIVE 提供)。

在 C 程式設計語言中，參數宣告為 void 的指標; 任何資料類型的位址都可以指定為參數。

如果 *BufferLength* 參數為零，則不會參照 *Buffer* ; 在此情況下，以 C 或 System/390 組合器撰寫的程式所傳遞的參數位址可以是空值。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_WARNING
警告 (局部完成)。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_WARNING:

MQRC_INCOMPLETE_GROUP
(2241 , X'8C1') 訊息群組不完整。

MQRC_INCOMPLETE_MSG
(2242 , X'8C2') 邏輯訊息不完整。

MQRC_INCONSISTENT_PERSISTENCE
(2185 , X'889 ') 不一致持續性規格。

MQRC_INCONSISTENT_UOW
(2245 , X'8C5') 工作單元規格不一致。

MQRC_MULTIPLE_REASONS
(2136 , X'858 ') 傳回多個原因碼。

MQRC_PRIORITY_EXCEEDS_MAXIMUM
(2049 , X'801 ') 訊息優先順序超出支援的最大值。

MQRC_UNKNOWN_REPORT_OPTION
(2104 , X'838 ') 無法辨識訊息描述子中的報告選項。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204 , X'89C') 配接卡無法使用。

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130 , X'852 ') 無法載入配接卡服務模組。

MQRC_ALIAS_TARGTYPE_changed
(2480 , X'09B0') 訂閱目標類型已從佇列變更為主題物件。

MQRC_API_EXIT_ERROR
(2374 , X' 946 ') API 結束程式失敗。

MQRC_API_EXIT_LOAD_ERROR
(2183 , X'887 ') 無法載入 API 結束程式。

MQRC_ASDID_MISMATCH
(2157 , X'86D') 主要和起始 ASID 不同。

MQRC_BACKED_OUT
(2003 , X'7D3') 工作單元已取消。

MQRC_BUFFER_ERROR
(2004 , X'7D4') 緩衝區參數無效。

MQRC_BUFFER_LENGTH_ERROR
(2005 , X'7D5') 緩衝區長度參數無效。

MQRC_CALL_IN_PROGRESS
(2219 , X'8AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CALL_INTERRUPTED
(2549 , X'9F5') MQPUT 或 MQCMIT 已岔斷，且重新連線處理程序無法重新建立明確的結果。

MQRC_CF_STRUC_FAILED
(2373 , X' 945 ') 連結機能結構失敗。

MQRC_CF_STRUC_IN_USE
(2346 , X'92A') 連結機能結構使用中。

MQRC_CFGR_ERROR
(2416 , X' 970 ') 訊息資料中的 PCF 群組參數結構 MQCFGR 無效。

MQRC_CFH_ERROR
(2235 , X'8BB') PCF 標頭結構無效。

MQRC_CFIF_ERROR
(2414 , X'96E') 訊息資料中的 PCF 整數過濾器參數結構無效。

MQRC_CFIL_ERROR
(2236 , X'8BC') PCF 整數清單參數結構或 PCIF*64 整數清單參數結構無效。

MQRC_CFIN_ERROR
(2237 , X'8BD') PCF 整數參數結構或 PCIF*64 整數參數結構無效。

MQRC_CFSF_ERROR
(2415 , X'96F') 訊息資料中的 PCF 字串過濾器參數結構無效。

MQRC_CFSL_ERROR
(2238 , X'8BE') PCF 字串清單參數結構無效。

MQRC_CFST_ERROR
(2239 , X'8BF') PCF 字串參數結構無效。

MQRC_CICS_WAIT_FAILED
(2140 , X'85C') 等待要求被 CICS 拒絕。

MQRC_CLUSTER_EXIT_ERROR
(2266 , X'8DA') 叢集工作量結束程式失敗。

MQRC_CLUSTER_RESOLUTION_ERROR
(2189 , X'88D') 叢集名稱解析失敗。

MQRC_CLUSTER_RESOURCE_ERROR
(2269 , X'8DD') 叢集資源錯誤。

MQRC_COD_NOT_VALID_FOR_XCF_Q
(2106 , X'83A') COD 報告選項對 XCF 佇列無效。

MQRC_CONNECTION_BROKEN
(2009 , X'7D9') 與佇列管理程式的連線遺失。

MQRC_CONNECTION_NOT_AUTHORIZED
(2217 , X'8A9') 未獲連線授權。

MQRC_CONNECTION_QUIESCING
(2202 , X'89A') 連線靜止。

MQRC_CONNECTION_STOPPING
(2203 , X'89B') 連線關閉。

MQRC_CONTENT_ERROR
2554 (X'09FA') 無法剖析訊息內容，以判斷是否應該將訊息遞送至具有延伸訊息選取器的訂閱者。

MQRC_CONTEXT_HANDLE_ERROR
(2097 , X'831 ') 所參照的佇列控點不會儲存環境定義。

MQRC_CONTEXT_NOT_AVAILABLE
(2098 , X'832 ') 環境定義無法用於所參照的佇列控點。

MQRC_DATA_LENGTH_ERROR
(2010 , X'7DA') 資料長度參數無效。

MQRC_DH_ERROR
(2135 , X'857 ') 配送標頭結構無效。

MQRC_DLH_ERROR
(2141 , X'85D') 無法傳送的郵件標頭結構無效。

MQRC_EPH_ERROR
(2420 , X' 974 ') 內嵌 PCF 結構無效。

MQRC_EXPIRY_ERROR
(2013 , X'7DD') 到期時間無效。

MQRC_FEEDBACK_ERROR
(2014 , X'7DE') 回饋碼無效。

MQRC_GLOBAL_UOW_CONFLICT
(2351 , X'92F') 廣域工作單元衝突。

MQRC_GROUP_ID_ERROR
(2258 , X'8D2') 群組 ID 無效。

MQRC_HANDLE_IN_USE_FOR_UOW
(2353 , X' 931 ') 用於廣域工作單元的控點。

MQRC_HCONN_ERROR
(2018 , X'7E2') 連線控點無效。

MQRC_HEADER_ERROR
(2142 , X'85E') MQ 標頭結構無效。

MQRC_HOBJ_ERROR
(2019 , X'7E3') 物件控點無效。

MQRC_IIH_ERROR
(2148 , X'864 ') IMS 資訊標頭結構無效。

MQRC_INCOMPLETE_GROUP
(2241 , X'8C1') 訊息群組不完整。

MQRC_INCOMPLETE_MSG
(2242 , X'8C2') 邏輯訊息不完整。

MQRC_INCONSISTENT_PERSISTENCE
(2185 , X'889 ') 不一致持續性規格。

MQRC_INCONSISTENT_UOW
(2245 , X'8C5') 工作單元規格不一致。

MQRC_LOCAL_UOW_CONFLICT
(2352 , X' 930 ') 廣域工作單元與本端工作單元衝突。

MQRC_MD_ERROR
(2026 , X'7EA') 訊息描述子無效。

MQRC_MDE_ERROR
(2248 , X'8C8') 訊息描述子延伸無效。

MQRC_MISSING_REPLY_TO_Q
(2027 , X'7EB') 遺漏回覆目的地佇列或使用 MQPMO_SUPPRESS_REPLYTO

MQRC_MISSING_WIH
(2332 , X'91C') 訊息資料不是以 MQWIH 開頭。

MQRC_MSG_FLAGS_ERROR
(2249 , X'8C9') 訊息旗標無效。

MQRC_MSG_SEQ_NUMBER_ERROR
(2250 , X'8CA') 訊息序號無效。

MQRC_MSG_TOO_BIG_FOR_Q
(2030 , X'7EE') 訊息長度大於佇列的上限。

MQRC_MSG_TOO_BIG_FOR_Q_MGR
(2031 , X'7EF') 訊息長度大於佇列管理程式的上限。

MQRC_MSG_TYPE_ERROR
(2029 , X'7ED') 訊息描述子中的訊息類型無效。

MQRC_MULTIPLE_REASONS
(2136 , X'858 ') 傳回多個原因碼。

MQRC_NO_DESTINATIONS_AVAILABLE
(2270 , X'8DE') 沒有可用的目的地佇列。

MQRC_NOT_OPEN_FOR_OUTPUT
(2039 , X'7F7') 佇列未開啟以供輸出。

MQRC_NOT_OPEN_FOR_PASS_ALL
(2093 , X'82D') 佇列未開啟以供傳遞所有環境定義。

MQRC_NOT_OPEN_FOR_PASS_IDENT
(2094 , X'82E') 佇列未針對傳遞身分環境定義開啟。

MQRC_NOT_OPEN_FOR_SET_ALL
(2095 , X'82F') 未針對設定所有環境定義開啟佇列。

MQRC_NOT_OPEN_FOR_SET_IDENT
(2096 , X'830 ') 佇列未針對設定身分環境定義開啟。

已變更 MQRC_OBJECT_CHANGED
(2041 , X'7F9') 物件定義自開啟以來已變更。

MQRC_OBJECT_DAMAGED
(2101 , X'835 ') 物件已損壞。

MQRC_OFFSET_ERROR
(2251 , X'8CB') 訊息區段偏移無效。

MQRC_OPEN_FAILED
(2137 , X'859 ') 物件未順利開啟。

MQRC_OPTIONS_ERROR
(2046 , X'7FE') 選項無效或不一致。

MQRC_ORIGINAL_LENGTH_ERROR
(2252 , X'8CC') 原始長度無效。

MQRC_PAGESET_ERROR
(2193 , X'891 ') 存取頁集資料集時發生錯誤。

MQRC_PAGESET_FULL
(2192 , X'890 ') 外部儲存媒體已滿。

MQRC_PCF_ERROR
(2149 , X'865 ') PCF 結構無效。

MQRC_PERSISTENCE_ERROR
(2047 , X'7FF') 持續性無效。

MQRC_PERSISTENT_NOT_ALLOWED
(2048 , X'800 ') 佇列不支援持續訊息。

MQRC_PMO_ERROR
(2173 , X'87D') Put-message 選項結構無效。

MQRC_PMO_RECORD_FLAGS_ERROR
(2158 , X'86E') 放置訊息記錄旗標無效。

MQRC_PRIORITY_ERROR
(2050 , X'802 ') 訊息優先順序無效。

MQRC_PUBLICATION_FAILURE
(2502 , X'9C6') 尚未將發佈遞送給任何訂閱者。

MQRC_PUT_INHIBITED
(2051 , X'803 ') 禁止佇列、此佇列所解析的佇列或主題的 Put 呼叫。

MQRC_PUT_MSG_RECORDS_ERROR
(2159 , X'86F') 放置訊息記錄無效。

MQRC_PUT_NOT_RETAINED
(2479 , X'09AF') 無法保留出版品

MQRC_Q_DELETED
(2052 , X'804 ') 已刪除佇列。

MQRC_Q_FULL
(2053 , X'805 ') 佇列已包含訊息數目上限。

MQRC_Q_MGR_NAME_ERROR
(2058 , X'80A') 佇列管理程式名稱無效或不明。

MQRC_Q_MGR_NOT_AVAILABLE
(2059 , X'80B') 佇列管理程式無法用於連線。

MQRC_Q_MGR_QUIESCING
(2161 , X'871 ') 佇列管理程式靜止中。

MQRC_Q_MGR_STOPPING
(2162 , X'872 ') 佇列管理程式關閉。

無法使用 MQRC_Q_SPACE_NOT_AVAILABLE
(2056 , X'808 ') 磁碟上沒有可供佇列使用的空間。

MQRC_RECONNECT_FAILED
(2548 , X'9F4') 重新連接之後，發生錯誤，恢復可重新連接連線的控點。

MQRC_RECS_PRESENT_ERROR
(2154 , X'86A') 存在的記錄數無效。

MQRC_REPORT_OPTIONS_ERROR
(2061 , X'80D') 訊息描述子中的報告選項無效。

MQRC_RESOURCE_PROBLEM
(2102 , X'836 ') 可用的系統資源不足。

MQRC_RESPONSE_RECORDS_ERROR
(2156 , X'86C') 回應記錄無效。

MQRC_RFH_ERROR
(2334 , X'91E') MQRFH 或 MQRFH2 結構無效。

MQRC_RMH_ERROR
(2220 , X'8AC') 參照訊息標頭結構無效。

MQ RC_SEGMENT_LENGTH_ZERO
(2253 , X'8CD') 訊息區段中的資料長度為零。

不支援 MQRC_SEGMENTS_NOT_SUPPORTED
(2365 , X'93D') 不支援區段。

MQRC_SELECTION_NOT_AVAILABLE
2551 (X'09F7') 發佈資訊的可能訂閱者已存在，但佇列管理程式無法檢查是否要將發佈資訊傳送給訂閱者。

MQRC_STOPPED_BY_CLUSTER_EXIT
(2188 , X'88C') 叢集工作量結束程式拒絕呼叫。

MQRC_STORAGE_CLASS_ERROR
(2105 , X'839 ') 儲存類別錯誤。

MQRC_STORAGE_MEDIUM_FULL
(2192 , X'890 ') 外部儲存媒體已滿。

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'817 ') 可用的儲存體不足。

MQRC_SUPPRESSED_BY_EXIT
(2109 , X'83D') 跳出程式暫停呼叫。

MQRC_SYNCPOINT_LIMIT_REACHED
(2024 , X'7E8') 在現行工作單元內無法處理更多訊息。

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072 , X'818 ') 無法使用同步點支援。

MQRC_TM_ERROR
(2265 , X'8D9') 觸發訊息結構無效。

MQRC_TMC_ERROR
(2191 , X'88F') 字元觸發訊息結構無效。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

MQRC_UOW_ENLISTMENT_ERROR
(2354 , X' 932 ') 在廣域工作單元中列入失敗。

不支援 MQRC_UOW_MIX_NOT_SUPPORTED
(2355 , X' 933 ') 不支援混合工作單元呼叫。

MQRC_UOW_NOT_AVAILABLE
(2255 , X'8CF') 佇列管理程式無法使用的工作單元。

MQRC_WIH_ERROR
(2333 , X'91D') MQWIH 結構無效。

MQRC_WRONG_MD_VERSION
(2257 , X'8D1') 提供的 MQMD 版本錯誤。

MQRC_XQH_ERROR
(2260 , X'8D4') 傳輸佇列標頭結構無效。

如需這些代碼的詳細資訊，請參閱 原因碼。

主題使用注意事項

1. 下列注意事項適用於主題的使用:

   a. 使用 MQPUT 來發佈主題的訊息時，如果該主題的一個以上訂閱者由於其訂閱者佇列的問題 (例如已滿) 而無法取得發佈，則傳回 MQPUT 呼叫的原因碼及遞送行為取決於 TOPIC 上 PMSGDLV 或 NPMSGDLV 屬性的設定。 當指定 MQRO_DEAD_LETTER_Q 時，請注意將發佈資訊遞送至無法傳送的郵件佇列，或在指定 MQRO_DISCARD_MSG 時捨棄訊息，會被視為順利遞送訊息。 如果未遞送任何發佈，則 MQPUT 會傳回 MQRC_PUBLICATION_FAILURE。 這在下列情況下有可能發生:

   - 將訊息發佈至 PMSGDLV 或 NPMSGDLV (視訊息的持續性而定) 設為 ALL 的 TOPIC ，且任何訂閱 (可延續或不可延續) 都有無法接收發佈的佇列。
   - 將訊息發佈至具有 PMSGDLV 或 NPMSGDLV (視訊息的持續性而定) 設為 ALLDURR 的 TOPIC ，且可延續訂閱具有無法接收發佈的佇列。

   MQPUT 可以傳回 MQRC_NONE ，即使在下列情況下無法將發佈遞送給部分訂閱者:

   - 將訊息發佈至將 PMSGDLV 或 NPMSGDLV (視訊息的持續性而定) 設為 ALLAVAIL 的 TOPIC ，且任何可延續或不可延續的訂閱都有無法接收發佈的佇列。
   - 將訊息發佈至具有 PMSGDLV 或 NPMSGDLV (視訊息的持續性而定) 的 TOPIC ，且不可延續訂閱具有無法接收發佈的佇列。

   您可以使用 USEDLQ 主題屬性來判定當發佈訊息無法遞送至其正確的訂閱者佇列時，是否使用無法傳送郵件的佇列。 如需使用 USEDLQ 的相關資訊，請參閱 DEFINE TOPIC。

   b. 如果所使用主題沒有任何訂閱者，則發佈的訊息不會傳送至任何佇列，且會捨棄。 不論訊息是持續還是非持續，或它是否具有無限制期限或到期時間，如果沒有訂閱者，仍會捨棄它。 如果要保留訊息，則例外，在此情況下，雖然不會將訊息傳送至任何訂閱者的佇列，但會針對要遞送至任何新訂閱或使用 MQSUBRQ 要求保留發佈的任何訂閱者的主題儲存訊息。

MQPUT 和 MQPUT1

您可以同時使用 MQPUT 和 MQPUT1 呼叫，將訊息放置在佇列上; 要使用的呼叫視情況而定

- 使用 MQPUT 呼叫將多則訊息放在 相同的 佇列上。

  會先發出指定 MQOO_OUTPUT 選項的 MQOPEN 呼叫，然後再發出一個以上 MQPUT 要求，以將訊息新增至佇列; 最後以 MQCLOSE 呼叫關閉佇列。 這提供比重複使用 MQPUT1 呼叫更好的效能。

- 使用 MQPUT1 呼叫，只將 一則 訊息放置在佇列上。

  此呼叫會將 MQOPEN、MQPUT 及 MQCLOSE 呼叫封裝成單一呼叫，以將必須發出的呼叫數減至最少。

目的地佇列

下列注意事項適用於使用目的地佇列:

1. 如果應用程式將一連串訊息放置在相同佇列上，而不使用訊息群組，如果滿足詳細條件，則會保留這些訊息的順序。 部分條件同時適用於本端及遠端目的地佇列; 其他條件僅適用於遠端目的地佇列。

   適用於本端及遠端目的地佇列的條件

   - 所有 MQPUT 呼叫都在相同的工作單元內，或都不在工作單元內。

     請注意，當將訊息放入單一工作單元內的特定佇列時，來自其他應用程式的訊息可能會與佇列上的訊息序列混雜在一起。
   - 所有 MQPUT 呼叫都使用相同的物件控點 *Hobj* 來進行。

     在某些環境中，如果是從相同的應用程式進行呼叫，則在使用不同的物件控點時，也會保留訊息順序。 相同應用程式 的意義取決於環境:

     – 在 z/OS 上，應用程式為:

       - 對於 CICS， CICS 作業
       - 對於 IMS，作業
       - 若為 z/OS 批次，作業

     – 在 IBM i 上，應用程式是工作。

     – 在 Windows 及 UNIX 系統上，應用程式是執行緒。
   - 這些訊息都具有相同的優先順序。
   - 這些訊息不會放入已指定 MQOO_BIND_NOT_FIXED 的叢集佇列 (或在 DefBind 佇列屬性具有值 MQBND_BIND_NOT_FIXED 時生效的 MQOO_BIND_AS_Q_DEF)。

   適用於遠端目的地佇列的其他條件

   - 從傳送端佇列管理程式到目的地佇列管理程式只有一個路徑。

     如果序列中的部分訊息可能進入不同的路徑 (例如，因為重新配置、資料流量平衡或根據訊息大小來選取路徑) ，則無法保證訊息在目的地佇列管理程式中的順序。
   - 訊息不會暫時放置在傳送端、中間或目的地佇列管理程式的無法傳送郵件的佇列上。

     如果有一或多個訊息暫時放置在無法傳送郵件的佇列上 (例如，因為傳輸佇列或目的地佇列暫時已滿) ，則訊息可能會依序到達目的地佇列。
   - 訊息全部持續或全部非持續。

     如果傳送端佇列管理程式與目的地佇列管理程式之間路徑上的通道將其 *NonPersistentMsgSpeed* 屬性設為 MQNPMS_FAST ，則非持續訊息可以跳至持續訊息之前，導致相對於未保留非持續訊息的持續訊息順序。 不過，會保留持續訊息相對於彼此的順序，以及非持續訊息相對於彼此的順序。

   如果未滿足這些條件，您可以使用訊息群組來保留訊息順序，但這需要傳送及接收應用程式都使用訊息分組支援。 如需訊息群組的相關資訊，請參閱:

   - MQMD- MsgFlags 欄位
   - MQPMO_LOGICAL_ORDER
   - MQGMO_LOGICAL_ORDER

發佈清單

下列注意事項適用於配送清單的使用。

在下列環境中支援配送清單: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows，以及連接至這些系統的 WebSphere MQ MQI 用戶端。

1. 您可以使用 version-1 或 version-2 MQPMO，將訊息放置在配送清單中。 如果您使用 version-1 MQPMO (或 version-2 MQPMO *RecsPresent* 等於零)，則應用程式無法提供放置訊息記錄或回應記錄。 如果訊息順利傳送至配送清單中的某些佇列，而非其他佇列，則無法識別發生錯誤的佇列。

   如果應用程式提供放置訊息記錄或回應記錄，請將 *Version* 欄位設為 MQPMO_VERSION_2。

   您也可以使用 version-2 MQPMO，透過確保 *RecsPresent* 為零，將訊息傳送至不在配送清單中的單一佇列。

2. 完成碼和原因碼參數設定如下:

   - 如果發佈清單中的佇列全部都成功或失敗，則會設定完成碼和原因碼參數來說明一般結果。 在此情況下，不會設定 MQRR 回應記錄 (如果應用程式提供的話)。

     例如，如果每個放置成功，則完成碼及原因碼會設為 MQCC_OK 及 MQRC_NONE; 如果每個放置失敗，因為所有佇列都禁止放置，則參數會設為 MQCC_FAILED 及 MQRC_PUT_INHIBITED。

   - 如果配送清單中佇列的放置並非全部成功或失敗，請採取相同方式:

     – 如果至少有一個放置成功，則完成碼參數會設為 MQCC_WARNING，如果全部失敗，則會設為 MQCC_FAILED。

     – 原因碼參數設為 MQRC_MULTIPLE_REASONS。

     – 回應記錄 (如果由應用程式提供) 會設為配送清單中佇列的個別完成碼及原因碼。

     如果由於開啟目的地失敗而導致放置目的地失敗，則回應記錄中的欄位會設為 MQCC_FAILED 及 MQRC_OPEN_FAILED; 該目的地包括在 *InvalidDestCount* 中。

3. 如果配送清單中的目的地解析為本端佇列，則會以一般格式 (亦即，不是配送清單訊息) 將訊息放置在該佇列中。 如果多個目的地解析為相同的本端佇列，則會針對每個此類目的地在佇列上放置一則訊息。

   如果配送清單中的目的地解析為遠端佇列，則會將訊息放置在適當的傳輸佇列上。 當數個目的地解析為相同的傳輸佇列時，即使這些目的地在應用程式所提供的目的地清單中並不相鄰，也可以將包含那些目的地的單一配送清單訊息放在傳輸佇列中。 不過，只有在傳輸佇列支援配送清單訊息時，才能這麼做 (請參閱 DistLists)。

   如果傳輸佇列不支援配送清單，則會針對使用該傳輸佇列的每一個目的地，在傳輸佇列上放置一個正常格式的訊息副本。

   如果具有應用程式訊息資料的配送清單對傳輸佇列而言太大，則配送清單訊息會分割成較小的配送清單訊息，每一個包含較少目的地。 如果應用程式訊息資料只適合佇列，則完全無法使用配送清單訊息，且佇列管理程式會針對使用該傳輸佇列的每一個目的地，以正常形式產生一個訊息副本。

   如果不同的目的地具有不同的訊息優先順序或訊息持續性 (當應用程式指定 MQPRI_PRIORITY_AS_Q_DEF 或 MQPER_PERSISTENCE_AS_Q_DEF 時可能會發生此情況)，則訊息不會保留在相同的配送清單訊息中。 相反地，佇列管理程式會產生所需數量的配送清單訊息，以容納不同的優先順序和持續性值。

4. 放置到配送清單可能會導致:

   - 單一配送清單訊息，或
   - 一些較小的配送清單訊息，或
   - 混合使用配送清單訊息及一般訊息，或
   - 僅限正常訊息。

   發生上述哪些情況取決於是否:

   - 清單中的目的地是本端、遠端或混合。
   - 目的地具有相同的訊息優先順序和訊息持續性。
   - 傳輸佇列可以保留配送清單訊息。
   - 傳輸佇列的訊息長度上限足以容納配送清單形式的訊息。

   不過，不論上述哪一則發生，在下列情況下，所產生的每則 實體 訊息 (亦即，放置所產生的每則一般訊息或配送清單訊息) 只會計為 一則 訊息:

- 檢查應用程式是否已超出工作單元中允許的訊息數上限 (請參閱 *MaxUncommittedMsgs* 佇列管理程式屬性)。
- 正在檢查是否滿足觸發條件。
- 遞增佇列深度並檢查是否將超出佇列的佇列深度上限。

5. 如果個別開啟佇列 (例如，解析路徑中的變更)，則對佇列定義所做的任何變更會導致控點變成無效，不會導致配送清單控點變成無效。 不過，當在後續 MQPUT 呼叫中使用配送清單控點時，它會導致該特定佇列失敗。

標頭

如果在應用程式訊息資料開頭放置含有一或多個 WebSphere MQ 標頭結構的訊息，佇列管理程式會對標頭結構執行某些檢查，以驗證它們是否有效。 如果佇列管理程式偵測到錯誤，則呼叫會失敗，並傳回適當的原因碼。 所執行的檢查會根據所呈現的特定結構而有所不同:

- 僅當在 MQPUT 或 MQPUT1 呼叫上使用 version-2 或更新版本 MQMD 時，才會執行檢查。 如果使用 version-1 MQMD，則即使訊息資料開始時出現 MQMDE，也不會執行檢查。
- 不會驗證本端佇列管理程式不支援的結構，以及訊息中第一個 MQDLH 之後的結構。
- 佇列管理程式會完全驗證 MQDH 及 MQMDE 結構。
- 佇列管理程式會局部驗證其他結構 (並非會檢查所有欄位)。

佇列管理程式所執行的一般檢查包括下列各項:

- *StrucId* 欄位必須有效。
- *Version* 欄位必須有效。
- *StrucLength* 欄位必須指定足夠大的值，以包含結構，以及構成結構一部分的任何可變長度資料。
- *CodedCharSetId* 欄位不得為零，或無效的負值 (MQCCSI_DEFAULT、MQCCSI_EMBEDDED、MQCCSI_Q_MGR 及 MQCCSI_UNDEFINED 在大部分 WebSphere MQ 標頭結構中 無效 。
- 呼叫的 *BufferLength* 參數必須指定一個足夠大的值來包含結構 (結構不得延伸超過訊息結尾)。

除了對結構進行一般檢查之外，還必須滿足下列條件:

- PCF 訊息中結構的長度總和必須等於 MQPUT 或 MQPUT1 呼叫上 *BufferLength* 參數指定的長度。 PCF 訊息是格式名稱為 MQFMT_ADMIN、MQFMT_EVENT 或 MQFMT_PCF 的訊息。
- WebSphere MQ 結構不得截斷，除非下列情況允許截斷結構:
  - 報告訊息的訊息。
  - PCF 訊息。
  - 包含 MQDLH 結構的訊息。 (第一個 MQDLH 之後 的結構可以截斷; MQDLH 之前的結構無法截斷。)
- WebSphere MQ 結構不得分割為兩個以上區段; 該結構必須完全包含在一個區段內。

緩衝區

對於 Visual Basic 程式設計語言，下列要點適用:

- 如果 *Buffer* 參數的大小小於 *BufferLength* 參數指定的長度，則呼叫會失敗，原因碼為 MQRC_BUFFER_LENGTH_ERROR。
- *Buffer* 參數宣告為 `String` 類型。 如果要放置在佇列上的資料不是 `String` 類型，請使用 MQPUTAny 呼叫取代 MQPUT。

  MQPUTAny 呼叫具有與 MQPUT 呼叫相同的參數，但 *Buffer* 參數宣告為類型 Any，容許將任何類型的資料放置在佇列上。 不過，這表示無法檢查 *Buffer*，以確定其大小至少為 *BufferLength* 個位元組。

C 呼叫

```
MQPUT (Hconn, Hobj, &MsgDesc, &PutMsgOpts, BufferLength, Buffer,
      &CompCode, &Reason);
```

宣告參數如下:

```
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 呼叫

```
     CALL 'MQPUT' USING HCONN, HOBJ, MSGDESC, PUTMSGOPTS, BUFFERLENGTH,
                        BUFFER, COMPCODE, REASON.
```

宣告參數如下:

```
**   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 呼叫

```
call MQPUT (Hconn, Hobj, MsgDesc, PutMsgOpts, BufferLength, Buffer,
           CompCode, Reason);
```

宣告參數如下:

```
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 */
```

High Level Assembler 呼叫

```
CALL MQPUT,(HCONN,HOBJ,MSGDESC,PUTMSGOPTS,BUFFERLENGTH, X
               BUFFER,COMPCODE,REASON)
```

宣告參數如下:

```
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 呼叫

```
MQPUT Hconn, Hobj, MsgDesc, PutMsgOpts, BufferLength, Buffer, CompCode,
      Reason
```

宣告參數如下:

```
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 -放置一則訊息

MQPUT1 呼叫會將一則訊息放置在佇列或配送清單中，或放置在主題中。

佇列、配送清單或主題不需要開啟。

語法

MQPUT1 (*Hconn*、 *ObjDesc*、 *MsgDesc*、 *PutMsgOpts*、 *BufferLength*、 *Buffer*、 *CompCode*、 *Reason*)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 前一個 MQCONN 或 MQCONNX 呼叫已傳回 *Hconn* 的值。

在 z/OS (適用於 CICS 應用程式) 及 IBM i (適用於以相容模式執行的應用程式) 上，可以省略 MQCONN 呼叫，並為 *Hconn* 指定下列值:

MQHC_DEF_HCONN
預設連線控點。

ObjDesc
類型 :MQOD-輸入/輸出

這是一種結構，可識別訊息新增至其中的佇列，或訊息發佈至其中的主題。 請參閱第 413 頁的『MQOD-物件描述子』，以取得詳細資料。

如果結構是佇列，則必須授權使用者開啟佇列以進行輸出。 佇列 **不得** 是模型佇列。

MsgDesc
類型 :MQMD-輸入/輸出

此結構說明所傳送訊息的屬性，並在完成放置要求之後接收回饋資訊。 請參閱第 363 頁的『MQMD-訊息描述子』，以取得詳細資料。

如果應用程式提供 version-1 MQMD ，則訊息資料可以使用 MQMDE 結構作為字首，以指定存在於 version-2 MQMD 但不存在於 version-1 中的欄位值。 將 MQMD 中的 格式 欄位設為 MQFMT_MD_EXTENSION ，以指出 MQMDE 存在。 如需詳細資料，請參閱 第 405 頁的『MQMDE-訊息描述子延伸』。

如果在 MQGMO 結構的 *MsgHandle* 欄位中或在 MQPMO 結構的 *OriginalMsgHandle* 或 *NewMsgHandle* 欄位中提供有效的訊息控點，則應用程式不需要提供 MQMD 結構。 如果在其中一個欄位中未提供任何內容，則會從與訊息控點相關聯的描述子取得訊息的描述子。

***PutMsg* 選項**
類型 :MQPMO-輸入/輸出

請參閱第 432 頁的『MQPMO-放置訊息選項』， 以取得詳細資料。

BufferLength
類型 :MQLONG-輸入

Buffer 中訊息的長度。 零是有效的，表示訊息不包含任何應用程式資料。 上限取決於各種因素; 如需進一步詳細資料，請參閱 MQPUT 呼叫的 *BufferLength* 參數說明。

緩衝區
類型: MQBYTExBuffer 長度-輸入

這是包含要傳送之應用程式訊息資料的緩衝區。 在適合訊息中資料本質的界限上對齊緩衝區。 4 位元組對齊方式適用於大部分訊息 (包括包含 WebSphere MQ 標頭結構的訊息) ，但部分訊息可能需要更嚴格的對齊方式。 例如，包含 64 位元二進位整數的訊息可能需要 8 位元組對齊。

如果 *Buffer* 包含字元或數值資料，請將 *MsgDesc* 參數中的 *CodedCharSetId* 及 *Encoding* 欄位設為適合資料的值; 這可讓訊息接收端將資料 (必要的話) 轉換為接收端所使用的字集及編碼。

註: MQPUT1 呼叫的所有其他參數都必須採用本端佇列管理程式的字集及編碼 (由 *CodedCharSetId* 佇列管理程式屬性及 MQENC_NATIVE 提供)。

在 C 程式設計語言中，參數宣告為 void 的指標; 任何資料類型的位址都可以指定為參數。

如果 *BufferLength* 參數為零，則不會參照 *Buffer* ; 在此情況下，以 C 或 System/390 組合器撰寫的程式所傳遞的參數位址可以是空值。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_WARNING
警告 (局部完成)。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_WARNING:

MQRC_MULTIPLE_REASONS
(2136 ， X'858 ') 傳回多個原因碼。

MQRC_INCOMPLETE_GROUP
(2241 ， X'8C1') 訊息群組不完整。

MQRC_INCOMPLETE_MSG
(2242 ， X'8C2') 邏輯訊息不完整。

MQRC_PRIORITY_EXCEEDS_MAXIMUM
(2049 ， X'801 ') 訊息優先順序超出支援的最大值。

MQRC_UNKNOWN_REPORT_OPTION
(2104 , X'838 ') 無法辨識訊息描述子中的報告選項。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204 , X'89C') 配接卡無法使用。

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130 , X'852 ') 無法載入配接卡服務模組。

MQRC_ALIAS_BASE_Q_TYPE_ERROR
(2001 , X'7D1') 別名基本佇列不是有效的類型。

MQRC_API_EXIT_ERROR
(2374 , X' 946 ') API 結束程式失敗。

MQRC_API_EXIT_LOAD_ERROR
(2183 , X'887 ') 無法載入 API 結束程式。

MQRC_ASDID_MISMATCH
(2157 , X'86D') 主要和起始 ASID 不同。

MQRC_BACKED_OUT
(2003 , X'7D3') 工作單元已取消。

MQRC_BUFFER_ERROR
(2004 , X'7D4') 緩衝區參數無效。

MQRC_BUFFER_LENGTH_ERROR
(2005 , X'7D5') 緩衝區長度參數無效。

MQRC_CALL_IN_PROGRESS
(2219 , X'8AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

無法使用 MQRC_CF_NOT_AVAILABLE
(2345 , X' 929 ') 連結機能無法使用。

MQRC_CF_STRUC_AUTH_FAILED
(2348 , X'92C') 連結機能結構授權檢查失敗。

MQRC_CF_STRUC_ERROR
(2349 , X'92D') 連結機能結構無效。

MQRC_CF_STRUC_FAILED
(2373 , X' 945 ') 連結機能結構失敗。

MQRC_CF_STRUC_IN_USE
(2346 , X'92A') 連結機能結構使用中。

MQRC_CF_STRUC_LIST_HDR_IN_USE
(2347 , X'92B') 連結機能結構清單標頭使用中。

MQRC_CFGR_ERROR
(2416 , X' 970 ') 訊息資料中的 PCF 群組參數結構 MQCFGR 無效。

MQRC_CFH_ERROR
(2235 , X'8BB') PCF 標頭結構無效。

MQRC_CFIF_ERROR
(2414 , X'96E') 訊息資料中的 PCF 整數過濾器參數結構無效。

MQRC_CFIL_ERROR
(2236 , X'8BC') PCF 整數清單參數結構或 PCIF*64 整數清單參數結構無效。

MQRC_CFIN_ERROR
(2237 , X'8BD') PCF 整數參數結構或 PCIF*64 整數參數結構無效。

MQRC_CFSF_ERROR
(2415 , X'96F') 訊息資料中的 PCF 字串過濾器參數結構無效。

MQRC_CFSL_ERROR
(2238 , X'8BE') PCF 字串清單參數結構無效。

MQRC_CFST_ERROR
(2239 , X'8BF') PCF 字串參數結構無效。

MQRC_CICS_WAIT_FAILED
(2140 , X'85C') 等待要求被 CICS 拒絕。

MQRC_CLUSTER_EXIT_ERROR
(2266 , X'8DA') 叢集工作量結束程式失敗。

MQRC_CLUSTER_RESOLUTION_ERROR
(2189 , X'88D') 叢集名稱解析失敗。

MQRC_CLUSTER_RESOURCE_ERROR
(2269 , X'8DD') 叢集資源錯誤。

MQRC_COD_NOT_VALID_FOR_XCF_Q
(2106 , X'83A') COD 報告選項對 XCF 佇列無效。

MQRC_CONNECTION_BROKEN
(2009 , X'7D9') 與佇列管理程式的連線遺失。

MQRC_CONNECTION_NOT_AUTHORIZED
(2217 , X'8A9') 未獲連線授權。

MQRC_CONNECTION_QUIESCING
(2202 , X'89A') 連線靜止。

MQRC_CONNECTION_STOPPING
(2203 , X'89B') 連線關閉。

MQRC_CONTENT_ERROR
2554 (X'09FA') 無法剖析訊息內容來判斷訊息是否可以遞送至具有延伸訊息選取器的訂閱者。

MQRC_CONTEXT_HANDLE_ERROR
(2097 , X'831 ') 所參照的佇列控點不會儲存環境定義。

MQRC_CONTEXT_NOT_AVAILABLE
(2098 , X'832 ') 環境定義無法用於所參照的佇列控點。

MQRC_DATA_LENGTH_ERROR
(2010 , X'7DA') 資料長度參數無效。

MQRC_DB2_NOT_AVAILABLE
(2342 , X' 926 ') DB2 子系統無法使用。

MQRC_DEF_XMIT_Q_TYPE_ERROR
(2198 , X'896 ') 預設傳輸佇列不是本端。

MQRC_DEF_XMIT_Q_USAGE_ERROR
(2199 , X'897 ') 預設傳輸佇列使用錯誤。

MQRC_DH_ERROR
(2135 , X'857 ') 配送標頭結構無效。

MQRC_DLH_ERROR
(2141 , X'85D') 無法傳送的郵件標頭結構無效。

MQRC_EPH_ERROR
(2420 , X' 974 ') 內嵌 PCF 結構無效。

MQRC_EXPIRY_ERROR
(2013 , X'7DD') 到期時間無效。

MQRC_FEEDBACK_ERROR
(2014 , X'7DE') 回饋碼無效。

MQRC_GLOBAL_UOW_CONFLICT
(2351 , X'92F') 廣域工作單元衝突。

MQRC_GROUP_ID_ERROR
(2258 , X'8D2') 群組 ID 無效。

MQRC_HANDLE_IN_USE_FOR_UOW
(2353 , X' 931 ') 用於廣域工作單元的控點。

MQRC_HANDLE_NOT_AVAILABLE
(2017 , X'7E1') 沒有可用的控點。

MQRC_HCONN_ERROR
(2018 , X'7E2') 連線控點無效。

MQRC_HEADER_ERROR
(2142 , X'85E') WebSphere MQ 標頭結構無效。

MQRC_IIH_ERROR
(2148 , X'864 ') IMS 資訊標頭結構無效。

MQRC_LOCAL_UOW_CONFLICT
(2352 , X' 930 ') 廣域工作單元與本端工作單元衝突。

MQRC_MD_ERROR
(2026 , X'7EA') 訊息描述子無效。

MQRC_MDE_ERROR
(2248 , X'8C8') 訊息描述子延伸無效。

MQRC_MISSING_REPLY_TO_Q
(2027 , X'7EB') 遺漏回覆目的地佇列。

MQRC_MISSING_WIH
(2332 , X'91C') 訊息資料不是以 MQWIH 開頭。

MQRC_MSG_FLAGS_ERROR
(2249 , X'8C9') 訊息旗標無效。

MQRC_MSG_SEQ_NUMBER_ERROR
(2250 , X'8CA') 訊息序號無效。

MQRC_MSG_TOO_BIG_FOR_Q
(2030 , X'7EE') 訊息長度大於佇列的上限。

MQRC_MSG_TOO_BIG_FOR_Q_MGR
(2031 , X'7EF') 訊息長度大於佇列管理程式的上限。

MQRC_MSG_TYPE_ERROR
(2029 , X'7ED') 訊息描述子中的訊息類型無效。

MQRC_MULTIPLE_REASONS
(2136 , X'858 ') 傳回多個原因碼。

MQRC_NO_DESTINATIONS_AVAILABLE
(2270 , X'8DE') 沒有可用的目的地佇列。

MQRC_NOT_AUTHORIZED
(2035 , X'7F3') 未獲授權存取。

MQRC_OBJECT_DAMAGED
(2101 , X'835 ') 物件已損壞。

MQRC_OBJECT_IN_USE
(2042 , X'7FA') 已使用衝突選項開啟物件。

MQRC_OBJECT_LEVEL_INCOMPATIBLE
(2360 , X' 938 ') 物件層次不相容。

MQRC_OBJECT_NAME_ERROR
(2152 , X'868 ') 物件名稱無效。

MQRC_OBJECT_NOT_UNIQUE
(2343 , X' 927 ') 物件不是唯一的。

MQRC_OBJECT_Q_MGR_NAME_ERROR
(2153 , X'869 ') 物件佇列管理程式名稱無效。

MQRC_OBJECT_RECORDS_ERROR
(2155 , X'86B') 物件記錄無效。

MQRC_OBJECT_TYPE_ERROR
(2043 , X'7FB') 物件類型無效。

MQRC_OD_ERROR
(2044 , X'7FC') 物件描述子結構無效。

MQRC_OFFSET_ERROR
(2251 , X'8CB') 訊息區段偏移無效。

MQRC_OPTIONS_ERROR
(2046 , X'7FE') 選項無效或不一致。

MQRC_ORIGINAL_LENGTH_ERROR
(2252 , X'8CC') 原始長度無效。

MQRC_PAGESET_ERROR
(2193 , X'891 ') 存取頁集資料集時發生錯誤。

MQRC_PAGESET_FULL
(2192 , X'890 ') 外部儲存媒體已滿。

MQRC_PCF_ERROR
(2149 , X'865 ') PCF 結構無效。

MQRC_PERSISTENCE_ERROR
(2047 , X'7FF') 持續性無效。

MQRC_PERSISTENT_NOT_ALLOWED
(2048 , X'800 ') 佇列不支援持續訊息。

MQRC_PMO_ERROR
(2173 , X'87D') Put-message 選項結構無效。

MQRC_PMO_RECORD_FLAGS_ERROR
(2158 , X'86E') 放置訊息記錄旗標無效。

MQRC_PRIORITY_ERROR
(2050 , X'802 ') 訊息優先順序無效。

MQRC_PUBLICATION_FAILURE
(2502 , X'9C6') 尚未將發佈遞送給任何訂閱者。

MQRC_PUT_INHIBITED
(2051 , X'803 ') 佇列禁止放置呼叫。

MQRC_PUT_MSG_RECORDS_ERROR
(2159 , X'86F') 放置訊息記錄無效。

MQRC_Q_DELETED
(2052 , X'804 ') 已刪除佇列。

MQRC_Q_FULL
(2053 , X'805 ') 佇列已包含訊息數目上限。

MQRC_Q_MGR_NAME_ERROR
(2058 , X'80A') 佇列管理程式名稱無效或不明。

MQRC_Q_MGR_NOT_AVAILABLE
(2059 , X'80B') 佇列管理程式無法用於連線。

MQRC_Q_MGR_QUIESCING
(2161 , X'871 ') 佇列管理程式靜止中。

MQRC_Q_MGR_STOPPING
(2162 , X'872 ') 佇列管理程式關閉。

無法使用 MQRC_Q_SPACE_NOT_AVAILABLE
(2056 , X'808 ') 磁碟上沒有可供佇列使用的空間。

MQRC_Q_TYPE_ERROR
(2057 , X'809 ') 佇列類型無效。

MQRC_RECS_PRESENT_ERROR
(2154 , X'86A') 存在的記錄數無效。

MQRC_REMOTE_Q_NAME_ERROR
(2184 , X'888 ') 遠端佇列名稱無效。

MQRC_REPORT_OPTIONS_ERROR
(2061 , X'80D') 訊息描述子中的報告選項無效。

MQRC_RESOURCE_PROBLEM
(2102 , X'836 ') 可用的系統資源不足。

MQRC_RESPONSE_RECORDS_ERROR
(2156 , X'86C') 回應記錄無效。

MQRC_RFH_ERROR
(2334 , X'91E') MQRFH 或 MQRFH2 結構無效。

MQRC_RMH_ERROR
(2220 , X'8AC') 參照訊息標頭結構無效。

MQRC_SECURITY_ERROR
(2063 , X'80F') 發生安全錯誤。

MQ RC_SEGMENT_LENGTH_ZERO
(2253 , X'8CD') 訊息區段中的資料長度為零。

MQRC_SELECTION_NOT_AVAILABLE
2551 (X'09F7') 發佈資訊的可能訂閱者已存在，但佇列管理程式無法檢查是否要將發佈資訊傳送給訂閱者。

MQRC_STOPPED_BY_CLUSTER_EXIT
(2188 , X'88C') 叢集工作量結束程式拒絕呼叫。

MQRC_STORAGE_CLASS_ERROR
(2105 , X'839 ') 儲存類別錯誤。

MQRC_STORAGE_MEDIUM_FULL
(2192 , X'890 ') 外部儲存媒體已滿。

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'817 ') 可用的儲存體不足。

MQRC_SUPPRESSED_BY_EXIT
(2109 , X'83D') 跳出程式暫停呼叫。

MQRC_SYNCPOINT_LIMIT_REACHED
(2024 , X'7E8') 在現行工作單元內無法處理更多訊息。

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072 , X'818 ') 無法使用同步點支援。

MQRC_TM_ERROR
(2265 , X'8D9') 觸發訊息結構無效。

MQRC_TMC_ERROR
(2191 , X'88F') 字元觸發訊息結構無效。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

MQRC_UNKNOWN_ALIAS_BASE_Q
(2082 , X'822 ') 不明別名基本佇列。

MQRC_UNKNOWN_DEF_XMIT_Q
(2197 , X'895 ') 不明預設傳輸佇列。

MQRC_UNKNOWN_OBJECT_NAME
(2085 , X'825 ') 不明物件名稱。

MQRC_UNKNOWN_OBJECT_Q_MGR
(2086 , X'826 ') 不明物件佇列管理程式。

MQRC_UNKNOWN_REMOTE_Q_MGR
(2087、X'827 ') 不明遠端佇列管理程式。

MQRC_UNKNOWN_XMIT_Q
(2196、X'894 ') 不明傳輸佇列。

MQRC_UOW_ENLISTMENT_ERROR
(2354 , X' 932 ') 在廣域工作單元中列入失敗。

不支援 MQRC_UOW_MIX_NOT_SUPPORTED
(2355 , X' 933 ') 不支援混合工作單元呼叫。

MQRC_UOW_NOT_AVAILABLE
(2255 , X'8CF') 佇列管理程式無法使用的工作單元。

MQRC_WIH_ERROR
(2333 , X'91D') MQWIH 結構無效。

MQRC_WRONG_CF_LEVEL
(2366 , X'93E') 連結機能結構層次錯誤。

MQRC_WRONG_MD_VERSION
(2257 , X'8D1') 提供的 MQMD 版本錯誤。

MQRC_XMIT_Q_TYPE_ERROR
(2091 , X'82B') 傳輸佇列不是本端。

MQRC_XMIT_Q_USAGE_ERROR
(2092 , X'82C') 使用錯誤的傳輸佇列。

MQRC_XQH_ERROR
(2260 , X'8D4') 傳輸佇列標頭結構無效。

如需這些代碼的詳細資訊，請參閱 原因碼。

使用注意事項

1. MQPUT 和 MQPUT1 呼叫可用來將訊息放置在佇列上; 使用的呼叫取決於下列情況:
   - 使用 MQPUT 呼叫將多則訊息放在 相同的 佇列上。

     會先發出指定 MQOO_OUTPUT 選項的 MQOPEN 呼叫，然後再發出一個以上 MQPUT 要求，以將訊息新增至佇列; 最後以 MQCLOSE 呼叫關閉佇列。 這提供比重複使用 MQPUT1 呼叫更好的效能。
   - 使用 MQPUT1 呼叫，只將 一則 訊息放置在佇列上。

     此呼叫會將 MQOPEN、MQPUT 及 MQCLOSE 呼叫封裝成單一呼叫，以將必須發出的呼叫數減至最少。
2. 如果應用程式將一連串訊息放置在相同佇列上，而不使用訊息群組，如果滿足特定條件，則會保留這些訊息的順序。 不過，在大部分環境中， MQPUT1 呼叫無法滿足這些條件，因此不會保留訊息順序。 必須在這些環境中改用 MQPUT 呼叫。 如需詳細資料，請參閱 MQPUT 使用注意事項 。
3. MQPUT1 呼叫可用來將訊息放入配送清單。 如需此情況的一般資訊，請參閱 MQOPEN 和 MQPUT 呼叫的使用注意事項。

   在下列環境中支援配送清單: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows，以及連接至這些系統的 WebSphere MQ 用戶端。

   使用 MQPUT1 呼叫時， 下列差異適用:

   a. 如果應用程式提供 MQRR 回應記錄，則必須使用 MQOD 結構來提供它們; 無法使用 MQPMO 結構來提供它們。

   b. MQPUT1 絕不會在回應記錄中傳回原因碼 MQRC_OPEN_FAILED; 如果佇列無法開啟，則該佇列的回應記錄會包含開啟作業所產生的原因碼。

      如果佇列的開啟作業成功，且完成碼為 MQCC_WARNING ，則該佇列回應記錄中的完成碼和原因碼會取代為 put 作業所產生的完成碼和原因碼。

      如同 MQOPEN 和 MQPUT 呼叫，只有在配送清單中所有佇列的呼叫結果都不同時，佇列管理程式才會設定回應記錄 (如果有提供的話); 這是由呼叫完成，原因碼為 MQRC_MULTIPLE_REASONS 來指出。
4. 如果使用 MQPUT1 呼叫將訊息放置在叢集佇列上，則呼叫的行為會如同 MQOPEN 呼叫上已指定 MQOO_BIND_NOT_FIXED 一樣。

5. 如果在應用程式訊息資料開頭放置含有一或多個 WebSphere MQ 標頭結構的訊息，佇列管理程式會對標頭結構執行某些檢查，以驗證它們是否有效。 如需此作業的相關資訊，請參閱 MQPUT 呼叫的使用注意事項。
6. 如果發生多個警告狀況 (請參閱 *CompCode* 參數) ，則傳回的原因碼是下列清單中適用的 第一個 :
   a. MQRC_MULTIPLE_REASONS
   b. MQRC_INCOMPLETE_MSG
   c. MQRC_INCOMPLETE_GROUP
   d. MQRC_PRIORITY_EXCEEDS_MAXIMUM 或 MQRC_UNKNOWN_REPORT_OPTION
7. 對於 Visual Basic 程式設計語言， 下列要點適用:
   - 如果 *Buffer* 參數的大小小於 *BufferLength* 參數指定的長度，則呼叫會失敗，原因碼為 MQRC_BUFFER_LENGTH_ERROR。
   - *Buffer* 參數宣告為 String 類型。 如果要放置在佇列上的資料不是 String 類型，請使用 MQPUT1Any 呼叫取代 MQPUT1。

     MQPUT1Any 呼叫具有與 MQPUT1 呼叫相同的參數，但 *Buffer* 參數宣告為類型 Any，容許將任何類型的資料放置在佇列上。 不過，這表示無法檢查 *Buffer* ，以確定其大小至少為 *BufferLength* 個位元組。
8. 使用 MQPMO_SYNCPOINT 發出 MQPUT1 呼叫時，預設行為會變更，以便以非同步方式完成放置作業。這可能會導致部分應用程式的行為發生變更，這些應用程式依賴所傳回 MQOD 及 MQMD 結構中的特定欄位，但現在包含未定義的值。 應用程式可以指定 MQPMO_SYNC_RESPONSE ，以確保同步執行放置作業，並完成所有適當的欄位值。

C 呼叫

```
MQPUT1 (Hconn, &ObjDesc, &MsgDesc, &PutMsgOpts,
       BufferLength, Buffer, &CompCode, &Reason);
```

宣告參數如下:

```
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 呼叫

```
     CALL 'MQPUT1' USING HCONN, OBJDESC, MSGDESC, PUTMSGOPTS,
                         BUFFERLENGTH, BUFFER, COMPCODE, REASON.
```

宣告參數如下:

```
**   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 呼叫

```
call MQPUT1 (Hconn, ObjDesc, MsgDesc, PutMsgOpts, BufferLength, Buffer,
            CompCode, Reason);
```

宣告參數如下:

```
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 */
```

High Level Assembler 呼叫

```
         CALL MQPUT1,(HCONN,OBJDESC,MSGDESC,PUTMSGOPTS,BUFFERLENGTH,   X
               BUFFER,COMPCODE,REASON)
```

宣告參數如下:

```
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 呼叫

```
MQPUT1 Hconn, ObjDesc, MsgDesc, PutMsgOpts, BufferLength, Buffer,
       CompCode, Reason
```

宣告參數如下:

```
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-設定物件屬性

使用 MQSET 呼叫來變更控點所代表的物件屬性。 物件必須是佇列。

語法

MQSET (*Hconn*、 *Hobj*、 *SelectorCount*、 *Selector*、 *IntAttrCount*、 *IntAttrs*、 *CharAttrLength*、 *CharAttrs*、 *Compcode*、 *Reason*)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 前一個 MQCONN 或 MQCONNX 呼叫已傳回 *Hconn* 的值。

在 z/OS (適用於 CICS 應用程式) 及 IBM i (適用於以相容模式執行的應用程式) 上，可以省略 MQCONN 呼叫，並為 *Hconn* 指定下列值:

MQHC_DEF_HCONN
預設連線控點。

HOBJ
類型 :MQHOBJ-輸入

此控點代表具有要設定之屬性的佇列物件。 控點是由先前指定 MQOO_SET 選項的 MQOPEN 呼叫所傳回。

SelectorCount
類型 :MQLONG-輸入

這是 *Selectors* 陣列中提供的選取元計數。 它是要設定的屬性數目。 零是有效值。 容許的數目上限為 256。

選取器
類型: MQLONGxSelector 計數-輸入

這是 *SelectorCount* 屬性選取元的陣列; 每一個選取元都會識別具有要設定的值的屬性 (整數或字元)。

每一個選取器都必須對 *Hobj* 所代表的佇列類型有效。 只容許某些 MQIA_ * 和 MQCA_ * 值; 如稍後所列。

可以按任何順序指定選取元。 對應於整數屬性選取元 (MQIA_ * 選取元) 的屬性值必須以這些選取元在 *Selectors* 中出現的相同順序在 *IntAttrs* 中指定。 對應於字元屬性選取元 (MQCA_ * 選取元) 的屬性值必須以這些選取元的出現順序在 *CharAttrs* 中指定。 MQIA_ * 選取元可以與 MQCA_ * 選取元交錯; 只有每一個類型內的相對順序很重要。

您可以多次指定相同的選取器; 如果這樣做，則為特定選取器指定的最後一個值會生效。

註:

1. 整數和字元屬性選取元在兩個不同範圍內配置 :MQIA_* 選取元位於 MQIA_FIRST 至 MQIA_LAST 範圍內， MQCA_ * 選取元位於 MQCA_FIRST 至 MQCA_LAST 範圍內。

   對於每一個範圍， MQIA_LAST_USED 和 MQCA_LAST_USED 常數會定義佇列管理程式接受的最高值。
2. 如果所有 MQIA_ * 選取元都先出現，則可以使用相同的元素號碼來處理 *Selectors* 和 *IntAttrs* 陣列中的對應元素。
3. 如果 *SelectorCount* 參數為零，則不會參照 *Selectors* ; 在此情況下，以 C 或 System/390 組合器撰寫的程式所傳遞的參數位址可能是空值。

下表列出可設定的屬性。 無法使用此呼叫來設定其他屬性。 對於 MQCA_ * 屬性選取器，在括弧中提供了定義 *CharAttrs* 中所需字串的長度 (以位元組為單位) 的常數。

表 *571:* 佇列的 *MQSET* 屬性選取器

選取元	說明	附註
MQ 卡 _ 觸發程式資料	觸發資料 (MQ_TRIGGER_DATA_LENGTH)。	

表 571: 佇列的 MQSET 屬性選取器 (繼續)		
選取元	**說明**	**附註**
MQIA_DIST_清單	配送清單支援。	1
MQIA_INHIT_get	是否容許取得作業。	
MQIA_INHIT_PLT	是否容許放置作業。	
MQIA_TRIGGER_CONTROL	觸發控制。	
MQIA_TRIGGER_DEPTH	觸發深度。	
MQIA_TRIGGER_MSG_PRIORITY	觸發程式的臨界值訊息優先順序。	
MQIA_TRIGGER_TYPE	觸發程式類型。	

附註:

1. 僅在 AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows 以及連接至這些系統的 WebSphere MQ MQI 用戶端上受支援。

IntAttrCount
類型 :MQLONG-輸入

這是 *IntAttrs* 陣列中的元素數目，且必須至少為 *Selectors* 參數中 MQIA_ * 選取器的數目。 如果沒有任何值，則零是有效值。

IntAttrs
類型: MQLONGxIntAttrCount -輸入

這是 *IntAttrCount* 整數屬性值的陣列。 這些屬性值必須與 *Selectors* 陣列中 MQIA_ * 選取器的順序相同。

如果 *IntAttrCount* 或 *SelectorCount* 參數為零，則不會參照 *IntAttrs* ; 在此情況下，以 C 或 System/390 組合器撰寫的程式所傳遞的參數位址可能是空值。

***CharAttr* 長度**
類型 :MQLONG-輸入

這是 *CharAttrs* 參數的長度 (以位元組為單位) ，且至少必須是 *Selectors* 陣列中所指定字元屬性的長度總和。 如果 *Selectors* 中沒有 MQCA_ * 選取器，則零是有效值。

CharAttrs
類型: MQCHARxCharAttrLength -輸入

這是包含字元屬性值連結在一起的緩衝區。 緩衝區的長度由 *CharAttrLength* 參數提供。

字元屬性的指定順序必須與 *Selectors* 陣列中 MQCA_ * 選取器的指定順序相同。 每一個字元屬性的長度都是固定的 (請參閱 *Selectors*)。 如果要為屬性設定的值包含的非空白字元數少於屬性定義的長度，請在右側以空白填補 *CharAttrs* 中的值，使屬性值符合屬性定義的長度。

如果 *CharAttrLength* 或 *SelectorCount* 參數為零，則不會參照 *CharAttrs* ; 在此情況下，以 C 或 System/390 組合器撰寫的程式所傳遞的參數位址可能是空值。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204 , X'89C') 配接卡無法使用。

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130 , X'852 ') 無法載入配接卡服務模組。

MQRC_API_EXIT_ERROR
(2374 , X' 946 ') API 結束程式失敗。

MQRC_API_EXIT_LOAD_ERROR
(2183 , X'887 ') 無法載入 API 結束程式。

MQRC_ASDID_MISMATCH
(2157 , X'86D') 主要和起始 ASID 不同。

MQRC_CALL_IN_PROGRESS
(2219 , X'8AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CF_STRUC_FAILED
(2373 , X' 945 ') 連結機能結構失敗。

MQRC_CF_STRUC_IN_USE
(2346 , X'92A') 連結機能結構使用中。

MQRC_CF_STRUC_LIST_HDR_IN_USE
(2347 , X'92B') 連結機能結構清單標頭使用中。

MQRC_CHAR_ATTR_LENGTH_ERROR
(2006 , X'7D6') 字元屬性的長度無效。

MQRC_CHAR_ATTRS_ERROR
(2007 , X'7D7') 字元屬性字串無效。

MQRC_CICS_WAIT_FAILED
(2140 , X'85C') 等待要求被 CICS 拒絕。

MQRC_CONNECTION_BROKEN
(2009 , X'7D9') 與佇列管理程式的連線遺失。

MQRC_CONNECTION_NOT_AUTHORIZED
(2217 , X'8A9') 未獲連線授權。

MQRC_CONNECTION_STOPPING
(2203 , X'89B') 連線關閉。

MQRC_DB2_NOT_AVAILABLE
(2342 , X' 926 ') DB2 子系統無法使用。

MQRC_HCONN_ERROR
(2018 , X'7E2') 連線控點無效。

MQRC_HOBJ_ERROR
(2019 , X'7E3') 物件控點無效。

MQRC_INHIBIT_VALUE_ERROR
(2020 , X'7E4') 禁止-取得或禁止-放置佇列屬性的值無效。

MQRC_INT_ATTR_COUNT_ERROR
(2021 , X'7E5') 整數屬性計數無效。

MQRC_INT_ATTRS_ARRAY_ERROR
(2023 , X'7E7') 整數屬性陣列無效。

MQRC_NOT_OPEN_FOR_SET
(2040 , X'7F8') 佇列未開啟以供設定。

已變更 MQRC_OBJECT_CHANGED
(2041 , X'7F9') 物件定義自開啟以來已變更。

MQRC_OBJECT_DAMAGED
(2101 , X'835 ') 物件已損壞。

MQRC_PAGESET_ERROR
(2193 , X'891 ') 存取頁集資料集時發生錯誤。

MQRC_Q_DELETED
(2052 , X'804 ') 已刪除佇列。

MQRC_Q_MGR_NAME_ERROR
(2058 , X'80A') 佇列管理程式名稱無效或不明。

MQRC_Q_MGR_NOT_AVAILABLE
(2059 , X'80B') 佇列管理程式無法用於連線。

MQRC_Q_MGR_STOPPING
(2162 , X'872 ') 佇列管理程式關閉。

MQRC_RESOURCE_PROBLEM
(2102 , X'836 ') 可用的系統資源不足。

MQRC_SELECTOR_COUNT_ERROR
(2065 , X'811 ') 選取元計數無效。

MQRC_SELECTOR_ERROR
(2067 , X'813 ') 屬性選取器無效。

已超出 MQRC_SELECTOR_LIMIT_EXCEEDED
(2066 , X'812 ') 選取元計數太大。

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'817 ') 可用的儲存體不足。

MQRC_SUPPRESSED_BY_EXIT
(2109 , X'83D') 跳出程式暫停呼叫。

MQRC_TRIGGER_CONTROL_ERROR
(2075 , X'81B') 觸發控制屬性的值無效。

MQRC_TRIGGER_DEPTH_ERROR
(2076 , X'81C') 觸發深度屬性的值無效。

MQRC_TRIGGER_MSG_PRIORITY_ERR
(2077 , X'81D') trigger-message-priority 屬性值無效。

MQRC_TRIGGER_TYPE_ERROR
(2078 , X'81E') trigger-type 屬性的值無效。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

如需這些代碼的詳細資訊，請參閱 原因碼。

使用注意事項

1. 使用此呼叫，應用程式可以指定整數屬性陣列及/或字元屬性字串集合。 如果未發生任何錯誤，則會同步設定所有指定的屬性。 如果發生錯誤 (例如，如果選取元無效，或嘗試將屬性設為無效的值) ，則呼叫會失敗，且不會設定任何屬性。
2. 可以使用 MQINQ 呼叫來判斷屬性值; 如需詳細資料，請參閱 第 618 頁的『MQINQ-查詢物件屬性』 。

   註: 並非所有具有可使用 MQINQ 呼叫來查詢之值的屬性都可以使用 MQSET 呼叫來變更其值。 例如，此呼叫無法設定任何處理程序物件或佇列管理程式屬性。
3. 在重新啟動佇列管理程式時，會保留屬性變更 (除了暫時動態佇列的變更之外，這些變更不會在重新啟動佇列管理程式之後繼續存在)。
4. 您無法使用 MQSET 呼叫來變更模型佇列的屬性。 不過，如果您使用 MQOPEN 呼叫搭配 MQOO_SET 選項來開啟模型佇列，則可以使用 MQSET 呼叫來設定 MQOPEN 呼叫所建立之動態本端佇列的屬性。

5. 如果要設定的物件是叢集佇列，則必須有叢集佇列的本端實例，才能順利開啟。

如需物件屬性的相關資訊，請參閱:

- 第 731 頁的『佇列的屬性』
- 第 758 頁的『名稱清單的屬性』
- 第 760 頁的『程序定義的屬性』
- 第 701 頁的『佇列管理程式的屬性』

C 呼叫

```
MQSET (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs,
      CharAttrLength, CharAttrs, &CompCode, &Reason);
```

宣告參數如下:

```
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 呼叫

```
CALL 'MQSET' USING HCONN, HOBJ, SELECTORCOUNT, SELECTORS-TABLE,
                   INTATTRCOUNT, INTATTRS-TABLE, CHARATTRLENGTH,
                   CHARATTRS, COMPCODE, REASON.
```

宣告參數如下:

```
**   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 呼叫

```
call MQSET (Hconn, Hobj, SelectorCount, Selectors, IntAttrCount,
           IntAttrs, CharAttrLength, CharAttrs, CompCode, Reason);
```

宣告參數如下:

```
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 */
```

High Level Assembler 呼叫

```
CALL MQSET,(HCONN,HOBJ,SELECTORCOUNT,SELECTORS,INTATTRCOUNT,  X
             INTATTRS,CHARATTRLENGTH,CHARATTRS,COMPCODE,REASON)
```

宣告參數如下:

```
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 呼叫

```
MQSET Hconn, Hobj, SelectorCount, Selectors, IntAttrCount, IntAttrs,
      CharAttrLength, CharAttrs, CompCode, Reason
```

宣告參數如下:

```
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-設定訊息內容

使用 MQSET 呼叫來設定或修改訊息控點的內容。

語法

MQSETMP (*Hconn*、 *Hmsg*、 *SetPropOpts*、 名稱、 *PropDesc*、 類型、 *ValueLength*、 值、 *Compcode*、 原因)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。

此值必須符合用來建立 *Hmsg* 參數中所指定訊息控點的連線控點。 如果使用 MQHC_UNASSOCIATED_HCONN 建立訊息控點，則必須在設定訊息控點內容的執行緒上建立有效連線，否則呼叫會失敗，原因碼為 MQRC_CONNECTION_BROKEN。

Hmsg
類型 :MQHMSG-輸入

這是要修改的訊息控點。 此值是由前一個 MQCRTMH 呼叫所傳回。

***SetProp* 選項**
類型 :MQSMPO-輸入

控制如何設定訊息內容。

此結構可讓應用程式指定控制如何設定訊息內容的選項。 此結構是 MQSETMP 呼叫的輸入參數。 如需進一步資訊，請參閱 MQSMPO 。

name
類型 :MQCHARV-輸入

這是要設定的內容名稱。

如需使用內容名稱的進一步相關資訊，請參閱 內容名稱 及 內容名稱限制 。

PropDesc
類型 :MQPD-輸入/輸出

此結構用來定義內容的屬性，包括:

- 如果不支援內容會發生什麼情況
- 內容所屬的訊息環境定義
- 內容在流動時複製到哪些訊息

如需此結構的進一步相關資訊，請參閱 MQPD 。

type
類型 :MQLONG-輸入

所設定內容的資料類型。 它可以是下列其中一項:

MQ 類型 _ 布林
布林。 *ValueLength* 必須是 4。

MQTYPE_BYTE_STRING
位元組字串。 *ValueLength* 必須為零或大於零。

MQTYPE_INT8
8 位元帶正負號的整數。 *ValueLength* 必須是 1。

MQTYPE_INT16
16 位元帶正負號的整數。 *ValueLength* 必須是 2。

MQTYPE_INT32
32 位元帶正負號的整數。*ValueLength* 必須是 4。

MQTYPE_INT64
64 位元帶正負號的整數。*ValueLength* 必須是 8。

MQTYPE_FLOAT32
32 位元浮點數字。 *ValueLength* 必須是 4。

附註: 使用 IBM COBOL for z/OS 的應用程式不支援此類型。

MQTYPE_FLOAT64
64 位元浮點數字。 *ValueLength* 必須是 8。

附註: 使用 IBM COBOL for z/OS 的應用程式不支援此類型。

MQTYPE_STRING
字串。 *ValueLength* 必須為零或以上，或特殊值 MQVL_NULL_TERMINATED。

MQTYPE_NULL
內容存在，但具有空值。*ValueLength* 必須為零。

ValueLength
類型 :MQLONG-輸入

值 參數中內容值的長度 (以位元組為單位)。 零僅對空值或字串或位元組字串有效。 零表示內容存在，但值不包含字元或位元組。

如果 *Type* 參數已設定 MQTYPE_STRING ，則此值必須大於或等於零或下列特殊值:

MQVL_NULL_TERMINATED
該值由字串中發現的第一個空值定界。 字串中不包含空值。 如果未同時設定 MQTYPE_STRING ，則此值無效。

附註: 如果設定 MQVL_NULL_TERMINATED ，則用來終止字串的空值字元是「值」字集中的空值。

value
類型: MQBYTExValue 長度-輸入

要設定的內容值。 緩衝區必須在適合值中資料本質的界限上對齊。

在 C 程式設計語言中，參數宣告為 void 的指標; 任何資料類型的位址都可以指定為參數。

如果 *ValueLength* 為零，則不會參照 值 。 在此情況下，以 C 或 System/390 組合器撰寫的程式所傳遞的參數位址可以是空值。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_WARNING:

MQRC_RFH_FORMAT_ERROR
(2421 ， X'0975 ') 無法剖析包含內容的 MQRFH2 資料夾。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204 ， X'089C') 配接卡無法使用。

MQRC_ADAPTER_SERV_LOAD_ERROR
(2130 ， X'852 ') 無法載入配接卡服務模組。

MQRC_ASDID_MISMATCH
(2157 ， X'86D') 主要和起始 ASID 不同。

MQRC_BUFFER_ERROR
(2004 , X'07D4') 值參數無效。

MQRC_BUFFER_LENGTH_ERROR
(2005 , X'07D5') 值長度參數無效。

MQRC_CALL_IN_PROGRESS
(2219 , X'08AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_HMSG_ERROR
(2460 , X'099C') 訊息控點指標無效。

MQRC_MSG_HANDLE_IN_USE
(2499 , X'09C3') 訊息控點已在使用中。

MQRC_OPTIONS_ERROR
(2046 , X'07FE') 選項無效或不一致。

MQRC_PD_ERROR
(2482 , X'09B2') 內容描述子結構無效。

MQRC_PROPERTY_NAME_ERROR
(2442 , X'098A') 內容名稱無效。

MQRC_PROPERTY_TYPE_ERROR
(2473 , X'09A9') 內容資料類型無效。

MQRC_PROP_NUMBER_FORMAT_ERROR
(2472 , X'09A8') 在值資料中發現數字格式錯誤。

MQRC_SMPO_ERROR
(2463 , X'099F') 設定訊息內容選項結構無效。

MQRC_SOURCE_CCSID_ERROR
(2111 , X'083F') 內容名稱編碼字集 ID 無效。

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'817 ') 可用的儲存體不足。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

如需這些代碼的詳細資訊，請參閱 原因碼。

C 呼叫

```
MQSETMP (Hconn, Hmsg, &SetPropOpts, &Name, &PropDesc, Type,
ValueLength, &Value, &CompCode, &Reason);
```

宣告參數如下:

```
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 呼叫

```
     CALL 'MQSETMP' USING HCONN, HMSG, SETMSGOPTS, NAME, PROPDESC, TYPE,
                          VALUELENGTH, VALUE, COMPCODE, REASON.
```

宣告參數如下:

```
**   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 CMQCHRVV.
**   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 呼叫

```
call MQSETMP (Hconn, Hmsg, SetPropOpts, Name, PropDesc, Type, ValueLength,
              Value, CompCode, Reason);
```

宣告參數如下:

```
dcl Hconn       fixed bin(31);  /* Connection handle */
dcl Hmsg        fixed bin(63);  /* Message handle */
dcl SetPropOpts like MQSMPO;    /* 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 */
```

High Level Assembler 呼叫

```
        CALL MQSETMP,(HCONN,HMSG,SETMSGHOPTS,NAME,PROPDESC,TYPE,VALUELENGTH,
              VALUE,COMPCODE,REASON)
```

宣告參數如下:

```
HCONN        DS         F     Connection handle
HMSG         DS         D     Message handle
SETMSGOPTS   CMQSMPOA   ,     Options that control the action of MQSETMP
NAME         CMQCHRVA   ,     Property name
PROPDESC     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-擷取狀態資訊

使用 MQSTAT 呼叫來擷取狀態資訊。 傳回的狀態資訊類型由呼叫上指定的「類型」值決定。

語法

MQSTAT (*Hconn*、 類型、 *Stat* 、 *Compcode*、 *Reason*)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 前一個 MQCONN 或 MQCONNX 呼叫已傳回 *Hconn* 的值。

在 z/OS (適用於 CICS 應用程式) 及 IBM i (適用於以相容模式執行的應用程式) 上，可以省略 MQCONN 呼叫，並為 *Hconn* 指定下列值:

MQHC_DEF_HCONN
預設連線控點。

type
類型 :MQLONG-輸入

所要求的狀態資訊類型。 > 有效值為:

MQSTAT_TYPE_ASYNC_ERROR
傳回先前非同步放置作業的相關資訊。

MQSTAT_TYPE_RECONNECTION

傳回重新連線的相關資訊。 如果連線正在重新連接或無法重新連接，則資訊會說明導致連線開始重新連接的失敗。

此值僅適用於用戶端連線。 對於其他類型的連線，呼叫失敗，原因碼為 **MQRC_ENVIRONMENT_ERROR**

MQSTAT_TYPE_RECONNECTION_ERROR

傳回與重新連接相關的前一個失敗的相關資訊。 如果連線無法重新連接，資訊會說明導致重新連線失敗的失敗。

此值僅適用於用戶端連線。 對於其他類型的連線，呼叫失敗，原因碼為 **MQRC_ENVIRONMENT_ERROR** 。

Stat
類型 :MQSTS-輸入/輸出

狀態資訊結構。 請參閱第 513 頁的『MQSTS-狀態報告結構』，以取得詳細資料。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_API_EXIT_ERROR
(2374 , X' 946 ') API 結束程式失敗

MQRC_API_EXIT_LOAD_ERROR
(2183 , X'887 ') 無法載入 API 結束程式。

MQRC_CALL_IN_PROGRESS
(2219 , X'8AB') 在前一個呼叫完成之前輸入的 MQI 呼叫。

MQRC_CONNECTION_BROKEN
(2009 , X'7D9') 與佇列管理程式的連線遺失。

MQRC_CONNECTION_STOPPING
(2203 , X'89B') 連線關閉。

MQRC_FUNCTION_NOT_SUPPORTED
(2298 , X'8FA') 在現行環境中無法使用所要求的功能。

MQRC_HCONN_ERROR
(2018 , X'7E2') 連線控點無效。

MQRC_Q_MGR_STOPPING
(2162,X'872'-佇列管理程式停止中

MQRC_RESOURCE_PROBLEM
(2102 , X'836 ') 可用的系統資源不足。

MQRC_STAT_TYPE_ERROR
(2430 , X'97E' MQSTAT 類型發生錯誤

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'817 ') 可用的儲存體不足。

MQRC_STS_ERROR
(2426 , X'97A') MQSTS 結構發生錯誤

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

如需這些代碼的詳細資訊，請參閱 原因碼。

使用注意事項

1. 對 MQSTAT 指定 MQSTAT_TYPE_ASYNC_ERROR 類型的呼叫會傳回先前非同步 MQPUT 及 MQPUT1 作業的相關資訊。 從 MQSTAT 呼叫傳回時傳回的 MQSTS 結構包含該連線第一個記錄的非同步警告或錯誤資訊。 如果第一個之後有進一步的錯誤或警告，它們通常不會變更這些值。 不過，如果發生錯誤，完成碼為 MQCC_WARNING，則會改為傳回後續失敗，完成碼為 MQCC_FAILED 。
2. 如果自建立連線之後或自前次呼叫 MQSTAT 之後未發生任何錯誤，則會在 MQSTS 結構中傳回 MQCC_OK 的 CompCode 及 MQRC_NONE 的「原因」。
3. 透過三個計數器欄位 (PutSuccessCount、 PutWarningCount 及 PutFailureCount) 傳回在連線控點下已處理的非同步呼叫數。 每次順利處理非同步作業、發出警告或失敗時，佇列管理程式都會增加這些計數器 (請注意，基於統計目的，每個目的地佇列的放置配送清單計數一次，而不是每個配送清單一次)。 計數器不會增加到超過最大值 AMQ_LONG_MAX。
4. 成功呼叫 MQSTAT 會導致重設任何先前的錯誤資訊或計數。
5. MQSTAT 的行為視您提供的 MQSTAT Type 參數值而定。
6. **MQSTAT_TYPE_ASYNC_ERROR**
    a. 對 MQSTAT 指定 MQSTAT_TYPE_ASYNC_ERROR 類型的呼叫會傳回先前非同步 MQPUT 及 MQPUT1 作業的相關資訊。 從 MQSTAT 呼叫傳回時傳回的 MQSTS 結構包含該連線第一個記錄的非同步警告或錯誤資訊。 如果第一個之後有進一步的錯誤或警告，它們通常不會變更這些值。 不過，如果發生錯誤，完成碼為 MQCC_WARNING，則會改為傳回後續失敗，完成碼為 MQCC_FAILED 。
    b. 如果自建立連線之後或自前次呼叫 MQSTAT 之後未發生任何錯誤，則會在 MQSTS 結構中傳回 MQCC_OK 的 CompCode 及 MQRC_NONE 的「原因」。
    c. 透過三個計數器欄位 (PutSuccessCount、 PutWarningCount 及 PutFailureCount) 傳回在連線控點下已處理的非同步呼叫數。 每次順利處理非同步作業、發出警告或失敗時，佇列管理程式都會增加這些計數器 (請注意，基於統計目的，每個目的地佇列的放置配送清單計數一次，而不是每個配送清單一次)。 計數器不會增加到超過最大值 AMQ_LONG_MAX。

d. 成功呼叫 MQSTAT 會導致重設任何先前的錯誤資訊或計數。

MQSTAT_TYPE_RECONNECTION

假設您在重新連線期間在事件處理程式內呼叫 MQSTAT，並將 Type 設為 MQSTAT_TYPE_RECONNECTION。請考量下列範例。

用戶端正在嘗試重新連線或無法重新連接。
MQSTS 結構中的 CompCode 是 MQCC_FAILED，Reason 可能是 MQRC_CONNECTION_BROKEN 或 MQRC_Q_MGR_QUIESCING。ObjectType 是 MQOT_Q_MGR，ObjectName 是佇列管理程式的名稱，ObjectQMgrName 是空白。

用戶端已順利完成重新連線，或從未中斷連線。
MQSTS 結構中的 CompCode 是 MQCC_OK，而 Reason 是 MQRC_NONE

後續呼叫 MQSTAT 會傳回相同的結果。

MQSTAT_TYPE_RECONNECTION_ERROR

假設您呼叫 MQSTAT，並將 Type 設為 MQSTAT_TYPE_RECONNECTION_ERROR，以回應接收 MQRC_RECONNECT_FAILED MQI 呼叫。請考量下列範例。

在重新連線至不同佇列管理程式期間重新開啟佇列時，發生授權失敗。
MQSTS 結構中的 CompCode 是 MQCC_FAILED，而 Reason 是重新連線失敗的原因，例如 MQRC_NOT_AUTHORIZED。ObjectType 是造成問題的物件類型，例如 MQOT_QUEUE，ObjectName 是佇列名稱，ObjectQMgrName 是擁有佇列的佇列管理程式名稱。

重新連線期間發生 Socket 連線錯誤。
MQSTS 結構中的 CompCode 是 MQCC_FAILED，而 Reason 是重新連線失敗的原因，例如 MQRC_HOST_NOT_AVAILABLE。ObjectType 是 MQOT_Q_MGR，ObjectName 是佇列管理程式的名稱，ObjectQMgrName 是空白。

後續呼叫 MQSTAT 會傳回相同的結果。

C 呼叫

```
MQSTAT (Hconn, StatType, &Stat, &CompCode, &Reason);
```

宣告參數如下:

```
MQHCONN Hconn;        /* Connection Handle */
MQLONG StatType;    /* Status type */
MQSTS Stat;            /* Status information structure */
MQLONG CompCode;    /* Completion code */
MQLONG Reason;        /* Reason code qualifying CompCode */
```

COBOL 呼叫

```
CALL 'MQSTAT' USING HCONN, STATTYPE, STAT, COMPCODE, REASON.
```

宣告參數如下:

```
**    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 呼叫

```
call MQSTAT (Hconn, StatType, Stat, Compcode, Reason);
```

宣告參數如下:

```
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 組譯器呼叫

```
CALL MQSTAT,(HCONN,STATTYPE,STAT,COMPCODE,REASON)
```

宣告參數如下:

```
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-登錄訂閱

使用 MQSUB 呼叫來登錄特定主題的應用程式訂閱。

語法

MQSUB (*Hconn*、 *SubDesc*、 *Hobj*、 *Hsub*、 *Compcode* 、 *Reason*)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 前一個 MQCONN 或 MQCONNX 呼叫已傳回 *Hconn* 的值。

在 z/OS (適用於 CICS 應用程式) 及 IBM i (適用於以相容模式執行的應用程式) 上，可以省略 MQCONN 呼叫，並為 *Hconn* 指定下列值:

MQHC_DEF_HCONN
預設連線控點。

SubDesc
類型 :MQSD-輸入/輸出

此結構可識別應用程式正在登錄的使用中物件。 如需相關資訊，請參閱第 490 頁的『MQSD-訂閱描述子』。

HOBJ
類型 :MQHOBJ-輸入/輸出

此控點代表為了取得傳送至此訂閱的訊息而建立的存取權。 這些訊息可以儲存在特定佇列上，或佇列管理程式可以管理其儲存體而不使用特定佇列。

若要使用特定佇列，您必須在建立訂閱時將它與訂閱相關聯。 您可以使用兩種方式來執行此動作:

- 透過使用 DEFINE SUB MQSC 指令，並提供該指令與佇列物件名稱。
- 使用 MQSO_CREATE 呼叫 MQSUB 時提供此控點

  如果提供此控點作為呼叫上的輸入參數，則它必須是從佇列的前一個 MQOPEN 呼叫中使用下列至少一個選項所傳回的有效物件控點:

– MQOO_INPUT_ *
– MQ 瀏覽
– MQOO_OUTPUT (如果佇列是遠端佇列)

如果不是這種情況，則呼叫會失敗，並產生 MQRC_HOBJ_ERROR。 它不能是解析為主題物件之別名佇列的物件控點。 如果是這樣，則呼叫會失敗，並產生 MQRC_HOBJ_ERROR。

如果佇列管理程式要管理傳送至此訂閱的訊息儲存體，則應該在建立訂閱時使用 MQSO_MANAGED 選項來設定此值。 然後，佇列管理程式會傳回此控點作為呼叫的輸出參數。 傳回的控點稱為受管理控點。 如果指定 MQHO_NONE 但未指定 MQSO_MANAGED ，則呼叫會失敗，並產生 MQRC_HOBJ_ERROR。

當佇列管理程式將受管理控點傳回給您時，您可以在具有或不具有瀏覽選項的 MQGET 或 MQCB 呼叫、MQINQ 呼叫或 MQCLOSE 上使用它。 您無法在 MQPUT、MQSUB、MQSET 上使用它; 嘗試這樣做會失敗，並產生 MQRC_NOT_OPEN_FOR_OUTPUT、MQRC_HOBJ_ERROR 或 MQRC_NOT_OPEN_FOR_SET。

如果正在使用 MQSD 結構中的 MQSO_RESUME 選項來回復此訂閱，則可以透過將 MQSO_MANAGED 設為 MQHO_NONE ，將控點傳回此參數中的應用程式。 不論訂閱是否使用受管理控點，您都可以執行此動作，而且提供使用 DEFINE SUB 所建立的訂閱與該指令上所定義的訂閱佇列控點。 在回復以管理方式建立的訂閱的情況下，佇列會以 MQOO_INPUT_AS_Q_DEF 和 MQOO_BROWSE 開啟。 如果您需要指定其他選項，應用程式必須明確開啟訂閱佇列，並在呼叫時提供物件控點。 如果開啟佇列時發生問題，則呼叫會失敗，並產生 MQRC_INVALID_DESTINATION。 如果提供 *Hobj* ，則它必須相當於原始 MQSUB 呼叫中的 *Hobj* 。 這表示如果提供從 MQOPEN 呼叫傳回的物件控點，則該控點必須與先前使用的佇列相同。 如果不是相同的佇列，則呼叫會失敗，並產生 MQRC_HOBJ_ERROR。

如果使用 MQSD 結構中的 MQSO_ALTER 選項來變更此訂閱，則可以提供不同的 *Hobj* 。 任何已遞送至佇列且先前透過此參數所識別的發佈都會留在該佇列上，而且如果 *Hobj* 參數現在代表不同的佇列，則應用程式會負責擷取那些訊息。

下表彙總此參數與各種訂閱選項的使用:

選項	***Hobj***	**說明**
MQSO_CREATE + MQSO_MANAGED	輸入時忽略	建立訂閱並儲存佇列管理程式所管理的訊息
MQSO_CREATE	有效的物件控點	建立訂閱，以提供特定佇列作為訊息的目的地。
MQ 回復	MQHO_NONE	回復先前建立的訂閱 (不論它是否受管理) ，並讓佇列管理程式傳回物件控點供應用程式使用。
MQ 回復	有效、相符、物件控點	回復先前建立的訂閱，使用特定佇列作為訊息的目的地，並使用具有特定開啟選項的物件控點。
MQSO_ALTER + MQSO_MANAGED	MQHO_NONE	變更先前使用特定佇列的現有訂閱，因此它現在是受管理訂閱。 無法變更目的地 (受管理或未受管理) 的類別。
MQSO_ALTER	有效的物件控點	變更現有訂閱 (不論是否受管理) ，以便它現在使用特定佇列。 未使用 MQSO_MANAGED 選項時，可以變更提供的佇列，但無法變更目的地的類別 (受管理或未受管理)。

不論是否已提供或傳回，都必須在後續的 MQGET 或 MQCB 呼叫上指定 *Hobj* ，以接收傳送至此訂閱的發佈訊息。

當對 *Hobj* 控點發出 MQCLOSE 呼叫時，或當定義控點範圍的處理單元終止時 (直到應用程式中斷連線為止)，該控點不再有效。 傳回的物件控點範圍與呼叫上指定的連線控點範圍相同。 如需控點範圍的相關資訊，請參閱 Hconn (MQHCONN)-output 。 *Hobj* 控點的 MQCLOSE 不會影響 *Hsub* 控點。

HSub
類型 :MQHOBJ-輸出

此控點代表已建立的訂閱。 它可以用於兩個進一步的作業:

- 它可以在後續的 MQSUBRQ 呼叫上使用，以要求在建立訂閱時使用 MQSO_PUBLICATIONS_ON_REQUEST 選項時傳送發佈。
- 它可以在後續的 MQCLOSE 呼叫中使用，以移除已建立的訂閱。 當發出 MQCLOSE 呼叫時，或當定義控點範圍的處理單元終止時， *Hsub* 控點即不再有效。 傳回的物件控點範圍與呼叫上指定的連線控點範圍相同。 *Hsub* 控點的 MQCLOSE 不會影響 *Hobj* 控點。

此控點無法傳遞至 MQGET 或 MQCB 呼叫。 您必須使用 *Hobj* 參數。 您無法在 MQCLOSE 或 MQSUBRQ 以外的任何 WebSphere MQ 呼叫上使用此控點。 將此控點傳遞至任何其他 WebSphere MQ 呼叫會導致 MQRC_HOBJ_ERROR。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成

MQCC_WARNING
警告 (局部完成)

MQCC_FAILED
通話失敗

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK，則原因碼如下:

MQRC_NONE
(0， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED，則原因碼為下列其中一項:

MQRC_CLUSTER_RESOLUTION_ERROR
(2189， X'88D') 叢集名稱解析失敗。

MQRC_DURABILITY_NOT_ALLOWED
2436 (X'0984 ') 使用 MQSO_DURABLE 選項的 MQSUB 呼叫失敗。

MQRC_FUNCTION_NOT_SUPPORTED
2298 (X'08FA') 在現行環境中無法使用所要求的功能。

MQRC_HOBJ_ERROR
2019 (X'07E3') 物件控點 Hobj 無效。

MQRC_IDENTITY_MISMATCH
2434 (X'0982 ') 訂閱名稱符合現有訂閱。

MQRC_NOT_AUTHORIZED
2035 (X'07F3') 使用者未獲授權執行作業。

MQRC_OBJECT_STRING_ERROR
2441 (X'0989 ') Objectstring 欄位無效。

MQRC_OPTIONS_ERROR
2046 (X'07FE') 選項參數或欄位包含無效的選項或無效的選項組合。

MQRC_Q_MGR_QUIESCING
2161 (X'0871 ') 佇列管理程式靜止中。

MQRC_RECONNECT_Q_MGR_REQD
2555 (X'09FB' X) 需要 MQCNO_RECONNECT_Q_MGR 選項。

MQRC_RETAINED_MSG_Q_ERROR
2525 (X'09DD') 無法擷取已訂閱主題字串的已保留發佈資訊。

MQRC_RETAINED_NOT_DELIVERED
2526 (X'09DE') 已訂閱主題字串的保留發佈資訊無法遞送至訂閱目的地佇列，也無法遞送至無法傳送郵件的佇列。

MQRC_SD_ERROR
2424 (X'0978 ') 訂閱描述子 (MQSD) 無效。

MQRC_SELECTION_NOT_AVAILABLE
2551 (X'09F7') 選取字串未遵循 WebSphere MQ 選取器語法，且沒有可用的延伸訊息選取提供者。

MQRC_SELECTION_STRING_ERROR
2519 (X'09D7') 必須依照 MQCHARV 結構文件的說明來指定選取字串。

MQRC_SELECTOR_SYNTAX_ERROR
2459 (X'099B') 已發出 MQOPEN、 MQPUT1 或 MQSUB 呼叫，但指定了包含語法錯誤的選取字串。

MQRC_SUB_USER_DATA_ERROR
2431 (X'097F') SubUser 資料欄位無效。

MQRC_SUB_NAME_ERROR
2440 (X'0988 ') SubName 欄位無效。

MQRC_SUB_ALREADY_EXISTS
2432 (X'0980 ') 訂閱已存在。

MQRC_SUB_USER_DATA_ERROR
2431 (X'097F') SubUser 資料欄位無效。

MQRC_TOPIC_STRING_ERROR
2425 (X'0979 ') 主題字串無效。

MQRC_UNKNOWN_OBJECT_NAME
2085 (X'0825 ') 找不到識別的物件。

如需這些代碼的詳細資訊，請參閱 原因碼。

使用注意事項

1. 訂閱是對主題進行，使用預先定義主題物件的簡稱、主題字串的完整名稱來命名，或由兩個部分連結而成。 請參閱 第 490 頁的『MQSD-訂閱描述子』中 *ObjectName* 和 *ObjectString* 的說明。
2. 當發出 MQSUB 呼叫時，佇列管理程式會執行安全檢查，以驗證執行應用程式的使用者 ID 在允許存取之前具有適當的權限層次。 適當的主題物件位於主題階層中，且會對此主題物件進行權限檢查，以確保已設定訂閱權限。 如果未使用 MQSO_MANAGED 選項，則會對目的地佇列進行權限檢查，以確保設定輸出的權限。 如果使用 MQSO_MANAGED 選項，則不會對受管理佇列進行輸出或查詢存取權的權限檢查。
3. 如果您未提供 Hobj 作為輸入， MQSUB 呼叫會配置兩個控點: 物件控點 (Hobj) 和訂閱控點 (Hsub)。
4. 當使用 MQSO_MANAGED 選項時，會在 MQSUB 呼叫中傳回 Hobj ，可以進行查詢，以找出諸如「取消」臨界值及「過多取消重新排入佇列」名稱之類的屬性。 您也可以查詢受管理佇列的名稱，但不得嘗試直接開啟此佇列。
5. 訂閱可以分組，即使多個群組符合發佈，也只容許將單一發佈遞送至訂閱群組。 訂閱會使用 MQSO_GROUP_SUB 選項進行分組，為了將訂閱分組，它們必須是
   - 在相同的佇列管理程式上使用相同的具名佇列 (不使用 MQSO_MANAGED 選項)-由 MQSUB 呼叫上的 Hobj 參數代表
   - 共用相同的 SubCorrelID
   - 屬於相同的 SubLevel

   這些屬性定義視為在群組中的訂閱集，同時也是在訂閱分組時無法變更的屬性。 變更 SubLevel 會導致 MQRC_SUBLEVEL_NOT_ALTERABLE ， 而變更任何其他 (如果未分組訂閱則可以變更) 會導致 MQRC_GROUPING_NOT_ALTERABLE。

6. 從使用 MQSO_RESUME 選項的 MQSUB 呼叫傳回時，會填寫 MQSD 中的欄位。 傳回的 MQSD 可以直接傳遞至 MQSUB 呼叫，該呼叫會使用 MQSO_ALTER 選項，且您需要對套用至 MQSD 的訂閱進行任何變更。 如表格中所述，部分欄位有特殊考量。

來自 MQSUB 的 MQSD 輸出	
MQSD 中的欄位名稱	**特殊考量**
存取或建立選項	從 MQSUB 呼叫返回時可以重設部分選項。 如果您隨後在 MQSUB 呼叫中重複使用 MQSD，則必須明確設定您需要的選項。
延續性選項、目的地選項、登錄選項及萬用字元選項	視需要設定這些選項
發佈選項	這些選項會適當地設定，但 MQSO_NEW_PUBLICATIONS_ONLY 除外，它只適用於 MQSO_CREATE。
其他選項	從 MQSUB 呼叫傳回時，這些選項保持不變。 它們控制如何發出 API 呼叫，且不會與訂閱一起儲存。 必須在重複使用 MQSD 的任何後續 MQSUB 呼叫上視需要設定它們。
ObjectName	從 MQSUB 呼叫傳回時，此僅輸入欄位保持不變。
ObjectString	從 MQSUB 呼叫傳回時，此僅輸入欄位保持不變。 如果提供緩衝區，則會在 *ResObjectString* 欄位中傳回使用的完整主題名稱。
AlternateUserId 和 AlternateSecurityId	從 MQSUB 呼叫傳回時，這些僅輸入欄位保持不變。 它們控制如何發出 API 呼叫，且不會與訂閱一起儲存。 它們必須在重複使用 MQSD 的任何後續 MQSUB 呼叫上視需要設定。
SubExpiry	使用 MQSO_RESUME 選項從 MQSUB 呼叫返回時，此欄位會設為訂閱的原始到期，而不是剩餘到期時間。 然後，如果您使用 MQSO_ALTER 選項在 MQSUB 呼叫中重複使用 MQSD，則會重設訂閱的期限，以重新開始倒數。
SubName	此欄位是 MQSUB 呼叫的輸入欄位，在輸出時不會變更。
SubUserData and SelectionString	如果提供緩衝區，則在 MQSUB 呼叫使用 MQSO_RESUME 選項輸出時，會傳回這些可變長度欄位，同時在 *VSBufSize* 中也會傳回正的緩衝區長度。 如果未提供任何緩衝區，則只會在 MQCHARV 的 *VSLength* 欄位中傳回長度。 如果提供的緩衝區小於傳回欄位所需的空間，則只會在提供的緩衝區中傳回 *VSBufSize* 個位元組。 然後，如果您使用 MQSO_ALTER 選項在 MQSUB 呼叫中重複使用 MQSD，且未提供緩衝區，但提供非零 *VSLength*，如果該長度符合欄位的現有長度，則不會對欄位進行任何變更。

來自 MQSUB 的 MQSD 輸出 *(繼續)*	
MQSD 中的欄位名稱	**特殊考量**
SubCorrelID 和 PubAccounting 記號	如果您不使用 MQSO_SET_CORREL_ID，則佇列管理程式會產生 *SubCorrelId*。 如果您不使用 MQSO_SET_IDENTITY_CONTEXT，則佇列管理程式會產生 *PubAccountingToken*。 使用 MQSO_RESUME 選項，在 MQSD 中從 MQSUB 呼叫傳回這些欄位。 如果由佇列管理程式產生，則會使用 MQSO_CREATE 或 MQSO_ALTER 選項，在 MQSUB 呼叫上傳回產生的值。
PubPriority, SubLevel & PubApplIdentityData	這些欄位會在 MQSD 中傳回。
ResObject 字串	如果提供緩衝區，則 MQSD 中只會傳回此輸出欄位。

C 呼叫

```
MQSUB (Hconn, &SubDesc, &Hobj, &Hsub, &CompCode, &Reason)
```

宣告參數如下:

```
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 呼叫

```
CALL 'MQSUB' USING HCONN, SUBDESC, HOBJ, HSUB, COMPCODE, REASON.
```

宣告參數如下:

```
**   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 呼叫

```
call MQSUB (Hconn, SubDesc, Hobj, Hsub, CompCode, Reason)
```

宣告參數如下:

```
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 */
```

High Level Assembler 呼叫

```
CALL MQSUB,(HCONN,SUBDESC,HOBJ,HSUB,COMPCODE,REASON)
```

宣告參數如下:

```
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-訂閱要求

當訂閱者已向 MQSO_PUBLICATIONS_ON_REQUEST 登錄時，請使用 MQSUBRQ 呼叫來要求保留的發佈。

語法

MQSUBRQ (*Hconn*、 *Hsub*、 動作、 *SubRqOpts*、 *Compcode*、 *Reason*)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。 前一個 MQCONN 或 MQCONNX 呼叫已傳回 *Hconn* 的值。

在 z/OS (適用於 CICS 應用程式) 及 IBM i (適用於以相容模式執行的應用程式) 上，可以省略 MQCONN 呼叫，並為 *Hconn* 指定下列值:

MQHC_DEF_HCONN
預設連線控點。

HSub
類型 :MQHOBJ-輸入

此控點代表要要求更新的訂閱。 前一個 MQSUB 呼叫已傳回 *Hsub* 值。

Action
類型 :MQLONG-輸入

此參數控制正在訂閱上要求的特定動作。 必須指定下列值:

MQSR_ACTION_PUBLICATION

此動作會要求傳送所指定主題的更新發佈資訊。 只有在訂閱者在進行訂閱時對 MQSUB 呼叫指定了 MQSO_PUBLICATIONS_ON_REQUEST 選項時，才能使用它。 如果佇列管理程式具有主題的保留發佈資訊，則會傳送至訂閱者。 否則，通話會失敗。 如果應用程式傳送已保留的發佈資訊，則該發佈資訊的 MQIsRetained 訊息內容會指出此情況。

因為 Hsub 參數所代表的現有訂閱中的主題可以包含萬用字元，所以訂閱者可能會收到多個保留的發佈。

***SubRq* 選項**
類型 :MQSRO-輸入/輸出

這些選項控制 MQSUBRQ 的動作，如需詳細資料，請參閱 第 511 頁的『MQSRO-訂閱要求選項』 。

如果不需要任何選項，則以 C 或 S/390 組合器撰寫的程式可以指定空值參數位址，而不是指定 MQSRO 結構的位址。

CompCode
類型 :MQLONG-輸出

完成碼; 它是下列其中一項:

MQCC_OK
順利完成

MQCC_WARNING
警告 (局部完成)

MQCC_FAILED
通話失敗

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_FUNCTION_NOT_SUPPORTED
2298 (X'08FA') 在現行環境中無法使用所要求的功能。

MQRC_NO_RETAINED_MSG
2437 (X'0985 ') 目前未儲存此主題的保留發佈資訊。

MQRC_OPTIONS_ERROR
2046 (X'07FE') 選項參數或欄位包含無效的選項或無效的選項組合。

MQRC_Q_MGR_QUIESCING
2161 (X'0871 ') 佇列管理程式靜止中。

MQRC_SRO_ERROR
2438 (X'0986 ') 在 MQSUBRQ 呼叫上，「訂閱要求選項」MQSRO 無效。

MQRC_RETAINED_MSG_Q_ERROR
2525 (X'09DD') 無法擷取已訂閱主題字串的已保留發佈資訊。

MQRC_RETAINED_NOT_DELIVERED
2526 (X'09DE') 已訂閱主題字串的保留發佈資訊無法遞送至訂閱目的地佇列，也無法遞送至無法傳送郵件的佇列。

如需這些代碼的詳細資訊，請參閱 原因碼。

使用注意事項

下列使用注意事項適用於使用動作碼 MQSR_ACTION_PUBLICATION:

1. 如果此動詞順利完成，則符合指定訂閱的保留發佈已傳送至訂閱，且可以使用建立訂閱的原始 MQSUB 動詞所傳回的 Hobj , 使用 MQGET 或 MQCB 來接收。
2. 如果建立訂閱的原始 MQSUB 動詞所訂閱的主題包含萬用字元，則可以傳送多個保留的發佈資訊。 由於此呼叫而傳送的發佈數記錄在 SubRqOpts 結構的 NumPubs 欄位中。
3. 如果此動詞完成且原因碼為 MQRC_NO_RETAINED_MSG , 則目前沒有指定主題的保留發佈。 #
4. 如果此動詞完成且原因碼為 MQRC_RETAINED_MSG_Q_ERROR 或 MQRC_RETAINED_NOT_DELIVERED , 則目前會保留所指定主題的發佈資訊，但發生錯誤，表示它們無法遞送。
5. 應用程式必須具有主題的現行訂閱，才能進行此呼叫。 如果已在應用程式的前一個實例中進行訂閱，且無法使用訂閱的有效控點，則應用程式必須先使用 MQSO_RESUB 選項來呼叫 MQSUB , 以取得其控點，以便在此呼叫中使用。
6. 發佈會傳送至已登錄要與此應用程式的現行訂閱搭配使用的目的地。 如果必須將發佈傳送至其他地方，則必須先使用 MQSUB 呼叫搭配 MQSO_ALTER 選項來變更訂閱。

C 呼叫

```
MQSUB (Hconn, Hsub, Action, &SubRqOpts, &CompCode, &Reason)
```

宣告參數如下:

```
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 呼叫

```
CALL 'MQSUBRQ' USING HCONN, HSUB, ACTION, SUBRQOPTS, COMPCODE, REASON.
```

宣告參數如下:

```
** 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 呼叫

```
call MQSUBRQ (Hconn, Hsub, Action, SubRqOpts, CompCode, Reason)
```

宣告參數如下:

```
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 MQSRO; /* Options that control the action of MQSUBRQ */
dcl CompCode fixed bin(31); /* Completion code */
dcl Reason fixed bin(31); /* Reason code qualifying CompCode */
```

High Level Assembler 呼叫

```
CALL MQSUBRQ,(HCONN, HSUB, ACTION, SUBRQOPTS,COMPCODE,REASON)
```

宣告參數如下:

```
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
```

物件的屬性

此主題集合僅列出可作為 MQINQ 函數呼叫主旨的那些 WebSphere MQ 物件，並提供可查詢的屬性及要使用的選取器的詳細資料。

佇列管理程式的屬性

部分佇列管理程式屬性對於特定實作是固定的; 其他屬性可以使用 MQSC 指令 ALTER QMGR 來變更。

也可以使用指令 DISPLAY QMGR 來顯示屬性。 開啟特殊 MQOT_Q_MGR 物件，並搭配使用 MQINQ 呼叫與傳回的控點，可以查詢大部分佇列管理程式屬性。

下表彙總特定於佇列管理程式的屬性。 屬性按字母順序說明。

註: 本節中顯示的屬性名稱是與 MQINQ 呼叫搭配使用的敘述性名稱; 這些名稱與 PCF 指令的名稱相同。 當使用 MQSC 指令來定義、變更或顯示屬性時，會使用替代簡稱; 如需相關資訊，請參閱 Script (MQSC) 指令 。

表 *572:* 佇列管理程式的屬性.
含有鏈結和簡要說明的佇列管理程式屬性清單

屬性	說明
AccountingConnOverride	置換帳戶設定。
AccountingInterval	寫入中間帳戶記錄的頻率。
ActivityConnOverride	置換活動設定。
ActivityTrace	控制 WebSphere MQ MQI 應用程式活動追蹤的收集。
AdoptNewMCACheck	已檢查元素以決定是否採用新的 MCA。
AdoptNewMCAType	是否要自動重新啟動特定通道類型 MCA 的孤立實例。
AlterationDate	前次變更定義的日期
AlterationTime	前次變更定義的時間
AuthorityEvent	控制是否產生授權 (未獲授權) 事件
BridgeEvent	橋接器事件的控制屬性。
ChannelAutoDef	控制是否允許自動通道定義
ChannelAutoDefEvent	控制是否產生通道自動定義事件
ChannelAutoDefExit	自動通道定義的使用者結束程式名稱
ChannelEvent	通道事件的控制屬性。
ChannelInitiatorControl	通道起始程式的控制屬性
ChannelMonitoring	通道的線上監視資料
ChannelStatistics	控制通道統計資料的收集。
ChinitAdapters	用於處理 WebSphere MQ 呼叫的配接器子作業數。
ChinitDispatchers	用於通道起始程式的分派器數目。
	保留供 IBM 使用。
ChinitTraceAutoStart	通道起始程式追蹤是否應該自動啟動。
ChinitTraceTableSize	通道起始程式追蹤資料空間的大小。
ClusterSenderMonitoringDefault	叢集傳送端通道的線上監視資料預設值
ClusterSender 統計資料	控制收集叢集傳送端通道的統計資料監視資訊。
ClusterWorkloadData	叢集工作量結束程式的使用者資料
ClusterWorkloadExit	叢集工作量管理的使用者結束程式名稱
ClusterWorkloadLength	傳遞至叢集工作量結束程式的訊息資料長度上限
CLWLMRUChannels	叢集工作量平衡最近使用的通道數
CLWLUseQ	叢集工作量使用遠端佇列。
CodedCharSetId	編碼字集 ID
CommandEvent	指令事件的控制屬性。
CommandInput 完整名稱屬性	指令輸入佇列名稱
CommandLevel	指令層次
CommandServerControl 屬性	指令伺服器的控制屬性。

表 *572:* 佇列管理程式的屬性.
含有鏈結和簡要說明的佇列管理程式屬性清單
(繼續)

屬性	說明
配置事件屬性	配置事件的控制屬性。
DeadLetterQName	無法傳送郵件的佇列名稱
DEFCLXQ	預設叢集傳輸佇列類型
DefXmitQName	預設傳輸佇列名稱
DistLists	配送清單支援
DNSGroup	使用「工作量管理程式動態網域名稱服務」支援時 TCP 接聽器的群組名稱。
DNSWLM	TCP 接聽器是否向 Workload Manager for Dynamic Domain Name Services 登錄。
ExpiryInterval	掃描過期訊息的間隔
IGQPutAuthority	內部群組佇列作業放置權限
IGQUserId	內部群組佇列作業使用者 ID
InhibitEvent	控制是否產生禁止 (禁止取得及禁止放置) 事件
IPAddressVersion	Internet Protocol 位址的版本
IntraGroupQueuing	內部群組佇列作業支援
ListenerTimer	在 APPC 或 TCP/IP 失敗之後，嘗試重新啟動接聽器之間的時間間隔。
LocalEvent	控制是否產生本端錯誤事件
LoggerEvent	控制是否產生日誌程式事件
LUGroupName	處理佇列共用群組之入埠傳輸的 LU 6.2 接聽器的一般 LU 名稱。
LUName	用於出埠 LU 6.2 傳輸的 LU 名稱。
LU62ARMSuffix	SYS1.PARMLIB 成員 APPCPMxx，指定此通道起始程式的 LUADD。
LU62Channels	使用 LU 6.2 的現行通道或已連接用戶端數目上限。
MaxActiveChannels	隨時可處於作用中的通道數上限。
MaxChannels	現行通道數上限。
MaxHandles	控點數目上限
MaxMsgLength	訊息長度上限（以位元組為單位）
MaxPriority 屬性	最大優先順序
MaxPropertiesLength	內容資料的長度上限 (以位元組為單位)
MaxUncommittedMsgs	工作單元內未確定的訊息數上限
MQIAccounting	控制 MQI 資料的帳戶資訊收集。
MQIStatistics	控制收集佇列管理程式的統計資料監視資訊。
MsgMarkBrowseInterval	間隔，在此之後佇列管理程式可以從瀏覽的訊息中移除標示。
OutboundPortMin	使用 *OutboundPortMin*，定義連結送出通道時要使用的埠號範圍。
OutboundPortMin	使用 *OutboundPortMax*，定義連結送出通道時要使用的埠號範圍。
PerformanceEvent	控制是否產生效能相關事件
平台	執行佇列管理程式的平台
PubSubNPInputMsg	是否捨棄 (或保留) 未遞送的輸入訊息
PubSubNPResponse	控制未遞送的行為
PubSubMaxMsgRetryCount	處理 (在同步點下) 失敗指令訊息時的重試次數
PubSubSyncPoint	是否只應在同步點下處理持續 (或所有) 訊息
PubSubMode	已排入佇列的發佈/訂閱介面是否在執行中
QMgrDesc	佇列管理程式說明
QMgrIdentifier	佇列管理程式的唯一內部產生 ID

表 572: 佇列管理程式的屬性.
含有鏈結和簡要說明的佇列管理程式屬性清單
(繼續)

屬性	說明
QMgrName	佇列管理程式名稱
QSGName	佇列共用群組的名稱
QueueAccounting	控制佇列的帳戶資訊收集。
QueueMonitoring	佇列的線上監視資料
QueueStatistics	控制收集佇列的統計資料。
ReceiveTimeout	在回到非作用中狀態之前， TCP/IP 通道等待資料的時間長度。
ReceiveTimeoutMin	*ReceiveTimeout* 的限定元。
ReceiveTimeoutType	TCP/IP 通道在回到非作用中狀態之前等待資料的時間下限。
RemoteEvent	控制是否產生遠端錯誤事件
RepositoryName	此佇列管理程式為其提供儲存庫服務的叢集名稱
RepositoryNamelist	包含此佇列管理程式為其提供儲存庫服務之叢集名稱的名單物件名稱
ScyCase	安全設定檔的案例
SharedQMgr 名稱	共用佇列佇列-管理程式名稱
第 728 頁的『SPLCAP』	WebSphere MQ Advanced 佇列管理程式的訊息安全保護已開啟或關閉。
SSLCRLNamelist 1	包含鑑別資訊物件名稱的名單物件名稱。
SSLCryptoHardware 1	加密硬體配置字串。
SSLEvent	SSL 事件的控制屬性。
SSLFIPSRequired	僅將 FIPS 認證的演算法用於加密法。
SSLKeyRepository 1	SSL 金鑰儲存庫的位置。
SSLKeyReset 計數	SSL 金鑰重設計數。
SSLTasks 1	處理 SSL 呼叫的伺服器子作業數。
StatisticsInterval	寫入統計資料監視資料的頻率。
StartStopEvent	控制是否產生啟動和停止事件
SyncPoint	同步點可用性
TCPChannels	使用 TCP/IP 的現行通道或已連接用戶端數目上限。
TCPKeepAlive	是否使用 TCP KEEPALIVE 來檢查連線的另一端。
TCPName	您正在使用的 TCP/IP 系統名稱。
TCPStackType	通道起始程式如何使用 TCP/IP 位址。
TraceRoute 記錄屬性	控制追蹤路徑資訊的記錄。
TriggerInterval	觸發程式-訊息間隔
版本	版本
XrCapability	指定是否支援遙測指令。

附註:

1. 無法使用 MQINQ 呼叫來查詢此屬性，本節中未說明此屬性。 如需此屬性的詳細資料，請參閱 變更佇列管理程式 。

相關工作

指定在執行時期於 MQI 用戶端上僅使用 FIPS 認證的 CipherSpecs

相關參考

UNIX、 Linux 和 Windows 的聯邦資訊存取安全標準 (FIPS)

AccountingConn 置換 (MQLONG)

這可讓應用程式置換佇列管理程式屬性中 ACCTMQI 及 ACCTQDATA 值的設定。

此值是下列其中一個:

已停用 MQMON_DISABLED
應用程式無法使用 MQCONNX 呼叫 MQCNO 結構中的「選項」欄位來置換 ACCTMQI 及 ACCTQ 佇列管理程式屬性的設定。 這是預設值。

已啟用 MQMON_ENABLED
應用程式可以使用 MQCNO 結構中的「選項」欄位來置換 ACCTQ 及 ACCTMQI 佇列管理程式屬性。

此值的變更僅對變更屬性之後的佇列管理程式連線有效。

此屬性僅在 IBM i、Unix 系統及 Windows 上受支援。

若要判定此屬性的值，請搭配使用 MQQIA_ACCOUNTING_CONN_OVERRIDE 選取器與 MQINQ 呼叫。

AccountingInterval (MQLONG)

這會指定寫入中間統計記錄之前的時間 (以秒為單位)。

此值是 0 到 604800 範圍內的整數，預設值為 1800 (30 分鐘)。 指定 0 以關閉中間記錄。

此屬性僅在 IBM i、 Windows、 UNIX 及 Linux 系統上受支援。

若要判定此屬性的值，請搭配使用 MQQIA_ACCOUNTING_INTERVAL 選取器與 MQINQ 呼叫。

ActivityConn 置換 *(MQLONG)*

這可讓應用程式置換佇列管理程式屬性中的 ACTVTRC 值設定。

此值是下列其中一個:

已停用 MQMON_DISABLED
應用程式無法使用 MQCONNX 呼叫 MQCNO 結構中的「選項」欄位來置換 ACTVTRC 佇列管理程式屬性的設定。 這是預設值。

已啟用 MQMON_ENABLED
應用程式可以使用 MQCNO 結構中的「選項」欄位來置換 ACTVTRC 佇列管理程式屬性。

此值的變更僅對變更屬性之後的佇列管理程式連線有效。

此屬性僅在 IBM i、Unix 系統及 Windows 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_ACTIVITY_CONN_OVERRIDE 選取器與 MQINQ 呼叫。

ActivityTrace (MQLONG)

這會控制 WebSphere MQ MQI 應用程式活動追蹤的收集。

此值是下列其中一個:

MQMON_ON
收集 WebSphere MQ MQI 應用程式活動追蹤。

MQMON_OFF
請勿收集 WebSphere MQ MQI 應用程式活動追蹤。 這是預設值。

如果您將佇列管理程式屬性 ACTVCONO 設為 ENABLED，則可能會使用 MQCNO 結構中的「選項」欄位來置換個別連線的這個值。

此值的變更僅對變更屬性之後的佇列管理程式連線有效。

此屬性僅在 IBM i、Unix 系統及 Windows 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_ACTIVITY_TRACE 選取元與 MQINQ 呼叫。

AdoptNewMCACheck (MQLONG)

這會定義要檢查的元素，以決定當偵測到新的入埠通道與已在作用中的 MCA 同名時，是否採用 MCA

此值是下列其中一個:

MQADOPT_CHECK_Q_MGR_NAME
請檢查佇列管理程式名稱。

MQADOPT_CHECK_NET_ADDR
請檢查網址。

MQADOPT_CHECK_ALL
請檢查佇列管理程式名稱及網址。 如果可能的話，請執行此檢查，以保護您的通道不會被關閉、不小心或惡意關閉。 這是預設值。

MQADOPT_CHECK_NONE
請勿檢查任何元素。

此屬性的變更會在下次通道嘗試採用通道時生效。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_ADOPTNEWMCA_CHECK 選取器與 MQINQ 呼叫。

AdoptNewMCAType (MQLONG)

這會指定當偵測到符合 AdoptNewMCACheck 屬性的新入埠通道要求時，是否要自動重新啟動特定通道類型的 MCA 孤立實例

它是下列其中一個值:

MQADOPT_TYPE_NO
不需要採用孤立通道實例。 這是預設值。

MQADOPT_TYPE_ALL
採用所有通道類型。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQQIA_ADAPTNEWMCA_TYPE 選取器與 MQINQ 呼叫。

AlterationDate (MQCHAR12)

這是前次變更定義的日期。 日期的格式為 YYYY-MM-DD，並以兩個尾端空白填補，使其長度為 12 個位元組。

若要判定此屬性的值，請搭配使用 MQCA_ALTERATION_DATE 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_DATE_LENGTH 提供。

AlterationTime (MQCHAR8)

這是前次變更定義的時間。 時間的格式為 HH.MM.SS。

若要判定此屬性的值，請搭配使用 MQCA_ALTERATION_TIME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_TIME_LENGTH 提供。

AuthorityEvent (MQLONG)

這會控制是否產生授權 (未獲授權) 事件。 它是下列其中一個值:

已停用 MQEVR_DISABLED
事件報告已停用。

已啟用 MQEVR_ENABLED
已啟用事件報告。

如需事件的相關資訊，請參閱 事件監視 。

若要判定此屬性的值，請搭配使用 MQIA_AUTHORITY_EVENT 選取器與 MQINQ 呼叫。

BridgeEvent (MQLONG)

這會指定是否產生 IMS 橋接器事件。

此值是下列其中一個:

已啟用 MQEVR_ENABLED
產生 IMS 橋接器事件，如下所示:

已啟動 MQRC_BRIDGE_STARTED
MQRC_BRIDGE_STOPPED

已停用 MQEVR_DISABLED
不產生 IMS 橋接器事件; 這是預設值。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQQIA_BRIDGE_Event 選取器與 MQINQ 呼叫。

ChannelAutoDef (MQLONG)

此屬性控制 MQCHT_RECEIPT 及 MQCHT_SVRCONN 類型的通道自動定義。 一律會啟用 MQCHT_CLUSSDR 通道的自動定義。 此值是下列其中一個:

MQCHAD_DISABLED
通道自動定義已停用。

已啟用 MQCHAD_ENABLED
已啟用通道自動定義。

此屬性僅在 AIX、 HP-UX、 IBM i、 Linux、Solaris 及 Windows 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_CHANNEL_AUTO_DEF 選取器與 MQINQ 呼叫。

ChannelAutoDefEvent (MQLONG)

這會控制是否產生通道自動定義事件。 它適用於 MQCHT_RECEIPT、MQCHT_SVRCONN 及 MQCHT_CLUSSDR 類型的通道。 此值是下列其中一個:

已停用 MQEVR_DISABLED
事件報告已停用。

已啟用 MQEVR_ENABLED
已啟用事件報告。

如需事件的相關資訊，請參閱 事件監視 。

此屬性僅在 AIX、 HP-UX、 IBM i、 Linux、Solaris 及 Windows 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_CHANNEL_AUTO_DEF_EVENT 選取器與 MQINQ 呼叫。

ChannelAutoDefExit (MQCHARn)

這是自動通道定義的使用者結束程式名稱。 如果此名稱非空白，且 *ChannelAutoDef* 具有值 MQCHAD_ENABLED ，則每次佇列管理程式即將建立通道定義時都會呼叫該結束程式。 這適用於 MQCHT_RECEIPT、MQCHT_SVRCONN 及 MQCHT_CLUSSDR 類型的通道。 然後，結束程式可以執行下列其中一項:

- 建立通道定義而不變更。
- 修改所建立通道定義的屬性。
- 完全暫停建立通道。

註: 此屬性的長度和值都是環境特有的。 如需各種環境中此屬性值的詳細資料，請參閱 第 923 頁的『MQCD-通道定義』 中 MQCD 結構簡介。

此屬性僅在 AIX、 HP-UX、 IBM i、 Linux、Solaris、 Windows 及 z/OS 上受支援。 在 z/OS 上，它僅適用於叢集傳送端和叢集接收端通道。

若要判定此屬性的值，請搭配使用 MQCA_CHANNEL_AUTO_DEF_EXIT 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_EXIT_NAME_LENGTH 提供。

ChannelEvent (MQLONG)
這指定是否產生通道事件。

它是下列其中一個值:

MQ 事件 _ 異常狀況
僅產生下列頻道事件:

- 已啟動 MQRC_CHANNEL_ACTIVATED
- MQRC_CHANNEL_CONV_ERROR
- 已啟動 MQRC_CHANNEL_NOT_ACTIVATED
- MQRC_CHANNEL_STOPPED，具有下列 ReasonQualifiers:

  MQRQ_CHANNEL_STOPPED_ERROR
  MQRQ_CHANNEL_STOPPED_RETRY
  MQRQ_CHANNEL_STOPPED_DISABLED

  MQ RC_CHANNEL_STOPPED_BY_USER

已啟用 MQEVR_ENABLED
產生所有頻道事件。 也就是說，除了 EXCEPTION 所產生的那些事件之外，還會產生下列通道事件:

- 已啟動 MQRC_CHANNEL_STARTED
- MQRC_CHANNEL_STOPPED，具有下列 ReasonQualifier:

  MQRQ_CHANNEL_STOPPED_OK

已停用 MQEVR_DISABLED
不產生通道事件; 這是預設值。

若要判定此屬性的值，請搭配使用 MQIA_CHANNEL_EVENT 選取器與 MQINQ 呼叫。

ChannelInitiator 控制項 *(MQLONG)*
這會指定當佇列管理程式啟動時，是否要啟動通道起始程式。

它是下列其中一個值:

MQSVC_CONTROL_MANUAL
通道起始程式不會自動啟動。

MQSVC_CONTROL_Q_MGR
當佇列管理程式啟動時，會自動啟動通道起始程式。

若要判定此屬性的值，請搭配使用 MQIA_CHINIT_CONTROL 選取器與 MQINQ 呼叫。

ChannelMonitoring (MQLONG)
這會指定通道的線上監視資料。

此值是下列其中一個:

MQMON_NONE
不論 MONCHL 通道屬性的設定為何，都停用所有通道的通道監視資料收集。 這是預設值。

MQMON_OFF
針對在 MONCHL 通道屬性中指定 QMGR 的通道，關閉監視資料收集。

MQMON_LOW
針對在 MONCHL 通道屬性中指定 QMGR 的通道，以低資料收集比例開啟監視資料收集。

MQ mon_MEDIT
針對在 MONCHL 通道屬性中指定 QMGR 的通道，以中等比例的資料收集開啟監視資料收集。

MQMON_HIGH
針對在 MONCHL 通道屬性中指定 QMGR 的通道，以高資料收集比例開啟監視資料收集。

若要判定此屬性的值，請搭配使用 MQQIA_XX_ENCODE_CASE_ONE monitoring_channel 選取器與 MQINQ 呼叫。

ChannelStatistics (MQLONG)

這會控制通道統計資料的收集。

此值是下列其中一個:

MQMON_NONE
不論 STATCHL 通道屬性的設定為何，都停用所有通道的通道統計資料資料收集。 這是預設值。

MQMON_OFF
關閉在 STATCHL 通道屬性中指定 QMGR 之通道的統計資料收集。

MQMON_LOW
針對在 STATCHL 通道屬性中指定 QMGR 的通道，以低資料收集比例開啟統計資料資料收集。

MQ mon_MEDIT
針對在 STATCHL 通道屬性中指定 QMGR 的通道，以中等比例的資料收集開啟統計資料收集。

MQMON_HIGH
針對在 STATCHL 通道屬性中指定 QMGR 的通道，以高資料收集比例開啟統計資料資料收集。

對於大部分系統，建議您使用 MEDIAL。 不過，對於每秒處理大量訊息的通道，您可能想要透過選取「低」來降低取樣層次。 此外，對於只處理少數訊息且最新資訊很重要的通道，您可能想要選取 HIGH。

此屬性僅在 IBM i、 UNIX 系統及 Windows 上受支援。

若要判定此屬性的值，請搭配使用 MQQIA_STATISTICS_CHANNEL 選取器與 MQINQ 呼叫。

ChinitAdapters (MQLONG)

這是用來處理 WebSphere MQ 呼叫的配接器子作業數目。 此值必須是 0-9999 ， 預設值為 8。

配接卡與分派器的比例 (ChinitDispatchers 屬性) 應該大約為 8 到 5。 不過，如果您只有少數通道，則不需要從預設值減少此參數的值。 您可以使用下列值: 若為測試系統，則為 8 (預設值); 若為正式作業系統，則為 20。 理想情況下，您應該有 20 個配接器，其提供更大的 WebSphere MQ 呼叫平行化。 這對持續訊息而言很重要。 對於非持續訊息而言，配接器越少越好。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQQIA_CHINIT_Adapters 選取器與 MQINQ 呼叫。

ChinitDispatchers (MQLONG)

這是用於通道起始程式的分派器數目。 此值必須是 0-9999 ， 預設值為 5。

作為準則，容許一個分派器用於 50 個現行通道。 不過，如果您只有少數通道，則不需要從預設值減少此屬性的值。 如果您使用 TCP/IP ，則用於 TCP/IP 通道的分派器最大數目是 100 ，即使您在這裡指定較大的值也一樣。 您可以使用下列設定: 測試系統 5 (預設值); 正式作業系統 20 (您需要 20 個分派器來處理最多 1000 個作用中通道)。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_CHINIT_DISPATCHERS 選取器與 MQINQ 呼叫。

ChinitTraceAutoStart (MQLONG)

這指定是否自動啟動通道起始程式追蹤。

此值是下列其中一個:

MQTRAXSTR_YES
自動啟動通道起始程式追蹤。 這是預設值。

MQTRAXSTR_NO
請勿自動啟動通道起始程式追蹤。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_CHINIT_TRACE_AUTO_START 選取器與 MQINQ 呼叫。

ChinitTraceTableSize (MQLONG)
這是通道起始程式的追蹤資料空間大小 (MB)。

此值必須在 0 到 2048 範圍內，預設值為 2。

註: 每當您使用大型 z/OS 資料空間時，請確保系統上有足夠的輔助儲存體可支援任何相關 z/OS 分頁活動。 您可能也需要增加 SYS1.DUMP 資料集的大小。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_CHINIT_TRACE_TABLE_SIZE 選取器與 MQINQ 呼叫。

ClusterSenderMonitoringDefault (MQLONG)
這會指定要替換自動定義之叢集傳送端通道的 ChannelMonitoring 屬性的值。

此值是下列其中一個:

MQMON_Q_MGR
連線監視資料的集合繼承自佇列管理程式 *ChannelMonitoring* 屬性的設定。 這是預設值。

MQMON_OFF
通道的監視已關閉

MQMON_LOW
除非 *ChannelMonitoring* 是 MQMON_NONE，否則會以低資料收集速率開啟監視，並對系統效能產生最小影響。 所收集的資料可能不是最新的。

MQ mon_MEDIT
除非 *ChannelMonitoring* 是 MQMON_NONE，否則會以中等速率的資料收集開啟監視，而對系統效能的影響有限。

MQMON_HIGH
除非 *ChannelMonitoring* 是 MQMON_NONE，否則會以高資料收集速率開啟監視，並可能影響系統效能。 收集的資料是最新的可用資料。

若要判定此屬性的值，請搭配使用 MQIA_MONITORING_AUTO_CLUSSDR 選取器與 MQINQ 呼叫。

ClusterSender 統計資料 *(MQLONG)*

因為可以從儲存庫中 CLUSRCVR 的定義自動定義叢集傳送端通道，所以您無法使用 ALTER 通道來變更這些自動定義叢集傳送端通道的 STATCHL 屬性設定。 對於這些通道，是否要收集連線監視資料的決策是根據此佇列管理程式屬性的設定。

此值是下列其中一個:

MQMON_Q_MGR
自動定義叢集傳送端通道的統計資料收集是根據佇列管理程式屬性 STATCHL 的值。 這是預設值。

MQMON_OFF
關閉自動定義叢集傳送端通道的統計資料收集。

MQMON_LOW
針對資料收集比例較低的自動定義叢集傳送端通道，開啟統計資料資料收集。

MQ mon_MEDIT
針對具有中等資料收集比例的自動定義叢集傳送端通道，開啟統計資料資料收集。

MQMON_HIGH
針對具有高資料收集比例的自動定義叢集傳送端通道，開啟統計資料資料收集。

對於大部分系統，我們建議使用 MEDIAL。 不過，對於每秒處理大量訊息的自動定義叢集傳送端通道，您可能想要選取「低」來降低取樣層次。 此外，對於只處理少數訊息且最新資訊很重要的通道，您可能想要選取 HIGH。

若要判定此屬性的值，請搭配使用 MQIA_STATISTICS_AUTO_CLUSSDR 選取器與 MQINQ 呼叫。

ClusterWorkload 資料 *(MQCHAR32)*

這是使用者定義的 32 位元組字串，會在呼叫叢集工作量結束程式時傳遞給它。 如果沒有要傳遞至結束程式的資料，則字串為空白。

此屬性僅在 AIX、 HP-UX、 IBM i、 Linux、Solaris、 Windows 及 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQCA_CLUSTER_WORKLOAD_DATA 選取器與 MQINQ 呼叫。

ClusterWorkload 結束程式 *(MQCHARn)*

這是叢集工作量管理的使用者結束程式名稱。 如果此名稱不是空白，則每次將訊息放入叢集佇列或從一個叢集傳送端佇列移至另一個叢集傳送端佇列時，都會呼叫結束程式。 然後，結束程式可以接受佇列管理程式所選取的佇列實例作為訊息的目的地，或選取另一個佇列實例。

註: 此屬性的長度和值都是環境特有的。

此屬性僅在 AIX、 HP-UX、 IBM i、 Linux、Solaris、 Windows 及 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQCA_CLUSTER_WORKLOAD_EXIT 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_EXIT_NAME_LENGTH 提供。

ClusterWorkload 長度 *(MQLONG)*

這是傳遞至叢集工作量結束程式的訊息資料長度上限。 傳遞至結束程式的資料實際長度下限如下:

- 訊息的長度。
- 佇列管理程式的 *MaxMsgLength* 屬性。
- *ClusterWorkloadLength* 屬性。

此屬性僅在 AIX、 HP-UX、 IBM i、 Linux、Solaris、 Windows 及 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQQIA_CLUSTER_WORKLOAD_length 選取器與 MQINQ 呼叫。

CLWLMRUChannels (MQLONG)

這會指定最近使用的叢集通道數目上限，叢集工作量選擇演算法會考慮使用此叢集通道。

這是在 1 到 999999999 範圍內的值。

若要判定此屬性的值，請搭配使用 MQQIA_CLWL_MRU_XX_ENCODE_CASE_ONE channels 選取器與 MQINQ 呼叫。

CLWLUseQ (MQLONG)

這會指定是否針對叢集工作量使用遠端佇列。

此值是下列其中一個:

MQCLWL_USEQ_ANY
同時使用本端及遠端佇列。

MQCLWL_USEQ_LOCAL
請勿使用遠端佇列。 這是預設值。

若要判定此屬性的值，請搭配使用 MQIA_CLWL_USEQ 選取器與 MQINQ 呼叫。

CodedCharSetId (MQLONG)

這會定義佇列管理程式在 MQI 中定義的所有字串欄位 (例如物件名稱及佇列建立日期和時間) 所使用的字集。 字集必須是物件名稱中有效字元的單位元組字元。 它不適用於訊息中所包含的應用程式資料。 此值視環境而定:

- 在 z/OS 上，當佇列管理程式啟動時，會從系統參數設定此值; 預設值為 500。
- 在 Windows 上，此值是建立佇列管理程式之使用者的主要 CODEPAGE。
- 在 IBM i 上，該值是第一次建立佇列管理程式時在環境中設定的值。
- 在 UNIX 系統上，該值是建立佇列管理程式之使用者的語言環境的預設 CODESET。

若要判定此屬性的值，請搭配使用 MQIA_CODED_CHAR_SET_ID 選取器與 MQINQ 呼叫。

CommandEvent (MQLONG)

這指定是否產生指令事件，如下所示:

已停用 MQEVR_DISABLED
不產生指令事件。 這是預設值。

已啟用 MQEVR_ENABLED
產生指令事件。

MQEVR_NO_DISPLAY
會針對 MQINQ 以外的所有成功指令產生指令事件。

若要判定此屬性的值，請搭配使用 MQQIA_COMMAND_Event 選取器與 MQINQ 呼叫。

CommandInput 完整名稱 *(MQCHAR48)*

這是定義在本端佇列管理程式上的指令輸入佇列名稱。 這是使用者可以傳送指令的佇列 (如果有授權的話)。 佇列名稱視環境而定:

- 在 z/OS 上，佇列的名稱是 SYSTEM.COMMAND.INPUT; MQSC 和 PCF 指令可以傳送給它。 如需 MQSC 指令的詳細資料，以及 可程式指令格式的定義 ，請參閱 MQSC 指令 ，以取得 PCF 指令的詳細資料。
- 在所有其他環境中，佇列名稱為 SYSTEM.ADMIN.COMMAND.QUEUE，且只能向其傳送 PCF 指令。 不過，如果 MQSC 指令含括在 MQCMD_ESCAPE 類型的 PCF 指令內，則可以將 MQSC 指令傳送至此佇列。 如需 Escape 指令的相關資訊，請參閱 Escape 。

若要判定此屬性的值，請搭配使用 MQCA_COMMAND_INPUT_Q_NAME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_Q_NAME_LENGTH 提供。

CommandLevel (MQLONG)

這指出佇列管理程式支援的系統控制指令層次。 這可以是下列其中一個值:

MQCMDL_LEVEL_1
系統控制指令的層次 1。

此值由下列版本的 IBM WebSphere MQ 傳回:

- MQSeries for AIX 2.2 版
- MQSeries ，適用於
  - 版本 1 版次 1.1
  - 版本 1 版次 1.2
  - 版本 1 版次 1.3
- MQSeries for OS/400
  - 2.3 版
  - 3.1 版
  - 版本 3 版次 6
- MQSeries for Windows 2.0 版

MQCMDL_LEVEL_101
MQSeries for Windows 版本 2 版次 0.1。

MQCMDL_LEVEL_110
MQSeries for Windows 2.1 版。

MQCMDL_LEVEL_114
MQSeries for Version 1 Release 1.4。

MQCMDL_LEVEL_120
MQSeries for Version 1 Release 2.0。

MQCMDL_LEVEL_200
MQSeries for Windows NT 2.0 版。

MQCMDL_LEVEL_210
MQSeries for OS/390 第 2 版版次 1.0。

MQCMDL_LEVEL_220
系統控制指令的層次 220。

此值由下列版本的 IBM WebSphere MQ 傳回:

- MQSeries for AT&T GIS UNIX Version 2 Release 2
- MQSeries for SINIX and DC/OSx 2.2 版
- MQSeries for SunOS 2.2 版
- MQSeries for Tandem NonStop Kernel 2.2 版

MQCMDL_LEVEL_221
系統控制指令的層次 221。

此值由 MQSeries for AIX 版本 2 版次 2.1 傳回

MQCMDL_LEVEL_320
系統控制指令的層次 320。

此值由下列版本的 IBM WebSphere MQ 傳回:

- MQSeries for OS/400
  - 3.2 版
  - 版本 3 版次 7

MQCMDL_LEVEL_420
系統控制指令的層次 420。

此值由下列版本的 IBM WebSphere MQ 傳回:

- MQSeries for IBM i
  - 第 4 版版次 2.0
  - 第 4 版版次 2.1

MQCMDL_LEVEL_500
系統控制指令的層次 500。

此值由下列版本的 IBM WebSphere MQ 傳回:

- IBM WebSphere MQ for AIX 版本 5 版次 0
- MQSeries for HP-UX 版本 5 版次 0
- MQSeries for Solaris 版本 5 版次 0
- MQSeries for Windows NT 版本 5 版次 0

MQCMDL_LEVEL_510
系統控制指令的層次 510。

此值由下列版本的 IBM WebSphere MQ 傳回:

- IBM WebSphere MQ for AIX 5.1 版
- MQSeries for AS/400 5.1 版
- MQSeries for HP-UX 5.1 版
- IBM WebSphere MQ for HP Integrity NonStop Server 5.3 版
- MQSeries for Compaq Tru64 UNIX 5.1 版
- MQSeries for Solaris 5.1 版
- MQSeries for Windows NT 5.1 版

MQCMDL_LEVEL_520
系統控制指令的層次 520。

此值由下列版本的 IBM WebSphere MQ 傳回:

- MQSeries for AIX ersion 5.2 版
- MQSeries for AS/400 5.2 版
- MQSeries for HP-UX 5.2 版
- MQSeries for Linux 5.2 版
- MQSeries for OS/390 5.2 版
- MQSeries for Sun Solaris 5.2 版
- MQSeries for Windows NT 5.2 版

MQCMDL_LEVEL_530
層次 530 系統控制指令。

此值由下列版本的 IBM WebSphere MQ 傳回:

- IBM WebSphere MQ for AIX5.3 版
- IBM WebSphere MQ for HP-UX5.3 版
- IBM WebSphere MQ for i/Series 5.3 版
- IBM WebSphere MQ for Linux for Intel 5.3 版
- IBM WebSphere MQ for Linux for zSeries 5.3 版
- IBM WebSphere MQ for Solaris5.3 版
- IBM WebSphere MQ for Windows5.3 版
- IBM WebSphere MQ for z/OS5.3 版

MQCMDL_LEVEL_600
系統控制指令的層次 600。

此值由下列版本的 IBM WebSphere MQ 傳回:

- IBM WebSphere MQ for AIX Version 6.0
- IBM WebSphere MQ for HP-UX Version 6.0
- IBM WebSphere MQ for i/Series Version 6.0
- IBM WebSphere MQ for Linux Version 6.0
- IBM WebSphere MQ for Solaris Version 6.0
- IBM WebSphere MQ for Windows Version 6.0
- IBM WebSphere MQ for z/OS Version 6.0

MQCMDL_LEVEL_700
系統控制指令層次 700。

此值由下列版本的 IBM WebSphere MQ 傳回:

- IBM WebSphere MQ for AIX Version 7.0
- IBM WebSphere MQ for HP-UX Version 7.0
- IBM WebSphere MQ for IBM i Version 7.0
- IBM WebSphere MQ for Linux Version 7.0
- IBM WebSphere MQ for Solaris Version 7.0
- IBM WebSphere MQ for Windows Version 7.0
- IBM WebSphere MQ for z/OS Version 7.0

MQCMDL_LEVEL_701
系統控制指令的層次 701。

此值由下列版本的 IBM WebSphere MQ 傳回:

- IBM WebSphere MQ for AIX Version 7.0.1
- IBM WebSphere MQ for HP-UX Version 7.0.1
- IBM WebSphere MQ for IBM i Version 7.0.1
- IBM WebSphere MQ for Linux Version 7.0.1
- IBM WebSphere MQ for Solaris Version 7.0.1
- IBM WebSphere MQ for Windows Version 7.0.1
- IBM WebSphere MQ for z/OS Version 7.0.1

MQCMDL_LEVEL_710
系統控制指令的層次 710。

此值由下列版本的 IBM WebSphere MQ 傳回:

- IBM WebSphere MQ for AIX Version 7.1
- IBM WebSphere MQ for HP-UX Version 7.1
- IBM WebSphere MQ for IBM i Version 7.1
- IBM WebSphere MQ for Linux Version 7.1
- IBM WebSphere MQ for Solaris Version 7.1
- IBM WebSphere MQ for Windows Version 7.1
- IBM WebSphere MQ for z/OS Version 7.1

MQCMDL_LEVEL_750
系統控制指令的層次 750。

此值由下列版本的 IBM WebSphere MQ 傳回:

- IBM WebSphere MQ for AIX Version 7.5
- IBM WebSphere MQ for HP-UX Version 7.5
- IBM WebSphere MQ for IBM i Version 7.5
- IBM WebSphere MQ for Linux Version 7.5
- IBM WebSphere MQ for Solaris Version 7.5
- IBM WebSphere MQ for Windows Version 7.5

對應於 *CommandLevel* 屬性特定值的系統控制指令集，會根據 *Platform* 屬性的值而不同; 兩者必須用來決定支援哪些系統控制指令。

若要判定此屬性的值，請搭配使用 MQIA_COMMAND_LEVEL 選取器與 MQINQ 呼叫。

CommandServer 控制項 *(MQLONG)*

指定在佇列管理程式啟動時是否要啟動指令伺服器。

值可以為:

MQSVC_CONTROL_MANUAL
指令伺服器不會自動啟動。

MQSVC_CONTROL_Q_MGR
當佇列管理程式啟動時，會自動啟動指令伺服器。

z/OS 不支援這個屬性。

若要判定此屬性的值，請搭配使用 MQIA_CMD_SERVER_CONTROL 選取器與 MQINQ 呼叫。

ConfigurationEvent (MQLONG)

控制是否產生配置事件。

若要判定此屬性的值，請搭配使用 MQIA_CONFIGURATION_EVENT 選取器與 MQINQ 呼叫。

值可以為:

已停用 MQEVR_DISABLED
事件報告已停用。

已啟用 MQEVR_ENABLED
已啟用事件報告。

DeadLetter 完整名稱 *(MQCHAR48)*

這是在本端佇列管理程式上定義為無法傳送郵件 (未遞送訊息) 佇列的佇列名稱。 如果訊息無法遞送至正確的目的地，則會傳送至這個佇列。

例如，在下列情況下，會將訊息放置在此佇列上:

- 訊息到達佇列管理程式，目的地是尚未定義在該佇列管理程式上的佇列
- 訊息到達佇列管理程式，但其目的地佇列無法接收該訊息，可能是因為:
  - 佇列已滿
  - 禁止放置要求
  - 傳送節點沒有將訊息放入佇列的權限

應用程式也可以將訊息放在無法傳送郵件的佇列中。

報告訊息的處理方式與一般訊息相同; 如果報告訊息無法遞送至其目的地佇列 (通常是原始訊息的訊息描述子中 *ReplyToQ* 欄位所指定的佇列) ，則會將報告訊息置於無法傳送郵件 (無法遞送的訊息) 佇列中。

註: 已超過到期時間 (請參閱 MQMD-期限欄位) 的訊息在捨棄時 **不會** 傳送至此佇列。 不過，如果傳送應用程式要求，則仍會產生到期報告訊息 (MQRO_EXPIRATION) 並傳送至 *ReplyToQ* 佇列。

當發出放置要求的應用程式已透過 MQPUT 或 MQPUT1 呼叫所傳回的原因碼同步通知問題時 (例如，將訊息放置在禁止放置要求的本端佇列上) ，不會將訊息放置在無法傳送郵件 (無法遞送的訊息) 佇列中。

無法傳送郵件 (未遞送訊息) 佇列上的訊息有時會以 MQDLH 結構作為其應用程式訊息資料的字首。 此結構包含額外資訊，指出訊息放置在無法傳送的郵件 (無法遞送的訊息) 佇列上的原因。 如需此結構的詳細資料，請參閱 第 306 頁的『MQDLH-無法傳送郵件的標頭』 。

此佇列必須是本端佇列，且 *Usage* 屬性為 MQUS_NORMAL。

如果佇列管理程式不支援無法傳送的郵件 (未遞送的訊息) 佇列，或尚未定義佇列管理程式，則名稱全為空白。 所有 WebSphere MQ 佇列管理程式都支援無法傳送郵件 (取消遞送訊息) 的佇列，但依預設不會定義它。

如果未定義無法傳送的郵件 (無法遞送的訊息) 佇列，因為其他原因而已滿或無法使用，則會在傳輸佇列上保留由訊息通道代理程式傳送給它的訊息。

若要判定此屬性的值，請搭配使用 MQCA_DEAD_LETTER_Q_NAME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_Q_NAME_LENGTH 提供。

DefClusterXmitQueue 類型 *(MQLONG)*

DefClusterXmitQueue **類型** 屬性 可控制叢集傳送端通道預設會選取要從中取得訊息，以將訊息傳送至叢集接收端通道的傳輸佇列。

DefClusterXmitQueueType 的值為 MQCLXQ_SCTQ 或 MQCLXQ_CHANNEL。

MQCLXQ_SCTQ

所有叢集傳送端通道都會從 SYSTEM.CLUSTER.TRANSMIT.QUEUE 傳送訊息。 放置在傳輸佇列上的訊息的 correlID，可識別該訊息的目的地是哪一個叢集傳送端通道。

SCTQ 在定義佇列管理程式時會設定 。 在早於 Version 7.5 的 IBM WebSphere MQ 版本中，此行為是隱含的。 在舊版中，佇列管理程式屬性 DefClusterXmitQueueType 不存在。

MQCLXQ_CHANNEL
每個叢集傳送端通道會從不同的傳輸佇列傳送訊息。 每一個傳輸佇列都會從模型佇列 SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE 建立為永久動態佇列。

z/OS 不支援此屬性。

如果佇列管理程式屬性 DefClusterXmitQueue **類型**設為 CHANNEL，則為 預設配置會變更為與個別叢集傳輸佇列相關聯的叢集傳送端通道。 傳輸佇列是從模型佇列 SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE 建立的永久動態佇列。 每一個傳輸佇列都與一個叢集傳送端通道相關聯。 作為一個叢集傳送端通道服務叢集傳輸佇列，傳輸佇列只包含一個叢集中一個佇列管理程式的訊息。 您可以配置叢集，使某個叢集中的每個佇列管理程式都只包含一個叢集佇列。 在此情況下，從一個佇列管理程式到每個叢集佇列的訊息資料流量將與其他佇列的訊息分開傳送。

若要查詢值，請呼叫 MQINQ，或傳送「查詢佇列管理程式 (MQCMD_INQUIRE_Q_MGR)」PCF 指令，並設定 MQIA_DEF_CLUSTER_XMIT_Q_TYPE 選取器。 若要變更此值，請傳送「變更佇列管理程式 (MQCMD_CHANGE_Q_MGR)」PCF 指令，並設定 MQIA_DEF_CLUSTER_XMIT_Q_TYPE 選取器。

相關參考
變更佇列管理程式
查詢佇列管理程式
第 618 頁的『MQINQ-查詢物件屬性』
MQINQ 呼叫會傳回整數陣列及一組包含物件屬性的字串。

DefXmit 完整名稱 (MQCHAR48)

這是傳輸佇列的名稱，用於將訊息傳輸至遠端佇列管理程式 (如果沒有其他指示要使用哪個傳輸佇列的話)。

如果沒有預設傳輸佇列，則名稱會完全空白。 此屬性的起始值為空白。

若要判定此屬性的值，請搭配使用 MQCA_DEF_XMIT_Q_NAME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_Q_NAME_LENGTH 提供。

DistLists (MQLONG)

這指出本端佇列管理程式是否支援 MQPUT 及 MQPUT1 呼叫的配送清單。 它是下列其中一個值:

支援 MQDL_SUPPORTED
支援的配送清單。

不支援 MQDL_NOT_SUPPORTED
不支援配送清單。

若要判定此屬性的值，請搭配使用 MQQIA_DIST_清單選取器與 MQINQ 呼叫。

DNSGroup (MQCHAR18)

這是 TCP 接聽器的群組名稱，用於處理使用「工作量管理程式動態網域名稱服務」支援時要結合之佇列共用群組的入埠傳輸。 最大長度為 18 個字元。 如果您將此名稱保留空白，則會使用佇列共用群組名稱。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQCA_DNS_GROUP 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_DNS_GROUP_NAME_LENGTH 提供。

DNSWLM (MQLONG)

這會指定處理佇列共用群組之入埠傳輸的 TCP 接聽器是否向「動態網域名稱服務的工作量管理程式」登錄

此值是下列其中一個:

MQDNSWLM_YES
接聽器會向「工作量管理程式」登錄。

MQDNSWLM_NO
接聽器未向「工作量管理程式」登錄。 這是預設值。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_DNS_WLM 選取器與 MQINQ 呼叫。

ExpiryInterval (MQLONG)

這指出佇列管理程式掃描佇列以尋找過期訊息的頻率。 它是範圍 1 到 99999 999 之間的時間間隔 (以秒為單位)，或下列特殊值:

MQEXPI_OFF
佇列管理程式不會掃描佇列，以尋找過期訊息。

若要判定此屬性的值，請搭配使用 MQQIA_EXPIRY_INTERVAL 選取器與 MQINQ 呼叫。

此屬性僅在 z/OS 上受支援。

IGQPutAuthority (MQLONG)

只有在本端佇列管理程式是佇列共用群組的成員時，這個屬性才適用。 它指出當本端內部群組佇列作業代理程式 (IGQ 代理程式) 從共用傳輸佇列中移除訊息並將訊息放入本端佇列時所執行的權限檢查類型。 此值是下列其中一個:

MQIGQPA_DEFAULT
當訊息位於共用傳輸佇列上時，檢查授權的使用者 ID 是與訊息相關聯的 個別 MQMD 中 *UserIdentifier* 欄位的值。 這是將訊息放置在共用傳輸佇列上之程式的使用者 ID，通常與執行遠端佇列管理程式的使用者 ID 相同。

如果 RESLEVEL 設定檔指出要檢查多個使用者 ID，則也會檢查本端 IGQ 代理程式 (*IGQUserId*) 的使用者 ID。

MQIGQPA_CONTEXT
當訊息位於共用傳輸佇列上時，檢查授權的使用者 ID 是與訊息相關聯的 個別 MQMD 中 *UserIdentifier* 欄位的值。 這是將訊息放置在共用傳輸佇列上之程式的使用者 ID，通常與執行遠端佇列管理程式的使用者 ID 相同。

如果 RESLEVEL 設定檔指出要檢查多個使用者 ID，則也會檢查本端 IGQ 代理程式 (*IGQUserId*) 的使用者 ID，以及 內嵌 MQMD 中 *UserIdentifier* 欄位的值。 後一個使用者 ID 通常是產生訊息之應用程式的使用者 ID。

MQIGQPA_ONLY_IGQ
檢查授權的使用者 ID 是本端 IGQ 代理程式 (*IGQUserId*) 的使用者 ID。

如果 RESLEVEL 設定檔指出要檢查多個使用者 ID，則會將此使用者 ID 用於所有檢查。

MQIGQPA_ALTERNATE_OR_IGQ
檢查授權的使用者 ID 是本端 IGQ 代理程式 (*IGQUserId*) 的使用者 ID。

如果 RESLEVEL 設定檔指出要檢查多個使用者 ID，則也會檢查 內嵌 MQMD 中的 *UserIdentifier* 欄位值。 此使用者 ID 通常是產生訊息之應用程式的使用者 ID。

若要判定此屬性的值，請搭配使用 MQQIA_IGQ_PUT_AUTHORITY 選取器與 MQINQ 呼叫。

此屬性僅在 z/OS 上受支援。

IGQUserId (MQLONG)

唯有當本端佇列管理程式隸屬於佇列共用群組時，這個屬性才適用。 它指定與本端內部群組佇列作業代理程式 (IGQ 代理程式) 相關聯的使用者 ID。 當 IGQ 代理程式將訊息放入本端佇列時，此 ID 是可以檢查授權的其中一個使用者 ID。 所檢查的實際使用者 ID 取決於 *IGQPutAuthority* 屬性的設定，以及外部安全選項。

如果 *IGQUserId* 為空白，則不會有任何使用者 ID 與 IGQ 代理程式相關聯，且不會執行對應的授權檢查 (雖然可能仍會檢查其他使用者 ID 的授權)。

若要判定此屬性的值，請搭配使用 MQCA_IGQ_USER_ID 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_USER_ID_LENGTH 提供。

此屬性僅在 z/OS 上受支援。

InhibitEvent (MQLONG)

這會控制是否產生禁止 (禁止取得及禁止放置) 事件。 此值是下列其中一個:

已停用 MQEVR_DISABLED
事件報告已停用。

已啟用 MQEVR_ENABLED
已啟用事件報告。

如需事件的相關資訊，請參閱 事件監視 。

若要判定此屬性的值，請搭配使用 MQQIA_INHIT_Event 選取器與 MQINQ 呼叫。

在 z/OS 上，您無法使用 MQINQ 呼叫來判定此屬性的值。

IntraGroup 佇列作業 (MQLONG)

只有在本端佇列管理程式是佇列共用群組的成員時，這個屬性才適用。 它指出是否針對佇列共用群組啟用內部群組佇列作業。 此值是下列其中一個:

已停用 MQIGQ_DISABLED
所有指定給佇列共用群組中其他佇列管理程式的訊息，都會使用慣用通道來傳輸。

已啟用 MQIGQ_ENABLED
如果滿足下列條件，則會使用共用傳輸佇列來傳輸針對佇列共用群組中其他佇列管理程式的訊息:

- 訊息資料加上傳輸標頭的長度不超過 63 KB (64 512 位元組)。

  建議為傳輸標頭配置比 MQXQH 大小更多的空間; 為此目的提供常數 MQ_MSG_HEADER_LENGTH。

如果不滿足此條件，則使用常規通道來傳輸訊息。

註: 當啟用組內排隊時，使用共享傳輸佇列傳送的訊息的順序相對於使用常規通道傳送的訊息的順序不被保留。

若要判定此屬性的值，請搭配使用 MQIA_INTRA_GROUP_QUEUING 選取器與 MQINQ 呼叫。

此屬性僅在 z/OS 上受支援。

IPAddressVersion (MQLONG)

指定使用哪個 IP 位址版本 (IPv4 或 IPv6)。

此屬性僅與同時執行 IPv4 及 IPv6 的系統相關，且只有在符合下列其中一個條件時，才會影響定義為具有 MQXPY_TCP *TransportType* 的通道:

- 通道的 *ConnectionName* 是同時解析為 IPv4 及 IPv6 位址的主機名稱，且未指定其 *LocalAddress* 參數。
- 通道的 *ConnectionName* 及 *LocalAddress* 都是同時解析為 IPv4 及 IPv6 位址的主機名稱。

值可以為:

MQIPADDR_IPV4
使用 IPv4 。

MQIPADDR_IPV6
使用 IPv6 。

若要判定此屬性的值，請搭配使用 MQIA_IP_ADDRESS_VERSION 選取器與 MQINQ 呼叫。

ListenerTimer (MQLONG)

這是在 APPC 或 TCP/IP 失敗的情況下， WebSphere MQ 嘗試重新啟動接聽器之間的時間間隔 (以秒為單位)。 此值必須介於 5 和 9999 之間，預設值為 60。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_LISTENER_TIMER 選取器與 MQINQ 呼叫。

LocalEvent (MQLONG)

這會控制是否產生本端錯誤事件。 此值是下列其中一個:

已停用 MQEVR_DISABLED
事件報告已停用。

已啟用 MQEVR_ENABLED
已啟用事件報告。

如需事件的相關資訊，請參閱 事件監視 。

若要判定此屬性的值，請搭配使用 MQIA_LOCAL_EVENT 選取器與 MQINQ 呼叫。

在 z/OS 上，您無法使用 MQINQ 呼叫來判定此屬性的值。

LoggerEvent (MQLONG)

這會控制是否產生回復日誌事件。 此值是下列其中一個:

已停用 MQEVR_DISABLED
事件報告已停用。

已啟用 MQEVR_ENABLED
已啟用事件報告。

如需事件的相關資訊，請參閱 事件監視 。

若要判定此屬性的值，請搭配使用 MQQIA_LOGGER_Event 選取器與 MQINQ 呼叫。

此屬性僅在 AIX、 HP-UX、 IBM i、 Linux、Solaris 及 Windows 上受支援。

LUGroupName (MQCHAR8)

這是 LU 6.2 接聽器的一般 LU 名稱，用於處理佇列共用群組的入埠傳輸。 如果您將此名稱保留空白，則無法使用此接聽器。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQCA_LU_GROUP_NAME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_LU_NAME_LENGTH 提供。

LUName (MQCHAR8)

這是用於出埠 LU 6.2 傳輸的 LU 名稱。 將此設為接聽器用於入埠傳輸的相同 LU。 如果您將此名稱保留空白，則會使用 APPC/MVS 預設 LU; 這是變數，因此如果您使用 LU6.2，請一律設定 LUName。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQCA_LU_NAME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_LU_NAME_LENGTH 提供。

LU62ARMSuffix (MQCHAR2)

這是 SYS1.PARMLIB 成員 APPCPMxx ，指定此通道起始程式的 LUADD。 當 ARM 重新啟動通道起始程式時，會發出 z/OS 指令 SET APPC=xx。 如果您將此名稱保留空白，則不會發出 SET APPC=xx。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQCA_LU62_ARM_SUFFIX 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_ARM_SUFFIX_LENGTH 指定。

LU62Channels (MQLONG)

這是使用 LU 6.2 傳輸通訊協定的現行或可連接的通道數上限。

此值必須在 0 到 9999 的範圍內，預設值為 200。 如果您將此設為零，則不會使用 LU 6.2 傳輸通訊協定。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_LU62_CHANNELS 選取器與 MQINQ 呼叫。

MaxActive 通道數 *(MQLONG)*

此屬性是隨時可以 作用中 的通道數上限。

預設值是指定給 MaxChannels 屬性的值。 對於 z/OS，該值必須在 1 到 9 999 的範圍內。 對於所有其他平台，此值必須在 1 到 65 535 的範圍內。

若要判定此屬性的值，請搭配使用 MQIA_ACTIVE_CHANNELS 選取器與 **MQINQ** 呼叫。

相關概念
通道狀態

MaxChannels (MQLONG)
此屬性是可以是 目前 的通道數上限 (包括具有已連接用戶端的伺服器連線通道)。

對於 z/OS，此值必須在 1 到 9 999 的範圍內，預設值為 200。 對於所有其他平台，此值必須在 1 到 65 535 的範圍內，預設值為 100。 忙於處理來自網路的連線的系統可能需要比預設值更高的數目。 請在測試期間觀察系統的行為，以判斷適合您環境的正確值。

對於 z/OS 以外的平台， MaxChannels 的值設定在個別佇列管理程式的 qm.ini 檔中。

若要判定此屬性的值，請搭配使用 MQIA_MAX_通道選取器與 **MQINQ** 呼叫。

相關概念
通道狀態

MaxHandles (MQLONG)

這是任何一個作業可以同時使用的開啟控點數目上限。 單一佇列 (或非佇列物件) 的每一個成功 MQOPEN 呼叫都使用一個控點。 當物件關閉時，該控點會變成可供重複使用。 不過，當開啟配送清單時，會為配送清單中的每一個佇列配置個別控點，因此 MQOPEN 呼叫會使用與配送清單中佇列一樣多的控點。 在決定 *MaxHandles* 的適當值時，必須考慮這一點。

MQPUT1 呼叫會在其處理過程中執行 MQOPEN 呼叫; 因此， MQPUT1 會使用與 MQOPEN 一樣多的控點，但控點只會在 MQPUT1 呼叫本身的期間使用。

在 z/OS 上， 作業 表示 CICS 作業、 MVS 作業或 IMS 相依區域。

此值在 1 到 999 999 999 的範圍內。 預設值由環境決定:

- 在 z/OS 上，預設值為 100。
- 在所有其他環境中，預設值為 256。

若要判定此屬性的值，請搭配使用 MQQIA_MAX_HANDLES 選取器與 MQINQ 呼叫。

MaxMsg 長度 (MQLONG)

這是佇列管理程式可以處理的最長 實體 訊息長度。 不過，因為 *MaxMsgLength* 佇列管理程式屬性可以獨立於 *MaxMsgLength* 佇列屬性來設定，所以可以放置在佇列上的最長實體訊息是這兩個值中較小的。

如果佇列管理程式支援分段，則應用程式可以放置比兩個 *MaxMsgLength* 屬性中較小者更長的 邏輯 訊息，但前提是應用程式在 MQMD 中指定 MQMF_SEGMENTATION_ALLOWED 旗標。 如果指定該旗標，則邏輯訊息的長度上限為 999 999 999 999 個位元組，但通常由作業系統或應用程式執行所在環境強制的資源限制會導致下限。

MaxMsgLength 屬性的下限為 32 KB (32 768 位元組)。 上限為 100 MB (104 857 600 位元組)。

若要判定此屬性的值，請搭配使用 MQQIA_MAX_MSG_length 選取器與 MQINQ 呼叫。

MaxPriority (MQLONG)

這是佇列管理程式支援的訊息優先順序上限。 優先順序範圍從零 (最低) 到 *MaxPriority* (最高)。

若要判定此屬性的值，請搭配使用 MQIA_MAX_PRIORITY 選取器與 MQINQ 呼叫。

MaxProperties 長度 (MQLONG)
這是用來控制可與訊息一起流動的內容大小。 這包括內容名稱 (以位元組為單位) ， 以及內容值的大小 (以位元組為單位)。

若要判定此屬性的值，請搭配使用 MQIA_MAX_PROPERTIES_LENGTH 選取器與 MQINQ 呼叫。

MaxUncommitted 訊息數 *(MQLONG)*

這是工作單元內可存在的未確定的訊息數上限。 未確定的訊息數是自現行工作單元啟動以來下列項目的總和:

- 應用程式使用 MQPMO_SYNCPOINT 選項所放置的訊息
- 應用程式使用 MQGMO_SYNCPOINT 選項擷取的訊息
- 針對使用 MQPMO_SYNCPOINT 選項放置的訊息，由佇列管理程式產生的觸發訊息及 COA 報告訊息
- 針對 MQGMO_SYNCPOINT 選項所擷取的訊息，由佇列管理程式產生的 COD 報告訊息

下列 不 計為未確定的訊息:

- 應用程式在工作單元外部放置或擷取的訊息
- 因工作單元外部放置或擷取訊息而由佇列管理程式產生的觸發訊息或 COA/COD 報告訊息
- 佇列管理程式所產生的到期報告訊息 (即使導致到期報告訊息的呼叫已指定 MQGMO_SYNCPOINT)
- 佇列管理程式所產生的事件訊息 (即使導致事件訊息的呼叫指定 MQPMO_SYNCPOINT 或 MQGMO_SYNCPOINT)

註:

1. 異常狀況報告訊息是由「訊息通道代理程式 (MCA)」或應用程式所產生，並以應用程式放置或擷取一般訊息的相同方式來處理。
2. 使用 MQPMO_SYNCPOINT 選項放置訊息或區段時，不論放置實際產生多少實體訊息，未確定的訊息數都會增加 1。 (如果佇列管理程式必須細分訊息或區段，則可能會產生多個實體訊息。)
3. 使用 MQPMO_SYNCPOINT 選項放置配送清單時，未確定的訊息數目會 針對產生的每一個實體訊息增加一個。 這可以小到一，也可以大到配送清單中的目的地數目。

此屬性的下限為 1; 上限為 999 999 999 999。 預設值為 10000。

若要判定此屬性的值，請搭配使用 MQIA_MAX_UNCOMMITTED_MSGS 選取器與 MQINQ 呼叫。

MQIAccounting (MQLONG)

這會控制 MQI 資料的帳戶資訊收集。

此值是下列其中一個:

MQMON_ON
收集 API 帳戶資料。

MQMON_OFF
不收集 API 帳戶資料。 這是預設值。

如果您將佇列管理程式屬性 ACCTCONO 設為 ENABLED，則可能會使用 MQCNO 結構中的「選項」欄位來置換個別連線的這個值。 此值的變更僅對在變更屬性之後發生的佇列管理程式連線有效。

此屬性僅在 IBM i、 UNIX 系統及 Windows 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_ACCOUNTING_MQI 選取器與 MQINQ 呼叫。

MQIStatistics (MQLONG)

這會控制佇列管理程式的統計資料監視資訊收集。

此值是下列其中一個:

MQMON_ON
收集 MQI 統計資料。

MQMON_OFF
不收集 MQI 統計資料。 這是預設值。

此屬性僅在 IBM i、 UNIX and Linux 系統及 Windows 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_STATISTICS_MQI 選取器與 MQINQ 呼叫。

MsgMarkBrowseInterval (MQLONG)
時間間隔 (毫秒)，在此之後佇列管理程式可以自動從瀏覽訊息中移除標記。

這是時間間隔 (毫秒)，在此時間間隔之後，佇列管理程式可以自動從瀏覽訊息中移除標示。

此屬性說明使用取得訊息選項 MQGMO_MARK_BROWSE_CO_OP，被 MQGET 呼叫標示為已瀏覽的訊息預期保持標示為已瀏覽的時間間隔。

當已標示的控點數目超過此近似間隔時，佇列管理程式可能會自動取消標示已標示為已瀏覽的協同作業控點訊息。

這不會影響任何標示為瀏覽的訊息的狀態，該訊息是透過使用取得訊息選項 MQGMO_MARK_BROWSE_HANDLE 來呼叫 MQGET 所取得。

最大值是 999 999 999 999，預設值是 5000。 *MsgMarkBrowseInterval* 的特殊值 -1 代表無限制時間間隔。

小心: 此值不應低於預設值 5000。

若要判定此屬性的值，請搭配使用 MQIA__MSG_MARK_BROWSE_INTERVAL 選取器與 MQINQ 呼叫。

OutboundPort 上限 (MQLONG)
這是要用來連結送出通道的埠號範圍中最高的埠號，由 OutboundPortMin 和 OutboundPortMax 所定義。

該值是 0 到 65535 範圍內的整數，且必須等於或大於 OutboundPort 最小值。 預設值為 0。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_OUTBOUND_PORT_MAX 選取器與 MQINQ 呼叫。

OutboundPort 下限 (MQLONG)
這是要用來連結送出通道的埠號範圍中的最低埠號，由 OutboundPortMin 和 OutboundPortMax 所定義。

此值是 0 到 65535 範圍內的整數，且必須等於或小於 OutboundPort 最大值。 預設值為 0。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_OUTBOUND_PORT_MIN 選取元與 MQINQ 呼叫。

PerformanceEvent (MQLONG)
這會控制是否產生效能相關事件。 它是下列其中一個值:

已停用 MQEVR_DISABLED
事件報告已停用。

已啟用 MQEVR_ENABLED
已啟用事件報告。

如需事件的相關資訊，請參閱 事件監視。

若要判定此屬性的值，請搭配使用 MQQIA_PERFORMANCE_Event 選取器與 MQINQ 呼叫。

平台 *(MQLONG)*
這指出佇列管理程式執行所在的作業系統:

MQPL_AIX
AIX (與 MQPL_UNIX 相同的值)。

MQPL_MVS
z/OS (與 MQPL_ZOS 相同的值)。

MQPL_NSK
HP Integrity NonStop Server.

MQPL_OS390
z/OS (與 MQPL_ZOS 相同的值)。

MQPL_OS400
IBM i.

MQPL_UNIX
UNIX 系統。

MQPL_WINDOWS_NT
Windows 系統。

MQPL_ZOS
z/OS.

若要判定此屬性的值，請搭配使用 MQQIA_PLATFORM 選取器與 MQINQ 呼叫。

PubSubNPInputMsg (MQLONG)
是否要捨棄或保留未遞送的輸入訊息。

此值是下列其中一個：

MQUNDELIVERED_DISCARD
如果無法處理非持續性輸入訊息，則可能會捨棄它們。

這是預設值。

MQUNDELIVERED_KEEP
如果無法處理非持續性輸入訊息，則不會捨棄它們。 在此狀況下，排入佇列的發佈/訂閱介面將繼續以適當的間隔重試程序，且不會繼續處理後續訊息。

若要判定此屬性的值，請搭配使用 MQIA_PUBSUB_NP_MSG 選取器與 MQINQ 呼叫。

PubSubNPResponse (MQLONG)
控制未遞送回應訊息的行為。

此值是下列其中一個：

MQUNDELIVERED_NORMAL
無法放置在回覆佇列上的非持續性回應會放置在無法傳送的郵件佇列上，如果無法放置在 DLQ 上，則會捨棄它們。

MQUNDELIVERED_SAFE
無法放置在回覆佇列上的非持續性回應會放置在無法傳送的郵件佇列上。 如果無法設定回應，且無法放置在 DLQ 上，則排入佇列的發佈/訂閱介面會回復現行作業，然後以適當的間隔重試，且不會繼續處理後續訊息。

MQUNDELIVERED_DISCARD
不會將非持續性回應放置在回覆佇列上，會捨棄這些非持續性回應。

這是新佇列管理程式的預設值。

MQUNDELIVERED_KEEP
非持續性回應不會放置在無法傳送的郵件佇列上或捨棄。 相反地，排入佇列的發佈/訂閱介面會取消現行作業，然後以適當的間隔重試。

若要判定此屬性的值，請搭配使用 MQIA_PUBSUB_NP_RESP 選取器與 MQINQ 呼叫。

已移轉佇列管理程式的預設值。
如果已從 WebSphere MQ V6.0 移轉佇列管理程式，則此屬性的起始值取決於移轉之前 DiscardNonPersistentResponse 及 DLQNonPersistent 回應的值，如下表所示。

		DLQNonPersistent 回應		
		是	否	未設定
DiscardNonPersistentResponse	是	MQUNDELIVERED_NORMAL	MQUNDELIVERED_DISCARD	MQUNDELIVERED_NORMAL
	否	MQUNDELIVERED_SAFE	MQUNDELIVERED_KEEP	MQUNDELIVERED_SAFE
	未設定	如果 SyncPoint 持續 = 否，則 MQUNDELIVERED_SAFE 否則 MQUNDELIVERED_NORMAL	如果 SyncPoint 持續 = 否，則 MQUNDELIVERED_KEEP ELSE MQUNDELIVERED_DISCARD	如果 SyncPoint 持續 = 否，則 MQUNDELIVERED_SAFE 否則 MQUNDELIVERED_NORMAL

PubSubMaxMsgRetryCount (MQLONG)
在同步點下處理失敗指令訊息時的重試次數。

此值是下列其中一個:

0 - 999 999 999
預設值為 5。

若要判定此屬性的值，請搭配使用 MQIA_PUBSUB_MAXMSG_RETRY_COUNT 選取器與 MQINQ 呼叫。

PubSubSyncPoint (MQLONG)
是否只在同步點下處理持續訊息或所有訊息。

此值是下列其中一個:

MQSYNCPOINT_IFPER
這會使排入佇列的發佈/訂閱介面在同步點之外接收非持續訊息。 如果常駐程式在同步點之外收到發佈，則常駐程式會將發佈轉遞給它在同步點之外已知的訂閱者。

這是預設值。

MQ 同步點 _ 是
這會讓排入佇列的發佈/訂閱介面接收同步點下的所有訊息。

若要判定此屬性的值，請搭配使用 MQIA_PUBSUB_SYNC_PT 選取器與 MQINQ 呼叫。

PubSub 模式 (MQLONG)
發佈/訂閱引擎及排入佇列的發佈/訂閱介面是否在執行中，因此容許應用程式使用應用程式設計介面及排入佇列的發佈/訂閱介面所監視的佇列來發佈/訂閱。

此值是下列其中一個:

MQPSM_COMPAT
發佈/訂閱引擎正在執行中。 因此，可以使用應用程式設計介面來發佈/訂閱。 已排入佇列的發佈/訂閱介面不在執行中，因此不會處理放入已排入佇列的發佈/訂閱介面所監視之佇列的任何訊息。 此設定用於與使用此佇列管理程式的 WebSphere Message Broker V6 或更早版本相容，因為它必須讀取已排入佇列的發佈/訂閱介面正常讀取的相同佇列。

已停用 MQPSM_DISABLED
發佈/訂閱引擎及排入佇列的發佈/訂閱介面不在執行中。 因此，無法使用應用程式設計介面來發佈/訂閱。 不會處理放入佇列發佈/訂閱介面所監視之佇列的任何發佈/訂閱訊息。

已啟用 MQPSM_ENABLED
發佈/訂閱引擎及排入佇列的發佈/訂閱介面正在執行中。 因此，可以使用應用程式設計介面及佇列發佈/訂閱介面所監視的佇列來發佈/訂閱。 這是佇列管理程式的起始預設值。

若要判定此屬性的值，請搭配使用 MQIA_PUBSUB_MODE 選取元與 MQINQ 呼叫。

QMgrDesc (MQCHAR64)

此欄位用於說明佇列管理程式的註解。 該欄位的內容對佇列管理程式不重要，但佇列管理程式可能需要該欄位只包含可顯示的字元。 它不能包含任何空值字元; 必要的話，會以空白填補右邊。 在 DBCS 安裝中，這個欄位可以包含 DBCS 字元 (欄位長度上限為 64 個位元組)。

註: 如果此欄位包含不在佇列管理程式字集中的字元 (如 *CodedCharSetId* 佇列管理程式屬性所定義) ，則當此欄位傳送至另一個佇列管理程式時，可能會不正確地轉換這些字元。

- 在 z/OS 上，預設值是產品名稱和版本號碼。
- 在所有其他環境中，預設值為空白。

若要判定此屬性的值，請搭配使用 MQCA_Q_MGR_DESC 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_Q_MGR_DESC_LENGTH 提供。

QMgrIdentifier (MQCHAR48)

這是內部產生的佇列管理程式唯一名稱。

若要判定此屬性的值，請搭配使用 MQCA_Q_MGR_IDENTIFIER 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_Q_MGR_IDENTIFIER_LENGTH 提供。

在下列環境中支援此屬性: AIX、 HP-UX、 z/OS、 IBM i、Solaris、 Linux、 Windows，以及連接至這些系統的 WebSphere MQ 用戶端。

QMgrName (MQCHAR48)

這是本端佇列管理程式的名稱，亦即應用程式所連接的佇列管理程式名稱。

名稱的前 12 個字元用於建構唯一訊息 ID (請參閱 MQMD- MsgId 欄位)。 因此，可以交互通訊的佇列管理程式必須具有前 12 個字元不同的名稱，以便訊息 ID 在佇列管理程式網路中是唯一的。

在 z/OS 上，名稱與子系統名稱相同，但限制為 4 個非空白字元。

若要判定此屬性的值，請搭配使用 MQCA_Q_MGR_NAME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_Q_MGR_NAME_LENGTH 提供。

QSGName (MQCHAR4)

這是本端佇列管理程式所屬的佇列共用群組名稱。 如果本端佇列管理程式不屬於佇列共用群組，則名稱為空白。

若要判定此屬性的值，請搭配使用 MQCA_QSG_NAME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_QSG_NAME_LENGTH 提供。

此屬性僅在 z/OS 上受支援。

QueueAccounting (MQLONG)

這會控制佇列的帳戶資訊收集。

此值是下列其中一個:

MQMON_NONE
不收集佇列的帳戶資料，不論佇列帳戶屬性 ACCTQ 的設定為何。 這是預設值。

MQMON_OFF
請勿收集在 ACCTQ 佇列屬性中指定 QMGR 之佇列的結算資料。

MQMON_ON
收集在 ACCTQ 佇列屬性中指定 QMGR 之佇列的結算資料。

此值的變更僅對在變更屬性之後發生的佇列管理程式連線有效。

若要判定此屬性的值，請搭配使用 MQIA_ACCOUNTING_Q 選取器與 MQINQ 呼叫。

QueueMonitoring (MQLONG)

這會指定佇列線上監視的預設值。

如果 *QueueMonitoring* 佇列屬性設為 MQMON_Q_MGR ，則此屬性指定通道所假設的值。 值可以為:

MQMON_OFF
已關閉線上監視資料收集。 這是佇列管理程式的起始預設值。

MQMON_NONE
不論佇列的 *QueueMonitoring* 屬性設定為何，都會關閉佇列的線上監視資料收集。

MQMON_LOW
已開啟線上監視資料收集，資料收集的比例較低。

MQ mon_MEDIT
已開啟線上監視資料收集，且資料收集比例中等。

MQMON_HIGH
線上監視資料收集已開啟，資料收集的比例很高。

若要判定此屬性的值，請搭配使用 MQIA_MONITORING_Q 選取器與 MQINQ 呼叫。

QueueStatistics (MQLONG)

這會控制佇列的統計資料收集。

它是下列其中一個值:

MQMON_NONE
不論 *QueueStatistics* 佇列屬性的設定為何，都不要收集佇列的佇列統計資料。 這是預設值。

MQMON_OFF
不要收集在 *QueueStatistics* 佇列屬性中指定「佇列管理程式」之佇列的統計資料。

MQMON_ON
收集在 *QueueStatistics* 佇列屬性中指定「佇列管理程式」之佇列的統計資料。

若要判定此屬性的值，請搭配使用 MQIA_STATISTICS_Q 選取器與 MQINQ 呼叫。

ReceiveTimeout (MQLONG)

這指定在回到非作用中狀態之前， TCP/IP 通道等待從其友機接收資料 (包括活動訊號) 的時間長度。 它只適用於訊息通道，不適用於 MQI 通道。

ReceiveTimeout 的確切意義由 ReceiveTimeout 類型中指定的值變更。 ReceiveTimeout 類型可以設為下列其中一項:

- MQRCVTIME_EQUAL-此值是通道等待的秒數。 請指定 0-999999 範圍內的值。
- MQRCVTIME_ADD-此值是新增至協議 HBINT 的秒數，它會決定通道等待的時間長度。 請指定 1-999999 範圍內的值。
- MQRCVTIME_MULTIPLY-此值是乘數，適用於協議的 HBINT。 請指定 0 值或 2-99 範圍內的值。

預設值為 0。

將 ReceiveTimeout 類型設為 MQRCVTIME_MULTIPLY 或 MQRCVTIME_EQUAL ， 並將 ReceiveTimeout 設為 0 ， 以停止通道等待從其夥伴接收資料的逾時。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_RECEIVE_TIMEOUT 選取元與 MQINQ 呼叫。

ReceiveTimeout 下限 *(MQLONG)*

這是 TCP/IP 通道在回到非作用中狀態之前，等待從其友機接收資料 (包括活動訊號) 的最短時間 (秒)。

它只適用於訊息通道，不適用於 MQI 通道。 此值必須在 0 到 999999 的範圍內，預設值為 0。

如果您使用 ReceiveTimeout 類型，指定要相對於 HBINT 的協議值來計算 TCP/IP 通道等待時間，且產生的值小於此參數的值，則會改用此值。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_RECEIVE_TIMEOUT_MIN 選取元與 MQINQ 呼叫。

ReceiveTimeout 類型 *(MQLONG)*

這是套用至 ReceiveTimeout 的限定元，定義 TCP/IP 通道在回到非作用中狀態之前等待從其友機接收資料 (包括活動訊號) 的時間長度。 它只適用於訊息通道，不適用於 MQI 通道。

此值是下列其中一個:

MQRCVTIME_MULTIPLY
ReceiveTimeout 是乘數，可套用至協議的 HBINT 值以決定通道等待的時間。 這是預設值。

MQRCVTIME_ADD
ReceiveTimeout 是要新增至協議 HBINT 值的值 (以秒為單位) ， 以決定通道等待的時間長度。

MQRCVTIME_EQUAL
ReceiveTimeout 是通道等待的值 (以秒為單位)。

若要停止通道等待從其夥伴接收資料逾時，請將 ReceiveTimeout 類型設為 MQRCVTIME_multiply 或 MQRCVTIME_EQUAL ， 並將 ReceiveTimeout 設為 0。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_RECEIVE_TIMEOUT_TYPE 選取元與 MQINQ 呼叫。

RemoteEvent (MQLONG)

這會控制是否產生遠端錯誤事件。 它是下列其中一個值:

已停用 MQEVR_DISABLED
事件報告已停用。

已啟用 MQEVR_ENABLED
已啟用事件報告。

如需事件的相關資訊，請參閱 事件監視 。

若要判定此屬性的值，請搭配使用 MQQIA_REMOTE_Event 選取器與 MQINQ 呼叫。

RepositoryName (MQCHAR48)

這是此佇列管理程式為其提供儲存庫管理程式服務的叢集名稱。 如果佇列管理程式為多個叢集提供此服務，則 *RepositoryNamelist* 會指定識別叢集的名單物件名稱，且 *RepositoryName* 為空白。 至少其中一個 *RepositoryName* 和 *RepositoryNamelist* 必須為空白。

此屬性僅在 AIX、 HP-UX、 IBM i、 Linux、Solaris、 Windows 及 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQCA_REPOSITORY_NAME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_Q_MGR_NAME_LENGTH 提供。

RepositoryNamelist (MQCHAR48)

這是名稱清單物件的名稱，包含此佇列管理程式為其提供儲存庫管理程式服務的叢集名稱。 如果佇列管理程式只為一個叢集提供此服務，則名單物件只會包含一個名稱。 或者， *RepositoryName* 可以用來指定叢集的名稱，在此情況下 *RepositoryNamelist* 為空白。 至少其中一個 *RepositoryName* 和 *RepositoryNamelist* 必須為空白。

此屬性僅在 AIX、 HP-UX、 IBM i、 Linux、Solaris、 Windows 及 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQCA_REPOSITORY_NAMELIST 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_NAMELIST_NAME_LENGTH 提供。

ScyCase(MQCHAR8)

指定佇列管理程式是否支援大小寫混合格式的安全設定檔名稱，或只支援大寫。

此值是下列其中一個:

MQ 循環 _ 上層
安全設定檔名稱必須是大寫。

MQ SCYC_MIXED
安全設定檔名稱可以大寫或大小寫混合格式。

在指定 *SecurityType(MQSECTYPE_CLASSES)* 的情況下執行「重新整理安全」指令時，此屬性的變更會生效。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQQIA_SECURITY_案例選取元與 MQINQ 呼叫。

SharedQMgr 名稱 (MQLONG)

這會指定當「 *ObjectQmgrName* 」是佇列共用群組中另一個佇列管理程式的「」時，是否應該在共用佇列的 MQOPEN 呼叫中使用或將「 *ObjectQmgrName* 」視為本端佇列管理程式。

值可以為:

MQSQQM_USE
使用 *ObjectQmgrName* 並開啟適當的傳輸佇列。

MQSQQM_IGNORE
如果目標佇列是共用的，且 *ObjectQmgrName* 是相同佇列共用群組中佇列管理程式的，則會在本端執行開啟。

此屬性僅適用於 z/OS。

若要判定此屬性的值，請搭配使用 MQQIA_SHARED_Q_Q_MGR_NAME 選取器與 MQINQ 呼叫。

SPLCAP
指出 WebSphere MQ Advanced Message Security 的安全功能是否可用於佇列管理程式。

支援 MQCAP_SUPPORTED
如果已針對執行佇列管理程式的安裝安裝安裝 WebSphere MQ AMS 元件，則這是預設值。

不支援 MQCAP_NOT_SUPPORTED

SSLEvent (MQLONG)
這指定是否產生 SSL 事件。

它是下列其中一個值:

已啟用 MQEVR_ENABLED
產生 SSL 事件，如下所示:

MQRC_CHANNEL_SSL_ERROR

已停用 MQEVR_DISABLED
不產生 SSL 事件; 這是預設值。

若要判定此屬性的值，請搭配使用 MQIA_SSL_EVENT 選取器與 MQINQ 呼叫。

SSLFIPSRequired (MQLONG)

這可讓您指定如果在 WebSphere MQ 中執行加密法，而不是在加密硬體中執行，則只會使用 FIPS 認證的演算法。 如果已配置加密硬體，則使用的加密法模組是硬體產品所提供的那些模組; 這些模組可能或可能未通過 FIPS 認證，達到特定層次，視使用中的硬體產品而定。

該值是下列其中一個值:

MQSSL_FIPS_NO
使用使用中平台所支援的任何 CipherSpec 。 此值為預設值。

MQSSL_FIPS_YES
在此佇列管理程式的所有 SSL 連線上，只容許在 CipherSpecs 中使用 FIPS 認證的加密演算法。

此參數僅適用於 UNIX、 Linux、 Windows 及 z/OS 平台。

若要判定此屬性的值，請搭配使用 MQQIA_SSL_FIPS_required 選取器與 MQINQ 呼叫。

相關工作
指定在執行時期於 MQI 用戶端上僅使用 FIPS 認證的 CipherSpecs
相關參考
UNIX、 Linux 和 Windows 的聯邦資訊存取安全標準 (FIPS)

SSLKeyReset 計數 *(MQLONG)*
這指定起始通訊的 SSL 通道訊息通道代理程式 (MCA) 何時重設通道上用於加密的秘密金鑰。

該值代表在重新協議秘密金鑰之前，在通道上傳送及接收的未加密位元組總數。 位元組數包括 MCA 所傳送的控制資訊。

此值是 0 到 999 999 999 範圍內的數字，預設值為 0。 如果您指定 1 位元組到 32 KB 範圍內的 SSL/TLS 秘密金鑰重設計數，則 SSL/TLS 通道將使用 32 KB 的秘密金鑰重設計數。 這是為了避免對小型 SSL/TLS 秘密金鑰重設值進行過多金鑰重設的處理成本。

當起始通道 MCA 所傳送及接收的未加密位元組總數超出指定的值時，或如果在通道活動訊號之後傳送或接收資料之前已啟用通道活動訊號，則會重新協議秘密金鑰 (以先發生者為準)。

重新協議所傳送及接收的位元組計數包括通道 MCA 所傳送及接收的控制資訊，而且每當重新協議發生時都會重設。

使用值 0 表示永不重新協議秘密金鑰。

若要判定此屬性的值，請搭配使用 MQIA_SSL_RESET_COUNT 選取器與 MQINQ 呼叫。

StartStop 事件 (MQLONG)

這會控制是否產生啟動和停止事件。 此值是下列其中一個:

已停用 MQEVR_DISABLED
事件報告已停用。

已啟用 MQEVR_ENABLED
已啟用事件報告。

如需事件的相關資訊，請參閱 事件監視 。

若要判定此屬性的值，請搭配使用 MQQIA_START_STOP_Event 選取器與 MQINQ 呼叫。

StatisticsInterval (MQLONG)

這會指定將統計資料監視資料寫入監視佇列的頻率 (以秒為單位)。

此值是 0 到 604800 範圍內的整數，預設值為 1800 (30 分鐘)。

若要判定此屬性的值，請搭配使用 MQQIA_STATISTICS_INTERVAL 選取器與 MQINQ 呼叫。

SyncPoint (MQLONG)

這指出本端佇列管理程式是否支援工作單元，以及與 MQGET、MQPUT 及 MQPUT1 呼叫同步。

MQSP_available
可用的工作單元和同步點。

MQSP_NOT_AVAILABLE
無法使用工作單元和同步點。

- 在 z/OS 上，永不會傳回此值。

若要判定此屬性的值，請搭配使用 MQIA_SYNCPOINT 選取器與 MQINQ 呼叫。

TCPChannels (MQLONG)

這是使用 TCP/IP 傳輸通訊協定的現行或可連接用戶端的通道數上限。

此值必須在 0 到 9999 的範圍內，預設值為 200。 如果您指定 0，則不會使用 TCP/IP。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQQIA_TCP_通道選取器與 MQINQ 呼叫。

TCPKeepAlive (MQLONG)

這指定是否使用 TCP KEEPALIVE 來檢查連線的另一端是否仍然可用。 如果無法使用，通道會關閉。

此值是下列其中一個:

MQTCPKEEP_YES
使用 TCP 設定檔配置資料集中指定的 TCP KEEPALIVE。 如果您指定通道屬性 KeepAliveInterval (KAINT)，則會使用它設定的值。

MQTCPKEEP_NO
請勿使用 TCP KEEPALIVE。 這是預設值。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQQIA_TCP_KEEP_ALIVE 選取器與 MQINQ 呼叫。

TCPName (MQCHAR8)

這是您正在使用的唯一或預設 TCP/IP 系統的名稱，視 TCPStackType 的值而定。 預設值為 TCPIP。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQCA_TCP_NAME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_TCP_NAME_LENGTH 提供。

TCPStackType (MQLONG)

這指定通道起始程式是否只能使用 TCPName 中指定的 TCP/IP 位址空間，或選擇性地連結至任何選取的 TCP/IP 位址

此值是下列其中一個：

MQTCPSTACK_SINGLE
通道起始程式只能使用 TCPName 中指定的 TCP/IP 位址空間。 這是預設值。

MQTCPSTACK_MULTIPLE
通道起始程式可以使用任何可用的 TCP/IP 位址空間。 如果沒有為通道或接聽器指定其他值，則它預設為 TCPName 中指定的值。

此屬性僅在 z/OS 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_TCP_STACK_TYPE 選取器與 MQINQ 呼叫。

TraceRoute 記錄 *(MQLONG)*

這會控制追蹤路徑資訊的記錄。

此值是下列其中一個：

已停用 MQRECORDING_DISABLED
不容許附加至追蹤路徑訊息。

MQRECORDING_Q
將追蹤路徑訊息放置到固定指名佇列。

MQRECORDING_MSG
將追蹤路徑訊息放置到使用訊息本身所決定的佇列。 這是預設值

若要判定此屬性的值，請搭配使用 MQIA_TRACE_ROUTE_RECORDING 選取器與 MQINQ 呼叫。

TriggerInterval (MQLONG)

這是用來限制觸發訊息數目的時間間隔 (毫秒)。 這只有在 *TriggerType* 是 MQTT_FIRST 時才相關。 在此情況下，通常只有在適當的訊息到達佇列且佇列先前是空的時，才會產生觸發訊息。 不過，在某些情況下，即使佇列不是空的，也可以使用 MQTT_FIRST 觸發來產生其他觸發訊息。 這些額外的觸發訊息不會比每 *TriggerInterval* 毫秒產生一次更頻繁。

如需觸發的相關資訊，請參閱 觸發通道 。

該值不小於 0 且不大於 999 999 999。 預設值是 999 999 999 999。

若要判定此屬性的值，請搭配使用 MQQIA_TRIGGER_INTERVAL 選取器與 MQINQ 呼叫。

TriggerInterval (MQLONG)

這是用來限制觸發訊息數目的時間間隔 (毫秒)。 這只有在 *TriggerType* 是 MQTT_FIRST 時才相關。 在此情況下，通常只有在適當的訊息到達佇列且佇列先前是空的時，才會產生觸發訊息。 不過，在某些情況下，即使佇列不是空的，也可以使用 MQTT_FIRST 觸發來產生其他觸發訊息。 這些額外的觸發訊息不會比每 *TriggerInterval* 毫秒產生一次更頻繁。

如需觸發的相關資訊，請參閱 觸發通道 。

該值不小於 0 且不大於 999 999 999。 預設值是 999 999 999 999。

若要判定此屬性的值，請搭配使用 MQQIA_TRIGGER_INTERVAL 選取器與 MQINQ 呼叫。

版本 *(MQCFST)*

這是 WebSphere MQ 程式碼的 VVRRMMFF 版本，其中:

VV-版本

RR-發行

MM-維護層次

FF-修正層次

XrCapability(MQLONG)

這會控制佇列管理程式是否支援 WebSphere MQ Telemetry 指令。

此值是下列其中一個:

支援 MQCAP_SUPPORTED
支援已安裝的 WebSphere MQ Telemetry 元件及遙測指令。

不支援 MQCAP_NOT_SUPPORTED
webSphere MQ Telemetry 元件。

此屬性僅在 IBM i、Unix 系統及 Windows 上受支援。

若要判定此屬性的值，請搭配使用 MQIA_XR_capability 選取器與 MQINQ 呼叫。

佇列的屬性

佇列定義有五種類型。 部分佇列屬性適用於所有類型的佇列; 其他佇列屬性僅適用於特定類型的佇列。

佇列類型

佇列管理程式支援下列類型的佇列定義:

本端佇列
您可以將訊息儲存在本端佇列上。 在 z/OS 上，您可以使它成為共用或專用佇列。

如果佇列是由程式連接到的佇列管理程式所擁有，則該佇列對程式而言是本端佇列。 您可以從本端佇列中取得訊息，以及將訊息放置在本端佇列上。

佇列定義物件會保留佇列的定義資訊，以及放置在該佇列上的實體訊息。

本端佇列管理程式佇列
佇列存在於本端佇列管理程式上。 在 z/OS 上，此佇列稱為專用佇列。

共用佇列 (僅限 z/OS)
佇列存在於共用儲存庫中，所有屬於擁有共用儲存庫之佇列共用群組的佇列管理程式都可以存取該共用儲存庫。

連接至佇列共用群組中任何佇列管理程式的應用程式可以將訊息放置在此類型的佇列上，並從中移除訊息。 這類佇列實際上與本端佇列相同。 *QType* 佇列屬性的值為 MQQT_LOCAL。

連接至本端佇列管理程式的應用程式可以在此類型的佇列中放置及移除訊息。 *QType* 佇列屬性的值為 MQQT_LOCAL。

叢集佇列

您可以在定義訊息的佇列管理程式上，將訊息儲存在叢集佇列中。 叢集佇列是由叢集佇列管理程式所管理的佇列，並可提供給叢集的其他佇列管理程式使用。 *QType* 佇列屬性的值為 MQQT_CLUSTER。

叢集佇列定義會通告至叢集中的其他佇列管理程式。 叢集中的其他佇列管理程式不需要相對應的遠端佇列定義，就可以將訊息放置在叢集佇列中。 可以使用叢集名稱清單在多個叢集中通告叢集佇列。

在通告佇列時，叢集中的任何佇列管理程式都可以將訊息置入該佇列中。 若要放置訊息，佇列管理程式必須從完整儲存庫中找出管理該佇列的位置。 然後，它會將一些遞送資訊新增到訊息中，並將訊息放置在叢集傳輸佇列上。

除了 z/OS 之外，佇列管理程式可以將叢集中其他佇列管理程式的訊息儲存在多個傳輸佇列上。 您可以採用兩種不同的方式來配置佇列管理程式，以將訊息儲存在多個叢集傳輸佇列上。 如果您將佇列管理程式屬性 DEFCLXQ 設定為 CHANNEL，則會從 SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE 中為每一個叢集傳送端通道自動建立一個不同的叢集傳輸佇列。 如果將 CLCHNAME 傳輸佇列選項設定為符合一個以上的叢集傳送端通道，則佇列管理程式可以將相符通道的訊息儲存在該傳輸佇列上。

叢集佇列可以是 IBM WebSphere MQ for z/OS 中的佇列共用群組成員所共用的佇列。

遠端佇列

遠端佇列不是實體佇列; 它是存在於遠端佇列管理程式上之佇列的本端定義。 遠端佇列的本端定義包含告知本端佇列管理程式如何將訊息遞送至遠端佇列管理程式的資訊。

連接至本端佇列管理程式的應用程式可以將訊息放置在這種類型的佇列上; 這些訊息會放置在本端傳輸佇列上，用來將訊息遞送至遠端佇列管理程式。 應用程式無法從遠端佇列移除訊息。 *QType* 佇列屬性的值為 MQQT_REMOTE。

您也可以將遠端佇列定義用於:

- 回覆佇列別名化

  在此情況下，定義的名稱是回覆目的地佇列的名稱。 如需相關資訊，請參閱 回覆目的地佇列別名及叢集。

- 佇列管理程式別名化

  在此情況下，定義的名稱是佇列管理程式的別名，而不是佇列的名稱。 如需相關資訊，請參閱 佇列管理程式別名及叢集。

別名佇列

這不是實體佇列; 它是本端佇列、共用佇列、叢集佇列或遠端佇列的替代名稱。 別名所解析成的佇列名稱是別名佇列定義的一部分。

連接至本端佇列管理程式的應用程式可以將訊息放置在這種類型的佇列上; 訊息會放置在別名所解析成的佇列上。 如果別名解析為本端佇列、共用佇列或具有本端實例的叢集佇列，則應用程式可以從這種類型的佇列中移除訊息。 *QType* 佇列屬性的值為 MQQT_ALIAS。

模型佇列

這不是實體佇列; 它是一組可從中建立本端佇列的佇列屬性。

訊息無法儲存在此類型的佇列上。

「佇列」屬性

部分佇列屬性適用於所有類型的佇列; 其他佇列屬性僅適用於特定類型的佇列。 套用屬性的佇列類型顯示在 第 732 頁的表 573 及後續表格中。

第 732 頁的表 573 彙總特定於佇列的屬性。 屬性按字母順序說明。

註: 本節中顯示的屬性名稱是與 MQINQ 及 MQSET 呼叫搭配使用的敘述性名稱; 這些名稱與 PCF 指令的名稱相同。 當使用 MQSC 指令來定義、變更或顯示屬性時，會使用替代簡稱; 如需詳細資料，請參閱 Script (MQSC) 指令 。

表 *573:* 佇列的屬性. 直欄套用如下:

- 本端佇列的直欄也適用於共用佇列。
- 模型佇列的直欄指出從模型佇列建立的本端佇列繼承哪些屬性。
- 叢集佇列的直欄指出在開啟叢集佇列以單獨查詢或查詢及輸出時可以查詢的屬性。 如果查詢任何其他屬性，則呼叫會傳回完成碼 MQCC_WARNING 及原因碼 MQRC_SELECTOR_NOT_FOR_TYPE (2068)。

  如果開啟叢集佇列以進行查詢，並加上一或多個輸入、瀏覽或設定，則會改為套用本端佇列的直欄。

  如果只開啟叢集佇列進行查詢，或開啟叢集佇列進行查詢及輸出，加上指定基本佇列管理程式名稱，則會改為套用本端佇列的直欄。

屬性	說明	本端	模型	別名	遠端	叢集
AlterationDate	前次變更定義的日期	✓		✓	✓	
AlterationTime	前次變更定義的時間	✓		✓	✓	
BackoutRequeueQName	取消重新排入佇列的佇列名稱過多	✓	✓			

表 573: 佇列的屬性. 直欄套用如下:

- 本端佇列的直欄也適用於共用佇列。
- 模型佇列的直欄指出從模型佇列建立的本端佇列繼承哪些屬性。
- 叢集佇列的直欄指出在開啟叢集佇列以單獨查詢或查詢及輸出時可以查詢的屬性。 如果查詢任何其他屬性，則呼叫會傳回完成碼 MQCC_WARNING 及原因碼 MQRC_SELECTOR_NOT_FOR_TYPE (2068)。

  如果開啟叢集佇列以進行查詢，並加上一或多個輸入、瀏覽或設定，則會改為套用本端佇列的直欄。

  如果只開啟叢集佇列進行查詢，或開啟叢集佇列進行查詢及輸出，加上指定基本佇列管理程式名稱，則會改為套用本端佇列的直欄。

(繼續)

屬性	說明	本端	模型	別名	遠端	叢集
BackoutThreshold	取消臨界值	✓	✓			
BaseQName	別名解析成的佇列名稱			✓		
CFStrucName	連結機能結構名稱	✓	✓			
CLCHNAME	叢集傳送端通道名稱	✓	✓			
ClusterName	佇列所屬叢集的名稱	✓		✓	✓	✓
ClusterNamelist	包含佇列所屬叢集名稱的名單物件名稱	✓		✓	✓	
CLWLQueuePriority	叢集工作量佇列優先順序	✓		✓	✓	✓
CLWLQueueRank	叢集工作量佇列等級	✓		✓	✓	✓
CLWLUseQ	使用遠端佇列	✓				
CreationDate	建立佇列的日期	✓				
CreationTime	建立佇列的時間	✓				
CurrentQDepth	現行佇列深度	✓				
DefaultPutResponse	預設放置回應	✓	✓	✓	✓	
DefBind	預設連結	✓		✓	✓	✓
DefinitionType attribute	佇列定義類型	✓	✓			
DefInputOpenOption	預設的輸入開啟選項	✓	✓			
DefPersistence	預設訊息持續性	✓	✓	✓	✓	✓
DefPriority	預設訊息優先順序	✓	✓	✓	✓	✓
DefReadAhead	預設先讀	✓	✓	✓		
DistLists	配送清單支援	✓	✓			
HardenGetBackout	是否維護精確的取消計數	✓	✓			
IndexType	索引類型	✓	✓			
InhibitGet	是否容許佇列的取得作業	✓	✓	✓		
InhibitPut	是否容許佇列的放置作業	✓	✓	✓	✓	✓
InitiationQName	起始佇列的名稱	✓	✓			
MaxMsgLength	訊息長度上限（以位元組為單位）	✓	✓			
MaxQDepth	佇列深度上限	✓	✓			
MsgDeliverySequence attribute	訊息遞送順序	✓	✓			
NonPersistentMessage Class	非持續訊息的可靠性目標	✓	✓			
OpenInputCount	針對輸入的開啟數目	✓				
OpenOutputCount	為了輸出而開啟的數目	✓				

表 573: 佇列的屬性. 直欄套用如下:

- 本端佇列的直欄也適用於共用佇列。
- 模型佇列的直欄指出從模型佇列建立的本端佇列繼承哪些屬性。
- 叢集佇列的直欄指出在開啟叢集佇列以單獨查詢或查詢及輸出時可以查詢的屬性。 如果查詢任何其他屬性，則呼叫會傳回完成碼 MQCC_WARNING 及原因碼 MQRC_SELECTOR_NOT_FOR_TYPE (2068)。

  如果開啟叢集佇列以進行查詢，並加上一或多個輸入、瀏覽或設定，則會改為套用本端佇列的直欄。

  如果只開啟叢集佇列進行查詢，或開啟叢集佇列進行查詢及輸出，加上指定基本佇列管理程式名稱，則會改為套用本端佇列的直欄。

(繼續)

屬性	說明	本端	模型	別名	遠端	叢集
PropertyControl	內容控制	✓	✓	✓		
ProcessName	處理程序名稱	✓	✓			
QDepthHighEvent attribute	是否產生「佇列深度高」事件	✓	✓			
QDepthHighLimit	佇列深度的高限制	✓	✓			
QDepthLowEvent attribute	是否產生「佇列深度低」事件	✓	✓			
QDepthLowLimit attribute	佇列深度的下限	✓	✓			
QDepthMaxEvent	是否產生「佇列已滿」事件	✓	✓			
QDesc	佇列說明	✓	✓	✓	✓	✓
QName	佇列名稱	✓		✓	✓	✓
QServiceInterval	佇列服務間隔的目標	✓	✓			
QServiceIntervalEvent attribute	是否產生「服務間隔高」或「服務間隔正常」事件	✓	✓			
QSGDisp attribute	佇列共用群組處置	✓		✓	✓	
QueueAccounting	佇列帳戶資料收集	✓	✓	✓	✓	✓
QueueMonitoring	佇列的線上監視資料	✓	✓			
QueueStatistics	佇列統計資料收集	✓	✓	✓	✓	✓
QType	佇列類型	✓		✓	✓	✓
RemoteQMgrName	遠端佇列管理程式的名稱				✓	
RemoteQName	遠端佇列的名稱				✓	
RetentionInterval	保留間隔	✓	✓			
Scope	佇列的項目是否也存在於 Cell 目錄中	✓		✓	✓	
Shareability	佇列可共用性	✓	✓			
StorageClass	佇列的儲存類別	✓	✓			
TriggerControl	觸發控制	✓	✓			
TriggerData	觸發資料	✓	✓			
TriggerDepth	觸發深度	✓	✓			
TriggerMsgPriority	觸發程式的臨界值訊息優先順序	✓	✓			
TriggerType	觸發類型	✓	✓			
Usage attribute	佇列使用情形	✓	✓			
XmitQName	傳輸佇列名稱				✓	

相關概念
叢集佇列數
本端佇列

AlterationDate (MQCHAR12)

前次變更定義的日期。

本端	模型	別名	遠端	叢集
X		X	X	

這是前次變更定義的日期。 日期的格式為 YYYY-MM-DD，並以兩個尾端空白填補，使長度為 12 個位元組 (例如， 1992-09-23¬¬，其中 ¬¬ 代表兩個空白字元)。

當佇列管理程式運作時，某些屬性 (例如， *CurrentQDepth*) 的值會變更。 這些屬性的變更不會影響 *AlterationDate*。

若要判定此屬性的值，請搭配使用 MQCA_ALTERATION_DATE 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_DATE_LENGTH 提供。

AlterationTime (MQCHAR8)

前次變更定義的時間。

本端	模型	別名	遠端	叢集
X		X	X	

這是前次變更定義的時間。 時間格式為 HH.MM.SS，使用 24 小時制，如果小時小於 10 (例如 09.10.20)，則為前導零。

- 在 z/OS 上，時間是格林威治標準時間 (GMT)，受精確設為 GMT 的系統時鐘影響。
- 在其他環境中，時間是當地時間。

當佇列管理程式運作時，某些屬性 (例如， *CurrentQDepth*) 的值會變更。 這些屬性的變更不會影響 *AlterationTime*。

若要判定此屬性的值，請搭配使用 MQCA_ALTERATION_TIME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_TIME_LENGTH 提供。

BackoutRequeue 完整名稱 *(MQCHAR48)*

這是過多的取消重新排入佇列佇列名稱。 除了容許查詢其值之外，佇列管理程式也不會根據此屬性的值採取任何動作。

本端	模型	別名	遠端	叢集
X	X			

在 WebSphere Application Server 內執行的應用程式，以及使用 WebSphere MQ Application Server 機能的應用程式，會使用這個屬性來判斷已取消的訊息應該送往何處。 對於所有其他應用程式，佇列管理程式不會根據屬性值採取任何動作。

WebSphere 適用於 JMS 的 MQ 類別會使用這個屬性來決定在何處傳送已取消 *BackoutThreshold* 屬性所指定次數上限的訊息。

若要判定此屬性的值，請搭配使用 MQCA_BACKOUT_REQ_Q_NAME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_Q_NAME_LENGTH 提供。

BackoutThreshold (MQLONG)
這是取消臨界值。 除了容許查詢其值之外，佇列管理程式也不會根據此屬性的值採取任何動作。

本端	模型	別名	遠端	叢集
X	X			

在 WebSphere Application Server 內執行的應用程式以及使用 WebSphere MQ Application Server 機能的應用程式將使用此屬性來判定是否應該取消訊息。 對於所有其他應用程式，佇列管理程式不會根據屬性值採取任何動作。

WebSphere 適用於 JMS 的 MQ 類別使用這個屬性來決定在將訊息傳送至 *BackoutRequeueQName* 屬性指定的佇列之前，容許取消訊息的次數。

若要判定此屬性的值，請搭配使用 MQIA_BACKOUT_THRESHOLD 選取器與 MQINQ 呼叫。

BaseQName (MQCHAR48)
這是定義給本端佇列管理程式的佇列名稱。

本端	模型	別名	遠端	叢集
		X		

(如需佇列名稱的相關資訊，請參閱 MQOD- ObjectName 欄位。) 佇列是下列其中一種類型:

本端 MQQT_LOCAL
本端佇列。

MQQT_REMOTE
遠端佇列的本端定義。

MQQT_CLUSTER
叢集佇列。

若要判定此屬性的值，請搭配使用 MQCA_BASE_Q_NAME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_Q_NAME_LENGTH 提供。

BaseType (MQCFIN)
別名所解析成的物件類型。

本端	模型	別名	遠端	叢集
		X		

它是下列其中一個值:

MQOT_Q
基本物件類型是佇列

MQOT_TOPIC
基本物件類型是主題

CFStrucName (MQCHAR12)
這是儲存佇列上訊息的連結機能結構名稱。 名稱的第一個字元是在範圍 A 到 Z 內，其餘字元是在範圍 A 到 Z、0 到 9 或空白內。

本端	模型	別名	遠端	叢集
X	X			

若要取得連結機能中結構的完整名稱，請以 *CFStrucName* 佇列屬性值作為 *QSGName* 佇列管理程式屬性值的字尾。

此屬性僅適用於共用佇列; 如果 *QSGDisp* 沒有值 MQQSGD_SHARED ，則會忽略此屬性。

若要判定此屬性的值，請搭配使用 MQCA_CF_STRUC_NAME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_CF_STRUC_NAME_LENGTH 提供。

此屬性僅在 z/OS 上受支援。

ClusterChannel 名稱 *(MQCHAR20)*

ClusterChannel **名稱** 是使用此佇列作為傳輸佇列之叢集傳送端通道的通用名稱。 該屬性指定哪些叢集傳送端通道將訊息從此叢集傳輸佇列傳送到叢集接收端通道。 z/OS 不支援 ClusterChannel **名稱** 。

本端	模型	別名	遠端	叢集
✓	✓			

預設佇列管理程式配置，可讓所有叢集傳送端通道從單一傳輸佇列 SYSTEM.CLUSTER.TRANSMIT.QUEUE 傳送訊息。 預設配置可以透過變更佇列管理程式屬性 DefClusterXmitQueueType 加以修改。 此屬性的預設值為 SCTQ。 您可以將此值變更為 CHANNEL。 如果將 DefClusterXmitQueueType 屬性設定為 CHANNEL，則每個叢集傳送端通道會預設為使用特定的叢集傳輸佇列 SYSTEM.CLUSTER.TRANSMIT.*ChannelName*。

您還可以手動將傳輸佇列屬性 ClusterChannelName 設定為叢集傳送端通道。 以叢集傳送端通道所連接的佇列管理程式為目的地的訊息，會儲存在識別叢集傳送端通道的傳輸佇列中。 它們不會儲存在預設叢集傳輸佇列中。 如果您將 ClusterChannelName 屬性設定為空白，當通道重新啟動時，通道會切換至預設叢集傳輸佇列。 預設佇列為 SYSTEM.CLUSTER.TRANSMIT.*ChannelName* 或 SYSTEM.CLUSTER.TRANSMIT.QUEUE，這取決於佇列管理程式 DefClusterXmitQueueType 屬性的值。

透過在 ClusterChannel **名稱**中指定星號 "*"，您可以將傳輸佇列與一組叢集傳送端通道相關聯。 星號可以位於通道名稱字串的開頭、結尾或中間任意位置。 ClusterChannelName 的長度限制為 20 個字元: MQ_CHANNEL_NAME_LENGTH。

ClusterName (MQCHAR48)

這是佇列所屬叢集的名稱。

本端	模型	別名	遠端	叢集
X		X	X	X

如果佇列屬於多個叢集，則 *ClusterNamelist* 會指定識別叢集的名單物件名稱，且 *ClusterName* 為空白。 至少其中一個 *ClusterName* 和 *ClusterNamelist* 必須為空白。

若要判定此屬性的值，請搭配使用 MQCA_CLUSTER_NAME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_CLUSTER_NAME_LENGTH 提供。

ClusterNamelist (MQCHAR48)

這是包含此佇列所屬叢集名稱的名單物件名稱。

本端	模型	別名	遠端	叢集
X		X	X	

如果佇列只屬於一個叢集，則名單物件只會包含一個名稱。 或者， *ClusterName* 可以用來指定叢集的名稱，在此情況下 *ClusterNamelist* 為空白。 至少其中一個 *ClusterName* 和 *ClusterNamelist* 必須為空白。

若要判定此屬性的值，請搭配使用 MQCA_CLUSTER_NAMELIST 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_NAMELIST_NAME_LENGTH 提供。

CLWLQueuePriority (MQLONG)
這是叢集工作量佇列優先順序，這是 0 到 9 範圍內的值，代表佇列的優先順序。

本端	模型	別名	遠端	叢集
X		X	X	X

如需相關資訊，請參閱 叢集佇列。

若要判定此屬性的值，請搭配使用 MQQIA_CLWL_Q_優先順序選取器與 MQINQ 呼叫。

CLWLQueueRank (MQLONG)
這是叢集工作量佇列等級，這是 0 到 9 範圍內的值，代表佇列等級。

本端	模型	別名	遠端	叢集
X		X	X	X

如需相關資訊，請參閱 叢集佇列。

若要判定此屬性的值，請搭配使用 MQIA_CLWL_Q_RANK 選取器與 MQINQ 呼叫。

CLWLUseQ (MQLONG)
這會定義當目標佇列同時具有本端實例及至少一個遠端叢集實例時， MQPUT 的行為。 如果放置源自叢集通道，則這個屬性不適用。

本端	模型	別名	遠端	叢集
X				

此值是下列其中一個：

MQCLWL_USEQ_ANY
使用遠端和本端佇列。

MQCLWL_USEQ_LOCAL
請勿使用遠端佇列。

MQCLWL_USEQ_AS_Q_MGR
從佇列管理程式的 MQIA_CLWL_USEQ 繼承定義。

如需相關資訊，請參閱 叢集佇列。

若要判定此屬性的值，請搭配使用 MQCA_CLWL_USEQ 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_CLWL_USEQ_LENGTH 提供。

CreationDate (MQCHAR12)
此為佇列的建立日期。

本端	模型	別名	遠端	叢集
X				

日期格式為 YYYY-MM-DD，並以兩個尾端空白填補，使長度為 12 個位元組 (例如， `2013-09-23¬¬`，其中 ¬¬ 代表 2 個空白字元)。

- 在 IBM i 上，佇列的建立日期可能與代表佇列的基礎作業系統實體 (檔案或使用者空間) 的建立日期不同。

若要判定此屬性的值，請搭配使用 MQCA_CREATION_DATE 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_CREATION_DATE_LENGTH 提供。

CreationTime (MQCHAR8)
這是建立佇列的時間。

本端	模型	別名	遠端	叢集
X				

時間格式為 HH.MM.SS，使用 24 小時制，如果小時小於 10 (例如 09.10.20)，則為前導零。

- 在 z/OS 上，時間是格林威治標準時間 (GMT)，受精確設為 GMT 的系統時鐘影響。
- 在其他環境中，時間是當地時間。
- 在 IBM i 上，佇列的建立時間可能與代表佇列的基礎作業系統實體 (檔案或使用者空間) 的建立時間不同。

若要判定此屬性的值，請搭配使用 MQCA_CREATION_TIME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_CREATION_TIME_LENGTH 提供。

CurrentQDepth (MQLONG)
這是佇列目前的訊息數量。

本端	模型	別名	遠端	叢集
X				

在 MQPUT 呼叫期間，以及在取消 MQGET 呼叫期間，它會增加。 在非瀏覽 MQGET 呼叫期間，以及在取消 MQPUT 呼叫期間，它會減少。 其效果是計數包括已放置在工作單元內佇列中，但尚未確定 (即使 MQGET 呼叫無法擷取它們) 的訊息。 同樣地，它會排除已使用 MQGET 呼叫在工作單元內擷取，但尚未確定的訊息。

計數也包括已超過到期時間但尚未捨棄的訊息，雖然這些訊息不適合擷取。 如需相關資訊，請參閱 MQMD-期限欄位 。

工作單元處理及訊息分段都可能導致 *CurrentQDepth* 超出 *MaxQDepth*。 不過，這不會影響訊息的可擷取性; 佇列上的 所有 訊息都可以正常使用 MQGET 呼叫來擷取。

此屬性的值會隨著佇列管理程式的運作而波動。

若要判定此屬性的值，請搭配使用 MQQIA_CURRENT_Q_DEPTH 選取器與 MQINQ 呼叫。

DefaultPut 回應 *(MQLONG)*
指定當應用程式指定 MQPMO_RESPONSE_AS_Q_DEF 時，要用於佇列放置作業的回應類型。

本端	模型	別名	遠端	叢集
X	X	X	X	

它是下列其中一個值:

MQPRT_SYNC_response
同步發出放置作業，並傳回回應。

MQPRT_ASYNC_RESPONSE
以非同步方式發出 put 作業，並傳回 MQMD 欄位子集。

DefBind (MQLONG)
這是在 MQOPEN 呼叫上指定 MQOO_BIND_AS_Q_DEF 且佇列是叢集佇列時使用的預設連結。

本端	模型	別名	遠端	叢集
X		X	X	X

此值是下列其中一個:

MQBND_BIND_ON_OPEN
MQOPEN 呼叫已修正連結。

MQBND_BIND_NOT_FIXED
未修正連結。

MQBND_BIND_ON_GROUP
容許應用程式要求將訊息群組全部配置給相同的目的地實例。 因為此值在 IBM WebSphere MQ Version 7.1 中是新的值，所以如果開啟此佇列的任何應用程式連接至 IBM WebSphere MQ Version 7.0.1 或更早版本的佇列管理程式，則不得使用此值。

若要判定此屬性的值，請搭配使用 MQIA_DEF_BIND 選取器與 MQINQ 呼叫。

DefinitionType (MQLONG)
這指出如何定義佇列。

本端	模型	別名	遠端	叢集
X	X			

此值是下列其中一個：

MQQDT_PREDEFINED
佇列是系統管理者所建立的永久佇列; 只有系統管理者可以刪除它。

預先定義的佇列是使用 DEFINE MQSC 指令所建立，且只能使用 DELETE MQSC 指令來刪除。 無法從模型佇列建立預先定義的佇列。

指令可以由操作員或將指令訊息傳送至指令輸入佇列的授權使用者發出 (如需相關資訊，請參閱 CommandInputQName 屬性)。

MQQDT_PERMANENT_DYNAMIC
佇列是由應用程式使用物件描述子 MQOD 中指定的模型佇列名稱發出 MQOPEN 呼叫所建立的永久佇列。 模型佇列定義具有 *DefinitionType* 屬性的 MQQDT_PERMANENT_DYNAMIC 值。

可以使用 MQCLOSE 呼叫來刪除此類型的佇列。 如需詳細資料，請參閱第 567 頁的『MQCLOSE-關閉物件』。

永久動態佇列的 *QSGDisp* 屬性值為 MQQSGD_Q_MGR。

MQQDT_TEMPORARY_DYNAMIC
佇列是由應用程式使用物件描述子 MQOD 中指定的模型佇列名稱發出 MQOPEN 呼叫所建立的暫時佇列。 模型佇列定義的 *DefinitionType* 屬性值為 MQQDT_TEMPORARY_DYNAMIC。

當 MQCLOSE 呼叫由建立此佇列的應用程式關閉時， MQCLOSE 呼叫會自動刪除此佇列類型。

暫時動態佇列的 *QSGDisp* 屬性值為 MQQSGD_Q_MGR。

MQQDT_SHARED_DYNAMIC
佇列是共用永久佇列，由應用程式使用物件描述子 MQOD 中指定的模型佇列名稱發出 MQOPEN 呼叫所建立。 模型佇列定義的 *DefinitionType* 屬性值為 MQQDT_SHARED_DYNAMIC。

可以使用 MQCLOSE 呼叫來刪除此類型的佇列。 如需詳細資料，請參閱第 567 頁的『MQCLOSE-關閉物件』。

共用動態佇列的 *QSGDisp* 屬性值是 MQQSGD_SHARED。

模型佇列定義中的這個屬性不會指出模型佇列的定義方式，因為模型佇列一律是預先定義的。 相反地，此屬性在模型佇列中的值是用來判定使用 MQOPEN 呼叫從模型佇列定義建立的每一個動態佇列的 *DefinitionType* 。

若要判定此屬性的值，請搭配使用 MQQIA_DEFINITION_TYPE 選取器與 MQINQ 呼叫。

DefInputOpenOption (MQLONG)
這是開啟佇列以供輸入的預設方式。

本端	模型	別名	遠端	叢集
X	X			

如果在開啟佇列時在 MQOPEN 呼叫上指定 MQOO_INPUT_AS_Q_DEF 選項，則適用。 此值是下列其中一個：

MQOO_INPUT_EXCLUSIVE
開啟佇列以取得具有專用存取權的訊息。

開啟佇列以與後續 MQGET 呼叫搭配使用。 如果佇列目前由這個或另一個應用程式開啟以進行任何類型的輸入 (MQOO_INPUT_SHARED 或 MQOO_INPUT_EXCLUSIVE) ，則呼叫會失敗，原因碼為 MQRC_OBJECT_IN_USE。

MQOO_INPUT_SHARED
開啟佇列以取得具有共用存取權的訊息。

開啟佇列以與後續 MQGET 呼叫搭配使用。 如果佇列目前由這個或另一個具有 MQOO_INPUT_SHARED 的應用程式開啟，則呼叫會成功，但如果佇列目前以 MQOO_INPUT_EXCLUSIVE 開啟，則會失敗，原因碼為 MQRC_OBJECT_IN_USE。

若要判定此屬性的值，請搭配使用 MQIA_DEF_INPUT_OPEN_OPTION 選取器與 MQINQ 呼叫。

DefPersistence (MQLONG)
這是佇列上訊息的預設持續性。 如果在放置訊息時在訊息描述子中指定 MQPER_PERSISTENCE_AS_Q_DEF ，則適用。

本端	模型	別名	遠端	叢集
X	X	X	X	X

如果佇列名稱解析路徑中有多個定義，則在 MQPUT 或 MQPUT1 呼叫時，會從路徑中 第一個 定義的這個屬性值取得預設持續性。 這可能是:

- 別名佇列
- 本端佇列
- 遠端佇列的本端定義
- 佇列管理程式別名
- 傳輸佇列 (例如， *DefXmitQName* 佇列)

此值是下列其中一個:

MQPER_PERSISTENT
訊息在系統失敗及佇列管理程式重新啟動之後仍然存在。 持續訊息無法放置在:

- 暫時動態佇列數
- 對映至 CFLEVEL (2) 或以下的 CFSTRUCT 物件，或 CFSTRUCT 物件定義為 RECOVER (NO) 的共用佇列。

持續訊息可以放置在永久動態佇列及預先定義佇列上。

MQPER_NOT_PERSISTENT
此訊息通常不會在系統失敗或佇列管理程式重新啟動之後繼續存在。 即使在佇列管理程式重新啟動期間，在輔助儲存體上找到完整的訊息副本，也會如此。

如果是共用佇列，非持續訊息 會 在佇列共用群組中的佇列管理程式重新啟動之後，但在共用佇列上用來儲存訊息的連結機能失敗之後，仍會繼續存在。

持續及非持續訊息都可以存在於相同的佇列中。

若要判定此屬性的值，請搭配使用 MQQIA_DEF_持續性選取器與 MQINQ 呼叫。

DefPriority (MQLONG)
這是佇列上訊息的預設優先順序。 當訊息放入佇列時，如果在訊息描述子中指定 MQPRI_PRIORITY_AS_Q_DEF ，則會套用此選項。

本端	模型	別名	遠端	叢集
X	X	X	X	X

如果佇列名稱解析路徑中有多個定義，則會從放置作業時路徑中 第一個 定義的這個屬性值取得訊息的預設優先順序。 這可能是:

- 別名佇列
- 本端佇列
- 遠端佇列的本端定義
- 佇列管理程式別名
- 傳輸佇列 (例如， *DefXmitQName* 佇列)

訊息放置在佇列上的方式取決於佇列的 *MsgDeliverySequence* 屬性值:

- 如果 *MsgDeliverySequence* 屬性為 MQMDS_PRIORITY ，則訊息放置在佇列上的邏輯位置取決於訊息描述子中 *Priority* 欄位的值。
- 如果 *MsgDeliverySequence* 屬性為 MQMDS_FIFO ，則不論訊息描述子中 *Priority* 欄位的值為何，都會將訊息放置在佇列上，彷彿它們的優先順序等於已解析佇列的 *DefPriority* 。 不過， *Priority* 欄位會保留放置訊息的應用程式所指定的值。 如需相關資訊，請參閱 MsgDelivery 順序屬性 。

優先順序介於範圍零 (最低) 到 *MaxPriority* (最高) 之間; 請參閱 MaxPriority 屬性。

若要判定此屬性的值，請搭配使用 MQQIA_DEF_優先順序選取器與 MQINQ 呼叫。

DefRead 自動搜尋 *(MQLONG)*

指定遞送至用戶端之非持續訊息的預設先讀行為。

本端	模型	別名	遠端	叢集
X	X	X		

DefReadAhead 可以設為下列其中一個值:

MQREADA_NO
在應用程式要求非持續訊息之前，它們不會先傳送至用戶端。 如果用戶端異常結束，最多可以遺失一則非持續訊息。

MQREADA_YES
非持續訊息會先傳送至用戶端，然後應用程式才會要求它們。 如果用戶端異常結束，或用戶端未耗用所傳送的所有訊息，則可能會遺失非持續訊息。

MQREADA_DISABLED
未針對此佇列啟用先讀非持續訊息。 不論用戶端應用程式是否要求先讀，訊息都不會先傳送至用戶端。

若要判定此屬性的值，請搭配使用 MQIA_DEF_READ_AHEAD 選取元與 MQINQ 呼叫。

DefPResp (MQLONG)

預設放置回應類型 (DEFPRESP) 屬性定義當 MQPMO 內的 PutResponse 類型已設為 MQPMO_RESPONSE_AS_Q_DEF 時，應用程式所使用的值。 此屬性適用於所有佇列類型。

本端	模型	別名	遠端	叢集
是	是	是	是	是

此值是下列其中一個:

同步
發出同步傳回回應的 put 作業。

ASYNC
以非同步方式發出 put 作業，並傳回 MQMD 欄位子集。

若要判定此屬性的值，請搭配使用 MQQIA_DEF_PUT_response_type 選取器與 MQINQ 呼叫。

DistLists (MQLONG)

這指出是否可以將配送清單訊息放置在佇列上。

本端	模型	別名	遠端	叢集
X	X			

訊息通道代理程式 (MCA) 會設定屬性來通知本端佇列管理程式，通道另一端的佇列管理程式是否支援配送清單。 後一個佇列管理程式 (稱為 結盟 佇列管理程式) 是在傳送的 MCA 從本端傳輸佇列中移除訊息之後，接下來會接收該訊息的佇列管理程式。

每當傳送端 MCA 建立與夥伴佇列管理程式上接收端 MCA 的連線時，傳送端 MCA 即會設定該屬性。 如此一來，傳送端 MCA 可能會導致本端佇列管理程式只將夥伴佇列管理程式可以正確處理的訊息放置在傳輸佇列上。

此屬性主要用於傳輸佇列，但不論為佇列定義的用法為何，都會執行所說明的處理 (請參閱 使用情形屬性)。

此值是下列其中一個：

支援 MQDL_SUPPORTED
配送清單訊息可以儲存在佇列上，並以該格式傳輸至夥伴佇列管理程式。 這會減少將訊息傳送至多個目的地所需的處理量。

不支援 MQDL_NOT_SUPPORTED
無法將配送清單訊息儲存在佇列上，因為夥伴佇列管理程式不支援配送清單。 如果應用程式放置配送清單訊息，且該訊息要放置在此佇列上，則佇列管理程式會分割配送清單訊息，並改為將個別訊息放置在佇列上。 這會增加將訊息傳送至多個目的地所需的處理量，但可確保夥伴佇列管理程式正確處理訊息。

若要判定此屬性的值，請搭配使用 MQQIA_DIST_清單選取器與 MQINQ 呼叫。 若要變更此屬性的值，請使用 MQSET 呼叫。

z/OS 不支援這個屬性。

HardenGet 取消 *(MQLONG)*
對於每一個訊息，會保留計數，指出工作單元內 MQGET 呼叫擷取訊息的次數，以及該工作單元隨後取消的次數。

本端	模型	別名	遠端	叢集
X	X			

在 MQGET 呼叫完成之後，此計數在訊息描述子的 *BackoutCount* 欄位中可用。

訊息取消計數在佇列管理程式重新啟動之後仍然存在。 不過，為了確保計數正確，每次 MQGET 呼叫擷取此佇列工作單元內的訊息時，都必須 強化 資訊 (記錄在磁碟或其他永久儲存裝置上)。 如果未這樣做，佇列管理程式會失敗，且 MQGET 呼叫會取消，計數可能會增加，也可能不會增加。

不過，強化工作單元內每一個 MQGET 呼叫的資訊會增加額外的處理成本，因此只有在計數精確很重要時，才將 *HardenGetBackout* 屬性設為 MQQA_BACKOUT_HARDENED。

在 IBM i、 UNIX 系統及 Windows 上，不論此屬性的設定為何，一律會強化訊息取消計數。

下列為可能的值：

MQQA_BACKOUT_HARDENED
「強化」是用來確保此佇列上訊息的取消計數是精確的。

MQQA_BACKOUT_NOT_HARDENED
不會使用「強化」來確保此佇列上訊息的取消計數是精確的。 因此，計數可能低於應該值。

若要判定此屬性的值，請搭配使用 MQIA_HARDEN_GET_BACKOUT 選取器與 MQINQ 呼叫。

IndexType (MQLONG)
這指定佇列管理程式針對佇列上的訊息所維護的索引類型。

本端	模型	別名	遠端	叢集
X	X			

所需的索引類型取決於應用程式擷取訊息的方式，以及佇列是共用佇列還是非共用佇列 (請參閱 QSGDisp 屬性)。 *IndexType* 可能有下列值:

無 MQIT_NONE
此佇列的佇列管理程式不會維護任何索引。 針對通常循序處理的佇列使用此值，亦即，在 MQGET 呼叫上不使用任何選取準則。

MQIT_MSG_ID
佇列管理程式會維護使用佇列上訊息之訊息 ID 的索引。 使用此值佇列，其中應用程式通常會使用訊息 ID 作為 MQGET 呼叫的選取準則來擷取訊息。

MQIT_CORREL_ID
佇列管理程式會維護一個索引，該索引使用佇列上訊息的相關性 ID。 對於應用程式通常使用相關性 ID 作為 MQGET 呼叫的選取準則來擷取訊息的佇列，請使用此值。

MQIT_MSG_TOKEN
佇列管理程式會維護一個索引，該索引使用佇列上訊息的訊息記號，以與 z/OS 的工作量管理程式 (WLM) 功能搭配使用。

您 必須 對 WLM 管理的佇列指定此選項; 請勿對任何其他類型的佇列指定此選項。 此外，對於應用程式未使用 z/OS 工作量管理程式函數的佇列，請勿使用此值，而是使用訊息記號作為 MQGET 呼叫的選取準則來擷取訊息。

MQIT_GROUP_ID
佇列管理程式會維護一個索引，以使用佇列上訊息的群組 ID。 此值 必須 用於應用程式在 MQGET 呼叫中使用 MQGMO_LOGICAL_ORDER 選項擷取訊息的佇列。

具有此索引類型的佇列不能是傳輸佇列。 必須定義具有此索引類型的共用佇列，以對映至 CFLEVEL (3) 或 CFLEVEL (4) 的 CFSTRUCT 物件。

註:

1. 未定義索引類型為 MQIT_GROUP_ID 之佇列上訊息的實體順序，因為在 MQGET 呼叫上使用 MQGMO_LOGICAL_ORDER 選項來最佳化佇列以有效地擷取訊息。 這表示訊息的實體順序通常不是訊息抵達佇列的順序。
2. 如果 MQIT_GROUP_ID 佇列具有 *MsgDeliverySequence* MQMDS_PRIORITY ，則佇列管理程式會使用訊息優先順序 0 及 1 來最佳化邏輯順序的訊息擷取。 因此，群組中第一個訊息的優先順序不得為零或一; 如果是，則會將訊息當作優先順序為 2 來處理。 MQMD 結構中的 *Priority* 欄位不變。

如需訊息群組的相關資訊，請參閱 MQGMO-選項欄位中群組和區段選項的說明。

在各種情況下應該使用的索引類型顯示在 第 744 頁的表 574 和 第 745 頁的表 575 中。

表 *574:* 未指定 *MQGMO_LOGICAL_ORDER* 時佇列索引類型的建議值或必要值

MQGET 呼叫的選取準則	**非共用佇列的索引類型**	**共用佇列的索引類型**
無	任意	任意
使用一個 ID 的選項:		
訊息 ID	建議 MQIT_MSG_ID	需要 MQIT_NONE 或 MQIT_MSG_ID; 建議使用 MQIT_MSG_ID
相關性 ID	建議 MQIT_CORREL_ID	需要 MQIT_CORREL_ID
群組 ID	建議 MQIT_GROUP_ID	需要 MQIT_GROUP_ID
使用兩個 ID 的選項:		

表 *574:* 未指定 *MQGMO_LOGICAL_ORDER* 時佇列索引類型的建議值或必要值 *(*繼續*)*		
MQGET 呼叫的選取準則	**非共用佇列的索引類型**	**共用佇列的索引類型**
訊息 ID 加上相關性 ID	建議 MQIT_MSG_ID 或 MQIT_CORREL_ID	需要 MQIT_NONE 或 MQIT_MSG_ID 或 MQIT_CORREL_ID (為了提高效率，建議選擇索引類型以符合將具有最明確索引鍵的 MQMD 欄位)
訊息 ID 加上群組 ID	建議 MQIT_MSG_ID 或 MQIT_GROUP_ID	不支援
相關性 ID 加上群組 ID	建議 MQIT_CORREL_ID 或 MQIT_GROUP_ID	不支援
使用三個 ID 的選項:		
訊息 ID 加上相關性 ID 加上群組 ID	建議 MQIT_MSG_ID 或 MQIT_CORREL_ID 或 MQIT_GROUP_ID	不支援
使用群組相關準則的選項:		
群組 ID 加上訊息序號	需要 MQIT_GROUP_ID	需要 MQIT_GROUP_ID
訊息序號 (必須是 1)	需要 MQIT_GROUP_ID	需要 MQIT_GROUP_ID
使用訊息記號的選項:		
應用程式使用的訊息記號	不要使用 MQIT_MSG_TOKEN	
WLM 使用的訊息記號	需要 MQIT_MSG_TOKEN	不支援

表 *575:* 指定 *MQGMO_LOGICAL_ORDER* 時佇列索引類型的建議值或必要值		
MQGET 呼叫的選取準則	**非共用佇列的索引類型**	**共用佇列的索引類型**
無	需要 MQIT_GROUP_ID	需要 MQIT_GROUP_ID
使用一個 ID 的選項:		
訊息 ID	需要 MQIT_GROUP_ID	不支援
相關性 ID	需要 MQIT_GROUP_ID	不支援
群組 ID	需要 MQIT_GROUP_ID	需要 MQIT_GROUP_ID
使用兩個 ID 的選項:		
訊息 ID 加上相關性 ID	需要 MQIT_GROUP_ID	不支援
訊息 ID 加上群組 ID	需要 MQIT_GROUP_ID	不支援
相關性 ID 加上群組 ID	需要 MQIT_GROUP_ID	不支援
使用三個 ID 的選項:		
訊息 ID 加上相關性 ID 加上群組 ID	需要 MQIT_GROUP_ID	不支援

若要判定此屬性的值，請搭配使用 MQIA_INDEX_TYPE 選取元與 MQINQ 呼叫。

此屬性僅在 z/OS 上受支援。

InhibitGet (MQLONG)
這會控制是否容許此佇列的取得作業。

本端	模型	別名	遠端	叢集
X	X	X		

如果佇列是別名佇列，則在取得作業時，必須同時容許別名及基本佇列的取得作業， MQGET 呼叫才會成功。 此值是下列其中一個：

MQQA_GET_INHIBITED
禁止取得作業。

MQGET 呼叫失敗，原因碼為 MQRC_GET_INHIBITED。 這包括指定 MQGMO_BROWSE_FIRST 或 MQGMO_BROWSE_NEXT 的 MQGET 呼叫。

註: 如果在工作單元內運作的 MQGET 呼叫順利完成，則後續將 *InhibitGet* 屬性值變更為 MQQA_GET_INHIBITED 並不會阻止確定工作單元。

MQQA_GET_ALLOWED
容許取得作業。

若要判定此屬性的值，請搭配使用 MQQIA_INHIT_GET 選取元與 MQINQ 呼叫。 若要變更此屬性的值，請使用 MQSET 呼叫。

InhibitPut (MQLONG)
這會控制是否容許此佇列的放置作業。

本端	模型	別名	遠端	叢集
X	X	X	X	X

如果佇列名稱解析路徑中有多個定義，則在執行放置作業時，路徑中的 每一個 定義 (包括任何佇列管理程式別名定義) 都必須容許放置作業， MQPUT 或 MQPUT1 呼叫才能成功。 此值是下列其中一個：

MQQA_PUT_INHIBITED
禁止放置作業。

MQPUT 及 MQPUT1 呼叫失敗，原因碼為 MQRC_PUT_INHIBITED。

註: 如果在工作單元內運作的 MQPUT 呼叫順利完成，則後續將 *InhibitPut* 屬性值變更為 MQQA_PUT_INHIBITED 並不會阻止確定工作單元。

容許 MQQA_PUT_ALLOWED
容許放置作業。

若要判定此屬性的值，請搭配使用 MQQIA_INHIT_put 選取器與 MQINQ 呼叫。 若要變更此屬性的值，請使用 MQSET 呼叫。

InitiationQName (MQCHAR48)
這是定義在本端佇列管理程式上的佇列名稱; 佇列類型必須是 MQQT_LOCAL。

本端	模型	別名	遠端	叢集
X				

當由於訊息到達此屬性所屬的佇列而需要啟動應用程式時，佇列管理程式會將觸發訊息傳送至起始佇列。 在收到觸發訊息之後，啟動適當應用程式的觸發監視器應用程式必須監視起始佇列。

若要判定此屬性的值，請搭配使用 MQCA_INITIATION_Q_NAME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_Q_NAME_LENGTH 提供。

MaxMsg 長度 *(MQLONG)*
這是可以放置在佇列上的最長 實體 訊息長度的上限。

本端	模型	別名	遠端	叢集
X	X			

不過，因為 *MaxMsgLength* 佇列屬性可以獨立於 *MaxMsgLength* 佇列管理程式屬性來設定，所以可以放置在佇列上的最長實體訊息的實際長度上限是這兩個值中較小的值。

如果佇列管理程式支援分段，則只有在 MQMD 中指定 MQMF_SEGMENTATION_ALLOWED 旗標時，應用程式才能放置比兩個 *MaxMsgLength* 屬性中較小者更長的 邏輯 訊息。 如果指定該旗標，則邏輯訊息的長度上限為 999 999 999 999 個位元組，但通常由作業系統或應用程式執行所在環境強制的資源限制會導致下限。

嘗試在佇列上放置太長的訊息失敗，原因碼如下:

- 如果訊息對佇列而言太大，則為 MQRC_MSG_TOO_BIG_FOR_Q
- 如果訊息對佇列管理程式而言太大，但對佇列而言不是太大，則為 MQRC_MSG_TOO_BIG_FOR_Q_MGR

MaxMsgLength 屬性的下限為零; 上限為 100 MB (104 857 600 位元組)。

如需相關資訊，請參閱 MQPUT- BufferLength 參數。

若要判定此屬性的值，請搭配使用 MQQIA_MAX_MSG_length 選取器與 MQINQ 呼叫。

MaxQDepth (MQLONG)
這是已定義同時存在於佇列上的實體訊息數上限。

本端	模型	別名	遠端	叢集
X	X			

嘗試將訊息放置在已包含 *MaxQDepth* 訊息的佇列中失敗，原因碼為 MQRC_Q_FULL。

工作單元處理及訊息分段都可能導致佇列上的實際實體訊息數超出 *MaxQDepth*。 不過，這不會影響訊息的可擷取性; 佇列上的 所有 訊息都可以使用 MQGET 呼叫來擷取。

此屬性的值為零或大於零。 上限由環境決定:

- 在 AIX、 HP-UX、 z/OS、Solaris、 Linux 及 Windows 上，此值不能超過 999 999。
- 在 IBM i 上，此值不能超過 640 000。

註: 即使佇列上的訊息少於 *MaxQDepth* ，佇列可用的儲存體空間也可能會耗盡。

若要判定此屬性的值，請搭配使用 MQQIA_MAX_Q_DEPTH 選取器與 MQINQ 呼叫。

MsgDelivery 順序 *(MQLONG)*

本端	模型	別名	遠端	叢集
X	X			

這決定 MQGET 呼叫將訊息傳回應用程式的順序:

MQMDS_FIFO
以 FIFO 順序 (先進先出) 傳回訊息。

不論訊息的優先順序為何， MQGET 呼叫會傳回 第一個 訊息，其滿足呼叫中指定的選取準則。

MQMDS_PRIORITY
以優先順序傳回訊息。

MQGET 呼叫會傳回滿足呼叫上指定的選取準則的 最高優先順序 訊息。 在每一個優先順序層次內，會以 FIFO 順序 (先進先出) 傳回訊息。

- 在 z/OS 上，如果佇列具有 *IndexType* MQIT_GROUP_ID，則 *MsgDeliverySequence* 屬性會指定訊息群組傳回至應用程式的順序。 傳回群組的特定順序取決於每一個群組中第一個訊息的位置或優先順序。 未定義佇列上訊息的實體順序，因為佇列已最佳化，可使用 MQGET 呼叫上的 MQGMO_LOGICAL_ORDER 選項有效地擷取訊息。
- 在 z/OS 上，如果 *IndexType* 是 MQIT_GROUP_ID 且 *MsgDeliverySequence* 是 MQMDS_PRIORITY，則佇列管理程式會使用訊息優先順序零和一，以邏輯順序最佳化訊息的擷取。 因此，群組中第一個訊息的優先順序不得為零或一; 如果是，則會將訊息當作優先順序為 2 來處理。 MQMD 結構中的 *Priority* 欄位不變。

如果在佇列上有訊息時變更相關屬性，則遞送順序如下:

- MQGET 呼叫傳回訊息的順序由訊息到達佇列時對佇列有效的 *MsgDeliverySequence* 及 *DefPriority* 屬性值決定:
  - 當訊息到達時，如果 *MsgDeliverySequence* 是 MQMDS_FIFO，則會將訊息放在佇列上，如同其優先順序是 *DefPriority* 一樣。 這不會影響訊息的訊息描述子中 *Priority* 欄位的值; 該欄位會保留第一次放置訊息時所擁有的值。
  - 當訊息到達時，如果 *MsgDeliverySequence* 為 MQMDS_PRIORITY，則會將訊息放置在佇列上的適當位置，以符合訊息描述子中 *Priority* 欄位提供的優先順序。

  如果在佇列中有訊息時變更 *MsgDeliverySequence* 屬性的值，則不會變更佇列中訊息的順序。

  如果在佇列上有訊息時變更 *DefPriority* 屬性的值，則即使 *MsgDeliverySequence* 屬性設為 MQMDS_FIFO，也不一定會以 FIFO 順序遞送訊息; 會先遞送放在佇列上優先順序較高的訊息。

若要判定此屬性的值，請搭配使用 MQQIA_MSG_DELIVERY_SEQUENCE 選取器與 MQINQ 呼叫。

NonPersistentMessageClass (MQLONG)
非持續訊息的可靠性目標。

本端	模型	別名	遠端	叢集
X	X			

這會指定捨棄放置在此佇列上的非持續訊息的情況:

MQNPM_CLASS_NORMAL
非持續訊息限制為佇列管理程式階段作業的生命期限; 如果佇列管理程式重新啟動，則會捨棄這些訊息。 這僅適用於非共用佇列，並且是預設值。

MQNPM_CLASS_HIGH
佇列管理程式會嘗試在佇列的生命期限內保留非持續訊息。 如果失敗，非持續訊息可能仍會遺失。 此值是針對共用佇列施行。

若要判定此屬性的值，請搭配使用 MQIA_NPM_CLASS 選取器與 MQINQ 呼叫。

OpenInput 計數 *(MQLONG)*
這是目前透過 MQGET 呼叫從佇列中移除訊息有效的控點數。

本端	模型	別名	遠端	叢集
X				

它是 本端 佇列管理程式已知的這類控點總數。 如果佇列是共用佇列，則計數不包括針對本端佇列管理程式所屬佇列共用群組中其他佇列管理程式的佇列執行之輸入的開啟數。

計數包括已開啟解析成此佇列之別名佇列以供輸入的控點。 此計數不包括針對未包含輸入之動作開啟佇列的控點 (例如，僅開啟用於瀏覽的佇列)。

此屬性的值會隨著佇列管理程式的運作而波動。

若要判定此屬性的值，請搭配使用 MQIA_OPEN_INPUT_COUNT 選取器與 MQINQ 呼叫。

OpenOutput 計數 *(MQLONG)*
這是目前透過 MQPUT 呼叫將訊息新增至佇列有效的控點數。

本端	模型	別名	遠端	叢集
X				

它是 本端 佇列管理程式已知的這類控點總數; 它不包括在遠端佇列管理程式中針對此佇列執行之輸出的開啟。 如果佇列是共用佇列，則計數不包括針對本端佇列管理程式所屬佇列共用群組中其他佇列管理程式的佇列執行之輸出的開啟數。

此計數包括開啟解析成此佇列的別名佇列以供輸出的控點。 該計數不包括針對不包括輸出之動作開啟佇列的控點 (例如，僅開啟查詢的佇列)。

此屬性的值會隨著佇列管理程式的運作而波動。

若要判定此屬性的值，請搭配使用 MQQIA_OPEN_output_COUNT 選取器與 MQINQ 呼叫。

ProcessName (MQCHAR48)
這是定義在本端佇列管理程式上的處理程序物件名稱。 處理程序物件識別可處理佇列的程式。

本端	模型	別名	遠端	叢集
X	X			

若要判定此屬性的值，請搭配使用 MQCA_PROCESS_NAME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_PROCESS_NAME_LENGTH 提供。

PropertyControl (MQLONG)
指定如何針對使用 MQGMO_PROPERTIES_AS_Q_DEF 選項的 MQGET 呼叫從佇列擷取的訊息處理訊息內容。

本端	模型	別名	遠端	叢集
X	X	X		

此值是下列其中一個:

MQPROP_ALL
當訊息遞送至應用程式時，訊息的所有內容都會包含在訊息中。 這些內容（訊息描述子或延伸中的除外）會放在訊息資料內的一或多個 MQRFH2 標頭中。 如果提供訊息控點，則行為是在訊息控點中傳回內容。

MQPROP_COMPATIBILITY
如果訊息包含字首為 mcd. 的內容， jms. ， 使用者 或 mqext. ， 所有訊息內容都會遞送至 MQRFH2 標頭中的應用程式。 否則訊息的所有內容（訊息描述子或延伸中的除外）都會被捨棄，而不再能供應用程式存取。 這是預設值；它允許預期 JMS 相關內容位於訊息資料之 MQRFH2 標頭中的應用程式，可以不需修改繼續運作。 如果提供訊息控點，則行為是傳回訊息控點中的內容。

MQPROP_FORCE_MQRFH2
不論應用程式是否指定訊息控點，一律會在 MQRFH2 標頭的訊息資料中傳回內容。 在 MQGET 呼叫上 MQGMO 結構的 MsgHandle 欄位中提供的有效訊息控點會被忽略。 訊息的內容無法透過訊息控點存取。

MQPROP_NONE
在將訊息遞送至應用程式之前，訊息的所有內容 (訊息描述子或延伸中的除外) 都會從訊息中移除。 如果提供訊息控點，則行為是在訊息控點中傳回內容。

此參數適用於「本端」、「別名」及「模型」佇列。 若要判定其值，請搭配使用 MQIA_PROPERTY_CONTROL 選取器與 MQINQ 呼叫。

QDepthHigh 事件 *(MQLONG)*
這會控制是否產生「佇列深度高」事件。

本端	模型	別名	遠端	叢集
X	X			

「佇列深度高」事件指出應用程式已將訊息放置在佇列上，這會導致佇列上的訊息數目大於或等於佇列深度高臨界值 (請參閱 *QDepthHighLimit* 屬性)。

註: 此屬性的值可以動態變更。

此值是下列其中一個:

已停用 MQEVR_DISABLED
事件報告已停用。

已啟用 MQEVR_ENABLED
已啟用事件報告。

如需事件的相關資訊，請參閱 事件監視 。

若要判定此屬性的值，請搭配使用 MQQIA_Q_DEPTH_HIGH_Event 選取器與 MQINQ 呼叫。

此屬性在 z/OS 上受支援，但 MQINQ 呼叫無法用來判斷其值。

QDepthHigh 限制 *(MQLONG)*
這是用來比較佇列深度以產生「佇列深度高」事件的臨界值。

本端	模型	別名	遠端	叢集
X	X			

此事件指出應用程式已在佇列上放置訊息，且這已導致佇列上的訊息數變成大於或等於佇列深度高臨界值。請參閱 QDepthHigh 事件屬性。

該值以佇列深度上限 (*MaxQDepth* 屬性) 百分比表示，且大於或等於 0 且小於或等於 100。 預設值為 80。

若要判定此屬性的值，請搭配使用 MQIA_Q_DEPTH_HIGH_LIMIT 選取器與 MQINQ 呼叫。

此屬性在 z/OS 上受支援，但 MQINQ 呼叫無法用來判斷其值。

QDepthLow 事件 *(MQLONG)*
這會控制是否產生「佇列深度低」事件。

本端	模型	別名	遠端	叢集
X	X			

「佇列深度低」事件指出應用程式已從佇列擷取訊息，且這已導致佇列上的訊息數變成小於或等於佇列深度低臨界值 (請參閱 QDepthLow 限制屬性)。

註: 此屬性的值可以動態變更。

此值是下列其中一個:

已停用 MQEVR_DISABLED
事件報告已停用。

已啟用 MQEVR_ENABLED
已啟用事件報告。

如需事件的相關資訊，請參閱 事件監視 。

若要判定此屬性的值，請搭配使用 MQIA_Q_DEPTH_LOW_EVENT 選取器與 MQINQ 呼叫。

此屬性在 z/OS 上受支援，但 MQINQ 呼叫無法用來判斷其值。

QDepthLow 限制 *(MQLONG)*
這是用來比較佇列深度以產生「佇列深度低值」事件的臨界值。

本端	模型	別名	遠端	叢集
X	X			

此事件指出應用程式已從佇列擷取訊息，且已導致佇列上的訊息數變成小於或等於佇列深度低臨界值。 請參閱 QDepthLow 事件屬性。

該值以佇列深度上限 (*MaxQDepth* 屬性) 百分比表示，且大於或等於 0 且小於或等於 100。 預設值為 20。

若要判定此屬性的值，請搭配使用 MQIA_Q_DEPTH_LOW_LIMIT 選取器與 MQINQ 呼叫。

此屬性在 z/OS 上受支援，但 MQINQ 呼叫無法用來判斷其值。

QDepthMax 事件 *(MQLONG)*

這會控制是否產生「佇列已滿」事件。 「佇列已滿」事件指出因佇列已滿，即佇列深度已達到其最大值，而拒絕放置至佇列。

本端	模型	別名	遠端	叢集
X	X			

註: 此屬性的值可以動態變更。

此值是下列其中一個:

已停用 MQEVR_DISABLED
事件報告已停用。

已啟用 MQEVR_ENABLED
已啟用事件報告。

如需事件的相關資訊，請參閱 事件監視 。

若要判定此屬性的值，請搭配使用 MQQIA_Q_DEPTH_MAX_Event 選取器與 MQINQ 呼叫。

此屬性在 z/OS 上受支援，但 MQINQ 呼叫無法用來判斷其值。

QDesc (MQCHAR64)

使用此欄位作為敘述性註解。

本端	模型	別名	遠端	叢集
X	X	X	X	X

該欄位的內容對佇列管理程式不重要，但佇列管理程式可能需要該欄位只包含可顯示的字元。 它不能包含任何空值字元; 必要的話，會以空白填補右邊。 在 DBCS 安裝中，欄位可以包含 DBCS 字元 (欄位長度上限為 64 個位元組)。

註: 如果此欄位包含不在佇列管理程式字集中的字元 (如 *CodedCharSetId* 佇列管理程式屬性所定義)，則當此欄位傳送至另一個佇列管理程式時，可能會不正確地轉換這些字元。

若要判定此屬性的值，請搭配使用 MQCA_Q_DESC 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_Q_DESC_LENGTH 提供。

完整名稱 *(MQCHAR48)*

這是定義在本端佇列管理程式上的佇列名稱。

本端	模型	別名	遠端	叢集
X		X	X	X

佇列管理程式上定義的所有佇列都共用相同的佇列名稱空間。 因此， MQQT_LOCAL 佇列和 MQQT_ALIAS 佇列不能同名。

若要判定此屬性的值，請搭配使用 MQCA_Q_NAME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_Q_NAME_LENGTH 提供。

QServiceInterval (MQLONG)
這是用於比較以產生「服務間隔高」及「服務間隔正常」事件的服務間隔。

本端	模型	別名	遠端	叢集
X	X			

請參閱 QServiceInterval 事件屬性。

該值以毫秒為單位，且大於或等於零，且小於或等於 999 999 999 999。

若要判定此屬性的值，請搭配使用 MQIA_Q_SERVICE_INTERVAL 選取器與 MQINQ 呼叫。

此屬性在 z/OS 上受支援，但 MQINQ 呼叫無法用來判斷其值。

QServiceInterval 事件 *(MQLONG)*
這會控制是否產生「服務間隔高」或「服務間隔確定」事件。

本端	模型	別名	遠端	叢集
X	X			

- 當檢查指出至少在 *QServiceInterval* 屬性所指示的時間內未從佇列中擷取任何訊息時，會產生「服務間隔高」事件。
- 當檢查指出在 *QServiceInterval* 屬性指出的時間內已從佇列擷取訊息時，會產生「服務間隔正常」事件。

註: 此屬性的值可以動態變更。

此值是下列其中一個:

MQQSIE_高
已啟用佇列服務間隔高事件。

- 「佇列服務間隔高」事件 **已啟用** 及
- **已停用**「佇列服務間隔正常」事件。

MQQSIE_OK
已啟用「佇列服務間隔確定」事件。

- 「佇列服務間隔高」事件為 **已停用** 及
- 「佇列服務間隔確定」事件 **已啟用**。

MQQSIE_NONE
未啟用佇列服務間隔事件。

- 「佇列服務間隔高」事件為 **已停用** 及
- 「佇列服務間隔正常」事件也會 **停用**。

若為共用佇列，則會忽略此屬性的值; 採用 MQQSIE_NONE 值。

如需事件的相關資訊，請參閱 事件監視 。

若要判定此屬性的值，請搭配使用 MQIA_Q_SERVICE_INTERVAL_EVENT 選取器與 MQINQ 呼叫。

在 z/OS 上，您無法使用 MQINQ 呼叫來判定此屬性的值。

QSGDisp (MQLONG)
這會指定佇列的處置方式。

本端	模型	別名	遠端	叢集
X		X	X	

此值是下列其中一個：

MQQSGD_Q_MGR
物件具有佇列管理程式處置。 這表示只有本端佇列管理程式可以識別物件定義; 佇列共用群組中的其他佇列管理程式無法識別此定義。

佇列共用群組中的每一個佇列管理程式都可以具有與現行物件具有相同名稱及類型的物件，但這些是個別物件，且它們之間沒有相關性。 它們的屬性不會限制為彼此相同。

MQQSGD_COPY
物件是存在於共用儲存庫中之主要物件定義的本端副本。 佇列共用群組中的每一個佇列管理程式都可以有自己的物件副本。 一開始，所有副本都具有相同的屬性，但透過使用 MQSC 指令，您可以變更每一個副本，使其屬性與其他副本的屬性不同。 當共用儲存庫中的主要定義變更時，副本的屬性會重新同步化。

MQQSGD_SHARED
物件具有共用處置。 這表示共用儲存庫中存在佇列共用群組中所有佇列管理程式已知的單一物件實例。 當群組中的佇列管理程式存取物件時，它會存取物件的單一共用實例。

若要判定此屬性的值，請搭配使用 MQIA_QSG_DISP 選取器與 MQINQ 呼叫。

此屬性僅在 z/OS 上受支援。

QueueAccounting (MQLONG)

本端	模型	別名	遠端	叢集
X	X	X	X	

這會控制佇列的帳戶資料收集。 若要收集此佇列的帳戶資料，也必須使用 MQCONNX 呼叫 MQCNO 結構中的 QMGR 屬性 ACCTQ 或「選項」欄位來啟用此連線的帳戶資料。

此屬性具有下列其中一個值：

MQMON_Q_MGR
此佇列的帳戶資料是根據 QMGR 屬性 ACCTQ 的設定來收集。 這是預設的設定。

MQMON_OFF
不收集此佇列的帳戶資料。

MQMON_ON
收集此佇列的帳戶資料。

若要判定此屬性的值，請搭配使用 MQIA_ACCOUNTING_Q 選取器與 MQINQ 呼叫。

QueueMonitoring (MQLONG)
控制收集佇列的線上監視資料。

本端	模型	別名	遠端	叢集
X	X			

此值是下列其中一個：

MQMON_Q_MGR
根據 *QueueMonitoring* 佇列管理程式屬性的設定來收集監視資料。 這是預設值。

MQMON_OFF
已關閉此佇列的連線監視資料收集。

MQMON_LOW
如果 *QueueMonitoring* 佇列管理程式屬性的值不是 MQMON_NONE，則會開啟連線監視資料收集，且此佇列的資料收集速率較低。

MQ mon_MEDIT
如果 *QueueMonitoring* 佇列管理程式屬性值不是 MQMON_NONE，則會開啟連線監視資料收集，且此佇列的資料收集速率中等。

MQMON_HIGH
如果 *QueueMonitoring* 佇列管理程式屬性的值不是 MQMON_NONE，則會開啟連線監視資料收集，且此佇列的資料收集速率會很高。

若要判定此屬性的值，請搭配使用 MQIA_MONITORING_Q 選取器與 MQINQ 呼叫。

QueueStatistics (MQCHAR12)

本端	模型	別名	遠端	叢集
X	X	X	X	

這會控制佇列的統計資料收集。

此屬性具有下列其中一個值:

MQMON_Q_MGR
根據 QMGR 屬性 STATQ 的設定收集此佇列的結算資料。 這是預設的設定。

MQMON_OFF
關閉此佇列的統計資料收集。

MQMON_ON
開啟此佇列的統計資料收集。

QType (MQLONG)

本端	模型	別名	遠端	叢集
X		X	X	X

這是佇列的類型; 它具有下列其中一個值:

MQQT_ALIAS
別名佇列定義。

MQQT_CLUSTER
叢集佇列。

本端 MQQT_LOCAL
本端佇列。

MQQT_REMOTE
遠端佇列的本端定義。

若要判定此屬性的值，請搭配使用 MQIA_Q_TYPE 選取器與 MQINQ 呼叫。

RemoteQMgr 名稱 *(MQCHAR48)*

本端	模型	別名	遠端	叢集
			X	

這是在其中定義佇列 *RemoteQName* 的遠端佇列管理程式名稱。 如果 *RemoteQName* 佇列具有 *QSGDisp* 值 MQQSGD_COPY 或 MQQSGD_SHARED，則 *RemoteQMgrName* 可以是擁有 *RemoteQName* 的佇列共用群組名稱。

如果應用程式開啟遠端佇列的本端定義，則 *RemoteQMgrName* 不得為空白，且不得是本端佇列管理程式的名稱。 如果 *XmitQName* 為空白，則會使用與 *RemoteQMgrName* 同名的本端佇列作為傳輸佇列。 如果沒有名稱為 *RemoteQMgrName* 的佇列，則會使用 *DefXmitQName* queue-manager 屬性所識別的佇列。

如果此定義用於佇列管理程式別名，則 *RemoteQMgrName* 是正在建立別名之佇列管理程式的名稱。 它可以是本端佇列管理程式的名稱。 否則，如果開啟時 *XmitQName* 為空白，則必須存在名稱與 *RemoteQMgrName* 相同的本端佇列; 此佇列將用作傳輸佇列。

如果此定義用於回覆別名，則此名稱是要作為 *ReplyToQMgr* 的佇列管理程式名稱。

註: 當建立或修改佇列定義時，不會對指定給這個屬性的值執行任何驗證。

若要判定此屬性的值，請搭配使用 MQCA_REMOTE_Q_MGR_NAME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_Q_MGR_NAME_LENGTH 提供。

RemoteQName (MQCHAR48)

本端	模型	別名	遠端	叢集
			X	

這是在遠端佇列管理程式 *RemoteQMgrName* 上已知的佇列名稱。

如果應用程式開啟遠端佇列的本端定義，當開啟時 *RemoteQName* 不得空白。

如果此定義用於佇列管理程式別名定義，則開啟時 *RemoteQName* 必須為空白。

如果定義用於回覆別名，則此名稱是要作為 *ReplyToQ* 的佇列名稱。

註: 當建立或修改佇列定義時，不會對指定給這個屬性的值執行任何驗證。

若要判定此屬性的值，請搭配使用 MQCA_REMOTE_Q_NAME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_Q_NAME_LENGTH 提供。

RetentionInterval (MQLONG)

這是保留佇列的時段。 過了這個時間之後，就可以刪除佇列。

本端	模型	別名	遠端	叢集
X	X			

時間以小時為測量單位，從建立佇列的日期和時間開始計算。 佇列的建立日期和時間記錄在 *CreationDate* 和 *CreationTime* 屬性中。

提供此資訊是為了讓內部管理應用程式或操作員識別並刪除不再需要的佇列。

註: 佇列管理程式絕不會根據這個屬性採取任何動作來刪除佇列，或防止刪除保留間隔尚未過期的佇列; 使用者必須負責採取任何必要的動作。

使用真實的保留間隔來防止永久動態佇列累積 (請參閱 DefinitionType 屬性)。 不過，這個屬性也可以與預先定義的佇列搭配使用。

若要判定此屬性的值，請搭配使用 MQQIA_RETENTION_INTERVAL 選取器與 MQINQ 呼叫。

範圍 *(MQLONG)*

這會控制這個佇列的項目是否也存在於 Cell 目錄中。

本端	模型	別名	遠端	叢集
X		X	X	

Cell 目錄由可安裝的名稱服務提供。 此值是下列其中一個:

MQSCO_Q_MGR
佇列定義具有佇列管理程式範圍: 佇列的定義不會延伸到擁有它的佇列管理程式之外。 若要開啟佇列以從其他佇列管理程式輸出，必須指定擁有端佇列管理程式的名稱，或其他佇列管理程式必須具有佇列的本端定義。

MQSCO_CELL
佇列定義有 Cell 範圍: 佇列定義也會放在 Cell 中所有佇列管理程式所能使用的 Cell 目錄中。 您可以指定佇列的名稱來開啟佇列，以便從 Cell 中的任何佇列管理程式輸出; 不需要指定擁有佇列的佇列管理程式名稱。 不過，如果 Cell 中的任何佇列管理程式也有該名稱的佇列本端定義，則該佇列定義無法使用，因為本端定義優先。

Cell 目錄由可安裝的名稱服務提供。

模型和動態佇列不能有 Cell 範圍。

只有在已配置支援 Cell 目錄的名稱服務時，這個值才有效。

若要判定此屬性的值，請搭配使用 MQIA_SCOPE 選取器與 MQINQ 呼叫。

此屬性的支援遵循下列限制:

- 在 IBM i 上，支援此屬性，但只有 MQSCO_Q_MGR 有效。
- 在 z/OS 上，不支援此屬性。

可共用性 *(MQLONG)*
這指出佇列是否可以同時開啟多次以供輸入。

本端	模型	別名	遠端	叢集
X	X			

此值是下列其中一個:

MQQA_SHAREABLE
佇列可共用。

容許多個具有 MQOO_INPUT_SHARED 選項的開啟。

MQQA_NOT_SHAREABLE
佇列不可共用。

具有 MQOO_INPUT_SHARED 選項的 MQOPEN 呼叫會被視為 MQOO_INPUT_EXCLUSIVE。

若要判定此屬性的值，請搭配使用 MQQIA_SHAREATability 選取器與 MQINQ 呼叫。

StorageClass (MQCHAR8)
這是使用者定義的名稱，定義用來保留佇列的實體儲存體。 實際上，只有在訊息需要頁出其記憶體緩衝區時，才會將訊息寫入磁碟。

本端	模型	別名	遠端	叢集
X	X			

如果要判斷這個屬性的值，請搭配使用 MQCA_STORAGE_CLASS 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_STORAGE_CLASS_LENGTH 提供。

此屬性僅在 z/OS 上受支援。

TriggerControl (MQLONG)
這會控制是否將觸發訊息寫入起始佇列，以啟動應用程式來處理佇列。

本端	模型	別名	遠端	叢集
X	X			

這是下列其中一項:

MQTC_OFF
不寫入此佇列的觸發訊息。 在此情況下， *TriggerType* 的值不相關。

MQTC_ON
當發生適當的觸發事件時，會寫入此佇列的觸發訊息。

若要判定此屬性的值，請搭配使用 MQQIA_TRIGGER_Control 選取器與 MQINQ 呼叫。 若要變更此屬性的值，請使用 MQSET 呼叫。

TriggerData (MQCHAR64)
當到達此佇列的訊息導致觸發訊息寫入起始佇列時，這是佇列管理程式插入觸發訊息中的可用格式資料。

本端	模型	別名	遠端	叢集
X	X			

此資料的內容對佇列管理程式不重要。 它對處理起始佇列的觸發監視器應用程式或觸發監視器啟動的應用程式有意義。

字串不得包含任何空值。 必要的話，它會以空白填補在右側。

若要判定此屬性的值，請搭配使用 MQCA_TRIGGER_DATA 選取器與 MQINQ 呼叫。 若要變更此屬性的值，請使用 MQSET 呼叫。 此屬性的長度由 MQ_TRIGGER_DATA_LENGTH 提供。

TriggerDepth (MQLONG)

本端	模型	別名	遠端	叢集
X	X			

這是在寫入觸發訊息之前，必須在佇列上且優先順序為 *TriggerMsgPriority* 或更高的訊息數。 當 *TriggerType* 設為 MQTT_DEPTH 時適用。 *TriggerDepth* 的值為 1 或更大。 否則不會使用此屬性。

若要判定此屬性的值，請搭配使用 MQQIA_TRIGGER_DEPTH 選取器與 MQINQ 呼叫。 若要變更此屬性的值，請使用 MQSET 呼叫。

TriggerMsg 優先順序 *(MQLONG)*
這是訊息優先順序，低於此優先順序的訊息不會參與觸發訊息的產生 (亦即，在決定是否產生觸發訊息時，佇列管理程式會忽略這些訊息)。

本端	模型	別名	遠端	叢集
X	X			

TriggerMsgPriority 可以在範圍零 (最低) 到 *MaxPriority* (最高; 請參閱 MaxPriority 屬性); 值零會導致所有訊息參與產生觸發訊息。

若要判定此屬性的值，請搭配使用 MQQIA_TRIGGER_MSG_優先順序選取器與 MQINQ 呼叫。 若要變更此屬性的值，請使用 MQSET 呼叫。

TriggerType (MQLONG)
這會控制因訊息到達此佇列而寫入觸發訊息的條件。

本端	模型	別名	遠端	叢集
X	X			

它具有下列其中一個值:

MQTT_NONE
不會因為此佇列上的訊息而寫入任何觸發訊息。 這與將 *TriggerControl* 設為 MQTC_OFF 的效果相同。

MQTT_FIRST
每當佇列上優先順序 *TriggerMsgPriority* 或更高的訊息數從 0 變更為 1 時，即會寫入觸發訊息。

MQTT_EVERY
每當優先順序為 *TriggerMsgPriority* 或更高的訊息到達佇列時，即會寫入觸發訊息。

MQ tt_DEPTH
每當佇列上優先順序 *TriggerMsgPriority* 或更高的訊息數目等於或超過 *TriggerDepth* 時，即會寫入觸發訊息。 在寫入觸發訊息之後， *TriggerControl* 會設為 MQTC_OFF ，以防止進一步觸發，直到再次明確開啟它為止。

若要判定此屬性的值，請搭配使用 MQQIA_TRIGGER_TYPE 選取器與 MQINQ 呼叫。 若要變更此屬性的值，請使用 MQSET 呼叫。

用法 *(MQLONG)*
這指出佇列的用途。

本端	模型	別名	遠端	叢集
X	X			

此值是下列其中一個:

MQUS_NORMAL
這是應用程式在放置及取得訊息時使用的佇列; 該佇列不是傳輸佇列。

MQ 使用者 _ 傳輸
這是用來保留以遠端佇列管理程式為目的地之訊息的佇列。 當應用程式將訊息傳送至遠端佇列時，本端佇列管理程式會以特殊格式將訊息暫時儲存在適當的傳輸佇列中。 然後，訊息通道代理程式會從傳輸佇列讀取訊息，並將訊息傳輸至遠端佇列管理程式。 如需傳輸佇列的相關資訊，請參閱 定義傳輸佇列。

只有特許應用程式可以開啟 MQOO_OUTPUT 的傳輸佇列，以直接將訊息放置在其上。 通常，只有公用程式應用程式才會執行此動作。 請確定訊息資料格式正確 (請參閱 第 539 頁的『MQXQH-傳輸佇列標頭』)，或在傳輸程序期間可能發生錯誤。 除非指定其中一個 MQPMO_ * _CONTEXT 環境定義選項，否則不會傳遞或設定環境定義。

若要判定此屬性的值，請搭配使用 MQIA_USAGE 選取器與 MQINQ 呼叫。

XmitQName (MQCHAR48)
這是傳輸佇列名稱。 當開啟時，如果這個屬性不是空白，不論是針對遠端佇列或佇列管理程式別名定義，它都會指定要用來轉遞訊息的本端傳輸佇列名稱。

本端	模型	別名	遠端	叢集
			X	

如果 *XmitQName* 為空白，則會使用名稱與 *RemoteQMgrName* 相同的本端佇列作為傳輸佇列。 如果沒有名稱為 *RemoteQMgrName* 的佇列，則會使用 *DefXmitQName* queue-manager 屬性所識別的佇列。

如果使用定義作為佇列管理程式別名，且 *RemoteQMgrName* 是本端佇列管理程式的名稱，則會忽略此屬性。 如果使用定義作為回覆目的地佇列別名定義，則也會忽略它。

若要判定此屬性的值，請搭配使用 MQCA_XMIT_Q_NAME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_Q_NAME_LENGTH 提供。

名稱清單的屬性

下表彙總名稱清單特定的屬性。 屬性按字母順序說明。

所有 WebSphere MQ 系統，以及連接至這些系統的 WebSphere MQ MQI 用戶端，都支援名稱清單。

註: 本節顯示的屬性名稱是與 MQINQ 和 MQSET 呼叫搭配使用的敘述性名稱; 這些名稱與 PCF 指令的名稱相同。 當使用 MQSC 指令來定義、變更或顯示屬性時，會使用替代簡稱; 如需相關資訊，請參閱 Script (MQSC) 指令 。

表 576: 名稱清單的屬性	
屬性	**說明**
AlterationDate	前次變更定義的日期
AlterationTime	前次變更定義的時間
NameCount	名單中的名稱數目
NamelistDesc	名單說明
NamelistName	名稱清單名稱
名稱	*NameCount* 名稱清單
NamelistType	名稱清單類型
QSGDisp	佇列共用群組處置

AlterationDate (MQCHAR12)

這是前次變更定義的日期。 日期的格式為 YYYY-MM-DD，並以兩個尾端空白填補，使其長度為 12 個位元組。

若要判定此屬性的值，請搭配使用 MQCA_ALTERATION_DATE 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_DATE_LENGTH 提供。

AlterationTime (MQCHAR8)

這是前次變更定義的時間。 時間的格式為 HH.MM.SS。

若要判定此屬性的值，請搭配使用 MQCA_ALTERATION_TIME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_TIME_LENGTH 提供。

NameCount (MQLONG)

這是名稱清單中的名稱數。 它大於或等於零。 下列是已定義的值:

MQNC_MAX_NAMELIST_NAME_COUNT
名單中的名稱數目上限。

若要判定此屬性的值，請搭配使用 MQIA_NAME_COUNT 選取器與 MQINQ 呼叫。

NamelistDesc (MQCHAR64)

使用此欄位作為敘述性註解; 其值是由定義程序所建立。 該欄位的內容對佇列管理程式不重要，但佇列管理程式可能需要該欄位只包含可顯示的字元。 它不能包含任何空值字元; 必要的話，會以空白填補右邊。 在 DBCS 安裝中，這個欄位可以包含 DBCS 字元 (欄位長度上限為 64 個位元組)。

註: 如果此欄位包含不在佇列管理程式字集中的字元 (如 *CodedCharSetId* 佇列管理程式屬性所定義) ，則當此欄位傳送至另一個佇列管理程式時，可能會不正確地轉換這些字元。

若要判定此屬性的值，請搭配使用 MQCA_NAMELIST_DESC 選取器與 MQINQ 呼叫。

此屬性的長度由 MQ_NAMELIST_DESC_LENGTH 指定。

NamelistName (MQCHAR48)

這是在本端佇列管理程式上定義的名單名稱。 如需名單名稱的相關資訊，請參閱 其他物件名稱 一節。

每一個名稱清單的名稱都不同於屬於佇列管理程式的其他名稱清單名稱，但可能會複製不同類型 (例如，佇列) 的其他佇列管理程式物件名稱。

若要判定此屬性的值，請搭配使用 MQCA_NAMELIST_NAME 選取器與 MQINQ 呼叫。

此屬性的長度由 MQ_NAMELIST_NAME_LENGTH 提供。

NamelistType (MQLONG)

這會指定名單中名稱的本質，並指出如何使用名單。 它是下列其中一個值:

MQNT_NONE
沒有已指派類型的名單。

MQNT_Q
包含佇列名稱的名單。

MQNT_CLUSTER
包含叢集名稱的名單。

MQNT_AUTH_INFO
包含鑑別資訊物件名稱的名單。

若要判定此屬性的值，請搭配使用 MQIA_NAMELIST_TYPE 選取元與 MQINQ 呼叫。

此屬性僅在 z/OS 上受支援。

名稱 *(MQCHAR48xNameCount)*

這是 *NameCount* 名稱的清單，其中每一個名稱都是定義給本端佇列管理程式的物件名稱。 如需物件名稱的相關資訊，請參閱 命名 IBM WebSphere MQ 物件的規則 。

若要判定此屬性的值，請搭配使用 MQCA_NAMES 選取器與 MQINQ 呼叫。

清單中每一個名稱的長度由 MQ_OBJECT_NAME_LENGTH 提供。

QSGDisp (MQLONG)

這會指定名稱清單的處置。 此值是下列其中一個:

MQQSGD_Q_MGR
物件具有佇列管理程式處置: 只有本端佇列管理程式才知道物件定義; 佇列共用群組中的其他佇列管理程式並不知道此定義。

佇列共用群組中的每一個佇列管理程式都可以具有與現行物件具有相同名稱及類型的物件，但這些是個別物件，且它們之間沒有相關性。 它們的屬性不會限制為彼此相同。

MQQSGD_COPY
物件是存在於共用儲存庫中之主要物件定義的本端副本。 佇列共用群組中的每一個佇列管理程式都可以有自己的物件副本。 一開始，所有副本都有相同的屬性，但您可以使用 MQSC 指令來變更每一個副本，使其屬性與其他副本的屬性不同。 當共用儲存庫中的主要定義變更時，副本的屬性會重新同步化。

若要判定此屬性的值，請搭配使用 MQIA_QSG_DISP 選取器與 MQINQ 呼叫。

此屬性僅在 z/OS 上受支援。

程序定義的屬性

下表彙總處理程序定義特定的屬性。 屬性按字母順序說明。

註: 本節中的屬性名稱是與 MQINQ 及 MQSET 呼叫搭配使用的敘述性名稱; 這些名稱與 PCF 指令的名稱相同。 當使用 MQSC 指令來定義、變更或顯示屬性時，會使用替代簡稱; 如需相關資訊，請參閱 Script (MQSC) 指令 。

表 *577:* 程序定義的屬性

屬性	說明
AlterationDate	前次變更定義的日期
AlterationTime	前次變更定義的時間
ApplId	應用程式 ID
ApplType	應用程式類型
EnvData	環境資料
ProcessDesc	程序說明
ProcessName	處理程序名稱
QSGDisp	佇列共用群組處置

表 577: 程序定義的屬性 (繼續)	
屬性	**說明**
UserData	使用者資料

AlterationDate (MQCHAR12)

這是前次變更定義的日期。 日期的格式為 YYYY-MM-DD，並以兩個尾端空白填補，使其長度為 12 個位元組。

若要判定此屬性的值，請搭配使用 MQCA_ALTERATION_DATE 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_DATE_LENGTH 提供。

AlterationTime (MQCHAR8)

這是前次變更定義的時間。 時間的格式為 HH.MM.SS。

若要判定此屬性的值，請搭配使用 MQCA_ALTERATION_TIME 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_TIME_LENGTH 提供。

ApplId (MQCHAR256)

這是識別要啟動之應用程式的字串。 此資訊供處理起始佇列上的訊息的觸發監視器應用程式使用; 此資訊會作為觸發訊息的一部分傳送至起始佇列。

ApplId 的意義由觸發監視器應用程式決定。 WebSphere MQ 所提供的觸發監視器需要 *ApplId* 是可執行程式的名稱。 下列注意事項適用於指出的環境:

- 在 z/OS 上， *ApplId* 必須:
  - CICS 交易 ID ，適用於使用 CICS 觸發監視器交易 CKTI 啟動的應用程式
  - IMS 交易 ID ，適用於使用 IMS 觸發監視器 CSQQTRMN 啟動的應用程式
- 在 Windows 系統上，程式名稱可以字首為磁碟機及目錄路徑。
- 在 UNIX 系統上，程式名稱的字首可以是目錄路徑。

字串不能包含任何空值。 必要的話，它會以空白填補在右側。

若要判定此屬性的值，請搭配使用 MQCA_APPL_ID 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_PROCESS_APPL_ID_LENGTH 提供。

ApplType (MQLONG)

這會識別為了回應觸發訊息的接收而啟動之程式的本質。 此資訊供處理起始佇列上的訊息的觸發監視器應用程式使用; 此資訊會作為觸發訊息的一部分傳送至起始佇列。

ApplType 可以具有任何值，但對於標準類型，建議使用下列值; 將使用者定義的應用程式類型限制為 MQAT_USER_FIRST 到 MQAT_USER_LAST 範圍內的值:

MQAT_AIX
AIX 應用程式 (與 MQAT_UNIX 的值相同)。

MQ 批次
批次應用程式 (batch application)

MQ 屬性分配管理系統
分配管理系統應用程式

MQAT_CICS
CICS 交易。

MQAT_CICS_BRIDGE
CICS 橋接器應用程式。

MQAT_CICS_VSE
CICS/VSE 交易。

MQAT_DOS
PC DOS 上的 WebSphere MQ MQI 用戶端應用程式。

MQAT_IMS
IMS 應用程式。

MQAT_IMS_BRIDGE
IMS 橋接器應用程式。

MQAT_JAVA
Java 應用程式。

MQAT_MVS
MVS 或 TSO 應用程式 (與 MQAT_ZOS 的值相同)。

MQAT_NOTES_AGENT
Lotus Notes 代理程式應用程式。

MQAT_NSK
HP Integrity NonStop Server 應用程式。

MQAT_OS390
OS/390 應用程式 (與 MQAT_ZOS 相同值)。

MQAT_OS400
IBM i 應用程式。

MQ 屬性 _ rrS_BATCH
RRS 批次應用程式。

MQAT_UNIX
UNIX 應用程式。

MQAT_UNKNOWN
應用程式類型不明。

MQAT_USER
使用者應用程式。

MQAT_VOS
Stratus VOS 應用程式。

MQ 視窗
16 位元 Windows 應用程式。

MQAT_WINDOWS_NT
32 位元 Windows 應用程式。

MQAT_WLM
z/OS 工作量管理程式應用程式。

MQAT_XCF
XCF。

MQAT_ZOS
z/OS 應用程式。

MQAT_USER_FIRST
使用者定義應用程式類型的最低值。

MQAT_USER_LAST
使用者定義應用程式類型的最高值。

若要判定此屬性的值，請搭配使用 MQIA_APPL_TYPE 選取元與 MQINQ 呼叫。

EnvData (MQCHAR128)

這是一個字串，包含與要啟動之應用程式相關的環境相關資訊。 此資訊供處理起始佇列上的訊息的觸發監視器應用程式使用; 此資訊會作為觸發訊息的一部分傳送至起始佇列。

EnvData 的意義由觸發監視器應用程式決定。 WebSphere MQ 所提供的觸發監視器會將 *EnvData* 附加至傳遞至已啟動應用程式的參數清單。 參數清單由 MQTMC2 結構組成，後面接著一個空白，後面接著 *EnvData* ，並移除尾端空白。 下列注意事項適用於指出的環境:

- 在 z/OS 上:
  - WebSphere MQ 所提供的觸發監視器應用程式不會使用 *EnvData* 。
  - 如果 ApplType 是 MQAT_WLM ，您可以在 EnvData 中提供工作資訊標頭 (MQWIH) 中 ServiceName 及 ServiceStep 欄位的預設值。
- 在 UNIX 系統上， *EnvData* 可以設為 & 字元，以在背景中執行已啟動的應用程式。

字串不能包含任何空值。 必要的話，它會以空白填補在右側。

若要判定此屬性的值，請搭配使用 MQCA_ENV_DATA 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_PROCESS_ENV_DATA_LENGTH 提供。

ProcessDesc (MQCHAR64)

使用此欄位作為敘述性註解。 該欄位的內容對佇列管理程式不重要，但佇列管理程式可能需要該欄位只包含可顯示的字元。 它不能包含任何空值字元; 必要的話，會以空白填補右邊。 在 DBCS 安裝中，欄位可以包含 DBCS 字元 (欄位長度上限為 64 個位元組)。

註: 如果此欄位包含不在佇列管理程式字集中的字元 (如 *CodedCharSetId* 佇列管理程式屬性所定義) ，則當此欄位傳送至另一個佇列管理程式時，可能會不正確地轉換這些字元。

若要判定此屬性的值，請搭配使用 MQCA_PROCESS_DESC 選取器與 MQINQ 呼叫。

此屬性的長度由 MQ_PROCESS_DESC_LENGTH 提供。

ProcessName (MQCHAR48)

這是定義在本端佇列管理程式上的程序定義名稱。

每一個程序定義的名稱都不同於屬於佇列管理程式之其他程序定義的名稱。 但程序定義的名稱可能與其他不同類型 (例如，佇列) 的佇列管理程式物件名稱相同。

若要判定此屬性的值，請搭配使用 MQCA_PROCESS_NAME 選取器與 MQINQ 呼叫。

此屬性的長度由 MQ_PROCESS_NAME_LENGTH 提供。

QSGDisp (MQLONG)

這會指定程序定義的處置方式。 此值是下列其中一個:

MQQSGD_Q_MGR
物件具有佇列管理程式處置: 只有本端佇列管理程式才知道物件定義; 佇列共用群組中的其他佇列管理程式並不知道此定義。

佇列共用群組中的每一個佇列管理程式都可以具有與現行物件具有相同名稱及類型的物件，但這些是個別物件，且它們之間沒有相關性。 它們的屬性不會限制為彼此相同。

MQQSGD_COPY
物件是存在於共用儲存庫中之主要物件定義的本端副本。 佇列共用群組中的每一個佇列管理程式都可以有自己的物件副本。 一開始，所有副本都有相同的屬性，但您可以使用 MQSC 指令來變更每一個副本，使其屬性與其他副本的屬性不同。 當共用儲存庫中的主要定義變更時，副本的屬性會重新同步化。

若要判定此屬性的值，請搭配使用 MQIA_QSG_DISP 選取器與 MQINQ 呼叫。

此屬性僅在 z/OS 上受支援。

UserData (MQCHAR128)

UserData 是一個字串，包含與要啟動之應用程式相關的使用者資訊。 此資訊供處理起始佇列上訊息的觸發監視器應用程式使用，或由觸發監視器啟動的應用程式使用。 資訊會作為觸發訊息的一部分傳送至起始佇列。

UserData 的意義由觸發監視器應用程式決定。 WebSphere MQ 所提供的觸發監視器會將 *UserData* 傳遞至已啟動的應用程式，作為參數清單的一部分。 參數清單包含 MQTMC2 結構 (包含 *UserData*)，後面接著一個空白，後面接著 *EnvData*，並移除尾端空白。

字串不能包含任何空值。 必要的話，它會以空白填補在右側。 對於 Microsoft Windows，如果要將程序定義傳遞至 **runmqtrm**，則字串不得包含雙引號。

若要判定此屬性的值，請搭配使用 MQCA_USER_DATA 選取器與 MQINQ 呼叫。 此屬性的長度由 MQ_PROCESS_USER_DATA_LENGTH 提供。

回覆碼

對於每一個 WebSphere MQ 訊息佇列介面 (MQI) 及 WebSphere MQ 管理介面 (MQAI) 呼叫，佇列管理程式或結束常式會傳回 **完成** 碼及 **原因** 碼，以指出呼叫成功或失敗。

除非特別註明，否則應用程式不得依賴以特定順序檢查的錯誤。 如果呼叫可能產生多個完成碼或原因碼，則所報告的特定錯誤取決於實作。

在 WebSphere MQ API 呼叫之後，應用程式檢查是否順利完成必須一律檢查完成碼。 請勿根據原因碼的值來假設完成碼值。

完成碼

完成碼參數 (*CompCode*) 可讓呼叫者快速查看呼叫是否已順利完成、局部完成或失敗。下列是完成碼的清單，其詳細資料比通話說明中提供的詳細資料還多:

MQCC_OK
呼叫已完全完成; 已設定所有輸出參數。 在此情況下， *Reason* 參數一律具有值 MQRC_NONE。

MQCC_WARNING
呼叫已局部完成。 除了 *CompCode* 和 *Reason* 輸出參數之外，可能還設定了部分輸出參數。 *Reason* 參數提供部分完成的相關資訊。

MQCC_FAILED
呼叫處理未完成。 除非特別註明，否則佇列管理程式的狀態不會變更。 已設定 *CompCode* 和 *Reason* 輸出參數; 除非另有說明，否則其他參數保持不變。

原因可能是應用程式中的錯誤，或可能是程式外部的某個狀況所導致，例如使用者的權限可能已被撤銷。 *Reason* 參數提供錯誤的其他相關資訊。

原因碼

原因碼參數 (*Reason*) 限定完成碼參數 (*CompCode*)。

如果沒有特殊原因要報告，則會傳回 MQRC_NONE。 成功呼叫會傳回 MQCC_OK 和 MQRC_NONE。

如果完成碼為 MQCC_WARNING 或 MQCC_FAILED，則佇列管理程式一律會報告合格原因; 詳細資料會在每一個呼叫說明下提供。

當使用者結束常式設定完成碼及原因時，它們必須遵守這些規則。 此外，使用者結束程式所定義的任何特殊原因值都必須小於零，以確保它們不會與佇列管理程式所定義的值衝突。 在適當的情況下，結束程式可以設定佇列管理程式已定義的原因。

原因碼也會出現在:

- MQDLH 結構的 *Reason* 欄位
- MQMD 結構的 *Feedback* 欄位

如需原因碼的完整說明，請參閱 原因碼 。

MQI 選項的驗證規則

本節列出從 MQOPEN、MQPUT、 MQPUT1、MQGET、MQCLOSE 或 MQSUB 呼叫產生 MQRC_OPTIONS_ERROR 原因碼的狀況。

MQOPEN 呼叫

對於 MQOPEN 呼叫的選項:

- 至少必須指定下列其中一項 :
  - MQ 瀏覽
  - MQOO_INPUT_EXCLUSIVE[1]
  - MQOO_INPUT_SHARED[1]
  - MQOO_INPUT_AS_Q_DEF[1]
  - MQOO_INQUIRE
  - MQOO_OUTPUT
  - MQOO_SET
  - MQOO_BIND_ON_OPEN[2]
  - MQOO_BIND_NOT_FIXED[2]
  - MQOO_BIND_ON_GROUP[2]
  - MQOO_BIND_AS_Q_DEF[2]
- 只容許下列其中 一項 :
  - MQOO_READ_AHEAD
  - MQOO_NO_READ_AHEAD
  - MQOO_READ_AHEAD_AS_Q_DEF

1. 只容許下列其中 一項 :
   - MQOO_INPUT_EXCLUSIVE
   - MQOO_INPUT_SHARED
   - MQOO_INPUT_AS_Q_DEF
2. 只容許下列其中 一項 :
   - MQOO_BIND_ON_OPEN
   - MQOO_BIND_NOT_FIXED
   - MQ OO_BIND_ON_GROUP
   - MQOO_BIND_AS_Q_DEF

   註: 上述選項互斥。 不過，因為 MQOO_BIND_AS_Q_DEF 的值是零，所以使用其他兩個連結選項之一來指定它不會導致原因碼 MQRC_OPTIONS_ERROR。 提供 MQOO_BIND_AS_Q_DEF 以輔助程式文件。

- 如果指定 MQOO_SAVE_ALL_CONTEXT ，則也必須指定其中一個 MQOO_INPUT_ * 選項。
- 如果指定其中一個 MQOO_SET_ * _CONTEXT 或 MQOO_PASS_ * _CONTEXT 選項，則也必須指定 MQOO_OUTPUT。
- 如果指定 MQOO_CO_OP ，則也必須指定 MQOO_BROWSE
- 如果指定 MQOO_NO_多重播送，則也必須指定 MQOO_OUTPUT。

MQPUT 呼叫

對於 put-message 選項:

- 不容許 MQPMO_SYNCPOINT 與 MQPMO_NO_SYNCPOINT 的組合。
- 只容許下列其中 一項 :
  - MQPMO_DEFAULT_CONTEXT
  - MQPMO_NO_CONTEXT
  - MQPMO_PASS_ALL_CONTEXT

  - MQPMO_PASS_IDENTITY_CONTEXT
  - MQPMO_SET_ALL_CONTEXT
  - MQPMO_SET_IDENTITY_CONTEXT
- 只容許下列其中 一項 :
  - MQPMO_ASYNC_RESPONSE
  - MQPMO_SYNC_RESPONSE
  - MQPMO_RESPONSE_AS_TOPIC_DEF
  - MQPMO_RESPONSE_AS_Q_DEF
- 不容許 MQPMO_ALTERNATE_USER_AUTHORITY (它僅在 MQPUT1 呼叫上有效)。

MQPUT1 呼叫

對於 put-message 選項，規則與 MQPUT 呼叫的規則相同，但下列各項除外:

- 容許 MQPMO_ALTERNATE_USER_AUTHORITY。
- 不 接受 MQPMO_LOGICAL_ORDER。

MQGET 呼叫

對於 get-message 選項:

- 只容許下列其中 一項 :
  - MQGMO_NO_SYNCPOINT
  - MQ GMO_同步點
  - MQGMO_SYNCPOINT_IF_PERSISTENT
- 只容許下列其中 一項 :
  - MQGMO_BROWSE_FIRST
  - MQGMO_BROWSE_MSG_UNDER_CURSOR
  - MQGMO_BROWSE_NEXT
  - MQGMO_MSG_UNDER_CURSOR
- 不容許 MQGMO_SYNCPOINT 與下列任何一項搭配使用:
  - MQGMO_BROWSE_FIRST
  - MQGMO_BROWSE_MSG_UNDER_CURSOR
  - MQGMO_BROWSE_NEXT
  - MQGMO_LOCK
  - MQGMO_UNLOCK
- 不容許 MQGMO_SYNCPOINT_IF_PERSISTENT 與下列任何一項搭配使用:
  - MQGMO_BROWSE_FIRST
  - MQGMO_BROWSE_MSG_UNDER_CURSOR
  - MQGMO_BROWSE_NEXT
  - MQ GMO_COMPLETE_MSG
  - MQGMO_UNLOCK
- MQGMO_MARK_SKIP_BACKOUT 需要指定 MQGMO_SYNCPOINT。
- 不容許 MQGMO_WAIT 與 MQGMO_SET_信號的組合。
- 如果指定 MQGMO_LOCK ， 也必須指定下列其中一項:
  - MQGMO_BROWSE_FIRST

  - MQGMO_BROWSE_MSG_UNDER_CURSOR
  - MQGMO_BROWSE_NEXT
- 如果指定 MQGMO_UNLOCK，則只容許下列項目:
  - MQGMO_NO_SYNCPOINT
  - MQGMO_NO_WAIT

MQCLOSE 呼叫

對於 MQCLOSE 呼叫的選項:

- 不容許 MQCO_DELETE 和 MQCO_DELETE_PURGE 的組合。
- 只容許下列其中一項:
  - MQCO_KEEP_SUB
  - MQCO_REMOVE_SUB

MQSUB 呼叫

對於 MQSUB 呼叫的選項:

- 至少必須指定下列其中一項:
  - MQSO_ALTER
  - MQ 回復
  - MQSO_CREATE
- 只容許下列其中一項:
  - MQ 可延續
  - MQSO_NON_DURABLE

  註: 上述選項互斥。 不過，因為 MQSO_NON_DURABLE 的值是零，所以使用 MQSO_DURABLE 指定它不會導致原因碼 MQRC_OPTIONS_ERROR。 提供 MQSO_NON_DURABLE 以協助程式說明文件。
- 不容許 MQSO_GROUP_SUB 與 MQSO_MANAGED 的組合。
- MQSO_GROUP_SUB 需要指定 MQSO_SET_CORREL_ID。
- 只容許下列其中一項:
  - MQSO_ANY_USERID
  - MQSO_FIXED_USERID
- MQSO_NEW_PUBLICATIONS_ONLY 只能與 MQSO_CREATE 一起使用。
- 不容許 MQSO_PUBLICATIONS_ON_REQUEST 與 SubLevel 大於 1 的組合。
- 只容許下列其中一項:
  - MQSO_WILDCARD_CHAR
  - MQSO_WILDCARD_TOPIC
- MQSO_NO_MULTICAST 需要指定 MQSO_MANAGED。

排入佇列的發佈/訂閱指令訊息

應用程式可以使用 MQRFH2 指令訊息來控制已排入佇列的發佈/訂閱應用程式。

使用 MQRFH2 進行發佈/訂閱的應用程式可以將下列指令訊息傳送至 SYSTEM.BROKER.CONTROL.QUEUE:

- 第 768 頁的『刪除發佈訊息』
- 第 769 頁的『取消登錄訂閱者訊息』

- 第 772 頁的『發佈訊息』
- 第 774 頁的『登錄訂閱者訊息』
- 第 778 頁的『要求更新訊息』

如果您正在撰寫排入佇列的發佈/訂閱應用程式，則必須瞭解這些訊息、佇列管理程式回應訊息，以及訊息描述子 (MQMD); 請參閱下列資訊:

- 第 779 頁的『佇列管理程式回應訊息』
- 第 784 頁的『佇列管理程式轉遞之發佈的 MQMD 設定』
- 第 785 頁的『佇列管理程式回應訊息中的 MQMD 設定』
- 第 781 頁的『發佈/訂閱原因碼』

指令包含在 MQRFH2 標頭的 **NameValueData** 欄位中的 <psc> 資料夾中。 分配管理系統可以傳送以回應指令訊息的訊息包含在 <pscr> 資料夾中。

每一個指令的說明都會列出可包含在資料夾中的內容。 除非另有指定，否則內容是選用的，且只能出現一次。

內容名稱會顯示為 <Command>。
值必須是字串格式，例如: Publish。
代表內容值的字串常數會顯示在括弧中，例如: (MQPSC_PUBLISH)。
字串常數定義在佇列管理程式隨附的標頭檔 cmqpsc.h 中。

刪除發佈訊息

Delete Publication 指令訊息會從發佈者傳送至佇列管理程式，或從另一個佇列管理程式傳送至佇列管理程式，以告知佇列管理程式刪除指定主題的任何保留發佈資訊。

此訊息會傳送至佇列管理程式的佇列發佈/訂閱介面所監視的佇列。

輸入佇列應該是原始發佈資訊傳送至其中的佇列。

如果您對 **Delete Publication** 指令訊息中指定的部分 (而非全部) 主題具有權限，則只會刪除那些主題。 **Broker Response** 訊息指出未刪除哪些主題。

同樣地，如果 **Publish** 指令包含多個主題，則 **Delete Publication** 指令只會刪除 **Delete Publication** 指令中所指定主題的發佈。

如需將指令訊息傳送至佇列管理程式時所需的訊息描述子 (MQMD) 參數的詳細資料，請參閱 第 784 頁的『佇列管理程式轉遞之發佈的 MQMD 設定』。

內容

<Command> (*MQPSC_COMMAND*)
值為 DeletePub(*MQPSC_DELETE_PUBLICATION*)。

必須指定此內容。

<Topic> (*MQPSC_TOPIC*)
該值是包含要刪除其保留發佈資訊之主題的字串。 字串中可以包含萬用字元，以刪除多個主題的發佈資訊。

必須指定此內容; 可以視需要針對任意數目的主題重複此內容。

<DelOpt> (*MQPSC_DELETE_OPTION*)
刪除選項內容可以採用下列其中一個值:

本端 (*MQPSC_LOCAL*)

不論是否使用 **本端** 選項發佈，指定主題的所有保留發佈資訊都會在本端佇列管理程式 (亦即，傳送此訊息至其中的佇列管理程式) 中刪除。

其他佇列管理程式的發佈不受影響。

無 (*MQPSC_NONE*)

所有選項都採用其預設值。 這與省略 DelOpt 內容具有相同的效果。 如果同時指定其他選項，則會忽略 **無** 。

如果省略此內容，則預設值是在網路中的所有佇列管理程式上刪除所指定主題的所有保留發佈，而不論它們是否使用 **本端** 選項發佈。

範例

以下是 **Delete Publication** 指令訊息的 NameValue **資料** 範例。 此範例應用程式用來在本端佇列管理程式中刪除包含 Team1 與 Team2 之間相符項中最新評分的保留發佈資訊。

```
<psc>
 <Command>DeletePub</Command>
 <Topic>Sport/Soccer/State/LatestScore/Team1 Team2</Topic>
 <DelOpt>Local</DelOpt>
</psc>
```

取消登錄訂閱者訊息

Deregister Subscriber 指令訊息由訂閱者或代表訂閱者的另一個應用程式傳送至佇列管理程式，以指出它不再想要接收符合給定參數的訊息。

此訊息會傳送至 SYSTEM.BROKER.CONTROL.QUEUE，佇列管理程式的控制佇列。 使用者必須具有將訊息放入此佇列的必要權限。

如需將指令訊息傳送至佇列管理程式時所需之訊息描述子 (MQMD) 參數的詳細資料，請參閱 佇列管理程式所轉遞發佈的 MQMD 設定 。

透過指定原始訂閱的對應主題、訂閱點及過濾器值，可以取消登錄個別訂閱。 如果未在原始訂閱中指定任何值 (亦即，它們採用預設值) ，則在取消登錄訂閱時應該省略這些值。

可以使用 DeregAll 選項來取消登錄訂閱者或訂閱者群組的所有訂閱。 例如，如果指定 DeregAll 與訂閱點 (但沒有主題或過濾器) 一起，則不論主題及過濾器為何，都會取消登錄指定訂閱點上訂閱者的所有訂閱。 容許主題、過濾器及訂閱點的任何組合; 如果全部三個都指定，則只能有一個訂閱相符，且會忽略 DeregAll 選項。

訊息必須由登錄訂閱的訂閱者傳送; 這會透過檢查訂閱者的使用者 ID 來確認。

系統管理者也可以使用 MQSC 或 PCF 指令來取消登錄訂閱。 不過，登錄於暫時動態佇列的訂閱會與佇列相關聯，而不只是佇列名稱。 如果明確刪除佇列，或由應用程式切斷與佇列管理程式的連線，則無法再使用 **Deregister Subscriber** 指令來取消登錄該佇列的訂閱。 訂閱可以使用開發人員工作台取消登錄，佇列管理程式會在下次符合訂閱的發佈時，或下次佇列管理程式重新啟動時，自動移除訂閱。 在正常情況下，應用程式應該在刪除佇列之前取消登錄其訂閱，或從佇列管理程式中斷連線。

如果訂閱者傳送訊息以取消登錄訂閱，並收到回應訊息以指出已順利處理此訂閱，則在取消登錄訂閱的同時，如果佇列管理程式正在處理部分發佈，則它們可能仍會到達訂閱者佇列。 如果未從佇列中移除訊息，則訂閱者佇列上可能有未處理的訊息累積。 在休眠一段時間之後，如果應用程式執行的迴圈包含具有適當 CorrelId 的 MQGET 呼叫，則會清除佇列中的這些訊息。

同樣地，如果訂閱者使用永久動態佇列，並在 MQCLOSE 呼叫上使用 *MQCO_DELETE_PURGE* 選項取消登錄並關閉佇列，則佇列可能不是空的。 如果在刪除佇列時尚未確定來自佇列管理程式的任何發佈， MQCLOSE 呼叫會發出 MQRC_Q_NOT_EMPTY 回覆碼。 應用程式可以透過不時休眠並重新發出 MQCLOSE 呼叫來避免此問題。

內容

<Command> (*MQPSC_COMMAND*)
值為 DeregSub (*MQPSC_DEREGISTER_SUBSCRIBER*)。

必須指定此內容。

<Topic> (*MQPSC_TOPIC*)
值是包含要取消登錄之主題的字串。

如果要取消登錄多個主題，則可以選擇性地重複此內容。 如果在 <RegOpt>中指定 DeregAll ，則可以省略它。

如果訂閱者想要保留其他主題的訂閱，則指定的主題可以是已登錄主題的子集。 容許使用萬用字元，但包含萬用字元的主題字串必須完全符合 **Deregister Subscriber** 指令訊息中指定的對應字串。

<SubPoint> (*MQPSC_SUBSCRIPTION_POINT*)
此值是一個字串，指定要從其中分離訂閱的訂閱點。

此內容不得重複。 如果指定 <Topic> ，或在 <RegOpt>中指定 DeregAll ，則可以省略它。 如果您省略此內容，則會發生下列情況:

- 如果您 **不** 指定 DeregAll，則會從預設訂閱點取消登錄符合 <Topic> 內容 (以及 <Filter> 內容 (如果有的話)) 的訂閱。
- 如果您指定 DeregAll，則會從所有訂閱點取消登錄所有訂閱 (如果有的話，符合 <Topic> 和 <Filter> 內容)。

請注意，您無法明確指定預設訂閱點。 因此，無法僅從這個訂閱點取消登錄所有訂閱; 您必須指定主題。

<SubIdentity> (*MQPSC_SUBSCRIPTION_IDENTITY*)
這是長度上限為 64 個字元的可變長度字串。 它用來代表對訂閱感興趣的應用程式。 佇列管理程式會維護每一個訂閱的一組訂閱者身分。 每一個訂閱都可以容許其身分集只保留單一身分，或不限數目的身分。

如果 SubIdentity 位於訂閱的身分集中，則會從該集中移除它。 如果身分集因此而變成空的，則會從佇列管理程式中移除訂閱，除非將 LeaveOnly 指定為 RegOpt 內容的值。 如果身分集仍包含其他身分，則不會從佇列管理程式中移除訂閱，也不會岔斷發佈流程。

如果指定 SubIdentity ，但 SubIdentity 不在訂閱的身分集內，則 **Deregister Subscriber** 指令會失敗，回覆碼為 *MQRCCF_SUB_IDENTITY_ERROR*。

<Filter> (*MQPSC_FILTER*)
此值是一個字串，指定要取消登錄的過濾器。 它必須完全符合 (包括大小寫及任何空格) 先前已登錄的訂閱過濾器。

如果要取消登錄多個過濾器，則可以選擇性地重複此內容。 如果指定 <Topic> ，或在 <RegOpt>中指定 DeregAll ，則可以省略它。

如果訂閱者想要保留其他過濾器的訂閱，則指定的過濾器可以是已登錄過濾器的子集。

<RegOpt> (*MQPSC_REGISTRATION_OPTION*)
登錄選項內容可以採用下列值:

DeregAll
(*MQPSC_DEREGISTER_ALL*)

將取消登錄針對此訂閱者登錄的所有相符訂閱。

如果您指定 DeregAll:

- 可以省略 <Topic>、 <SubPoint>和 <Filter> 。
- 必要的話，可以重複 <Topic> 和 <Filter> 。
- <SubPoint> 不得重複。

如果您 **不** 指定 DeregAll:

- 必須指定 <Topic> ，必要的話可以重複。
- <SubPoint> 和 <Filter> 可以省略。
- <SubPoint> 不得重複。
- 必要的話，可以重複 <Filter> 。

如果主題和過濾器都重複，則會移除符合這兩者的所有組合的所有訂閱。 例如，指定三個主題及三個過濾器的 **Deregister Subscriber** 指令將嘗試移除九個訂閱。

CorrelAsID
(*MQPSC_CORREL_ID_AS_IDENTITY*)

訊息描述子 (MQMD) 中的 CorrelId (不得為零) 用來識別訂閱者。 它必須符合原始訂閱中使用的 CorrelId 。

FullResp
(*MQPSC_FULL_RESPONSE*)

當指定 FullResp 時，如果指令未失敗，則會在回應訊息中傳回訂閱的所有屬性。

當指定 FullResp 時， **Deregister Subscriber** 指令中不允許 DeregAll 。 也無法指定多個主題。 在這兩種情況下，指令都會失敗，回覆碼為 *MQRCCF_REG_OPTIONS_ERROR*。

LeaveOnly
(*MQPSC_LEAVE_ONLY*)

當您在訂閱的身分集中指定 SubIdentity 時，會從訂閱的身分集中移除 SubIdentity 。 即使產生的身分集是空的，也不會從佇列管理程式移除訂閱。 如果 SubIdentity 值不在身分中，則指令會失敗，回覆碼為 *MQRCCF_SUB_IDENTITY_ERROR*。

如果指定 LeaveOnly 時未指定 SubIdentity，則指令會失敗，回覆碼為 *MQRCCF_REG_OPTIONS_ERROR*。

如果既未指定 LeaveOnly 也未指定 SubIdentity ，則不論為訂閱設定的身分內容為何，都會移除訂閱。

NONE
(MQPSC_NONE)

所有選項都採用其預設值。 這與省略登錄選項內容具有相同的效果。 如果同時指定其他選項，則會忽略 **無** 。

VariableUserID
(*MQPSC_VARIABLE_USER_ID*)

指定時，訂閱者 (佇列、佇列管理程式及相關性) 的身分不受限於單一使用者 ID。 這與佇列管理程式的現有行為不同，該佇列管理程式會將原始登錄訊息的使用者 ID 與訂閱者的身分相關聯，然後再開啟，以防止任何其他使用者使用該身分。 如果新的訂閱者嘗試使用相同的身分，則會傳回回覆碼 *MQRCCF_DUPLICATE_SUBSCRIPTION* 。

任何使用者都可以在具有適當權限時修改或取消登錄訂閱，避免現有的檢查使用者 ID 必須符合原始訂閱者的使用者 ID。

若要將此選項新增至現有訂閱，指令必須來自與原始訂閱本身相同的使用者 ID。

如果要取消登錄的訂閱已設定 VariableUserId ，則必須在取消登錄時設定此項，以指出要取消登錄的訂閱。 否則，會使用 **Deregister Subscriber** 指令的使用者 ID 來識別訂閱。 如果提供了訂閱名稱，則會置換此訂閱者 ID 及其他訂閱者 ID。

如果省略此內容，則預設值是不設定任何登錄選項。

<QMgrName> (*MQPSC_Q_MGR_NAME*)
此值是訂閱者佇列的佇列管理程式名稱。 它必須符合原始訂閱中使用的 QMgrName 。

如果省略此內容，則預設值為訊息描述子 (MQMD) 中的 ReplyToQMgr 名稱。 如果產生的名稱空白，則會預設為佇列管理程式的名稱。

<QName> (*MQPSC_Q_NAME*)
值是訂閱者佇列的名稱。 它必須符合原始訂閱中使用的 **完整名稱** 。

如果省略此內容，則預設值為訊息描述子 (MQMD) 中的 ReplyToQ 名稱，其不得為空白。

<SubName> (*MQPSC_SUBSCRIPTION_NAME*)
如果您在 **Deregister Subscriber** 指令上指定 SubName ，除非在訂閱本身設定 VariableUserId ，否則 SubName 值優先於使用者 ID 以外的所有其他 ID 欄位。 如果未設定 VariableUserId ，則只有在指令訊息的使用者 ID 符合訂閱的使用者 ID 時， **Deregister Subscriber** 指令才會成功; 如果未設定，則指令會失敗，回覆碼為 *MQRCCF_DUPLICATE_IDENTITY*。

如果存在符合此指令傳統身分的訂閱，但沒有 SubName，則 **Deregister Subscriber** 指令會失敗，回覆碼為 *MQRCCF_SUB_NAME_ERROR*。 如果嘗試使用符合傳統身分但未指定 SubName 的指令訊息來取消登錄具有 SubName 的訂閱，則指令會成功。

<SubUserData> (*MQPSC_SUBSCRIPTION_USER_DATA*)
這是可變長度字串。 此值由具有訂閱的佇列管理程式儲存，但不會影響將發佈遞送至訂閱者。 可以使用新值重新登錄至相同的訂閱來變更此值。 此屬性用於應用程式。

如果存在 SubUser 資料，則會在訂閱的 Metaopic 資訊 (MQCACF_REG_SUB_USER_DATA) 中傳回資料。

範例

以下是 **Deregister Subscriber** 指令訊息的 NameValue **資料** 範例。 在此範例中，範例應用程式取消登錄其對主題的訂閱，這些主題包含所有相符項的最新評分。 訂閱者的身分 (包括 CorrelId) 取自 MQMD 中的預設值。

```
<psc>
 <Command>DeregSub</Command>
 <RegOpt>CorrelAsId</RegOpt>
 <Topic>Sport/Soccer/State/LatestScore/#</Topic>
</psc>
```

發佈訊息

Publish 指令訊息會放入佇列中，或從佇列管理程式放入訂閱者中，以發佈指定一或多個主題的相關資訊。

必須具備將訊息放入佇列的權限，以及發佈指定一或多個主題之相關資訊的權限。

如果使用者有權發佈部分 (而非全部) 主題的相關資訊，則只會使用那些主題來發佈; 警告回應會指出哪些主題未用來發佈。

如果訂閱者有任何相符的訂閱，佇列管理程式會將 **Publish** 訊息轉遞至對應 **Register Subscriber** 指令訊息中所定義的訂閱者佇列。

如需將指令訊息傳送至佇列管理程式時所需的訊息描述子 (MQMD) 參數的詳細資料，以及在佇列管理程式將發佈資訊轉遞至訂閱者時使用的詳細資料，請參閱 佇列管理程式回應訊息 。

除非 **Publish** 訊息是本端發佈，否則佇列管理程式會將它轉遞至網路中具有相符訂閱的其他佇列管理程式。

發佈資料 (如果有的話) 包含在訊息內文中。 資料可以在 MQRFH2 標頭的 NameValue **資料** 欄位中的 <mcd> 資料夾中說明。

內容

<Command> (*MQPSC_COMMAND*)
值為 Publish(*MQPSC_PUBLISH*)。

必須指定此內容。

<Topic> (*MQPSC_TOPIC*)
此值是包含分類此出版品之主題的字串。 不接受萬用字元。

您必須將主題新增至名單 SYSTEM.QPUBSUB.QUEUE.NAMELIST，請參閱 新增串流，以取得如何完成此作業的指示。

必須指定此內容，並且可以視需要選擇性地針對多個主題重複此內容。

<SubPoint> (*MQPSC_SUBSCRIPTION_POINT*)
發佈的訂閱點。

在 WebSphere Event Broker V6 中， <SubPoint> 內容的值是處理發佈之 Publication 節點的「訂閱點」屬性值。

在 WebSphere MQ V7.0.1 中， <SubPoint> 內容的值必須符合訂閱點的名稱。 請參閱 新增訂閱點 。

<PubOpt> (*MQPSC_PUBLICATION_OPTION*)
發佈選項內容可以採用下列值:

RetainPub
(*MQPSC_RETAIN_PUB*)

佇列管理程式會保留發佈資訊的副本。 如果未設定此選項，則只要佇列管理程式將發佈傳送至其所有現行訂閱者，即會刪除發佈。

IsRetained 發佈
(*MQPSC_IS_RETAINED_PUB*)

(只能由佇列管理程式設定。) 佇列管理程式已保留此發佈資訊。 佇列管理程式會設定此選項，以通知訂閱者此發佈資訊已發佈且已保留，前提是已使用 InformIf **保留** 選項登錄訂閱。 只有在回應 Register Subscriber 或 Request Update 指令訊息時，才會設定它。 直接傳送至訂閱者的保留發佈未設定此選項。

本端
(*MQPSC_LOCAL*)

此選項會告知佇列管理程式，不得將此發佈資訊傳送至其他佇列管理程式。 如果在此佇列管理程式中登錄的所有訂閱者都有相符的訂閱，則會收到此發佈。

OtherSubs 僅
(*MQPSC_OTHER_SUBS_ONLY*)

此選項容許更簡單的處理會議類型應用程式，其中發佈者也是相同主題的訂閱者。 它告訴佇列管理程式不要將發佈傳送至發佈者的訂閱者佇列，即使它有相符的訂閱。 發佈者的訂閱者佇列由其 QMgrName、 **完整名稱**及選用 CorrelId 組成，如下列清單中所述。

CorrelAsID
(*MQPSC_CORREL_ID_AS_IDENTITY*)

在發佈者也是訂閱者的應用程式中， MQMD (不得為零) 中的 CorrelId 是發佈者訂閱者佇列的一部分。

NONE
(*MQPSC_NONE*)

所有選項都採用其預設值。 這與省略發佈選項內容具有相同的效果。 如果同時指定其他選項，則會忽略 **無** 。

您可以透過引入其他 <PubOpt> 元素來具有多個發佈選項。

如果省略此內容，則預設值是不設定發佈選項。

<PubTime> (*MQPSC_PUBLISH_TIMESTAMP*)
此值是發佈者所設定的選用發佈時間戳記。 其長度為 16 個字元，格式如下:

```
YYYYMMDDHHMMSSTH
```

使用「通用時間」。 在傳送給訂閱者之前，佇列管理程式不會檢查此資訊。

<SeqNum> (*MQPSC_SEQUENCE_NUMBER*)
此值是發佈者所設定的選用序號。

每一個發佈都必須以 1 為增量。 不過，佇列管理程式不會檢查此情況，它只會將此資訊傳輸給訂閱者。

如果將相同主題上的發佈發佈發佈至不同的交互連接佇列管理程式，發佈者必須負責確保序號 (如果使用的話) 是有意義的。

<QMgrName> (*MQPSC_Q_MGR_NAME*)
在發佈者也是訂閱者的應用程式中 (請參閱 OtherSubs **僅**) ，此值是一個字串，包含發佈者訂閱者佇列的佇列管理程式名稱。

如果省略此內容，則預設值為訊息描述子 (MQMD) 中的 ReplyToQMgr 名稱。 如果產生的名稱空白，則會預設為佇列管理程式的名稱。

<QName> (*MQPSC_Q_NAME*)
在發佈者也是訂閱者的應用程式中，此值是包含發佈者的訂閱者佇列名稱的字串 (請參閱 OtherSubs **僅**)。

如果省略此內容，則預設值為訊息描述子 (MQMD) 中的 ReplyToQ 名稱，如果設定 OtherSubsOnly ，則此名稱不得為空白。

範例

以下是 **Publish** 指令訊息的部分 *NameValue* 資料 範例。

第一個範例是針對範例應用程式中相符模擬器所傳送的發佈，指出已啟動相符項。

```
<psc>
 <Command>Publish</Command>
 <Topic>Sport/Soccer/Event/MatchStarted</Topic>
</psc>
```

第二個範例適用於保留的發佈。 會發佈 Team1 與 Team2 之間相符的最新評分。

```
<psc>
 <Command>Publish</Command>
 <PubOpt>RetainPub</PubOpt>
 <Topic>Sport/Soccer/State/LatestScore/Team1 Team2</Topic>
</psc>
```

登錄訂閱者訊息

Register Subscriber 指令訊息由訂閱者傳送至佇列管理程式，或由另一個應用程式代表訂閱者傳送至佇列管理程式，以指出它想要在訂閱點訂閱一個以上主題。 也可以指定訊息內容過濾器。

在發佈/訂閱過濾表示式中，巢狀括弧會導致效能指數減少。 避免巢狀括弧至大於大約 6 的深度。

訊息會傳送至 SYSTEM.BROKER.CONTROL.QUEUE，這是佇列管理程式的控制佇列。 除了訂閱中主題或主題的存取權 (由佇列管理程式系統管理者設定) 之外，還需要將訊息放入此佇列的權限。

如果使用者對部分 (而非全部) 主題具有權限，則只會登錄具有權限的主題; 警告回應會指出未登錄的主題。

如需將指令訊息傳送至佇列管理程式時所需的訊息描述子 (MQMD) 參數的詳細資料，請參閱 第 783 頁的『傳送至佇列管理程式之指令訊息中的 MQMD 設定』 。

如果回覆目的地佇列是暫時動態佇列，當佇列關閉時，佇列管理程式會自動取消登錄訂閱。

內容

<Command> (*MQPSC_COMMAND*)
值為 RegSub (*MQPSC_REGISTER_SUBSCRIBER*)。 必須指定此內容。

<Topic> (*MQPSC_TOPIC*)
訂閱者想要接收其發佈的主題。 萬用字元可以指定為主題的一部分。

如果您使用 MQSC 指令 **display sub** 來檢查以此方式建立的訂閱，則 <Topic> 標籤的值會顯示為訂閱的 TOPICSTR 內容。

這是必要內容，可以視需要選擇性地針對多個主題重複此內容。

<SubPoint> (*MQPSC_SUBSCRIPTION_POINT*)
此值是附加訂閱的訂閱點。

如果省略此內容，則會使用預設訂閱點。

在 WebSphere 事件分配管理系統 V6 中， <SubPoint> 內容的值必須符合訂閱之 Publication 節點的「訂閱點」屬性值。

在 WebSphere MQ V7.0.1 中， <SubPoint> 內容的值必須符合訂閱點的名稱。 請參閱 新增訂閱點 。

<Filter> (*MQPSC_FILTER*)
此值是一個 SQL 表示式，用來作為發佈訊息內容的過濾器。 如果指定主題上的發佈資訊符合過濾器，則會將它傳送至訂閱者。 此內容對應於 MQSUB 和 MQOPEN 呼叫中使用的「選取字串」。 如需相關資訊，請參閱 在訊息內容上選取

如果省略此內容，則不會進行內容過濾。

<RegOpt> (*MQPSC_REGISTRATION_OPTION*)
此「登錄選項」內容可以採用下列值:

AddName
(*MQPSC_ADD_NAME*)

針對符合此「登錄訂閱」指令的傳統身分，但沒有現行 SubName 值的現有訂閱指定時，此指令中指定的 SubName 會新增至訂閱。

如果指定 AddName ，則 SubName 欄位是必要的，否則會傳回 MQRCCF_REG_OPTIONS_ERROR。

CorrelAsID
(*MQPSC_CORREL_ID_AS_IDENTITY*)

將相符發佈傳送至訂閱者佇列時，會使用訊息描述子 (MQMD) 中的 CorrelId 。 CorrelId 不能是零，

FullResp
(*MQPSC_FULL_RESPONSE*)

指定時，如果指令未失敗，則會在回應訊息中傳回訂閱的所有屬性。

只有在指令訊息參照單一訂閱時， FullResp 才有效。 因此，指令中只允許一個主題; 否則指令會失敗，回覆碼為 *MQRCCF_REG_OPTIONS_ERROR*。

InformIfRet
(*MQPSC_INFORM_IF_RETAINED*)

當佇列管理程式傳送「發佈」訊息來回應 **Register Subscriber** 或 **Request Update** 指令訊息時，會通知訂閱者是否保留發佈。 佇列管理程式會在訊息中包括 IsRetainedPub 發佈選項來執行此動作。

JoinExcl
(*MQPSC_JOIN_EXCLUSIVE*)

此選項指出應該將指定的 SubIdentity 新增為訂閱的身分集的專用成員，且無法將其他身分新增至該身分集。

如果身分已加入 'shared' 且是集合中的唯一項目，則該集合會變更為此身分所保留的專用鎖定。 否則，如果訂閱目前在身分集中具有其他身分 (具有共用存取權) ，則指令會失敗，回覆碼為 *MQRCCF_SUBSCRIPTION_IN_USE*。

JoinShared
(*MQPSC_JOIN_SHARED*)

此選項指出指定的 SubIdentity 應該新增至訂閱的身分集。

如果目前只鎖定訂閱 (使用 JoinExcl 選項) ，則指令會失敗，回覆碼為 *MQRCCF_SUBSCRIPTION_LOCKED*，除非鎖定訂閱的身分與此指令訊息中的身分相同。 在此情況下，會自動將鎖定修改為共用鎖定。

本端
(*MQPSC_LOCAL*)

訂閱是本端訂閱，且不會配送至網路中的其他佇列管理程式。 其他佇列管理程式的發佈不會遞送至這個訂閱者，除非它也有對應的廣域訂閱。

NewPubs
(*MQPSC_NEW_PUBS_ONLY*)

登錄訂閱時存在的保留發佈不會傳送至訂閱者; 只會傳送新的發佈。

如果訂閱者重新登錄並變更此選項，使其不再設定，則可能會重新傳送已傳送至其中的發佈。

NoAlter
(*MQPSC_NO_ALTER*)

不會變更現有相符訂閱的屬性。

建立訂閱時，會忽略此選項。 指定的所有其他選項都適用於新訂閱。

如果 SubIdentity 也指定了其中一個結合選項 (JoinExcl 或 JoinShared)，則不論是否指定 NoAlter，都會將身分新增至身分集。

NONE
(*MQPSC_NONE*)

所有登錄選項都會採用其預設值。

如果訂閱者已登錄，其選項會重設為其預設值 (請注意，這 不會 與省略登錄選項內容有相同的影響)，且訂閱期限會從 **Register Subscriber** 訊息的 MQMD 更新。

如果同時指定其他登錄選項，則會忽略 **無** 。

NonPers
(*MQPSC_NON_PERSISTENT*)

符合此訂閱的發佈會以非持續訊息形式遞送至訂閱者。

珀斯
(*MQPSC_PERSISTENT*)

符合此訂閱的發佈會以持續訊息形式遞送至訂閱者。

PersAs 發佈
(*MQPSC_PERSISTENT_AS_PUBLISH*)

符合此訂閱的發佈會以發佈者指定的持續性遞送至訂閱者。 這是預設行為。

PersAs 佇列
(*MQPSC_PERSISTENT_AS_Q*)

符合此訂閱的發佈會以訂閱者佇列上指定的持續性遞送至訂閱者。

PubOnReqOnly
(*MQPSC_PUB_ON_REQUEST_ONLY*)

除了回應 **Request Update** 指令訊息之外，佇列管理程式不會將發佈傳送至訂閱者。

VariableUserID
(*MQPSC_VARIABLE_USER_ID*)

指定時，訂閱者 (佇列、佇列管理程式及相關性) 的身分不受限於單一使用者 ID。 這與佇列管理程式的現有行為不同，該佇列管理程式會將原始登錄訊息的使用者 ID 與訂閱者的身分相關聯，然後再開啟，以防止任何其他使用者使用該身分。 如果新訂閱者嘗試使用相同的身分，則會傳回 *MQRCCF_DUPLICATE_SUBSCRIPTION* 。

這可讓任何使用者修改或取消登錄訂閱 (如果使用者具有適當的權限)。 因此，不需要檢查使用者 ID 是否符合原始訂閱者的使用者 ID。

若要將此選項新增至現有訂閱，指令必須來自與原始訂閱本身相同的使用者 ID。

如果 **Request Update** 指令的訂閱已設定 VariableUserId，則必須在要求更新時設定此項，以指出所參照的訂閱。 否則，會使用 **Request Update** 指令的使用者 ID 來識別訂閱。 如果提供了訂閱名稱，則會置換此訂閱者 ID 及其他訂閱者 ID。

如果沒有此選項集的 **Register Subscriber** 指令訊息參照已設定此選項的現有訂閱，則會從此訂閱中移除此選項，且現在已修正訂閱的使用者 ID。 如果已存在具有相同身分 (佇列、佇列管理程式及相關性 ID) 但具有與它相關聯的不同使用者 ID 的訂閱者，則指令會失敗，回覆碼為 *MQRCCF_DUPLICATE_IDENTITY*，因為只能有一個使用者 ID 與訂閱者身分相關聯。

如果省略登錄選項內容且已登錄訂閱者，則不會變更其登錄選項，且會從 **Register Subscriber** 訊息的 MQMD 更新訂閱期限。

如果訂閱者尚未登錄，則會建立新的訂閱，且所有登錄選項都採用其預設值。

預設值為 PersAs **發佈** ，且未設定其他選項。

<QMgrName> (*MQPSC_Q_MGR_NAME*)
此值是訂閱者佇列的佇列管理程式名稱，佇列管理程式會將相符的發佈傳送至該佇列管理程式。

如果省略此內容，則預設值為訊息描述子 (MQMD) 中的 ReplyToQMgr 名稱。 如果產生的名稱空白，則會預設為佇列管理程式的 QMgrName。

<QName> (*MQPSC_Q_NAME*)
該值是訂閱者佇列的名稱，佇列管理程式會將符合的發佈傳送至其中。

如果省略此內容，則預設值為訊息描述子 (MQMD) 中的 ReplyToQ 名稱，在此情況下不得為空白。

如果佇列是暫時動態佇列，則必須在 <RegOpt> 內容中指定非持續性發佈遞送 (NonPers)。

如果佇列是暫時動態佇列，當佇列關閉時，佇列管理程式會自動取消登錄訂閱。

<SubName> (*MQPSC_SUBSCRIPTION_NAME*)
這是提供給特定訂閱的名稱。 您可以使用它而非佇列管理程式、佇列及選用 correlId 來參照訂閱。

如果已存在具有此 **SubName** 的訂閱，則訂閱的任何其他屬性 (Topic、 QMgrName、QName、CorrelId、 UserId、 RegOpts、 UserSubData 及 Expiry) 都會置換為在新的 Register Subscriber 指令訊息中傳遞的屬性 (如果指定的話)。 不過，如果在未指定完整名稱欄位的情況下使用 **SubName** ，且在 MQMD 標頭中指定 ReplyToQ ，則訂閱者佇列會變更為 ReplyToQ。

如果已存在符合此指令傳統身分的訂閱，但沒有 **SubName**，則除非指定 **AddName** 選項，否則登錄指令會失敗，回覆碼為 *MQRCCF_DUPLICATE_SUBSCRIPTION*。

如果您嘗試使用另一個指定相同 **SubName** 的 Register Subscriber 指令來變更現有的具名訂閱，且新指令中的 Topic、 QMgrName、QName 及 CorrelId 值符合不同的現有訂閱，且已定義或未定義 SubName ，則指令會失敗，回覆碼為 *MQRCCF_DUPLICATE_SUBSCRIPTION*。 這可防止兩個訂閱名稱參照相同的訂閱。

<SubIdentity> (*MQPSC_SUBSCRIPTION_IDENTITY*)
此字串用來代表對訂閱感興趣的應用程式。 它是長度上限為 64 個字元的可變長度字串，並且是選用項目。 佇列管理程式會維護每一個訂閱的一組訂閱者身分。 每一個訂閱都可以容許其身分集只包含一個身分，或不限數目的身分 (請參閱 **JoinShared** 及 **JoinExcl** 選項)。

指定 **JoinShared** 或 **JoinExcl** 選項的訂閱指令會將 **SubIdentity** 新增至訂閱的身分集 (如果它尚未存在，且現有身分集容許此類動作); 亦即，沒有其他訂閱者專門加入或身分集是空的。

由於指定 **SubIdentity** 的 Register Subscriber 指令而對訂閱屬性進行的任何變更，只有在它是此訂閱的身分集的唯一成員時才會成功。 否則指令會失敗，回覆碼為 *MQRCCF_SUBSCRIPTION_IN_USE*。 這可防止訂閱的屬性變更，而不會讓其他感興趣的訂閱者知道。

如果您指定長度超過 64 個字元的字串，則指令會失敗，回覆碼為 *MQRCCF_SUB_IDENTITY_ERROR*。

<SubUserData> (*MQPSC_SUBSCRIPTION_USER_DATA*)
這是可變長度字串。 此值由佇列管理程式與訂閱一起儲存，但對發佈遞送至訂閱者沒有影響。 可以使用新值重新登錄至相同的訂閱來變更此值。 這個屬性是供應用程式使用。

如果訂閱存在，則會在 Metaopic 資訊 (*MQCACF_REG_SUB_USER_DATA*) 中傳回 **SubUser 資料** 。

如果您指定多個登錄選項值 NonPers, PersAsPub, PersAsQueue, and Pers，則只會使用最後一個登錄選項值。 您無法在個別訂閱中結合這些選項。

範例

以下是 **Register Subscriber** 指令訊息的 NameValue **資料** 範例。 在範例應用程式中，結果服務會使用此訊息來登錄主題的訂閱，其中包含所有相符項中的最新評分，並設定「持續作為發佈」選項。 訂閱者的身分 (包括 CorrelId) 取自 MQMD 中的預設值。

```
<psc>
 <Command>RegSub</Command>
 <RegOpt>PersAsPub</RegOpt>
 <RegOpt>CorrelAsId</RegOpt>
 <Topic>Sport/Soccer/State/LatestScore/#</Topic>
</psc>
```

要求更新訊息

Request Update 指令訊息會從訂閱者傳送至佇列管理程式，以要求符合給定 (選用) 過濾器之指定主題及訂閱點的現行保留發佈資訊。

此訊息會傳送至 *SYSTEM.BROKER.CONTROL.QUEUE*，佇列管理程式的控制佇列。 除了要求更新中主題的存取權之外，還需要將訊息放入此佇列的權限; 這是由佇列管理程式的系統管理者所設定。

如果訂閱者在登錄時指定選項 PubOnReqOnly，則通常會使用此指令。 如果佇列管理程式具有任何相符的保留發佈資訊，則會將它們傳送至訂閱者。 如果佇列管理程式沒有相符的保留發佈資訊，則要求會失敗，回覆碼為 *MQRCCF_NO_RETAINED_MSG*。 要求者必須先前已使用相同的「主題」、 SubPoint 及「過濾器」值登錄訂閱。

內容

<Command> (*MQPSC_COMMAND*)
值為 ReqUpdate (*MQPSC_REQUEST_UPDATE*)。 必須指定此內容。

<Topic> (*MQPSC_TOPIC*)
此值是訂閱者要求的主題; 容許使用萬用字元。

必須指定此內容，但此訊息中只容許出現一次。

<SubPoint> (*MQPSC_SUBSCRIPTION_POINT*)
此值是附加訂閱的訂閱點。

如果省略此內容，則會使用預設訂閱點。

<Filter> (*MQPSC_FILTER*)
此值是 ESQL 表示式，用來作為發佈訊息內容的過濾器。 如果指定主題上的發佈資訊符合過濾器，則會將它傳送至訂閱者。

<Filter> 內容的值應該與您現在要求更新之原始訂閱上指定的值相同。

如果省略此內容，則不會進行內容過濾。

<RegOpt> (*MQPSC_REGISTRATION_OPTION*)
登錄選項內容可以採用下列值:

CorrelAsID
(*MQPSC_CORREL_ID_AS_IDENTITY*)

將相符發佈傳送至訂閱者佇列時，會使用訊息描述子 (MQMD) 中的 CorrelId (不得為零)。

NONE
(*MQPSC_NONE*)

所有選項都採用其預設值。 這與省略 <RegOpt> 內容具有相同的效果。 如果同時指定其他選項，則會忽略 **無** 。

VariableUserID
(*MQPSC_VARIABLE_USER_ID*)

指定時，訂閱者 (佇列、佇列管理程式及 correlid) 的身分不會限制為單一使用者 ID。 這與佇列管理程式的現有行為不同，該佇列管理程式會將原始登錄訊息的使用者 ID 與訂閱者的身分相關聯，然後

再開啟，以防止任何其他使用者使用該身分。 如果新訂閱者嘗試使用相同的身分，則指令會失敗，回覆碼為 *MQRCCF_DUPLICATE_SUBSCRIPTION*。

這可讓任何使用者在具有適當權限時修改或取消登錄訂閱。 因此，不需要檢查使用者 ID 是否符合原始訂閱者的使用者 ID。

若要將此選項新增至現有訂閱，指令必須來自與原始訂閱相同的使用者 ID。

如果 **Request Update** 指令的訂閱已設定 VariableUserId，則必須在要求更新時設定此項，以指出所參照的訂閱。 否則，會使用 **Request Update** 指令的使用者 ID 來識別訂閱。 如果提供了訂閱名稱，則會置換此訂閱者 ID 及其他訂閱者 ID。

如果省略此內容，則預設值是不設定任何登錄選項。

<QMgrName> (*MQPSC_Q_MGR_NAME*)
此值是訂閱者佇列的佇列管理程式名稱，佇列管理程式會將相符的保留發佈資訊傳送至該佇列管理程式。

如果省略此內容，則預設值為訊息描述子 (MQMD) 中的 ReplyToQMgr 名稱。 如果產生的名稱空白，則會預設為佇列管理程式的 QMgrName。

<QName> (*MQPSC_Q_NAME*)
此值是訂閱者佇列的名稱，佇列管理程式會將符合的保留發佈資訊傳送至該訂閱者佇列。

如果省略此內容，則預設值為訊息描述子 (MQMD) 中的 ReplyToQ 名稱，在此情況下不得為空白。

<SubName> (*MQPSC_SUBSCRIPTION_NAME*)
這是提供給特定訂閱的名稱。 如果在 **Request Update** 指令上指定，除非在訂閱本身上設定 VariableUserId，否則 SubName 值優先於使用者 ID 以外的所有其他 ID 欄位。 如果未設定 VariableUserId，則只有在指令訊息的使用者 ID 符合訂閱的使用者 ID 時， 要求更新 指令才會成功。 如果指令訊息的使用者 ID 不符合訂閱的使用者 ID，則指令會失敗，回覆碼為 *MQRCCF_DUPLICATE_IDENTITY*。

如果已設定 VariableUserId，且使用者 ID 與訂閱的使用者 ID 不同，則當新指令訊息的使用者 ID 有權瀏覽串流佇列並放入訂閱的訂閱者佇列時，指令會成功。 否則，指令會失敗，回覆碼為 *MQRCCF_NOT_AUTHORIZED*。

如果存在符合此指令傳統身分的訂閱，但沒有 SubName，則 **Request Update** 指令會失敗，回覆碼為 *MQRCCF_SUB_NAME_ERROR*。

如果嘗試使用符合傳統身分但未指定 SubName 的指令訊息，對具有 SubName 的訂閱要求更新，則指令會成功。

範例

以下是 **Request Update** 指令訊息的 NameValue **資料** 範例。 在範例應用程式中，結果服務會使用此訊息來要求保留的發佈，其中包含所有團隊的最新評分。 訂閱者的身分 (包括 CorrelId) 取自 MQMD 中的預設值。

```
<psc>
 <Command>ReqUpdate</Command>
 <RegOpt>CorrelAsId</RegOpt>
 <Topic>Sport/Soccer/State/LatestScore/#</Topic>
</psc>
```

佇列管理程式回應訊息

如果指令訊息描述子指定需要回應，則會將 **Queue Manager Response** 訊息從佇列管理程式傳送至發佈者或訂閱者的 ReplyToQ，以指出佇列管理程式所接收指令訊息的成功或失敗。

回應訊息包含在 <pscr> 資料夾中 MQRFH2 標頭的 NameValue **資料** 欄位內。

如果發生警告或錯誤，回應訊息會包含指令訊息中的 <psc> 資料夾，以及 <pscr> 資料夾。 訊息資料 (如果有的話) 未包含在佇列管理程式回應訊息中。 如果發生錯誤，則未處理任何導致錯誤的訊息; 如果發生警告，則部分訊息可能已順利處理。

如果傳送回應失敗:

- 對於發佈訊息，如果 MQPUT 失敗，佇列管理程式會嘗試將回應傳送至 WebSphere MQ 無法傳送郵件的佇列。 這可讓發佈傳送給訂閱者，即使回應無法傳回給發佈者。
- 對於其他訊息，或如果發佈回應無法傳送至無法傳送郵件的佇列，則會記載錯誤，且指令訊息通常會回復。 是否發生此情況取決於 MQInput 節點的配置方式。

內容

<Completion> (*MQPSCR_COMPLETION*)
完成碼，可以採用下列三個值之一:

確定
指令已順利完成

警告
指令已完成，但有警告

錯誤
指令失敗

<Response> (*MQPSCR_response*)
指令訊息的回應 (如果該指令產生完成碼 warning 或 error)。 它包含 <Reason> 內容，並且可能包含指出警告或錯誤原因的其他內容。

如果有一或多個錯誤，則只有一個回應資料夾，僅指出第一個錯誤的原因。 如果有一或多個警告，則每一個警告都有一個回應資料夾。

<Reason> (*MQPSCR_REASON*)
如果完成碼是 **警告** 或 **錯誤**，則為定義完成碼的原因碼。 它設為下列範例中列出的其中一個錯誤碼。 <Reason> 內容包含在 <Response> 資料夾內。 原因碼後面可以接著 <psc> 資料夾中的任何有效內容 (例如，主題名稱) ，指出錯誤或警告的原因。 如果您收到原因碼? ??? ， check the data for correctness, for example, matching angled brackets (< >).

範例

以下是 **Queue Manager Response** 訊息中 NameValue **資料** 的部分範例。 成功的回應可能如下:

```
<pscr>
  <Completion>ok</Completion>
</pscr>
```

以下是失敗回應的範例; 失敗是過濾器錯誤。 第一個 NameValue **資料** 字串包含回應; 第二個字串包含原始指令。

```
<pscr>
  <Completion>error</Completion>
  <Response>
    <Reason>3150</Reason>
  </Reponse>
</pscr>

<psc>
 ...
 command message (to which
 the queue manager is responding)
 ...
</psc>
```

以下是警告回應的範例 (由於未獲授權的主題)。 第一個 NameValue **資料** 字串包含回應; 第二個 NameValue **資料** 字串包含原始指令。

```
<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>
```

發佈/訂閱原因碼

這些原因碼可能會在發佈/訂閱回應 <pscr> 資料夾的 **原因** 欄位中傳回。 也會列出可用來在 C 或 C++ 程式設計語言中代表這些程式碼的常數。

MQRC_ 常數需要 WebSphere MQ cmqc.h 標頭檔。 MQRCCF_ 常數需要 WebSphere MQ cmqcfc.h 標頭檔 (除了需要 cmqpsc.h 標頭檔的 *MQRCCF_FILTER_ERROR* 和 *MQRCCF_WRONG_USER* 之外)。

原因碼和文字	說明	發出者
2336 MQRC_RFH_COMMAND_ERROR	<psc> 資料夾的 <Command> 欄位有效值為: RegSub、 DeregSub、 Publish、 DeletePub 及 ReqUpdate。 任何其他值都會導致發出此錯誤碼。	任何指令
2337 MQRC_RFH_ PARM_ERROR	<psc> 和 <mcd> 資料夾都有一組可以在其中指定的有效參數。 請檢查這些資料夾的說明，並確定您未指定不正確的參數。	任何指令
2338 MQRC_RFH_DUPLICATE_PARM	<psc> 資料夾內的部分參數 (例如, Topic) 可以重複，但其他參數 (例如, Command) 不能重複。 請確認您沒有複製不可重複的參數。	任何指令
2339 遺漏 MQRC_RFH_PARM_MISSING	<psc> 或 <mcd> 資料夾內的部分參數是選用項目，可以省略; 部分參數是必要項目，且不得省略。 確認您已在 <psc> 和 <mcd> 資料夾中併入所有必要參數。	任何指令
2551 MQRC_SELECTION_NOT_AVAILABLE	沒有延伸訊息選取提供者可用來判斷哪些訂閱者應該指定過濾器來接收發佈。	發佈、登錄訂閱者及要求更新
	沒有延伸訊息選取提供者可用來處理指定訂閱者的過濾器。	登錄訂閱者及要求更新
2554 MQRC_CONTENT_ERROR	延伸訊息選取提供者在現行或保留的發佈中發現錯誤。	發佈及要求更新
3008 MQRCF_COMMAND_FAILED	發生內部錯誤，導致指令無法正確執行。 如果重新發出指令，則可能會發生錯誤。 佇列管理程式的系統事件日誌包含向 IBM 報告問題時應該使用的資訊。	任何指令
3072 MQRCCF_TOPIC_ERROR	您為「主題」參數提供的一或多個值不正確。 請檢查主題的值是否符合指定的限制。	任何指令

原因碼和文字	說明	發出者
3073 MQRCCF_NOT_REGISTERED	您在 DeregSub 或 ReqUpdate 指令上指定的 SubPoint、Topic 及 Filter 的組合不是您先前已登錄的組合，或者對於 DeregSub 指令，如果指定 DeregAll 選項，則未使用其中一個 SubPoint、Topic 或 Filter 內容來取消登錄任何訂閱。	取消登錄「訂閱者」和「要求更新」指令
3074 MQRCCF_Q_MGR_NAME_ERROR	指定的佇列管理程式無效，或佇列管理程式無法使用或不存在。	「取消登錄訂閱者」、「發佈」、「登錄訂閱者」及「要求更新」指令
3076 MQRCCF_Q_NAME_ERROR	指定的佇列名稱無效，或佇列不存在於指定的佇列管理程式上。	「取消登錄訂閱者」、「發佈」、「登錄訂閱者」及「要求更新」指令
3077 MQRCCF_NO_RETAINED_MSG	您指定的主題沒有保留的訊息。 視應用程式的設計而定，這可能是或可能不是錯誤。	要求更新指令
3079 MQRCCF_INCORRECT_Q	RegSub、 DeregSub 及 ReqUpdate 指令一律會傳送至 SYSTEM.BROKER.CONTROL.QUEUE 佇列。 發佈及刪除發佈指令會傳送至特定發佈/訂閱訊息流程的輸入佇列; 這是在設計訊息流程時所決定。 如果指令傳送至錯誤佇列，則會傳回此錯誤碼。	任何指令
3080 MQRCCF_CORREL_ID_ERROR	您已指定 CorrelAsId 作為其中一個 RegOpt 參數。 不過， MQMD 的 CorrelId 欄位不包含有效的相關性 ID (亦即，設為 MQCI_NONE)。	取消登錄訂閱者和登錄訂閱者指令
3081 MQRCCF_NOT_AUTHORIZED	您未獲授權來執行所要求的動作。 佇列管理程式的授權設定由系統管理者使用「主題階層」編輯器來處理。	發佈和登錄訂閱者指令
3083 MQRCCF_REG_OPTIONS_ERROR	您在包含 RegSub 或 DeregSub 指令的 <psc> 資料夾中指定了無法辨識的 RegOpt 參數。	取消登錄訂閱者和登錄訂閱者指令
3084 MQRCCF_PUB_OPTIONS_ERROR	您已在包含「發佈」指令的 <psc> 資料夾中指定無法辨識的 PubOpt 參數。	發佈指令
3087 MQRCCF_DEL_OPTIONS_ERROR	您在包含 DeletePub 指令的 <psc> 資料夾中指定了無法辨識的 DelOpt 參數。	刪除發佈指令
3150 MQRCCF_FILTER_ERROR	指定給 Filter 參數的值無效。 請檢查說明過濾表示式有效語法的區段，並確定您的表示式符合。	取消登錄訂閱者、登錄訂閱者及要求更新指令
3151 MQRCCF_WRONG_USER	已存在符合所指定之訂閱的訂閱; 不過，它已由不同的使用者登錄。 訂閱只能由原先登錄它的使用者變更或取消登錄。	取消登錄訂閱者、登錄訂閱者及要求更新指令

原因碼和文字	說明	發出者
3152 MQRCCF_DUPLICATE_SUBSCRIPTION	已存在具有不同訂閱名稱的相符訂閱。	
3153 MQRCCF_SUB_NAME_ERROR	訂閱名稱的格式無效，或已存在沒有訂閱名稱的相符訂閱。	
3154 MQRCCF_SUB_IDENTITY_ERROR	訂閱身分參數發生錯誤。 提供的值超出容許的長度上限，或訂閱身分目前不是訂閱身分集的成員，且未指定「結合登錄」選項。	
3155 MQRCCF_SUBSCRIPTION_IN_USE	身分集的成員嘗試修改或取消登錄訂閱，但它不是此集的唯一成員。	
3156 MQRCCF_SUBSCRIPTION_LOCKED	訂閱目前由另一個身分專用鎖定。	
3157 MQRCCF_ALREADY_COMBED	已指定結合登錄選項，但訂閱者身分已是訂閱身分集的成員。	

傳送至佇列管理程式之指令訊息中的 MQMD 設定

將指令訊息傳送至佇列管理程式的應用程式會使用訊息描述子 (MQMD) 中的下列欄位設定。 這裡不會列出保留為預設值的欄位，也不會以一般方式列出可設為任何有效值的欄位。

報告
請參閱 MsgType 和 CorrelId。

MsgType
MsgType 應該設為 *MQMT_REQUEST* 或 *MQMT_DATAGRAM*。 如果 MsgType 未設為其中一個值，則將傳回 *MQRC_MSG_TYPE_ERROR* 。

如果一律需要回應，則 MsgType 應該針對指令訊息設定為 *MQMT_REQUEST* 。 在此情況下， **報告** 欄位中的 MQRO_PAN 和 MQRO_NAN 旗標並不重要。

如果 MsgType 設為 *MQMT_DATAGRAM*，則回應取決於 **報告** 欄位中 MQRO_PAN 和 MQRO_NAN 旗標的設定:

- 僅 MQRO_PAN 表示只有在指令成功時，佇列管理程式才會傳送回應。
- 僅 MQRO_NAN 表示只有在指令失敗時，佇列管理程式才會傳送回應。
- 如果指令完成但有警告，則會在設定 MQRO_PAN 或 MQRO_NAN 時傳送回應。
- MQRO_PAN + MQRO_NAN 表示不論指令成功或失敗，佇列管理程式都會傳送回應。 從佇列管理程式的視景中，這與將 MsgType 設為 MQMT_REQUEST 的效果相同。
- 如果未設定 MQRO_PAN 或 MQRO_NAN ，則不會傳送任何回應。

格式
設為 MQFMT_RF_HEADER_2

MsgId
此欄位通常設為 MQMI_NONE ，因此佇列管理程式會產生唯一值。

CorrelId
此欄位可以設為任何值。 如果傳送端身分包括 CorrelId，請在 **報告** 欄位中指定此值以及 MQRO_PASS_CORREL_ID ，以確保在佇列管理程式傳送給傳送端的所有回應訊息中設定此值。

ReplyToQ
此欄位定義要傳送回應的佇列 (如果有的話)。 這可能是傳送端的佇列; 優點是訊息中可以省略 **完整名稱** 參數。 不過，如果要將回應傳送至不同的佇列，則需要 **完整名稱** 參數。

回覆目的地佇列管理程式
此欄位定義回應的佇列管理程式。 如果您將此欄位保留空白 (預設值)，則本端佇列管理程式會在此欄位中放置自己的名稱。

佇列管理程式轉遞之發佈的 MQMD 設定

當佇列管理程式將發佈資訊傳送給訂閱者時，會使用訊息描述子 (MQMD) 中的這些欄位設定。 MQMD 中的所有其他欄位都設為其預設值。

報告
報告 設為 MQRO_NONE。

MsgType
MsgType 設定為 MQMT_DATAGRAM。

期限
期限 設定為從發佈者收到的 Publish 訊息中的值。 在保留訊息的情況下，未完成的時間會減少訊息已在佇列管理程式的大約時間。

格式
格式 設為 MQFMT_RF_HEADER_2

MsgId
MsgId 設定為唯一值。

CorrelId
如果 CorrelId 是訂閱者身分的一部分，則這是訂閱者在登錄時指定的值。 否則，它是佇列管理程式選擇的非零值。

優先順序
優先順序 會採用發佈者所設定的值，如果發佈者指定 MQPRI_PRIORITY_AS_Q_DEF，則會視為已解決。

持續性
持續性 會採用發佈者所設定的值，如果發佈者已指定 MQPER_PERSISTENCE_AS_Q_DEF，則會視為已解決，除非在此發佈資訊所傳送至的訂閱者的 Register Subscriber 訊息中另有指定。

ReplyToQ
ReplyToQ 設為空白。

回覆目的地佇列管理程式
ReplyTo **佇列管理程式** 設為佇列管理程式的名稱。

UserIdentifier
UserIdentifier 是訂閱者登錄時所設定的使用者 ID。

AccountingToken
AccountingToken 是訂閱者的帳戶記號，在第一次登錄訂閱者時設定。

ApplIdentityData
ApplIdentity **資料** 是訂閱者的應用程式身分資料，如訂閱者第一次登錄時所設定。

PutApplType
PutAppl **類型** 設為 MQAT_BROKER。

PutApplName
PutAppl **名稱** 設為佇列管理程式名稱的前 28 個字元。

PutDate
PutDate 是放置訊息的日期。

PutTime
PutTime 是放置訊息的時間。

ApplOriginData
ApplOrigin **資料** 設為空白。

佇列管理程式回應訊息中的 MQMD 設定

佇列管理程式在傳送發佈訊息的回覆時，會使用訊息描述子 (MQMD) 中的這些欄位設定。 MQMD 中的所有其他欄位都設為其預設值。

報告
報告 設為全部零。

MsgType
MsgType 設為 MQMT_REPLY。

格式
格式 設為 MQFMT_RF_HEADER_2

MsgId
MsgId 的設定取決於原始指令訊息中的 **報告** 選項。 依預設，它會設為 MQMI_NONE ，以便佇列管理程式產生唯一值。

CorrelId
CorrelId 的設定取決於原始指令訊息中的 **報告** 選項。 依預設，這表示 CorrelId 設為與指令訊息的 MsgId 相同的值。 這可用來使指令與其回應產生關聯。

優先順序
優先順序會設為原始指令訊息中的相同值。

持續性
持續性 設為原始指令訊息中所設定的值。

期限
期限 設定為與佇列管理程式所接收原始指令訊息中的值相同。

PutApplType
PutAppl **類型** 設為 MQAT_BROKER。

PutApplName
PutAppl **名稱** 設為佇列管理程式名稱的前 28 個字元。

其他環境定義欄位會設定為如同使用 MQPMO_PASS_IDENTITY_CONTEXT 產生一樣。

機器編碼

本節說明訊息描述子中 *Encoding* 欄位的結構。

如需結構中欄位的摘要，請參閱 第 363 頁的『MQMD-訊息描述子』 。

Encoding 欄位是 32 位元整數，分成四個不同的子欄位; 這些子欄位識別:

- 用於二進位整數的編碼
- 用於聚集十進位整數的編碼
- 用於浮點數字的編碼
- 保留位元

每個子欄位由位元遮罩識別，該遮罩在對應於子欄位的位置中具有 1 位元，而在其他位置中具有 0 位元。 位元會編號為位元 0 是最高有效位元，而位元 31 是最低有效位元。 下列是已定義的遮罩:

MQENC_INTEGER_MASK
二進位整數編碼的遮罩。

此子欄位在 *Encoding* 欄位內佔用位元位置 28 到 31。

MQENC_DECIMAL_MASK
壓縮十進位整數編碼的遮罩。

這個子欄位佔用 *Encoding* 欄位內的位元位置 24 到 27。

MQENC_FLOAT_MASK
浮點數編碼的遮罩。

此子欄位佔用 *Encoding* 欄位內的位元位置 20 到 23。

MQ 保留遮罩
保留位元的遮罩。

這個子欄位在 *Encoding* 欄位內佔據位元位置 0 到 19。

二進位整數編碼

下列值適用於二進位整數編碼:

未定義 MQENC_INTEGER_UNDEFINED
二進位整數使用未定義的編碼來表示。

MQENC_INTEGER_NORMAL
二進位整數以慣用方式表示:

- 數字中最低有效位元組具有數字中任何位元組的最高位址; 最高有效位元組具有最低位址
- 每個位元組中的最低有效位元與具有下一個較高位址的位元組相鄰; 每個位元組中的最高有效位元與具有下一個較低位址的位元組相鄰

已反轉 MQENC_INTEGER_REVERSED
二進位整數的表示方式與 MQENC_INTEGER_NORMAL 相同，但位元組以相反順序排列。 每個位元組內的位元排列方式與 MQENC_INTEGER_NORMAL 相同。

壓縮十進位整數編碼

下列值適用於 packed-decimal-integer 編碼:

未定義 MQENC_DECIMAL_UNDEFINED
使用未定義的編碼來代表聚集十進位整數。

MQENC_DECIMAL_NORMAL
壓縮十進位整數以慣用方式表示:

- 數字可列印格式的每一個十進位數，以聚集十進位數表示，範圍為 X' 0 '到 X' 9' 之間的單一十六進位數字。 每一個十六進位數字會佔用四個位元，因此壓縮十進位數中的每一個位元組都代表數字可列印格式的兩個十進位數。
- 壓縮十進位數中最低有效位元組是包含最低有效十進位數的位元組。 在該位元組內，最高有效的四個位元包含最低有效的十進位數，而最低有效的四個位元包含符號。 符號為 X'C '(正數)、X 'D' (負數) 或 X'F ' (不帶正負號)。
- 數字中的最低有效位元組具有數字中任何位元組的最高位址; 最高有效位元組具有最低位址。
- 每個位元組中的最低有效位元與具有下一個較高位址的位元組相鄰; 每個位元組中的最高有效位元與具有下一個較低位址的位元組相鄰。

MQENC_DECIMAL_REVERED
壓縮十進位整數的表示方式與 MQENC_DECIMAL_NORMAL 相同，但位元組以相反順序排列。 每個位元組內的位元排列方式與 MQENC_DECIMAL_NORMAL 相同。

浮點數編碼

下列值適用於浮點數編碼:

未定義 MQENC_FLOAT_UNDEFINED
浮點數是以未定義的編碼來表示。

MQENC_FLOAT_IEE_NORMAL
使用標準 IEEE 代表浮點數 [3] 浮點數格式，位元組排列如下:

- 假數中最低有效位元組具有數字中任何位元組的最高位址; 包含指數的位元組具有最低位址
- 每個位元組中的最低有效位元與具有下一個較高位址的位元組相鄰; 每個位元組中的最高有效位元與具有下一個較低位址的位元組相鄰

[3] 電氣和電子工程師學會

IEEE 浮點數編碼的詳細資料可在 IEEE 標準 754 中找到。

MQENC_FLOAT_IEE_REVERSED
浮點數字以與 MQENC_FLOAT_IEE_NORMAL 相同的方式表示，但位元組以相反順序排列。 每個位元組內的位元排列方式與 MQENC_FLOAT_IEE_NORMAL 相同。

MQENC_FLOAT_S390
使用標準 System/390 浮點數格式來代表浮點數; 這也由 System/370 使用。

建構編碼

若要在 MQMD 中建構 *Encoding* 欄位的值，說明必要編碼的相關常數可以是:

- 加在一起，或
- 使用位元 OR 運算結合 (如果程式設計語言支援位元運算)

無論使用哪一種方法，都只能結合其中一個 MQENC_INTEGER_ * 編碼與其中一個 MQENC_DECIMAL_ * 編碼及其中一個 MQENC_FLOAT_ * 編碼。

分析編碼

Encoding 欄位包含子欄位; 因此，需要檢查整數、聚集十進位或浮點數編碼的應用程式必須使用所述的其中一項技術。

使用位元作業

如果程式設計語言支援位元作業，請執行下列步驟:

1. 根據所需的編碼類型，選取下列其中一個值:
   - 二進位整數編碼的 MQENC_INTEGER_MASK
   - 用於壓縮十進位整數編碼的 MQENC_DECIMAL_MASK
   - 浮點數編碼的 MQENC_FLOAT_MASK

   呼叫值 A。
2. 使用位元 AND 運算將 *Encoding* 欄位與 A 結合; 呼叫結果 B。
3. B 是必要的編碼，可以測試每一個適用於該編碼類型的值是否相等。

使用算術

如果程式設計語言 不 支援位元作業，請使用整數算術執行下列步驟:

1. 根據所需的編碼類型，選取下列其中一個值:
   - 1 代表二進位整數編碼
   - 16 表示聚集十進位整數編碼
   - 256 表示浮點數編碼

   呼叫值 A。
2. 將 *Encoding* 欄位的值除以 A; 呼叫結果 B。
3. B 除以 16; 呼叫結果 C。
4. 將 C 乘以 16 並減去 B; 呼叫結果 D。
5. 將 D 乘以 A; 呼叫結果 E。
6. E 是必要的編碼，可以測試每一個適用於該編碼類型的值是否相等。

機器架構編碼的摘要

第 788 頁的表 578 中顯示機器架構的編碼。

表 *578:* 機器架構的編碼摘要			
機器架構	**二進位整數編碼**	**壓縮十進位整數編碼**	**浮點數編碼**
IBM i	正常	正常	IEEE 正常
Intel x86	反向	反向	IEEE 反轉
PowerPC	正常	正常	IEEE 正常
System/390	正常	正常	System/390

報告選項及訊息旗標

本節說明 *Report* 及 *MsgFlags* 欄位，這些欄位是 MQGET、MQPUT 及 MQPUT1 呼叫所指定訊息描述子 MQMD 的一部分。

本節中的主題說明:

- 報告欄位的結構以及佇列管理程式處理它的方式
- 應用程式分析報告欄位的方式
- message-flags 欄位的結構

如需 MQMD 訊息描述子的相關資訊，請參閱 第 363 頁的『MQMD-訊息描述子』。

報告欄位的結構

此資訊說明報告欄位的結構。

Report 欄位是一個 32 位元整數，分成三個不同的子欄位。 這些子欄位識別:

- 本端佇列管理程式無法辨識時被拒絕的報告選項
- 一律接受的報告選項，即使本端佇列管理程式無法辨識它們也一樣
- 只有在滿足某些其他條件時才接受的報告選項

每個子欄位由位元遮罩識別，該遮罩在對應於子欄位的位置中具有 1 位元，而在其他位置中具有 0 位元。 子欄位中的位元不一定相鄰。 位元會編號為位元 0 是最高有效位元，而位元 31 是最低有效位元。 下列是定義來識別子欄位的遮罩:

MQRO_REJECT_UNSUPP_MASK
此遮罩會識別 *Report* 欄位內的位元位置，其中本端佇列管理程式不支援的報告選項會導致 MQPUT 或 MQPUT1 呼叫失敗，完成碼為 MQCC_FAILED ，原因碼為 MQRC_REPORT_OPTIONS_ERROR。

這個子欄位會佔用位元位置 3 ， 以及 11 到 13。

MQRO_ACCEPT_UNSUPP_MASK
此遮罩可識別 *Report* 欄位內的位元位置，其中在 MQPUT 或 MQPUT1 呼叫中仍接受本端佇列管理程式不支援的報告選項。 在此情況下會傳回完成碼 MQCC_WARNING ， 原因碼為 MQRC_UNKNOWN_REPORT_OPTION。

這個子欄位會佔用位元位置 0 到 2、4 到 10 ， 以及 24 到 31。

此子欄位包含下列報告選項:

- 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_UNSUPP_IF_XMIT_MASK
此遮罩可識別 *Report* 欄位內的位元位置，在此欄位中，仍然接受 MQPUT 或 MQPUT1 呼叫本端佇列管理程式不支援的報告選項，前提是 滿足下列兩個條件:

- 訊息以遠端佇列管理程式為目的地。
- 應用程式不會將訊息直接放置在本端傳輸佇列上 (亦即，在 MQOPEN 或 MQPUT1 呼叫上指定的物件描述子中， *ObjectQMgrName* 及 *ObjectName* 欄位所識別的佇列不是本端傳輸佇列)。

如果滿足這些條件，則會傳回具有原因碼 MQRC_UNKNOWN_REPORT_OPTION 的完成碼 MQCC_WARNING，如果不滿足，則會傳回具有原因碼 MQRC_REPORT_OPTIONS_ERROR 的 MQCC_FAILED。

此子欄位佔用位元位置 14 到 23。

此子欄位包含下列報告選項:

- MQRO_COA
- MQRO_COA_WITH_DATA
- MQ Ro_COA_WITH_FULL_DATA
- MQRO_COD
- MQRO_COD_WITH_DATA
- MQRO_COD_WITH_FULL_DATA

如果在 *Report* 欄位中指定佇列管理程式無法辨識的任何選項，則佇列管理程式會依序使用位元 AND 運算，將 *Report* 欄位與該子欄位的遮罩結合起來，來檢查每一個子欄位。 如果該作業的結果不是零，則會傳回上述完成碼及原因碼。

如果傳回 MQCC_WARNING，則未定義存在其他警告狀況時傳回的原因碼。

當傳送含有 遠端 佇列管理程式所辨識及處理之報告選項的訊息時，指定及接受本端佇列管理程式無法辨識的報告選項的能力非常有用。

分析報告欄位

Report 欄位包含子欄位; 因此，需要檢查訊息傳送者是否要求特定報告的應用程式必須使用所述的其中一項技術。

使用位元作業

如果程式設計語言支援位元作業，請執行下列步驟:

1. 根據要檢查的報告類型，選取下列其中一個值:
   - COA 報告的 MQRO_COA_WITH_FULL_DATA
   - COD 報告的 MQRO_COD_WITH_FULL_DATA
   - 異常狀況報告的 MQRO_EXCEPTION_WITH_FULL_DATA
   - 到期報告的 MQRO_EXPIRATION_WITH_FULL_DATA

   呼叫值 A。

在 z/OS 上，請使用 MQRO_ * _WITH_DATA 值而非 MQRO_ * _WITH_FULL_DATA 值。

2. 使用位元 AND 運算將 *Report* 欄位與 A 結合; 呼叫結果 B。
3. 測試 B 是否具有該報告類型所可能使用的每一個值的相等性。

   例如，如果 A 是 MQRO_EXCEPTION_WITH_FULL_DATA ，請測試 B 是否與下列每一項相等，以判定訊息傳送者指定的內容:

   - MQRO_NONE
   - MQRO_Exception
   - MQRO_EXCEPTION_WITH_DATA
   - MQRO_EXCEPTION_WITH_FULL_DATA

   對於應用程式邏輯，可以用最方便的任何順序來執行測試。

請使用類似方法來測試 MQRO_PASS_MSG_ID 或 MQRO_PASS_CORREL_ID 選項; 選取這兩個常數中的任何一個作為值 A ，然後依照上述說明繼續進行。

使用算術

如果程式設計語言 不 支援位元作業，請使用整數算術執行下列步驟:

1. 根據要檢查的報告類型，選取下列其中一個值:

   - COA 報告的 MQRO_COA
   - COD 報告的 MQRO_COD
   - 異常狀況報告的 MQRO_EXCEPTION
   - 到期報告的 MQRO_EXPIRATION

   呼叫值 A。
2. 將 *Report* 欄位除以 A; 呼叫結果 B。
3. B 除以 8; 呼叫結果 C。
4. 將 C 乘以 8 並減去 B; 將結果稱為 D。
5. 將 D 乘以 A; 呼叫結果 E。
6. 測試 E 是否具有該報告類型所可能使用的每一個值的相等性。

   例如，如果 A 為 MQRO_EXCEPTION ，請測試 E 是否與下列各項相等，以判定訊息傳送者指定的內容:

   - MQRO_NONE
   - MQRO_Exception
   - MQRO_EXCEPTION_WITH_DATA
   - MQRO_EXCEPTION_WITH_FULL_DATA

   對於應用程式邏輯，可以用最方便的任何順序來執行測試。

下列虛擬碼說明異常狀況報告訊息的這項技術:

```
A = MQRO_EXCEPTION
B = Report/A
C = B/8
D = B - C*8
E = D*A
```

使用類似方法來測試 MQRO_PASS_MSG_ID 或 MQRO_PASS_CORREL_ID 選項; 選取 A 這兩個常數中的任何一個適當的值，然後依照上述說明繼續進行，但將上述步驟中的值 8 取代為值 2。

message-flags 欄位的結構

此資訊說明訊息旗標欄位的結構。

MsgFlags 欄位是一個 32 位元整數，分成三個不同的子欄位。 這些子欄位識別:

- 本端佇列管理程式無法辨識拒絕的訊息旗標
- 一律接受的訊息旗標，即使本端佇列管理程式無法辨識它們
- 只有在滿足某些其他條件時才接受的訊息旗標

註: *MsgFlags* 中的所有子欄位都保留給佇列管理程式使用。

每個子欄位由位元遮罩識別，該遮罩在對應於子欄位的位置中具有 1 位元，而在其他位置中具有 0 位元。 位元會編號為位元 0 是最高有效位元，而位元 31 是最低有效位元。 下列是定義來識別子欄位的遮罩:

MQMF_REJECT_UNSUPP_MASK
此遮罩會識別 *MsgFlags* 欄位內的位元位置，其中本端佇列管理程式不支援的訊息旗標會導致 MQPUT 或 MQPUT1 呼叫失敗，完成碼為 MQCC_FAILED ，原因碼為 MQRC_MSG_FLAGS_ERROR。

這個子欄位會佔用位元位置 20 到 31。

此子欄位包含下列訊息旗標:

- MQMF_LAST_MSG_IN_GROUP
- MQMF_LAST_SEGMENT
- MQMF_MSG_IN_GROUP
- MQMF_SEGMENT
- 容許 MQMF_SEGMENTATION_ALLOWED
- MQMF_SEGMENTATION_INHIBITED

MQMF_ACCEPT_UNSUPP_MASK
此遮罩可識別 *MsgFlags* 欄位內的位元位置，在此欄位中， MQPUT 或 MQPUT1 呼叫仍接受本端佇列管理程式不支援的訊息旗標。 完成碼為 MQCC_OK。

這個子欄位會佔用位元位置 0 到 11。

MQMF_ACCEPT_UNSUPP_IF_XMIT_MASK
此遮罩可識別 *MsgFlags* 欄位內的位元位置，其中在 MQPUT 或 MQPUT1 呼叫 前提 滿足下列兩個條件時，仍會接受本端佇列管理程式不支援的訊息旗標:

- 訊息以遠端佇列管理程式為目的地。
- 應用程式不會將訊息直接放置在本端傳輸佇列上 (亦即，在 MQOPEN 或 MQPUT1 呼叫上指定的物件描述子中， *ObjectQMgrName* 及 *ObjectName* 欄位所識別的佇列不是本端傳輸佇列)。

如果滿足這些條件，則會傳回完成碼 MQCC_OK ，如果未滿足，則會傳回 MQCC_FAILED ，原因碼為 MQRC_MSG_FLAGS_ERROR。

這個子欄位會佔用位元位置 12 到 19。

如果在 *MsgFlags* 欄位中指定佇列管理程式無法辨識的旗標，則佇列管理程式會依序使用位元 AND 運算來結合 *MsgFlags* 欄位與該子欄位的遮罩，以檢查每一個子欄位。 如果該作業的結果不是零，則會傳回上述完成碼及原因碼。

資料轉換結束程式

這個主題集合說明資料轉換結束程式的介面，以及需要資料轉換時佇列管理程式所執行的處理程序。

如需資料轉換的相關資訊，請參閱 *WebSphere MQ* 下的資料轉換 ，網址為 https://www.ibm.com/support/docview.wss?uid=swg27005729。

在處理 MQGET 呼叫時，會呼叫資料轉換結束程式，以便將應用程式訊息資料轉換成接收端應用程式所需的表示法。 應用程式訊息資料的轉換是選用的; 它需要在 MQGET 呼叫上指定 MQGMO_CONVERT 選項。

說明下列主題:

- 佇列管理程式為回應 MQGMO_CONVERT 選項而執行的處理; 請參閱 第 792 頁的『轉換處理』。
- 佇列管理程式在處理內建格式時所使用的處理慣例; 這些慣例也建議用於使用者撰寫的結束程式。 請參閱 第 793 頁的『處理慣例』。

- 轉換報告訊息的特殊考量; 請參閱 第 796 頁的『報告訊息的轉換』。
- 傳遞至資料轉換結束程式的參數; 請參閱 第 807 頁的『MQ_DATA_CONV_EXIT-資料轉換結束程式』。
- 可從結束程式用來在不同表示法之間轉換字元資料的呼叫; 請參閱 第 802 頁的『MQXCNVC-轉換字元』。
- 特定於結束程式的資料結構參數; 請參閱 第 796 頁的『MQDXP-資料轉換結束程式參數』。

轉換處理

此資訊說明佇列管理程式為回應 MQGMO_CONVERT 選項所執行的處理。

如果在 MQGET 呼叫中指定 MQGMO_CONVERT 選項，且有訊息要傳回給應用程式，則佇列管理程式會執行下列動作:

1. 如果下列一或多項為真，則不需要轉換:
   - 訊息資料已在發出 MQGET 呼叫的應用程式所需的字集及編碼中。 在發出呼叫之前，應用程式必須將 MQGET 呼叫的 *MsgDesc* 參數中的 *CodedCharSetId* 和 *Encoding* 欄位設為必要值。
   - 訊息資料的長度為零。
   - MQGET 呼叫的 *Buffer* 參數長度為零。

   在這些情況下，會傳回訊息，但不會轉換至發出 MQGET 呼叫的應用程式; *MsgDesc* 參數中的 *CodedCharSetId* 及 *Encoding* 值會設為訊息中控制資訊的值，且呼叫會完成，並具有下列其中一個完成碼及原因碼組合:

完成碼	原因碼
MQCC_OK	MQRC_NONE
MQCC_WARNING	MQRC_TRUNCATED_MSG_ACCEPTED
MQCC_WARNING	MQRC_TRUNCATED_MSG_FAILED

   只有在訊息資料的字集或編碼不同於 *MsgDesc* 參數中的對應值，且有資料要轉換時，才會執行下列步驟:
2. 如果訊息中控制資訊的 *Format* 欄位值為 MQFMT_NONE ，則會傳回未轉換的訊息，完成碼為 MQCC_WARNING ，原因碼為 MQRC_FORMAT_ERROR。

   在所有其他情況下，轉換處理繼續進行。
3. 訊息會從佇列中移除，並放置在與 *Buffer* 參數大小相同的暫時緩衝區中。 對於瀏覽作業，訊息會複製到暫時緩衝區中，而不是從佇列中移除。
4. 如果訊息必須截斷以適合緩衝區，則會執行下列動作:
   - 如果 未 指定 MQGMO_ACCEPT_TRUNCATED_MSG 選項，則會傳回未轉換的訊息，完成碼為 MQCC_WARNING ，原因碼為 MQRC_TRUNCATED_MSG_FAILED。
   - 如果指定 MQGMO_ACCEPT_TRUNCATED_MSG 選項 是 ，則完成碼會設為 MQCC_WARNING ，原因碼會設為 MQRC_TRUNCATED_MSG_ACCEPTED ，且轉換處理繼續進行。
5. 如果訊息可以容納在緩衝區中而不截斷，或已指定 MQGMO_ACCEPT_TRUNCATED_MSG 選項，則會執行下列動作:
   - 如果格式是內建格式，則緩衝區會傳遞至佇列管理程式的資料轉換服務。
   - 如果格式不是內建格式，則會將緩衝區傳遞給與格式同名的使用者撰寫結束程式。 如果找不到結束程式，則會傳回未轉換的訊息，完成碼為 MQCC_WARNING ，原因碼為 MQRC_FORMAT_ERROR。

   如果未發生任何錯誤，則來自資料轉換服務或來自使用者撰寫結束程式的輸出是已轉換的訊息，加上要傳回給發出 MQGET 呼叫之應用程式的完成碼及原因碼。
6. 如果轉換成功，佇列管理程式會將已轉換的訊息傳回應用程式。 在此情況下， MQGET 呼叫所傳回的完成碼和原因碼是下列其中一種組合:

完成碼	原因碼
MQCC_OK	MQRC_NONE

完成碼	原因碼
MQCC_WARNING	MQRC_TRUNCATED_MSG_ACCEPTED

不過，如果由使用者撰寫的結束程式執行轉換，則即使轉換成功，也可以傳回其他原因碼。

如果轉換失敗，佇列管理程式會將未轉換的訊息傳回應用程式，並將 *MsgDesc* 參數中的 *CodedCharSetId* 及 *Encoding* 欄位設為訊息中控制資訊的值，且完成碼為 MQCC_WARNING。

處理慣例

轉換內建格式時，佇列管理程式會遵循說明的處理慣例。

使用者撰寫的結束程式也應該遵循這些慣例，雖然佇列管理程式未強制執行此慣例。 佇列管理程式所轉換的內建格式如下:

- MQFMT_ADMIN
- MQFMT_CICS (僅限 z/OS)
- MQFMT_COMMAND_1
- MQFMT_COMMAND_2
- MQFMT_DEAD_LETTER_HEADER
- MQFMT_DIST_HEADER
- MQFMT_EVENT 第 1 版
- MQFMT_EVENT 第 2 版
- 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 (僅限 z/OS)
- MQFMT_XMIT_Q_HEADER

1. 如果訊息在轉換期間展開，且超出 *Buffer* 參數的大小，則會執行下列動作:
   - 如果 未 指定 MQGMO_ACCEPT_TRUNCATED_MSG 選項，則會傳回未轉換的訊息，完成碼為 MQCC_WARNING ，原因碼為 MQRC_CONVERTED_MSG_TOO_BIG。
   - 如果指定 MQGMO_ACCEPT_TRUNCATED_MSG 選項 是 ，則會截斷訊息，完成碼會設為 MQCC_WARNING ，原因碼會設為 MQRC_TRUNCATED_MSG_ACCEPTED ，並繼續執行轉換處理程序。
2. 如果發生截斷 (在轉換之前或轉換期間) ，則 *Buffer* 參數中傳回的有效位元組數可以 小於 緩衝區長度。

   例如，如果 4 位元組整數或 DBCS 字元跨緩衝區結尾，則會發生這種情況。 不轉換資訊的不完整元素，且所傳回訊息中的那些位元組不包含有效資訊。 如果在轉換期間轉換之前被截斷的訊息收縮，也會發生此情況。

   如果傳回的有效位元組數小於緩衝區的長度，則緩衝區結尾的未用位元組會設為空值。
3. 如果陣列或字串橫跨緩衝區結尾，則會盡可能轉換資料; 只會轉換不完整的特定陣列元素或 DBCS 字元; 會轉換之前的陣列元素或字元。

4. 如果發生截斷 (在轉換之前或轉換期間)，則 *DataLength* 參數傳回的長度是截斷之前 未轉換 訊息的長度。
5. 在單位元組字集 (SBCS)、雙位元組字集 (DBCS) 或多位元組字集 (MBCS) 之間轉換字串時，字串可以展開或縮小。
   - 在 PCF 格式 MQFMT_ADMIN、MQFMT_EVENT 及 MQFMT_PCF 中， MQCFST 及 MQCFSL 結構中的字串會根據需要展開或收縮，以在轉換之後容納字串。

     對於字串清單結構 MQCFSL，清單中的字串可能會展開或收縮不同數量。 如果發生這種情況，佇列管理程式會以空白來填補較短的字串，使其長度與轉換後最長的字串相同。
   - 在 MQFMT_REF_MSG_HEADER 格式中， *SrcEnvOffset*、 *SrcNameOffset*、 *DestEnvOffset* 和 *DestNameOffset* 欄位所處理的字串會視需要展開或縮小，以在轉換之後容納字串。
   - 在格式 MQFMT_RF_HEADER 中， *NameValueString* 欄位會根據需要展開或縮小，以在轉換之後容納名稱/值配對。
   - 在具有固定欄位大小的結構中，佇列管理程式容許字串在其固定欄位內展開或縮小，前提是不會遺失任何重要資訊。 在這方面，欄位中第一個空值字元之後的尾端空白和字元會被視為不顯著。
     - 如果字串展開，但只需要捨棄無意義字元即可在欄位中容納已轉換的字串，則轉換會成功，且呼叫會完成，並傳回 MQCC_OK 及原因碼 MQRC_NONE (假設沒有其他錯誤)。
     - 如果字串展開，但已轉換的字串需要捨棄有效字元才能放入欄位中，則會傳回未轉換的訊息，且呼叫會完成， MQCC_WARNING 及原因碼 MQRC_CONVERTED_STRING_TOO_BIG。

       註: 在此情況下，不論是否指定 MQGMO_ACCEPT_TRUNCATED_MSG 選項，都會產生原因碼 MQRC_CONVERTED_string_too_big 結果。
     - 如果字串合約，佇列管理程式會以空白填補字串到欄位的長度。
6. 對於由一個以上 MQ 標頭結構後面接著使用者資料所組成的訊息，可能會轉換一個以上標頭結構，但不會轉換其餘訊息。 不過 (有兩個例外)，每一個標頭結構中的 *CodedCharSetId* 和 *Encoding* 欄位一律正確指出遵循標頭結構之資料的字集和編碼。

   兩個例外是 MQCIH 及 MQIIH 結構，其中 *CodedCharSetId* 及 *Encoding* 欄位中的值並不顯著。 對於那些結構，結構後面的資料與 MQCIH 或 MQIIH 結構本身使用相同的字集及編碼。
7. 如果所擷取訊息的控制資訊中或 *MsgDesc* 參數中的 *CodedCharSetId* 或 *Encoding* 欄位指定未定義或不支援的值，則在轉換訊息時不需要使用未定義或不受支援的值時，佇列管理程式可能會忽略錯誤。

   例如，如果訊息中的 *Encoding* 欄位指定不受支援的浮點數編碼，但訊息只包含整數資料，或包含不需要轉換的浮點數資料 (因為來源與目標浮點數編碼相同)，則可能不會診斷錯誤。

   如果診斷錯誤，則會傳回未轉換的訊息，完成碼為 MQCC_WARNING，且其中一個 MQRC_SOURCE_ * _ERROR 或 MQRC_TARGET_ * _ERROR 原因碼 (適當的話); *MsgDesc* 參數中的 *CodedCharSetId* 及 *Encoding* 欄位會設為訊息中控制資訊的值。

   如果未診斷錯誤且轉換順利完成，則在 *MsgDesc* 參數的 *CodedCharSetId* 及 *Encoding* 欄位中傳回的值，是由發出 MQGET 呼叫的應用程式指定的值。
8. 無論如何，如果訊息傳回至未轉換的應用程式，則完成碼會設為 MQCC_WARNING，且 *MsgDesc* 參數中的 *CodedCharSetId* 及 *Encoding* 欄位會設為適用於未轉換資料的值。 這也針對 MQFMT_NONE 執行。

   Reason 參數設為代碼，指出無法執行轉換的原因，除非訊息也必須截斷; 與截斷相關的原因碼優先於與轉換相關的原因碼。 (若要判斷是否已轉換截斷訊息，請檢查 *MsgDesc* 參數中 *CodedCharSetId* 及 *Encoding* 欄位所傳回的值。)

   診斷錯誤時，會傳回特定原因碼或一般原因碼 MQRC_NOT_CONVERTED。 傳回的原因碼取決於基礎資料轉換服務的診斷功能。
9. 如果傳回完成碼 MQCC_WARNING，且有多個相關原因碼，則優先順序如下:
   a. 下列原因優先於所有其他原因; 只會出現此群組中的其中一個原因:
      - 已接受 MQRC_SIGNAL_REQUEST_ACCEPTED
      - MQRC_TRUNCATED_MSG_ACCEPTED

   b. 未定義其餘原因碼內的優先順序。

10. MQGET 呼叫完成時:
    - 下列原因碼指出已順利轉換訊息:
      - MQRC_NONE
    - 下列原因碼指出 可能 已順利轉換訊息 (請檢查 *MsgDesc* 參數中的 *CodedCharSetId* 及 *Encoding* 欄位以找出):
      - MQ RC_MSG_MARKED_BROWSE_CO_OP
      - MQRC_TRUNCATED_MSG_ACCEPTED
    - 所有其他原因碼都指出未轉換訊息。

    下列處理程序特定於內建格式; 它不適用於使用者定義的格式:
11. 除了下列格式之外:
    - MQFMT_ADMIN
    - MQFMT_COMMAND_1
    - MQFMT_COMMAND_2
    - MQFMT_EVENT
    - MQFMT_IMS_VAR_STRING
    - MQFMT_PCF
    - MQFMT_STRING

    沒有任何內建格式可以從佇列名稱中有效的字元轉換或轉換成沒有 SBCS 字元的字集。 如果嘗試執行這類轉換，則會傳回未轉換的訊息，並適當地傳回完成碼 MQCC_WARNING 及原因碼 MQRC_SOURCE_CCSID_ERROR 或 MQRC_TARGET_CCSID_ERROR。

    Unicode 字集 UCS-2 是佇列名稱中有效字元沒有 SBCS 字元的字集範例。
12. 如果已截斷內建格式的訊息資料，則 不會 調整訊息中包含字串長度或元素或結構計數的欄位，以反映實際傳回應用程式的資料長度; 訊息資料中針對此類欄位傳回的值是適用於訊息 截斷之前的值。

    處理訊息 (例如截斷的 MQFMT_ADMIN 訊息) 時，請確定應用程式不會嘗試存取所傳回資料結尾以外的資料。
13. 如果格式名稱為 MQFMT_DEAD_LETTER_HEADER ，則訊息資料會以 MQDLH 結構開頭，後面可能接著零個以上位元組的應用程式訊息資料。 應用程式訊息資料的格式、字集及編碼是由訊息開頭 MQDLH 結構中的 *Format*、 *CodedCharSetId* 及 *Encoding* 欄位所定義。 因為 MQDLH 結構及應用程式訊息資料可以具有不同的字集及編碼，所以 MQDLH 結構及應用程式訊息資料中的一個、另一個或兩者可能需要轉換。

    必要的話，佇列管理程式會先轉換 MQDLH 結構。 如果轉換成功，或 MQDLH 結構不需要轉換，則佇列管理程式會檢查 MQDLH 結構中的 *CodedCharSetId* 及 *Encoding* 欄位，以查看是否需要轉換應用程式訊息資料。 如果 需要 轉換，佇列管理程式會以 MQDLH 結構中 *Format* 欄位提供的名稱來呼叫使用者撰寫的結束程式，或自行執行轉換 (如果 *Format* 是內建格式的名稱)。

    如果 MQGET 呼叫傳回完成碼 MQCC_WARNING ，且原因碼是指出轉換不成功的原因碼之一，則適用下列其中一項:
    - 無法轉換 MQDLH 結構。 在此情況下，也不會轉換應用程式訊息資料。
    - 已轉換 MQDLH 結構，但未轉換應用程式訊息資料。

    應用程式可以檢查 *MsgDesc* 參數中 *CodedCharSetId* 及 *Encoding* 欄位中傳回的值，以及 MQDLH 結構中傳回的值，以判定上述哪些值適用。
14. 如果格式名稱為 MQFMT_XMIT_Q_HEADER ，則訊息資料會以 MQXQH 結構開頭，後面可能接著零個以上位元組的其他資料。 這個額外資料通常是應用程式訊息資料 (長度可能為零) ，但在額外資料開始時，也可能有一或多個進一步的 MQ 標頭結構存在。

    MQXQH 結構必須採用佇列管理程式的字集及編碼。 MQXQH 結構後面的資料的格式、字集及編碼是由 MQXQH 中 所包含的 MQMD 結構中的 *Format*、 *CodedCharSetId* 及 *Encoding* 欄位所提供。 對於

後續呈現的每一個 MQ 標頭結構，結構中的 *Format*、 *CodedCharSetId* 及 *Encoding* 欄位會說明遵循該結構的資料; 該資料是另一個 MQ 標頭結構或應用程式訊息資料。

如果為 MQFMT_XMIT_Q_HEADER 訊息指定 MQGMO_CONVERT 選項，則會轉換應用程式訊息資料及某些 MQ 標頭結構， 但 *MQXQH* 結構中的資料不是。 從 MQGET 呼叫傳回時，因此:

- *MsgDesc* 參數中 *Format*、 *CodedCharSetId* 及 *Encoding* 欄位的值說明 MQXQH 結構中的資料，而 不 說明應用程式訊息資料; 因此這些值與發出 MQGET 呼叫的應用程式所指定的值 不 相同。

  其效果是在每一個 MQGET 呼叫之前，反覆地從傳輸佇列取得訊息並指定 MQGMO_CONVERT 選項的應用程式必須將 *MsgDesc* 參數中的 *CodedCharSetId* 及 *Encoding* 欄位重設為應用程式訊息資料所需的值。
- 最後一個 MQ 標頭結構中的 *Format*、 *CodedCharSetId* 及 *Encoding* 欄位值說明應用程式訊息資料。 如果沒有其他 MQ 標頭結構，則 MQXQH 結構內 MQMD 結構中的這些欄位會說明應用程式訊息資料。 如果轉換成功，則值將與發出 MQGET 呼叫的應用程式在 *MsgDesc* 參數中指定的值相同。

如果訊息是配送清單訊息， MQXQH 結構後面會接著 MQDH 結構 (加上其 MQOR 及 MQPMR 記錄的陣列) ，接著可能會接著零個以上的進一步 MQ 標頭結構，以及零個以上位元組的應用程式訊息資料。 與 MQXQH 結構一樣， MQDH 結構必須採用佇列管理程式的字集及編碼，而且即使指定 MQGMO_CONVERT 選項，也不會在 MQGET 呼叫上轉換它。

上述 MQXQH 及 MQDH 結構的處理主要供訊息通道代理程式從傳輸佇列取得訊息時使用。

報告訊息的轉換

一般而言，根據原始訊息傳送者指定的報告選項，報告訊息可以包含不同數量的應用程式訊息資料。 不過，活動報告可以包含資料，但沒有在常數中提及 * _WITH_DATA 的報告選項。

尤其是報告訊息可以包含下列其中一項:

1. 沒有應用程式訊息資料
2. 原始訊息中的部分應用程式訊息資料

   當原始訊息的傳送者指定 MQRO_ * _WITH_DATA 且訊息長度超過 100 個位元組時，即會發生此情況。
3. 原始訊息中的所有應用程式訊息資料

   當原始訊息的傳送者指定 MQRO_ * _WITH_FULL_DATA ，或指定 MQRO_ * _WITH_DATA 且訊息為 100 位元組或更短時，即會發生此情況。

當佇列管理程式或訊息通道代理程式產生報告訊息時，它會將格式名稱從原始訊息複製到報告訊息中控制資訊的 *Format* 欄位。 因此，報告訊息中的格式名稱可能暗示資料長度不同於報告訊息中實際存在的長度 (上述案例 1 和 2)。

如果在擷取報告訊息時指定 MQGMO_CONVERT 選項:

- 對於上述案例 1 ，不會呼叫資料轉換結束程式 (因為報告訊息沒有資料)。
- 對於上述案例 3 ，格式名稱正確地暗示訊息資料的長度。
- 但對於上述案例 2 ，會呼叫資料轉換結束程式，以轉換 短於 格式名稱所隱含長度的訊息。

  此外，傳給結束程式的原因碼通常是 MQRC_NONE (也就是說，原因碼並不表示訊息已截斷)。 這是因為訊息資料已被報告訊息的 傳送端 截斷，而不是由接收端的佇列管理程式截斷，以回應 MQGET 呼叫。

由於這些可能性，資料轉換結束程式 不得 使用格式名稱來推斷傳給它的資料長度; 相反地，結束程式必須檢查提供的資料長度，並準備轉換 小於 格式名稱所隱含的資料長度。 如果資料可以順利轉換，結束程式必須傳回完成碼 MQCC_OK 和原因碼 MQRC_NONE。 要轉換的訊息資料長度會以 *InBufferLength* 參數傳遞至結束程式。

產品相關程式設計介面

MQDXP-資料轉換結束程式參數

當在處理 MQGET 呼叫的過程中呼叫結束程式來轉換訊息資料時， MQDXP 結構是佇列管理程式傳遞至資料轉換結束程式的參數。 如需資料轉換結束程式的詳細資料，請參閱 MQ_DATA_CONV_EXIT 呼叫的說明。

MQDXP 中的字元資料是本端佇列管理程式的字集; 這是由 *CodedCharSetId* 佇列管理程式屬性所提供。 MQDXP 中的數值資料採用原生機器編碼; 這是由 MQENC_NATIVE 提供。

結束程式只能變更 MQDXP 中的 *DataLength*、 *CompCode*、 *Reason* 及 *ExitResponse* 欄位; 其他欄位的變更會被忽略。 不過，如果要轉換的訊息是只包含部分邏輯訊息的區段，則 *DataLength* 欄位 無法 變更。

當控制從結束程式回到佇列管理程式時，佇列管理程式會檢查 MQDXP 中傳回的值。 如果傳回的值無效，佇列管理程式會繼續處理，如同結束程式在 *ExitResponse* 中已傳回 MQXDR_CONVERSION_FAILED 一樣; 不過，在此情況下，佇列管理程式會忽略結束程式所傳回 *CompCode* 及 *Reason* 欄位的值，並改為使用這些欄位在結束程式 輸入 上的值。 MQDXP 中的下列值會導致進行此處理:

- *ExitResponse* 欄位不是 MQXDR_OK ， 也不是 MQXDR_CONVERSION_FAILED
- *CompCode* 欄位不是 MQCC_OK 也不是 MQCC_WARNING
- *DataLength* 欄位小於零，或 *DataLength* 欄位在轉換的訊息是只包含部分邏輯訊息的區段時變更。

下表彙總結構中的欄位。

表 *579: MQDXP* 中的欄位

欄位	說明	主題
StrucId	結構 ID	StrucId
Version	結構版本號碼	版本
AppOptions	應用程式選項	AppOptions
Encoding	應用程式所需的數值編碼	編碼
CodedCharSetId	應用程式所需的字集	CodedCharSetId
DataLength	訊息資料的長度 (以位元組為單位)	DataLength
CompCode	完成碼	CompCode
Reason	原因碼限定 *CompCode*	原因
ExitResponse	結束的回應	ExitResponse
Hconn	連線控點	Hconn
pEntryPoints	MQIEP 結構的位址	pEntry 點

欄位

MQDXP 結構包含下列欄位; 這些欄位按字母順序說明。

AppOptions
類型 :MQLONG

這是發出 MQGET 呼叫之應用程式所指定 MQGMO 結構的 *Options* 欄位副本。 結束程式可能需要檢查這些，以確定是否已指定 MQGMO_ACCEPT_TRUNCATED_MSG 選項。

這是結束程式的輸入欄位。

CodedCharSetId
類型 :MQLONG

這是發出 MQGET 呼叫之應用程式所需的字集編碼字集 ID; 如需詳細資料，請參閱 MQMD 結構中的 *CodedCharSetId* 欄位。 如果應用程式在 MQGET 呼叫中指定特殊值 MQCCSI_Q_MGR ，則在呼叫結束程式之前，佇列管理程式會將此變更為佇列管理程式所使用字集的實際字集 ID。

如果轉換成功，結束程式必須將此複製到訊息描述子中的 *CodedCharSetId* 欄位。

這是結束程式的輸入欄位。

CompCode
類型 :MQLONG

當呼叫結束程式時，如果結束程式未執行任何動作，則會包含傳回給發出 MQGET 呼叫之應用程式的完成碼。 它一律是 MQCC_WARNING ，因為訊息已截斷，或訊息需要轉換，但尚未執行此動作。

在結束程式的輸出上，此欄位包含要在 MQGET 呼叫的 *CompCode* 參數中傳回給應用程式的完成碼; 只有 MQCC_OK 和 MQCC_WARNING 有效。 如需結束程式如何在輸出上設定此欄位的建議，請參閱 *Reason* 欄位的說明。

這是結束程式的輸入/輸出欄位。

DataLength
類型 :MQLONG

當呼叫結束程式時，此欄位包含應用程式訊息資料的原始長度。 如果訊息被截斷以適合應用程式所提供的緩衝區，則提供給結束程式的訊息大小 小於 *DataLength* 的值。 提供給結束程式的訊息大小一律由結束程式的 *InBufferLength* 參數提供，而不論發生任何截斷。

在結束程式的輸入上，具有值 MQRC_TRUNCATED_MSG_ACCEPTED 的 *Reason* 欄位會指出截斷。

大部分轉換都不需要變更此長度，但結束程式可以在必要時變更此長度; 結束程式所設定的值會在 MQGET 呼叫的 *DataLength* 參數中傳回給應用程式。 不過，如果要轉換的訊息是只包含部分邏輯訊息的區段，則 無法 變更此長度。 這是因為變更長度會導致邏輯訊息中後續區段的偏移不正確。

請注意，如果結束程式想要變更資料的長度，請注意佇列管理程式已根據 未轉換 資料的長度決定訊息資料是否放入應用程式的緩衝區中。 此決策決定是否從佇列中移除訊息 (或針對瀏覽要求移動瀏覽游標) ，且不會受到轉換對資料長度所造成的任何變更影響。 基於此原因，建議轉換結束程式不要造成應用程式訊息資料長度的變更。

如果字元轉換確實暗示長度變更，則可以將字串轉換成另一個長度相同的字串 (以位元組為單位)、截斷尾端空白，或視需要以空白填補。

如果訊息未包含應用程式訊息資料，則不會呼叫結束程式; 因此 *DataLength* 一律大於零。

這是結束程式的輸入/輸出欄位。

Encoding
類型 :MQLONG

應用程式所需的數值編碼。

這是發出 MQGET 呼叫的應用程式所需要的數值編碼; 如需詳細資料，請參閱 MQMD 結構中的 *Encoding* 欄位。

如果轉換成功，結束程式會將此複製到訊息描述子中的 *Encoding* 欄位。

這是結束程式的輸入欄位。

ExitOptions
類型 :MQLONG

這是保留欄位; 其值為 0。

ExitResponse
類型 :MQLONG

來自結束程式的回應。由結束程式設定，以指出轉換是否成功。 它必須是下列其中一項:

MQXDR_OK
轉換成功。

如果結束程式指定此值，佇列管理程式會將下列項目傳回給發出 MQGET 呼叫的應用程式:

- 從結束程式輸出時的 *CompCode* 欄位值
- 從結束程式輸出時的 *Reason* 欄位值
- 從結束程式輸出時的 *DataLength* 欄位值

- 結束程式輸出緩衝區 *OutBuffer* 的內容。 傳回的位元組數小於結束程式的 *OutBufferLength* 參數，以及結束程式輸出上 *DataLength* 欄位的值。

  如果結束程式訊息描述子參數中的 *Encoding* 及 *CodedCharSetId* 欄位 兩者 未變更，則佇列管理程式會傳回:

- 輸入 結束程式之 MQDXP 結構中的 *Encoding* 及 *CodedCharSetId* 欄位值。

  如果結束程式訊息描述子參數中的 *Encoding* 及 *CodedCharSetId* 欄位之一或兩者已變更，則佇列管理程式會傳回:

- 從結束程式輸出的結束程式訊息描述子參數中的 *Encoding* 及 *CodedCharSetId* 欄位值

MQXDR_CONVERsion_FAILED
未順利完成轉換。

如果結束程式指定此值，佇列管理程式會將下列項目傳回給發出 MQGET 呼叫的應用程式:

- 從結束程式輸出時的 *CompCode* 欄位值
- 從結束程式輸出時的 *Reason* 欄位值
- 結束程式 輸入 上的 *DataLength* 欄位值
- 結束程式輸入緩衝區 *InBuffer* 的內容。 傳回的位元組數由 *InBufferLength* 參數提供

如果結束程式已變更 *InBuffer*，則未定義結果。

ExitResponse 是結束程式的輸出欄位。

Hconn
類型 :MQHCONN

這是可在 MQXCNVC 呼叫上使用的連線控點。 此控點不一定與發出 MQGET 呼叫的應用程式所指定的控點相同。

pEntryPoints
類型 :PMQIEP

MQIEP 結構的位址，透過此結構可進行 MQI 及 DCI 呼叫。

Reason
類型 :MQLONG

定義 *CompCode* 的原因碼。

當呼叫結束程式時，如果結束程式選擇不執行任何動作，則會包含傳回給發出 MQGET 呼叫之應用程式的原因碼。 可能的值包括 MQRC_TRUNCATED_MSG_ACCEPTED ，指出訊息已截斷，以符合應用程式所提供的緩衝區，以及 MQRC_NOT_CONVERTED ，指出訊息需要轉換，但尚未執行此動作。

在結束程式的輸出上，此欄位包含要在 MQGET 呼叫的 *Reason* 參數中傳回給應用程式的原因; 建議如下:

- 如果 *Reason* 在輸入結束程式時具有值 MQRC_TRUNCATED_MSG_ACCEPTED ，則無論轉換成功還是失敗，都不得變更 *Reason* 和 *CompCode* 欄位。

  (如果 *CompCode* 欄位不是 MQCC_OK ，則擷取訊息的應用程式可以透過將訊息描述子中傳回的 *Encoding* 及 *CodedCharSetId* 值與所要求的值進行比較來識別轉換失敗; 相反地，應用程式無法從適合緩衝區的訊息中識別截斷的訊息。 基於此原因，必須優先傳回 MQRC_TRUNCATED_MSG_ACCEPTED ，而不是任何指出轉換失敗的原因。)

- 如果 *Reason* 在結束程式的輸入上有任何其他值:
  - 如果轉換成功， *CompCode* 必須設為 MQCC_OK ， *Reason* 必須設為 MQRC_NONE。
  - 如果轉換失敗，或訊息展開且必須截斷以適合緩衝區，則 *CompCode* 必須設為 MQCC_WARNING (或維持不變) ，且 *Reason* 必須設為列出的其中一個值，以指出失敗的本質。

    請注意，如果轉換後的訊息對緩衝區而言太大，則只有在發出 MQGET 呼叫的應用程式指定 MQGMO_ACCEPT_TRUNCATED_MSG 選項時，才必須截斷訊息:

    - 如果指定該選項，則會傳回原因 MQRC_TRUNCATED_MSG_ACCEPTED。

- 如果未指定該選項，則會傳回未轉換的訊息，原因碼為 MQRC_CONVERTED_MSG_TOO_BIG。

建議結束程式使用列出的原因碼，以指出轉換失敗的原因，但如果認為適當，結束程式可以從 MQRC_ * 代碼集傳回其他值。 此外，會配置 MQRC_APPL_FIRST 至 MQRC_APPL_LAST 值的範圍，以供結束程式使用，以指出結束程式想要與發出 MQGET 呼叫的應用程式通訊的條件。

註: 如果無法順利轉換訊息，則結束程式 必須 在 *ExitResponse* 欄位中傳回 MQXDR_CONVERSION_FAILED ，以便讓佇列管理程式傳回未轉換的訊息。 不論 *Reason* 欄位中傳回的原因碼為何，都是 true。

第一個 MQRC_APPL_FIRST
(900 ， X'384 ') 應用程式定義原因碼的最低值。

MQRC_APPL_LAST
(999 ， X'3E7') 應用程式定義原因碼的最高值。

MQ RC_CONVERTED_MSG_TOO_Big
(2120 ， X'848 ') 轉換的資料對緩衝區而言太大。

MQRC_NOT_CONVERTED
(2119 ， X'847 ') 訊息資料未轉換。

MQRC_SOURCE_CCSID_ERROR
(2111 ， X'83F') 來源編碼字集 ID 無效。

MQRC_SOURCE_DECIMAL_ENC_ERROR
(2113 ， X'841 ') 無法辨識訊息中的壓縮十進位編碼。

MQRC_SOURCE_FLOAT_ENC_ERROR
(2114 ， X'842 ') 無法辨識訊息中的浮點數編碼。

MQRC_SOURCE_INTEGER_ENC_ERROR
(2112 ， X'840 ') 無法辨識來源整數編碼。

MQRC_TARGET_CCSID_ERROR
(2115 ， X'843 ') 目標編碼字集 ID 無效。

MQRC_TARGET_DECIMAL_ENC_ERROR
(2117 ， X'845 ') 接收器指定的壓縮十進位編碼無法辨識。

MQRC_TARGET_FLOAT_ENC_ERROR
(2118 ， X'846 ') 接收器指定的浮點數編碼無法辨識。

MQRC_TARGET_INTEGER_ENC_ERROR
(2116 ， X'844 ') 無法辨識目標整數編碼。

MQRC_TRUNCATED_MSG_ACCEPTED
(2079 ， X'81F') 傳回截斷訊息 (處理完成)。

這是結束程式的輸入/輸出欄位。

StrucId
類型: MQCHAR4

結構 ID。值必須為:

MQDXP_STRUC_ID
資料轉換結束程式參數結構的 ID。

對於 C 程式設計語言，也會定義常數 MQDXP_STRUC_ID_ARRAY; 此值與 MQDXP_STRUC_ID 相同，但卻是字元陣列而非字串。

這是結束程式的輸入欄位。

Version
類型 :MQLONG

結構版本號碼。值必須為:

MQDXP_VERSION_1
資料轉換結束程式參數結構的版本號碼。

下列常數指定現行版本的版本號碼:

MQDXP_CURRENT_VERSION
資料轉換結束程式參數結構的現行版本。

註: 引進此結構的新版本時，現有組件的佈置不會變更。 因此，結束程式必須檢查 *Version* 欄位是否等於或大於包含結束程式需要使用之欄位的最低版本。

這是結束程式的輸入欄位。

C 宣告

```
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 宣告 (僅限 IBM i)

```
**   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 組譯器宣告

```
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)
```

MQXCNVC-轉換字元

MQXCNVC 呼叫會使用 C 程式設計語言，將字元從一個字集轉換成另一個字集。

此呼叫是 WebSphere MQ 資料轉換介面 (DCI) 的一部分，這是其中一個 WebSphere MQ 架構介面。

附註: 可以從應用程式及資料轉換結束程式環境使用該呼叫。

語法

MQXCNVC (*Hconn*、 選項、 *SourceCCSID*、 *SourceLength*、 *SourceBuffer*、 *TargetCCSID*、 *TargetLength*、 *TargetBuffer*、 *DataLength*、 *CompCode*、 *Reason*)

參數

Hconn
類型 :MQHCONN-輸入

此控點代表佇列管理程式的連線。

在資料轉換結束程式中， *Hconn* 通常是傳給 MQDXP 結構之 *Hconn* 欄位中的資料轉換結束程式的控點; 這個控點不一定與發出 MQGET 呼叫的應用程式所指定的控點相同。

在 IBM i 上，可以為 *Hconn* 指定下列特殊值:

MQHC_DEF_HCONN
預設連線控點。

如果您執行 CICS TS 3.2 或更高版本應用程式，請確定呼叫 MQXCNVC 呼叫的字元轉換結束程式定義為 OPENAPI。 此定義可防止不正確連線所導致的 2018 MQRC_HCONN_ERROR 錯誤，並容許 MQGET 完成。

選項
類型 :MQLONG-輸入

控制 MQXCNVC 動作的選項。

可以指定零個以上說明的選項。 如果需要多個值，則值可以是:

- 相加（請不要將同一常數相加多次），或者
- 使用位元 OR 運算結合 (如果程式設計語言支援位元運算)

預設轉換選項: 下列選項控制使用預設字元轉換:

MQDCC_DEFAULT_CONVERSION
預設轉換。

此選項指定如果不支援呼叫上指定的其中一個或兩個字集，則可以使用預設字元轉換。 這可讓佇列管理程式在轉換字串時使用安裝指定的預設字集，其近似指定的字集。

註: 使用近似字集來轉換字串的結果是部分字元可能轉換不正確。 您可以在字串中只使用指定字集和預設字集共用的字元，來避免此情況。

當安裝或重新啟動佇列管理程式時，配置選項會定義預設字集。

如果未指定 MQDCC_DEFAULT_CONVERSION ，則佇列管理程式只會使用指定的字集來轉換字串，如果不支援其中一個或兩個字集，則呼叫會失敗。

此選項在下列環境中受支援: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows。

填補選項: 下列選項可讓佇列管理程式以空白填補已轉換的字串，或捨棄無意義的尾端字元，使已轉換的字串符合目標緩衝區:

MQDCC_FILL_TARGET_BUFFER
填入目標緩衝區。

此選項要求以完全填入目標緩衝區的方式進行轉換:

- 如果轉換字串時字串合約，則會新增尾端空白，以填入目標緩衝區。
- 如果字串在轉換時展開，則會捨棄不重要的尾端字元，使轉換的字串符合目標緩衝區。 如果可以順利完成，則呼叫會完成， MQCC_OK 及原因碼 MQRC_NONE。

  如果不重要的尾端字元太少，則會在目標緩衝區中放置盡可能多的字串，且呼叫會完成，並產生 MQCC_WARNING 及原因碼 MQRC_CONVERTED_MSG_TOO_BIG。

  不顯著字元為:

  – 尾端空白

  – 字串中第一個空值字元之後的字元 (但排除第一個空值字元本身)
- 如果字串、 *TargetCCSID* 及 *TargetLength* 無法完全以有效字元設定目標緩衝區，則呼叫會失敗，並產生 MQCC_FAILED 及原因碼 MQRC_TARGET_LENGTH_ERROR。 當 *TargetCCSID* 是純 DBCS 字集 (例如 UCS-2) ，但 *TargetLength* 指定的長度是奇數位元組時，就會發生這種情況。
- *TargetLength* 可以小於或大於 *SourceLength*。 從 MQXCNVC 返回時， *DataLength* 具有與 *TargetLength* 相同的值。

如果未指定此選項:

- 視需要，容許在目標緩衝區內收縮或展開字串。 不新增或捨棄不顯著尾端字元。

  如果轉換的字串符合目標緩衝區，則呼叫會完成，且 MQCC_OK 及原因碼 MQRC_NONE。

  如果轉換的字串對目標緩衝區而言太大，則會將適合的字串放入目標緩衝區中，且呼叫會完成，MQCC_WARNING 及原因碼 MQRC_CONVERTED_MSG_TOO_BIG。 在此情況下，可以傳回少於 *TargetLength* 個位元組的附註。
- *TargetLength* 可以小於或大於 *SourceLength*。 從 MQXCNVC 返回時， *DataLength* 小於或等於 *TargetLength*。

此選項在下列環境中受支援: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows。

編碼選項: 說明的選項可用來指定來源和目標字串的整數編碼。 只有 當對應的字集 ID 指出主儲存體中字集的表示法取決於二進位整數所使用的編碼時，才會使用相關編碼。 這只會影響某些多位元組字集 (例如， UCS-2 字集)。

如果字集是單位元組字集 (SBCS) ，或在主儲存體中具有表示法且與整數編碼無關的多位元組字集，則會忽略編碼。

僅必須指定其中一個 MQDCC_SOURCE_ * 值，並結合其中一個 MQDCC_TARGET__ * 值:

MQDCC_SOURCE_ENC_NATIVE
來源編碼是環境和程式設計語言的預設值。

MQ DCC_SOURCE_ENC_NORMAL
來源編碼正常。

MQDCC_SOURCE_ENC_REVERSED
來源編碼已反轉。

MQDCC_SOURCE_ENC_UNDEFINED
未定義來源編碼。

MQDCC_TARGET_ENC_NATIVE
目標編碼是環境和程式設計語言的預設值。

MQDCC_TARGET_ENC_NORMAL
目標編碼正常。

MQDCC_TARGET_ENC_REVERSED
目標編碼已反轉。

未定義 MQDCC_TARGET_ENC_UNDEFINED
未定義目標編碼。

先前定義的編碼值可以直接新增至 *Options* 欄位。 不過，如果從 MQMD 或其他結構中的 *Encoding* 欄位取得來源或目標編碼，則必須完成下列處理:

1. 必須透過刪除浮點數和聚集十進位編碼，從 *Encoding* 欄位中擷取整數編碼; 如需如何執行此動作的詳細資料，請參閱 第 787 頁的『分析編碼』 。
2. 從步驟 1 產生的整數編碼必須乘以適當的因素，才能新增至 *Options* 欄位。 這些因素包括:
   - 來源編碼的 MQDCC_SOURCE_ENC_FACTOR
   - 目標編碼的 MQDCC_TARGET_ENC_FACTOR

下列程式碼範例說明如何以 C 程式設計語言撰寫此程式碼:

```
Options = (MsgDesc.Encoding & MQENC_INTEGER_MASK)
           * MQDCC_SOURCE_ENC_FACTOR
        + (DataConvExitParms.Encoding & MQENC_INTEGER_MASK)
           * MQDCC_TARGET_ENC_FACTOR;
```

如果未指定，編碼選項會預設為 undefined (MQDCC_ * _ENC_UNDEFINED)。 在大部分情況下，這不會影響 MQXCNVC 呼叫順利完成。 不過，如果對應的字集是多位元組字集，且具有相依於編碼的表示法 (例如， UCS-2 字集) ，則呼叫會失敗，原因碼為 MQRC_SOURCE_INTEGER_ENC_ERROR 或 MQRC_TARGET_INTEGER_ENC_ERROR (視情況而定)。

下列環境支援編碼選項: AIX、 HP-UX、 z/OS、 IBM i、Solaris、 Linux、 Windows。

預設選項: 如果未指定上述任何選項，則可以使用下列選項:

MQDCC_NONE
未指定選項。

MQDCC_NONE 定義為輔助程式文件。 此選項不會與任何其他選項一起使用，但由於其值為零，因此無法偵測此類使用。

SourceCCSID
類型 :MQLONG-輸入

這是 *SourceBuffer* 中輸入字串的編碼字集 ID。

SourceLength
類型 :MQLONG-輸入

這是 *SourceBuffer* 中輸入字串的長度 (以位元組為單位); 它必須為零或大於零。

SourceBuffer
類型: MQCHARxSource 長度-輸入

這是包含要從一個字集轉換成另一個字集之字串的緩衝區。

TargetCCSID
類型 :MQLONG-輸入

這是 *SourceBuffer* 要轉換成的字集的編碼字集 ID。

TargetLength
類型 :MQLONG-輸入

這是輸出緩衝區 *TargetBuffer* 的長度 (以位元組為單位); 它必須為零或大於零。 它可以小於或大於 *SourceLength*。

TargetBuffer
類型: MQCHARxTarget 長度-輸出

這是將字串轉換成 *TargetCCSID* 所定義的字集之後的字串。 轉換的字串可以短於或長於未轉換的字串。 *DataLength* 參數指出傳回的有效位元組數。

DataLength
類型 :MQLONG-輸出

這是在輸出緩衝區 *TargetBuffer* 中傳回的字串長度。 轉換的字串可以短於或長於未轉換的字串。

CompCode
類型 :MQLONG-輸出

它是下列其中一項:

MQCC_OK
順利完成。

MQCC_WARNING
警告 (局部完成)。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_WARNING:

MQ RC_CONVERTED_MSG_TOO_Big
(2120 , X'848 ') 轉換的資料對緩衝區而言太大。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_DATA_LENGTH_ERROR
(2010 , X'7DA') 資料長度參數無效。

MQRC_DBCS_ERROR
(2150 , X'866 ') DBCS 字串無效。

MQRC_HCONN_ERROR
(2018 , X'7E2') 連線控點無效。

MQRC_OPTIONS_ERROR
(2046 , X'7FE') 選項無效或不一致。

MQRC_RESOURCE_PROBLEM
(2102 , X'836 ') 可用的系統資源不足。

MQRC_SOURCE_BUFFER_ERROR
(2145 , X'861 ') 來源緩衝區參數無效。

MQRC_SOURCE_CCSID_ERROR
(2111 , X'83F') 來源編碼字集 ID 無效。

MQRC_SOURCE_INTEGER_ENC_ERROR
(2112 , X'840 ') 無法辨識來源整數編碼。

MQRC_SOURCE_LENGTH_ERROR
(2143 , X'85F') 來源長度參數無效。

MQRC_STORAGE_NOT_AVAILABLE
(2071 , X'817 ') 可用的儲存體不足。

MQRC_TARGET_BUFFER_ERROR
(2146 , X'862 ') 目標緩衝區參數無效。

MQRC_TARGET_CCSID_ERROR
(2115 , X'843 ') 目標編碼字集 ID 無效。

MQRC_TARGET_INTEGER_ENC_ERROR
(2116 , X'844 ') 無法辨識目標整數編碼。

MQRC_TARGET_LENGTH_ERROR
(2144 , X'860 ') 目標長度參數無效。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 發生非預期的錯誤。

如需這些代碼的詳細資訊，請參閱 原因碼。

C 呼叫

```
MQXCNVC (Hconn, Options, SourceCCSID, SourceLength, SourceBuffer,
        TargetCCSID, TargetLength, TargetBuffer, &DataLength,
        &CompCode, &Reason);
```

宣告參數如下:

```
MQHCONN  Hconn;            /* Connection handle */
MQLONG   Options;          /* Options that control the action of
                              MQXCNVC */
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 宣告 (僅限 IBM i)

```
     CALL 'MQXCNVC' USING HCONN, OPTIONS, SOURCECCSID, SOURCELENGTH,
                          SOURCEBUFFER, TARGETCCSID, TARGETLENGTH,
                          TARGETBUFFER, DATALENGTH, COMPCODE, REASON.
```

宣告參數如下:

```
**   Connection handle
 01  HCONN         PIC S9(9) BINARY.
**   Options that control the action of MQXCNVC
 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 組譯器宣告

```
         CALL MQXCNVC,(HCONN,OPTIONS,SOURCECCSID,SOURCELENGTH,         X
```

```
                SOURCEBUFFER,TARGETCCSID,TARGETLENGTH,TARGETBUFFER,     X
                DATALENGTH,COMPCODE,REASON)
```

宣告參數如下:

```
HCONN          DS  F       Connection handle
OPTIONS        DS  F       Options that control the action of MQXCNVC
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-資料轉換結束程式

MQ_DATA_CONV_EXIT 呼叫說明傳遞至資料轉換結束程式的參數。

佇列管理程式未提供稱為 MQ_DATA_CONV_EXIT 的進入點 (請參閱用法附註 11)。

此定義是 WebSphere MQ 「資料轉換介面 (DCI)」的一部分，它是其中一個 WebSphere MQ 架構介面。

語法

MQ_DATA_CONV_EXIT (*DataConvExitParms*、 *MsgDesc*、 *InBufferLength*、 *InBuffer*、 *OutBufferLength*、 *OutBuffer*)

參數

DataConvExitParms
類型 :MQDXP-輸入/輸出

此結構包含與呼叫結束程式相關的資訊。 結束程式會設定此結構中的資訊，以指出轉換的結果。 如需此結構中欄位的詳細資料，請參閱 第 796 頁的『MQDXP-資料轉換結束程式參數』 。

MsgDesc
類型 :MQMD-輸入/輸出

在結束程式的輸入時，這是與 *InBuffer* 參數中傳遞給結束程式的訊息資料相關聯的訊息描述子。

註: 傳遞至結束程式的 *MsgDesc* 參數一律是呼叫結束程式之佇列管理程式所支援的最新 MQMD 版本。 如果結束程式預期在不同環境之間可移植，則結束程式會檢查 *MsgDesc* 中的 *Version* 欄位，以驗證該結束程式需要存取的欄位是否存在於結構中。

在下列環境中，會傳遞 version-2 MQMD: AIX、 HP-UX、 IBM i、Solaris、 Linux、 Windows。 在支援資料轉換結束程式的所有其他環境中，會傳遞 version-1 MQMD 給結束程式。

在輸出上，如果轉換成功，結束程式會將 *Encoding* 和 *CodedCharSetId* 欄位變更為應用程式所要求的值; 這些變更會反映回應用程式。 會忽略結束程式對結構所做的任何其他變更; 它們不會反映回應用程式。

如果結束程式在 MQDXP 結構的 *ExitResponse* 欄位中傳回 MQXDR_OK ，但未變更訊息描述子中的 *Encoding* 或 *CodedCharSetId* 欄位，則佇列管理程式會針對那些欄位傳回 MQDXP 結構中對應欄位在結束程式輸入時所具有的值。

***InBuffer* 長度**
類型 :MQLONG-輸入

InBuffer 的長度 (以位元組為單位)。

這是輸入緩衝區的長度 *InBuffer*，並指定結束程式要處理的位元組數。 *InBufferLength* 是轉換之前的訊息資料長度，以及應用程式在 MQGET 呼叫中提供的緩衝區長度中的較小者。

該值一律大於零。

InBuffer
類型: MQBYTExInBufferLength -輸入

包含未轉換訊息的緩衝區。

這包含轉換之前的訊息資料。 如果結束程式無法轉換資料，在結束程式完成之後，佇列管理程式會將此緩衝區的內容傳回給應用程式。

註: 結束程式不應變更 *InBuffer*; 如果變更此參數，則未定義結果。

在 C 程式設計語言中，此參數定義為 void 的指標。

***OutBuffer* 長度**
類型 :MQLONG-輸入

OutBuffer 的長度 (以位元組為單位)。

這是輸出緩衝區 *OutBuffer* 的長度，與應用程式在 MQGET 呼叫中提供的緩衝區長度相同。

該值一律大於零。

OutBuffer
類型: MQBYTExOutBufferLength -輸出

包含已轉換訊息的緩衝區。

在結束程式的輸出上，如果轉換成功 (如 *DataConvExitParms* 參數的 *ExitResponse* 欄位中的 MQXDR_OK 值所指示)，則 *OutBuffer* 會以所要求的表示法包含要遞送至應用程式的訊息資料。 如果轉換不成功，則會忽略結束程式對此緩衝區所做的任何變更。

在 C 程式設計語言中，此參數定義為 void 的指標。

使用注意事項

1. 資料轉換結束程式是使用者撰寫的結束程式，在處理 MQGET 呼叫期間接收控制。 資料轉換結束程式所執行的函數由結束程式的提供者定義; 不過，結束程式必須符合這裡及相關聯參數結構 MQDXP 中說明的規則。

   可用於資料轉換結束程式的程式設計語言由環境決定。
2. 只有在下列 全部 為真時，才會呼叫結束程式:
   - MQGET 呼叫中指定 MQGMO_CONVERT 選項
   - 訊息描述子中的 *Format* 欄位不是 MQFMT_NONE
   - 訊息尚未在必要的表示法中; 也就是說，訊息的 *CodedCharSetId* 及/或 *Encoding* 與應用程式在 MQGET 呼叫所提供的訊息描述子中指定的值不同
   - 佇列管理程式尚未順利完成轉換
   - 應用程式緩衝區的長度大於零
   - 訊息資料的長度大於零
   - MQGET 作業期間到目前為止的原因碼是 MQRC_NONE 或 MQRC_TRUNCATED_MSG_ACCEPTED
3. 在寫入結束程式時，請考慮以容許它轉換已截斷的訊息的方式來撰寫結束程式。 截斷的訊息可能以下列方式產生:
   - 接收端應用程式提供的緩衝區小於訊息，但在 MQGET 呼叫上指定 MQGMO_ACCEPT_TRUNCATED_MSG 選項。

     在此情況下，結束程式輸入上 *DataConvExitParms* 參數中的 *Reason* 欄位值為 MQRC_TRUNCATED_MSG_ACCEPTED。
   - 訊息傳送者在傳送之前已截斷訊息。 例如，報告訊息可能會發生這種情況 (如需詳細資料，請參閱 第 796 頁的『報告訊息的轉換』)。

在此情況下，結束程式輸入之 *DataConvExitParms* 參數中的 *Reason* 欄位值為 MQRC_NONE (如果接收端應用程式提供的緩衝區足以容納訊息)。

因此，結束程式輸入上的 *Reason* 欄位值無法一律用來決定訊息是否已截斷。

截斷訊息的識別性質是在 *InBufferLength* 參數中提供給結束程式的長度 小於 訊息描述子中 *Format* 欄位所包含的格式名稱所隱含的長度。 因此，在嘗試轉換任何資料之前，結束程式應該先檢查 *InBufferLength* 的值; 結束程式 不應該 假設已提供格式名稱所隱含的完整資料量。

如果結束程式 未 寫入以轉換截斷訊息，且 *InBufferLength* 小於預期值，則結束程式會在 *DataConvExitParms* 參數的 *ExitResponse* 欄位中傳回 MQXDR_CONVERSION_FAILED ，且 *CompCode* 及 *Reason* 欄位會設為 MQCC_WARNING 及 MQRC_FORMAT_ERROR。

如果已寫入結束程式 已 來轉換截斷的訊息，則結束程式將盡可能轉換大量資料 (請參閱下一個用法附註) ，請注意不要嘗試檢查或轉換 *InBuffer* 結尾以外的資料。 如果轉換順利完成，結束程式會維持 *DataConvExitParms* 參數中的 *Reason* 欄位不變。 如果訊息被接收端佇列管理程式截斷，則會傳回 MQRC_TRUNCATED_MSG_ACCEPTED ，如果訊息被訊息傳送端截斷，則會傳回 MQRC_NONE。

訊息也可以展開 期間 轉換，使其大於 *OutBuffer*。 在此情況下，結束程式必須決定是否截斷訊息; *DataConvExitParms* 參數中的 *AppOptions* 欄位指出接收端應用程式是否指定 MQGMO_ACCEPT_TRUNCATED_MSG 選項。

4. 一般而言，會轉換提供給 *InBuffer* 中結束程式的訊息中的所有資料，或全部都不會轉換。 不過，如果在轉換之前或轉換期間截斷訊息，則會發生異常狀況; 在此情況下，緩衝區結尾可能有不完整的項目 (例如: 雙位元組字元的 1 個位元組，或 4 個位元組整數的 3 個位元組)。 在此狀況下，請考慮省略不完整的項目，並將 *OutBuffer* 中未用的位元組設為空值。 不過， 應該 轉換陣列或字串內的完整元素或字元。
5. 當第一次需要結束程式時，佇列管理程式會嘗試載入與格式 (副檔名除外) 同名的物件。 載入的物件必須包含處理具有該格式名稱之訊息的結束程式。 請考量讓結束程式名稱與包含結束程式的物件名稱相同，雖然並非所有環境都需要此名稱。
6. 自從應用程式連接至佇列管理程式之後，當應用程式嘗試擷取使用該 *Format* 的第一則訊息時，會載入新的結束程式副本。 若為 CICS 或 IMS 應用程式，這表示當 CICS 或 IMS 子系統連接至佇列管理程式時。 如果佇列管理程式已捨棄先前載入的副本，則也可以在其他時間載入新的副本。 因此，結束程式不得嘗試使用靜態儲存體將資訊從結束程式的一次呼叫傳送至下一次呼叫-可以在兩次呼叫之間卸載該結束程式。
7. 如果使用者提供的結束程式與佇列管理程式支援的其中一個內建格式同名，則使用者提供的結束程式不會取代內建轉換常式。 呼叫這類結束程式的唯一情況如下:
   - 如果內建轉換常式無法處理與所涉及 *CodedCharSetId* 或 *Encoding* 之間的轉換，或
   - 如果內建轉換常式無法轉換資料 (例如，因為有欄位或字元無法轉換)。
8. 結束程式的範圍與環境相關。 必須選擇 *Format* 名稱，以將與其他格式衝突的風險降至最低。 請考慮從識別定義格式名稱之應用程式的字元開始。
9. 資料轉換結束程式在類似發出 MQGET 呼叫之程式的環境中執行; 環境包括位址空間及使用者設定檔 (如果適用的話)。 程式可能是訊息通道代理程式，將訊息傳送至不支援訊息轉換的目的地佇列管理程式。 結束程式無法危害佇列管理程式的完整性，因為它不會在佇列管理程式的環境中執行。
10. 結束程式唯一可以使用的 MQI 呼叫是 MQXCNVC; 嘗試使用其他 MQI 呼叫會失敗，原因碼為 MQRC_CALL_IN_PROGRESS 或其他無法預期的錯誤。
11. 佇列管理程式未提供稱為 MQ_DATA_CONV_EXIT 的進入點。 不過，會以 C 程式設計語言為名稱 MQ_DATA_CONV_EXIT 提供 `typedef` ，並可用來宣告使用者撰寫的結束程式，以確保參數正確。 結束程式的名稱必須與格式名稱 (MQMD 中 *Format* 欄位包含的名稱) 相同，雖然並非所有環境都需要此名稱。

下列範例說明如何以 C 程式設計語言宣告處理 MYFORMAT 格式的結束程式:

```
#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. 在 z/OS 上，如果 API 交互結束程式也生效，則會在資料轉換結束程式之後呼叫它。

C 呼叫

```
exitname (&DataConvExitParms, &MsgDesc, InBufferLength,
         InBuffer, OutBufferLength, OutBuffer);
```

傳遞至結束程式的參數宣告如下:

```
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 宣告 (僅限 IBM i)

```
     CALL 'exitname' USING DATACONVEXITPARMS, MSGDESC, INBUFFERLENGTH,
                           INBUFFER, OUTBUFFERLENGTH, OUTBUFFER.
```

傳遞至結束程式的參數宣告如下:

```
**   Data-conversion exit parameter block
 01  DATACONVEXITPARMS.
     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 組譯器宣告

```
         CALL EXITNAME,(DATACONVEXITPARMS,MSGDESC,INBUFFERLENGTH,      X
               INBUFFER,OUTBUFFERLENGTH,OUTBUFFER)
```

傳遞至結束程式的參數宣告如下:

```
DATACONVEXITPARMS  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
```

指定為 MQRFH2 元素的內容

非訊息描述子內容可以指定為 MQRFH2 標頭資料夾中的元素。 將 MQRFH2 元素指定為內容的概觀。

這會保留與舊版 WebSphere MQ JMS 和 XMS 用戶端的相容性。 本節說明如何在 MQRFH2 標頭中指定內容。

若要使用 MQRFH2 元素作為內容，請依照 使用適用於 Java 的 WebSphere MQ 類別中的說明來指定元素。 此資訊補充 第 456 頁的『MQRFH2 -規則和格式化標頭 2』中說明的資訊。

將內容資料類型對映至 MQRFH2 資料類型

本主題提供對映至其對應 MQRFH2 資料類型之訊息內容類型的相關資訊。

表 *580:* 支援的 *MQRFH2* 資料類型

訊息內容類型	**MQRFH2 資料類型**
MQBYTE []	bin.hex
MQBOOL	布林值
MQINT8	i1
MQINT16	i2
MQINT32	i4
MQINT64	i8
MQFLOAT32	r4
MQFLOAT64	r8
MQCHAR []	弦

任何沒有資料類型的元素都會假設為 "string" 類型。

MQRFH2 資料類型 `int`(表示未指定大小的整數) 會被視為 `i8`。

元素屬性 `xsi:nil='true'`指出空值。 請勿將屬性 `xsi:nil='false'` 用於非空值。

例如，下列內容具有空值:

```
<NullProperty xsi:nil='true'></NullProperty>
```

位元組或字串內容可以有空值。 這由具有零長度元素值的 MQRFH2 元素表示。

例如，下列內容具有空值:

```
<EmptyProperty></EmptyProperty>
```

支援的 MQRFH2 資料夾

使用訊息描述子欄位作為內容的概觀。

MQRFH2 標頭和 JMS 中說明了資料夾 `<jms>`、 `<mcd>`、 `<mqext>`和 `<usr>` 。 `<usr>` 資料夾用來傳輸與訊息相關聯的任何 JMS 應用程式定義內容。 `<usr>` 資料夾中不容許群組。

MQRFH2 標頭和 JMS 支援下列其他資料夾:

- `<mq>`

  此資料夾用於並保留給 IBM WebSphere MQ 所使用的 MQ 定義內容。

- <mq_usr>

  這個資料夾可用來傳輸任何應用程式定義的內容，這些內容不會顯露成 JMS 使用者定義的內容，因為這些內容可能不符合 JMS 內容的需求。 此資料夾可以包含 <usr> 資料夾無法包含的群組。
- 以 content='properties' 屬性標示的任何資料夾。

  這類資料夾相當於內容中的 <mq_usr> 資料夾。
- <mqps>

  此資料夾用於 IBM WebSphere MQ 發佈/訂閱內容。

IBM WebSphere MQ 也支援下列已由 WAS/SIB 使用的資料夾:

- <sib>

  如果 WAS/SIB 系統訊息內容未公開為 JMS 內容，或對映至 JMS_IBM_ * 內容，但公開給 WAS/SIB 應用程式，則會使用及保留這個資料夾; 這些包括正向和反向遞送路徑內容。

  至少部分無法公開為 JMS 內容，因為它們是位元組陣列。 如果您的應用程式將內容新增至此資料夾，則會忽略或移除該值。
- <sib_usr>

  此資料夾用於 WAS/SIB 使用者訊息內容，因為它們不是受支援的類型，而無法公開為 JMS 使用者內容; 它們會向 WAS/SIB 應用程式公開。

  這些是您可以透過 SIMessage 介面取得或設定的使用者內容，但位元組陣列的內容對映至所需的內容值。

  如果 IBM WebSphere MQ 應用程式將任意 bin.hex 元素寫入資料夾，應用程式可能會收到 IOException，因為它不是預期要還原的格式。 如果您新增 bin.hex 元素以外的任何項目，則會收到 ClassCastException。

  請勿嘗試利用這個資料夾來提供內容給 WAS/SIB; 請改為使用 <usr> 資料夾來達到這個目的。
- <sib_context>

  這個資料夾用於未公開給 WAS/SIB 使用者應用程式或作為 JMS 內容的 WAS/SIB 系統訊息內容。 這些包括用於 Web 服務及類似項目的安全及交易式內容。

  您的應用程式不得將內容新增至此資料夾。
- <mqema>

  WAS/SIB 使用這個資料夾，而不是 <mqext> 資料夾。

MQRFH2 資料夾名稱區分大小寫。

下列資料夾是保留的，混合使用任何小寫或大寫字元:

- 任何以 mq 或 wmq 為字首的資料夾; 保留供 IBM WebSphere MQ 使用。
- 任何以 sib 為字首的資料夾; 保留供 WAS/SIB 使用。
- <Root> 和 <Body> 資料夾; 已保留但未使用。

下列資料夾無法辨識為包含訊息內容:

- <psc>

  由 WebSphere Message Broker 用來將發佈/訂閱指令訊息傳送至分配管理系統。
- <pscr>

  由 WebSphere Message Broker 用來包含來自分配管理系統的資訊，以回應發佈/訂閱指令訊息。
- 任何未由 WebSphere Message Broker 定義且未標示 content='properties' 屬性的資料夾。

請勿在 <psc> 或 <pscr> 資料夾上指定 content='properties' 。 如果您這麼做，這些資料夾會被視為內容，且 WebSphere Message Broker 可能會如預期停止運作。

如果您的應用程式正在建置含有內容的訊息，則在 MQRFH2 標頭中，要辨識為包含內容的 MQRFH2 標頭，該標頭必須在可鏈結於訊息標頭的標頭清單中。

MQRFH2 之前可以有任意數目的 MQH 標準標頭，或 MQCIH、MQDLH、MQIIH、MQTM、 MQTMC2 或 MQXQH。 字串或 MQCFH 會結束剖析，因為無法鏈結它們。

訊息可以包含多個 MQRFH2 標頭，所有標頭都包含訊息內容。 除非另有限制 (例如 WAS/SIB) ，否則同名的資料夾可以同時存在於不同的標頭中。 如果資料夾全部都在重要標頭中，則會將它們視為一個邏輯資料夾。

雖然重要標頭中的資料夾無法與非重要標頭中的那些資料夾合併，但重要標頭中具有相同名稱的資料夾可以合併，並移除任何衝突的內容。 您的應用程式不得相依於其訊息內的內容佈置。

MQRFH2 群組是針對使用者定義資料夾 (即 <wmq>、 <jms>、 <mcd>、 <usr>、 <mqext>、 <sib>、 <sib_usr>、 <sib_context>及 <mqema> 資料夾) 中的內容進行剖析。

IBM 定義的內容資料夾中的群組 (<wmq> 和 <mq> 資料夾除外) 會針對內容進行剖析。

MQRFH2 資料夾不能包含混合內容; 資料夾或群組可以包含群組或內容，也可以包含值，但不能同時包含兩者。

訊息區段 (第一個或後續區段) 不能包含 IBM WebSphere MQ 定義的內容，但訊息描述子中的那些內容除外。 因此，使用 MQMF_SEGMENT 或 MQMF_SEGMENTATION_ALLOWED 設定來放置包含此類內容的訊息，會導致放置失敗，並出現 MQRC_SEGMENTATION_NOT_ALLOWED。

不過，訊息群組可以包含 IBM WebSphere MQ 定義的內容。

產生 MQRFH2 標頭

如果 WebSphere MQ 將訊息內容轉換為其 MQRFH2 表示法，則必須將 MQRFH2 新增至訊息。 它會將 MQRFH2 新增為個別標頭，或將它與現有標頭合併。

由 WebSphere MQ 產生新的 MQRFH2 標頭可能會中斷訊息中的現有標頭。 為標頭剖析訊息緩衝區的應用程式必須知道在某些情況下，緩衝區中標頭的數目和位置可能會變更。 WebSphere MQ 會嘗試將訊息內容合併至現有的 MQRFH2 標頭 (可以在其中進行) ， 以將新增內容至訊息的影響降到最低。 它也會嘗試將產生的 MQRFH2 插入相對於訊息緩衝區中其他標頭的固定位置，以將影響降到最低。

產生的 MQRFH2 標頭會放在 MQMD 之後，以及任意數目的 MQXQH、 MQRFH 和 MQDLH 標頭 (不論它們的順序為何)。 產生的 MQRFH2 標頭會緊接在不是 MQMD、 MQXQH、 MQDLH 或 MQRFH 標頭的第一個標頭之前。

合併產生的 MQRFH2 的規則

下列規則適用於合併產生的 MQRFH2 與現有 MQRFH2。 在下列情況下，產生的 MQRFH2 標頭會與現有的 MQRFH2 標頭合併:

1. 現有 MQRFH2 位於相同的位置 WebSphere MQ 會將產生的 MQRFH2 或更早版本放置在標頭鏈中。
2. 所產生內容的 CCSID 與現有 MQRFH2 的 NameValueCCSID 相同。

否則，所產生的標頭會個別放置在緩衝區中的前面說明的位置。

在現有 MQRFH2 中合併資料夾的規則

如果訊息內容合併至現有的 MQRFH2，則會掃描現有的 MQRFH2 ，以找出符合訊息內容的資料夾，並合併它們。 如果相符資料夾不存在，則會將新的資料夾新增至現有資料夾的結尾。 如果相符資料夾確實存在，則會搜尋資料夾。 會改寫任何相符的內容。 資料夾尾端會新增任何新的項目。

MQRFH2 資料夾限制

MQRFH2 標頭中資料夾限制的概觀

MQRFH2 限制適用於下列資料夾:

- <usr> 資料夾中的元素名稱不得以字首 JMS 開頭; 這類內容名稱保留供 JMS 使用，且對使用者定義的內容無效。

  這類元素名稱不會導致 MQRFH2 剖析失敗，但 WebSphere MQ 訊息內容 API 無法存取。
- <usr> 資料夾中的元素名稱不得混合小寫或大寫、NULL、TRUE、FALSE、NOT、AND、OR、兩者之間、LIKE、IN、IS 及 ESCAPE。 這些名稱符合 SQL 關鍵字並使剖析選取元更難，因為當選取元中未指定特定內容的資料夾時， <usr> 是使用的預設資料夾。

這類元素名稱不會導致 MQRFH2 剖析失敗，但 WebSphere MQ 訊息內容 API 無法存取。

- 任何被視為包含訊息內容的資料夾中的元素名稱不得包含句點 (.) (Unicode 字元 U+002E)，因為在內容名稱中使用此字元來指示階層。

  這類元素名稱不會導致 MQRFH2 剖析失敗，但 WebSphere MQ 訊息內容 API 無法存取。

一般而言，雖然無法透過 WebSphere MQ 訊息內容 API 來存取 MQRFH2 的某些元素，但包含有效 XML 樣式資料的 MQRFH2 標頭可以由 WebSphere MQ 剖析而不會失敗。

MQRFH2 元素名稱衝突

MQRFH2 元素名稱內衝突的概觀。

訊息內容只能附加一個值。 如果嘗試存取內容導致值衝突，則會優先選擇一個值而不是另一個值。

如果資料夾不包含同名的元素，則用於存取 MQRFH2 元素的 WebSphere MQ 語法容許元素的唯一識別。 如果資料夾包含多個具有相同名稱的元素，則使用的內容值是最接近訊息標頭的值。

如果相同訊息內的不同重要 MQRFH2 標頭中包含兩個以上相同名稱的資料夾，則適用此情況。

在已兩次設定非訊息描述子內容之後處理 MQGET 呼叫時，可能會導致衝突: 不論是透過 MQSETMP 呼叫，還是直接在原始 MQRFH2 標頭中。

如果發生這種情況，則 API 呼叫與訊息相關聯的內容優先於訊息資料中的內容，即原始 MQRFH2 標頭中的內容。 如果發生衝突，則會將它視為在邏輯上位於訊息資料之前。

從內容名稱對映至 MQRFH2 資料夾和元素名稱

MQRFH2 標頭中內容名稱與元素名稱之間的差異概觀。

使用最終產生 MQRFH2 標頭的任何已定義 API 時，為了指定訊息內容 (例如， MQ JMS)，內容名稱不一定是 MQRFH2 資料夾中的元素名稱。

因此，會從內容名稱對映至 MQRFH2 元素，並以相反方式進行對映，同時考量包含該元素的資料夾名稱及元素名稱。 使用 Java 中已記載 IBM WebSphere MQ classes for JMS 中的部分範例。

內容名稱	MQRFH2 資料夾名稱	MQRFH2 元素名稱
JMSDestination	jms	Dst
JMSType	mcd	Type, Set, Fmt
xxx (使用者定義，其中 xxx 不是以 JMS 開頭)	usr	xxx

因此，當 JMS 應用程式存取 JMSDestination 內容時，這會對映至 <jms> 資料夾中的 Dst 元素。

將內容指定為 MQRFH2 元素時， IBM WebSphere MQ 定義其元素如下:

內容名稱	MQRFH2 資料夾名稱	MQRFH2 群組名稱	MQRFH2 元素名稱
<Property>	<usr>	不適用	<Property>
<folder>.<Property>	<folder>	不適用	<Property>
<folder>.<group>.<Property>	<folder>	<group>	<Property>

例如，當 IBM WebSphere MQ JMS 應用程式嘗試存取 Property1 內容時，這會對映至 <usr> 資料夾中的 Property1 元素。 wmq.Property2 內容對映至 <wmq> 資料夾中的 Property2 內容。

如果內容名稱包含多個。 字元，所使用的 MQRFH2 元素名稱是在最終名稱之後。 字元及 MQRFH2 群組用來形成階層; 允許巢狀 MQRFH2 群組。

IBM WebSphere MQ 應用程式會使用 使用適用於 Java 的 WebSphere MQ 類別中定義的簡稱，來存取 <mcd>、 <jms>及 <mqext> 資料夾中 MQRFH2 所包含的 JMS 標頭及提供者特定內容。

JMS 使用者定義內容是從 <usr> 資料夾存取。 IBM WebSphere MQ 應用程式可以將 <usr> 資料夾用於其應用程式內容 (如果 JMS 應用程式可接受該內容顯示為其使用者定義的內容之一)。

如果無法接受，請選擇另一個資料夾; 提供 <wmq_usr> 資料夾作為這類非 JMS 內容的標準位置。

您的應用程式可以指定並使用任何 MQRFH2 資料夾與明確定義的用法，如果您注意下列事項，則不會在 第 811 頁的『指定為 MQRFH2 元素的內容』 中記載:

1. 資料夾可能已在使用中，或者可能在未來由另一個應用程式使用，提供其內含內容的未定義存取權; 如需內容名稱的建議命名慣例，請參閱 內容名稱 。
2. 舊版 IBM WebSphere MQ classes for JMS 或 XMS 用戶端無法存取這些內容，該用戶端只能存取使用者定義內容的 <usr> 資料夾
3. 資料夾必須標示 content 屬性，並將值設為 properties，例如 content='properties'。

   第 684 頁的『MQSETMP-設定訊息內容』 會根據需要自動新增此屬性。 此屬性不得新增至任何 IBM 定義的資料夾，例如 <jms> 及 <usr>。 這樣做會導致 IBM WebSphere MQ classes for JMS 用戶端在 7.0 版之前使用 MessageFormatException 拒絕訊息。

因為 <usr> 資料夾是 <Property> 語法內容的預設位置，所以 IBM WebSphere MQ 應用程式和 JMS 應用程式會使用相同的名稱來存取相同的使用者定義內容值。

保留資料夾名稱

有數個保留資料夾名稱。 您無法使用此類名稱作為資料夾字首; 例如， Root.Property1 無法存取有效的內容，因為 Root 已保留。 下列清單包含保留資料夾名稱:

- 根
- 內文
- 內容
- 環境
- LocalEnvironment
- DestinationList
- ExceptionList
- InputBody
- InputRoot
- InputProperties
- InputLocal 環境
- InputDestination 清單
- InputException 清單
- OutputRoot
- OutputLocal 環境
- OutputDestination 清單
- OutputException 清單

將內容描述子欄位對映至 MQRFH2 標頭

當內容轉換為 MQRFH2 元素時，會使用下列元素屬性來指定內容描述子的有效欄位: 這說明 MQPD 欄位如何轉換為 MQRFH2 元素屬性。

支援

支援內容描述子欄位分割成三個元素屬性

- **sr** 元素屬性指定 MQPD_REJECT_UNSUPP_MASK 位元遮罩中的值。
- **sa** 元素屬性指定 MQPD_ACCEPT_UNSUPP_MASK 位元遮罩中的值。

- **sx** 元素屬性指定 MQPD_ACCEPT_UNSUPP_IF_XMIT_MASK 位元遮罩中的值。

These element attributes are only valid in the <mq> folder and are ignored if set on elements in the other folders containing properties.

支援值	MQRFH2 元素屬性	MQRFH2 屬性值
MQPD_SUPPORT_OPTIONAL	sa	選用 這是預設值。
MQPD_SUPPORT_REQUIRED	sr	必要
MQPD_SUPPORT_REQUIRED_IF_LOCAL	SX	本端

環境定義

使用 **context** 元素屬性來指出內容所屬的訊息環境定義。 僅使用一個值。 此元素屬性適用於任何包含內容之資料夾中的內容。

環境定義值	MQRFH2 屬性值
MQPD_NO_CONTEXT	無 這是預設值。
MQPD_USER _CONTEXT	user

CopyOptions

使用 **copy** 元素屬性來指出內容應該複製到其中的訊息。 可接受多個值; 以逗點區隔多個值。 例如, **copy='reply'** 和 **copy='publish,report'** 都是有效的。 此元素屬性適用於任何包含內容之資料夾中的內容。

註: 在屬性定義中，單引號或雙引號是有效的用法，例如 **copy='reply'** 或 **copy="report"**

CopyOption 值	MQRFH2 屬性值
MQPD_COPY_FORWARD	向前法
MQPD_COPY_REPLY	回覆
MQPD_COPY_REPORT	報告
MQPD_COPY_PUBLISH	發佈
MQPD_COPY_ALL	全部 請勿將此值與任何其他值一起指定。 與其他值一起使用時，這優先於 **none** 以外的任何值。
MQPD_COPY_DEFAULT	預設 這是預設值。 它相當於指定三個值 :MQCOPY_FORWARD、MQCOPY_REPORT 及 MQCOPY_PUBLISH。 請勿將此值與任何其他值一起指定。
MQPD_COPY_NONE	無 請勿將此值與任何其他值一起指定。 與另一個值一起使用時，優先使用。

Restrictions to the <mq> MQRFH2 folder

When a message is put on to a queue, it is searched for an <mq> folder so that the message can be processed according to its MQ-defined properties. 若要容許有效率地剖析 MQ 定義的內容，下列限制適用於資料夾:

- Only properties in the first significant <mq> folder in the message are acted upon by MQ; properties in any other <mq> folder in the message are ignored.
- 如果資料夾採用 UTF-8，則資料夾中只接受單位元組 UTF-8 字元。 資料夾中的多位元組字元可能會導致剖析失敗，並拒絕訊息。
- Do not include MQRFH2 groups in the <mq> folder. 內容值中存在 Unicode 字元 U+003C 將導致拒絕訊息。
- 請勿在資料夾中使用跳出字串。 跳出字串會被視為元素的實際值。
- 資料夾內只會將 Unicode 字元 U+0020 視為空格。 所有其他字元都會視為重要字元，且會導致剖析資料夾失敗，以及拒絕訊息。

If parsing of the <mq> folder fails, or if the folder does not observe these restrictions, the message is rejected with CompCode **MQCC_FAILED** and Reason **MQRC_RFH_RESTRICTED_FORMAT_ERR**.

無效的 MQRFH2 標頭

在 MQPUT、 MQPUT1 或 MQGET 呼叫處理程序時，可能會局部剖析訊息中的任何 MQRFH2 標頭，以檢查包含哪些資料夾，並判斷資料夾是否包含內容。 無效的 MQRFH2 標頭概觀。

如果訊息的局部剖析無法順利完成，因為結構無效，例如， `StrucLength` 欄位太小，則:

- 如果可以判定應用程式包括部分 WebSphere MQ 第 7 版選項，則 MQPUT 或 MQPUT1 呼叫會失敗，原因碼為 MQRC_RFH_ERROR ，因此現有應用程式不會失敗。
- MQGET 呼叫會順利傳回，而包含錯誤的 MQRFH2 會在您提供的緩衝區中傳回。

如果局部剖析失敗，因為無法偵測到特定資料夾是否包含內容 (例如，資料夾以 `<<jms` 開頭) ，因此在判定資料夾名稱之前剖析會失敗，則:

- 如果可以判定應用程式包含部分 WebSphere MQ 第 7 版選項，則 MQPUT 或 MQPUT1 呼叫會失敗，原因碼為 MQRC_RFH_FORMAT_ERROR ，因此現有應用程式不會失敗。
- MQGET 呼叫會順利傳回，而包含錯誤的 MQRFH2 會在您提供的緩衝區中傳回。
- 在佇列管理程式內部時，不會拒絕訊息，因為資料夾的格式不正確，但一律會將資料夾當作未包含任何內容來處理。

訊息可以在佇列管理程式網路中流動，其中的資料夾包含這類語法錯誤，但永不剖析及偵測，而訊息中的一個以上資料夾為:

- 有效
- 已順利剖析
- 用於處理訊息

因此，無法保證偵測。

如果您的其中一個應用程式使用 第 684 頁的『MQSETMP-設定訊息內容』或 MQINQMP 來存取內容，這樣做會導致完全剖析 MQRFH2 資料夾，偵測到無法完成剖析的錯誤，這會以 API 呼叫的適當回覆碼指出。 應用程式無法使用資料夾中的任何內容。

如果嘗試完整剖析 MQRFH2 資料夾，且剖析器找到無法辨識的元素屬性或無法辨識的資料類型，則剖析會繼續並順利完成，且不會發出任何警告; 這不會構成剖析錯誤。

字碼頁轉換

本節說明字碼集名稱和 CCSID、國家語言、 z/OS 轉換、 IBM i 轉換及 Unicode 轉換支援。

每一個國家語言區段列出下列資訊:

- 支援的原生 CCSID

- 不支援的字碼頁轉換

資訊中使用下列術語:

-8
指出 HP-UX 的 CCSID 適用於 HP-UX 定義的字碼集 *roman8*

AIX
指出 WebSphere MQ for AIX

HP-UX
指出 WebSphere MQ for HP-UX

Linux
指出 WebSphere MQ for Linux for Intel 及 WebSphere MQ for Linux for zSeries

HP Integrity NonStop Server
指出 WebSphere MQ for HP Integrity NonStop Server

OS/400
指出 WebSphere MQ for IBM i

Solaris
指出 WebSphere MQ for Solaris

Windows
指出 WebSphere MQ for Windows

z/OS
指出 WebSphere MQ for z/OS

資料轉換的預設值是在目標 (接收) 系統上執行轉換。

如果來源產品支援轉換，則可以透過在來源將通道屬性 CONVERT 設為 YES 來設定通道並交換資料。

註:

1. WebSphere MQ MQI 用戶端資訊的轉換是在伺服器中進行，因此伺服器必須支援從用戶端 CCSID 到伺服器 CCSID 的轉換。
2. 轉換可能包括 CSD/PTF 新增至最新版本 WebSphere MQ 的支援。 請檢查最新服務水準的內容，以查看您是否需要安裝 CSD/PTF 以啟用此轉換。

如需部分 CCSID 號碼與部分產業字碼集名稱之間的交互參照，請參閱 第 818 頁的表 581 。

字碼集名稱和 CCSID

WebSphere MQ for z/OS 提供比語言特定表格中所列更多的轉換。

表 *581:* 字碼集名稱和 *CCSID*

字碼集名稱	CCSID
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 (歐元)	923

表 *581:* 字碼集名稱和 *CCSID (*繼續*)*	
字碼集名稱	**CCSID**
big5	950
eucJP	954 5050 33722
eucKR	970
eucTW	964
eucCN	1383
pck	943
GBK	1386
koi8-r	878

國家語言

此資訊包含 WebSphere MQ 支援的語言。

WebSphere MQ 支援的語言如下:

- 美式英文-請參閱主題 第 820 頁的『美式英文』
- 德文-請參閱主題 第 820 頁的『德文』
- 丹麥文及挪威文-請參閱主題 第 821 頁的『丹麥文和挪威文』
- 芬蘭文和瑞典文-請參閱主題 第 821 頁的『芬蘭文和瑞典文』
- 義大利文-請參閱主題 第 822 頁的『義大利文』
- 西班牙文-請參閱主題 第 823 頁的『西班牙文』
- 英式英文/蓋爾語-請參閱主題 第 823 頁的『英式英文/蓋爾語』
- 法文-請參閱主題 第 824 頁的『法文』
- 多種語言-請參閱主題 第 824 頁的『多語言』
- 葡萄牙文-請參閱主題 第 825 頁的『葡萄牙文』
- 冰島文-請參閱主題 第 826 頁的『冰島文』
- 東歐語言-請參閱主題 第 826 頁的『東歐語』
- 斯拉夫語-請參閱主題 第 828 頁的『斯拉夫文』
- 愛沙尼亞文-請參閱主題 第 828 頁的『愛沙尼亞文』
- 拉脫維亞文和立陶宛文-請參閱主題 第 829 頁的『拉脫維亞文和立陶宛文』
- Ukranian-請參閱主題 第 830 頁的『烏克蘭文』
- 希臘文-請參閱主題 第 831 頁的『希臘文』
- 土耳其文-請參閱主題 第 831 頁的『土耳其文』
- 希伯來文-請參閱主題 第 832 頁的『猶太曆』
- Farsi-請參閱主題 第 834 頁的『波斯人』
- Urdu-請參閱主題 第 834 頁的『烏都文』
- 泰文-請參閱主題 第 834 頁的『泰文』
- 寮文-請參閱主題 第 834 頁的『寮文』
- 越南文-請參閱主題 第 835 頁的『越南文』
- 日文拉丁文 SBCS-請參閱主題 第 835 頁的『日文拉丁文 SBCS』
- 日文片假名 SBCS-請參閱主題 第 836 頁的『日文片假名 SBCS』

- 日文漢字/拉丁文混合-請參閱主題 第 838 頁的『日文漢字/拉丁文混合』
- 日文 Kanji/Katakana 混合-請參閱主題 第 839 頁的『日文漢字/片假名混合』
- 韓文-請參閱主題 第 841 頁的『韓文』
- 簡體中文-請參閱主題 第 841 頁的『簡體中文』
- 繁體中文-請參閱主題 第 842 頁的『繁體中文』

美式英文

美國英文 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上美式英文的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	37, 924, 1140
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	437, 850, 1252, 5348, 858
HP Integrity NonStop Server、 Solaris、 Linux	819, 923
Apple 用戶端	1275

所有非用戶端平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

IBM i

字碼頁:

37
不轉換為字碼頁 923、858

924
不轉換為字碼頁 437、858、1051、1140、1252、1275、5348

1140
不轉換為字碼頁 924、1051、1275

德文

德文的 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上德文的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	273, 924, 1141
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	437, 850, 858, 1252, 5348
HP Integrity NonStop Server、 Solaris、 Linux	819, 923
Apple 用戶端	1275

所有非用戶端平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

IBM i

字碼頁:

273
不轉換為字碼頁 858、923、924、1275

924
不轉換為字碼頁 273、437、858、1051、1141、1252、1275、5348

1141
不轉換為字碼頁 924、1051、1275

丹麥文和挪威文

丹麥文及挪威文的 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上丹麥文及挪威文的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	277, 924, 1142
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	850, 858, 865, 1252, 5348
HP Integrity NonStop Server、 Solaris、 Linux	819, 923
Apple 用戶端	1275

所有非用戶端平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

IBM i

字碼頁:

277
不轉換為字碼頁 858、923、924、1275

924
不轉換為字碼頁 277、858、865、1051、1142、1252、1275、5348

1142
不轉換為字碼頁 924、865、1051、1275

AIX

字碼頁:

819
不轉換為字碼頁 865

HP-UX

字碼頁:

1051
不轉換為字碼頁 865

Windows

字碼頁:

865
不轉換為字碼頁 1051、1275

芬蘭文和瑞典文

芬蘭文及瑞典文的 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上芬蘭文及瑞典文的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	278, 924, 1143
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	437, 850, 858, 865, 1252, 5348
HP Integrity NonStop Server、 Solaris、 Linux	819, 923
Apple 用戶端	1275

所有非用戶端平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

IBM i

字碼頁:

278
不轉換為字碼頁 858、923、924、1275

924
不轉換為字碼頁 278、437、858、865、1051、1143、1252、1275、5348

1143
不轉換為字碼頁 865、924、1051、1275

AIX

字碼頁:

819
不轉換為字碼頁 865

850
不轉換為字碼頁 865

HP-UX

字碼頁:

1051
不轉換為字碼頁 865

Windows

字碼頁:

865
不轉換為字碼頁 1051、1275

義大利文

義大利文 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上的義大利文原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	280, 924, 1144
AIX	819, 923, 5348

平台	原生 CCSID
HP-UX	819, 923, 1051
Windows	437, 850, 858, 1252, 5348
HP Integrity NonStop Server、 Solaris、 Linux	819, 923
Apple 用戶端	1275

所有非用戶端平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

IBM i

字碼頁：

280
不轉換為字碼頁 858、923、924、1275

924
不轉換為字碼頁 280、437、858、1051、1144、1252、1275、5348

1144
不轉換為字碼頁 924、1051、1275

西班牙文

西班牙文 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上西班牙文的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	284, 924, 1145
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	437, 850, 858, 1252, 5348
HP Integrity NonStop Server、 Solaris、 Linux	819, 923
Apple 用戶端	1275

所有非用戶端平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

IBM i

字碼頁：

284
不轉換為字碼頁 858、923、924、1275

924
不轉換為字碼頁 284、437、858、1051、1145、1252、1275、5348

1145
不轉換為字碼頁 924、1051、1275

英式英文/蓋爾語

英式英文/Gaelic 的 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上 UK English/Gaelic 的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	285, 924, 1146

平台	原生 CCSID
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	437, 850, 858, 1252, 5348
HP Integrity NonStop Server、 Solaris、 Linux	819, 923
Apple 用戶端	1275

所有非用戶端平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

IBM i

字碼頁：

285
不轉換為字碼頁 858、923、924、1275

924
不轉換為字碼頁 285、437、858、1051、1146、1252、1275、5348

1146
不轉換為字碼頁 924、1051、1275

法文

法文 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上法文的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	297, 924, 1147
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	437, 850, 858, 1252, 5348
HP Integrity NonStop Server、 Solaris、 Linux	819, 923
Apple 用戶端	1275

所有非用戶端平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

IBM i

字碼頁：

297
不轉換為字碼頁 858、923、924、1275、5348

924
不轉換為字碼頁 297、437、858、1051、1147、1252、1275、5348

1147
不轉換為字碼頁 924、1051、1275

多語言

「多種語言」的 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上多種語言轉換的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	500, 924, 1148
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	437, 850, 858, 1252, 5348
HP Integrity NonStop Server、 SINIX、Solaris、 Linux	819, 923
Apple 用戶端	1275

所有非用戶端平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

IBM i

字碼頁：

500
不轉換為字碼頁 858、923

924
不轉換為字碼頁 437、858、1051、1148、1252、1275、5348

1148
不轉換為字碼頁 924、1051、1275

葡萄牙文

葡萄牙文 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上葡萄牙文的原生 CCSID:

平台	原生 CCSID
IBM i	37, 500, 924, 1140
z/OS	500, 924, 1140
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	850, 858, 860, 1252, 5348
HP Integrity NonStop Server、 Solaris、 Linux	819, 923
Apple 用戶端	1275

所有非用戶端平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

IBM i

字碼頁：

37
不轉換為字碼頁 858、923、1275

500
不轉換為字碼頁 858、923、1275

924
不轉換為字碼頁 858、860、1051、1140、1252、1275、5348

1140
不轉換為字碼頁 860、924、1051、1275

HP-UX

字碼頁:

1051
不轉換為字碼頁 860

Windows

字碼頁:

860
不轉換為字碼頁 1051、1275

冰島文

冰島文 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上冰島文的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	871, 924, 1149
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	850, 858, 861, 1252, 5348
HP Integrity NonStop Server、 Solaris、 Linux	819, 923
Apple 用戶端	1275

所有非用戶端平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

IBM i

字碼頁:

871
不轉換為字碼頁 858、923、924、1275、5348

924
不轉換為字碼頁 858、861、871、1051、1149、1252、1275、5348

1149
不轉換為字碼頁 924、1051、1275

HP-UX

字碼頁:

1051
不轉換為字碼頁 861

Windows

字碼頁:

861
不轉換為字碼頁 1051、1275

東歐語

東歐語言 CCSID 及 CCSID 轉換的詳細資料。 使用這些 CCSID 的典型語言包括阿爾巴尼亞文、克羅埃西亞文、捷克文、匈牙利文、波蘭文、羅馬尼亞文、塞爾維亞文、斯洛伐克文及斯洛維尼亞文。

下表顯示受支援平台上東歐語言的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	870, 1153
Windows	852, 1250, 5346, 9044
AIX、 HP-UX、 HP Integrity NonStop Server、 Solaris、 Linux	912
東歐 Apple 用戶端	1282
羅馬尼亞文 Apple 用戶端	1285
克羅埃西亞文 Apple 用戶端	1284

所有非用戶端平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

z/OS

字碼頁:

870
不轉換為字碼頁 1284、1285

1153
不轉換為字碼頁 1250、1284、1285

IBM i

字碼頁:

870
不轉換為字碼頁 1284、1285、5346、9044

1153
不轉換為字碼頁 1282、1284、1285、5346、9044

HP-UX、Solaris、 Linux

字碼頁:

912
不轉換為字碼頁 1284、1285

HP Integrity NonStop Server

字碼頁:

912
不轉換為字碼頁 1153、1284、1285、9044

Windows

字碼頁:

852
不轉換為字碼頁 1284、1285

1250
不轉換為字碼頁 1284、1285

9044
不轉換為字碼頁 912、1282、1284、1285

斯拉夫文

斯拉夫語 CCSID 及 CCSID 轉換的詳細資料。 使用這些 CCSID 的典型語言包括白俄羅斯文、保加利亞文、馬其頓文、俄文和塞爾維亞文。

下表顯示受支援平台上斯拉夫語的原生 CCSID:

平台	原生 CCSID
z/OS	1025
IBM i	880, 1025
Windows	855, 866, 1131, 1251, 5347
Solaris	878, 915
AIX, HP-UX, Linux, HP Integrity NonStop Server	915
Apple 用戶端	1283

所有非用戶端平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

IBM i

字碼頁:

880
不轉換為字碼頁 855、866、878、1131、5347

1025
不轉換為字碼頁 878、5347

Windows

字碼頁:

855
不轉換為字碼頁 1131

866
不轉換為字碼頁 1131

1131
不轉換為字碼頁 855、866、880、1283

愛沙尼亞文

愛沙尼亞文的 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上愛沙尼亞文的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	1122, 1157
Windows	902, 922, 1257, 5353, 9449
AIX、 HP-UX、Solaris、 Linux	902, 922
HP Integrity NonStop Server	922

所有平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

z/OS

字碼頁:

1122
不轉換為字碼頁 902、1157、9449

1157
不轉換為字碼頁 922、1122、1257、9449

IBM i

字碼頁:

1122
不轉換為字碼頁 902、5353、9449

1157
不轉換為字碼頁 922、5353、9449

HP-UX、Solaris、 Linux

字碼頁:

902
不轉換為字碼頁 922、1122、9449

922
不轉換為字碼頁 902、1157、9449

Windows

字碼頁:

5353
不轉換為字碼頁 9449

9449
不轉換為字碼頁 902、922、1122、1157、1257、5353

902
不轉換為字碼頁 922、1122、9449

HP Integrity NonStop Server

字碼頁:

922
不轉換為字碼頁 902、1157、9449

拉脫維亞文和立陶宛文

拉脫維亞文和立陶宛文的 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上拉脫維亞文及立陶宛文的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	1112, 1156
Windows	901, 921, 1257, 5353, 9449
AIX、 HP-UX、Solaris、 Linux	901, 921
HP Integrity NonStop Server	921

所有平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

z/OS

字碼頁:

1112
不轉換為字碼頁 901、1156、9449

1156
不轉換為字碼頁 901、1156、9449

IBM i

字碼頁:

1112
不轉換為字碼頁 5353

1153
不轉換為字碼頁 921、5353、9449

HP-UX、Solaris、 Linux

字碼頁:

902
不轉換為字碼頁 921、1112、1257、9449

921
不轉換為字碼頁 901、1156、9449

Windows

字碼頁:

901
不轉換為字碼頁 921、1112、1257、9449

5355
不轉換為字碼頁 9449

9449
不轉換為字碼頁 901、921、1112、1156、1257

HP Integrity NonStop Server

字碼頁:

921
不轉換為字碼頁 901、1156、9449

烏克蘭文

烏克蘭文 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上 Ukranian 的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	1123
Windows	1124, 1125, 1251, 5347
AIX、 HP-UX、 HP Integrity NonStop Server、 Solaris、 Linux	1124

所有平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

IBM i

字碼頁:

1123
不轉換為字碼頁 5347

HP-UX

字碼頁:

1124
不轉換為字碼頁 5347

Windows

字碼頁:

1125
不轉換為字碼頁 1123

希臘文

希臘文 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上希臘文的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	875
HP-UX	813 (見附註)
Windows	869, 1253, 5349
AIX、NCR、 HP Integrity NonStop Server、 Solaris、 Linux	813
Apple 用戶端	1280
DOS 用戶端	737
註: HP-UX 只支援 ISO 字碼集。 HP-UX 專有 greek8 字碼集沒有已登錄的 CCSID ，因此不受支援。	

所有非用戶端平台都支援其原生 CCSID (其他平台的原生 CCSID) 之間的轉換，但有下列例外。

IBM i

字碼頁:

875
不轉換為字碼頁 5349

Windows

字碼頁:

1253
不轉換為字碼頁 737

5349
不轉換為字碼頁 737

土耳其文

土耳其文 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上土耳其文的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	1026

平台	原生 CCSID
HP-UX	920 (見附註)
Windows	857, 1254, 5350
AIX、 HP Integrity NonStop Server、 Solaris、 Linux	920
Apple 用戶端	1281
註: HP-UX 只支援 ISO 字碼集。 HP-UX 專有 turkish8 字碼集沒有已登錄的 CCSID , 因此不受支援。	

所有非用戶端平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

IBM i

字碼頁:

1026
不轉換為字碼頁 5350

猶太曆

希伯來文的 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上希伯來文的原生 CCSID:

平台	原生 CCSID
z/OS	424, 803, 4899, 12712
IBM i	424
AIX	916, 9048
HP-UX	916 (見附註)
Windows	1255, 5351
HP Integrity NonStop Server、 Solaris、 Linux	916
註: HP-UX 只支援 ISO 字碼集。 HP-UX 專有 greek8 字碼集沒有已登錄的 CCSID , 因此不受支援。	

所有平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

z/OS

字碼頁:

424
不轉換為字碼頁 867、4899、9048、12712

803
不轉換為字碼頁 867、4899、5351、9048、12712

4899
不轉換為字碼頁 424、803、856、862、916、1255

12712
不轉換為字碼頁 424、803、856、916、1255

IBM i

字碼頁:

424
不轉換為字碼頁 803、867、4899、5351、9048、12712

字碼頁 424 也會與 CCSID 4952 相互轉換，這是 856 的變式。

AIX

字碼頁：

916
不轉換為字碼頁 867、4899、9048、12712

9048
不轉換為字碼頁 424、803、856、862、916、1255

Windows

字碼頁：

1255
不轉換為字碼頁 867、4899、9048、12712

5351
不轉換為字碼頁 803

阿拉伯文

阿拉伯文 CCSID 及 CCSID 轉換的詳細資料

下表顯示受支援平台上阿拉伯文的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	420
AIX	1046, 1089
HP-UX	1089 (見附註)
Windows	720, 864, 1256, 5352
HP Integrity NonStop Server、 Solaris、 Linux	1089
註: HP-UX 只支援 ISO 字碼集。 HP-UX 專有 arabic8 字碼集沒有已登錄的 CCSID ， 因此不受支援。	

所有平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

IBM i

字碼頁：

420
不轉換為字碼頁 5352

HP-UX、Solaris、 Linux、 HP Integrity NonStop Server、 Tru64

字碼頁：

1089
不轉換為字碼頁 720

Windows

字碼頁：

720
不轉換為字碼頁 1089、5352

5352
不轉換為字碼頁 720

波斯人

波斯文 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上波斯文的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	1097
AIX、 HP-UX、 HP Integrity NonStop Server、 Solaris、 Linux Windows	1098 (見附註)
註: 這些平台的原生 CCSID 尚未標準化，且可能會變更。	

所有平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換。

烏都文

Urdu 的 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上 Urdu 的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	918
Windows	868
AIX、 HP-UX、 HP Integrity NonStop Server、 Solaris、 Linux	1006

所有平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

IBM i

字碼頁:

918
不轉換為字碼頁 1006

泰文

泰文 CCSID 及 CCSID 轉換的詳細資料。

下表顯示支援平台上泰文的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	838
AIX、 HP-UX、 HP Integrity NonStop Server、 Solaris、 Linux Windows	874 (見附註)
註: 這些平台的原生 CCSID 尚未標準化，且可能會變更。	

所有平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換。

寮文

寮文 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上寮文的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	1132
AIX、 HP-UX、 HP Integrity NonStop Server、 Solaris、 Linux Windows	1133

所有平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換。

越南文

越南文 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上越南文的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	1130
Windows	1258, 5354
AIX、 HP-UX、 HP Integrity NonStop Server、 Solaris、 Linux	1129

所有平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

IBM i

字碼頁:

1130
不轉換為字碼頁 1129、5354

日文拉丁文 *SBCS*

日文拉丁文 SBCS 的 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上日文拉丁文 SBCS 的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	1027
AIX	932、5050、33722 (請參閱附註 1)
Windows	932、943 (見附註 2 和 3)
Linux、 HP Integrity NonStop Server、 Solaris	943, 5050
HP-UX	不知道

註:

1. 5050 和 33722 是與 AIX 上基本字碼頁 954 相關的 CCSID。 作業系統所報告的 CCSID 是 33722。
2. Windows NT 使用字碼頁 932 ，但最好以 CCSID 943 表示。 不過，並非所有 WebSphere MQ 平台都支援此 CCSID。

   在 WebSphere MQ for Windows CCSID 932 上，用來代表字碼頁 932 ，但可以變更檔案 `../conv/table/ccsid.tbl` ，將使用的 CCSID 變更為 943。
3. WebSphere MQ 不支援以 JIS X 0213 (JIS2004) 標準為基礎的字碼頁。

所有平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

z/OS

字碼頁:

1027
不轉換為字碼頁 932、942、943、954、5050、33722

IBM i

字碼頁:

1027
不轉換為字碼頁 932

AIX

字碼頁:

932
不轉換為字碼頁 1027

5050
不轉換為字碼頁 1027

33722
不轉換為字碼頁 1027

Linux

字碼頁:

943
不轉換為字碼頁 1027

5050
不轉換為字碼頁 1027

Solaris

字碼頁:

943
不轉換為字碼頁 1027

5050
不轉換為字碼頁 1027

HP Integrity NonStop Server

字碼頁:

943
不轉換為字碼頁 1027

5050
不轉換為字碼頁 1027

日文片假名 *SBCS*

日文片假名 SBCS 的 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上日文片假名 SBCS 的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	290
HP-UX	897
AIX	932、5050、33722 (請參閱附註 1)

平台	原生 CCSID
Windows	932、943 (見附註 2 和 3)
Linux、 HP Integrity NonStop Server、 Solaris	943, 5050

註:

1. 5050 和 33722 是與 AIX 上基本字碼頁 954 相關的 CCSID。 作業系統所報告的 CCSID 是 33722。
2. Windows NT 使用字碼頁 932 ，但最好以 CCSID 943 表示。 不過，並非所有 WebSphere MQ 平台都支援此 CCSID。

   在 WebSphere MQ for Windows CCSID 932 上，用來代表字碼頁 932 ，但可以變更檔案 `../conv/table/ccsid.tbl` ，將使用的 CCSID 變更為 943。
3. WebSphere MQ 不支援以 JIS X 0213 (JIS2004) 標準為基礎的字碼頁。
4. 除了上述轉換之外， AIX、 HP-UX、Solaris、 Linux 及 Tru64 上的 WebSphere MQ 產品還支援從 CCSID 897 轉換至 CCSID 37、273、277、278、280、284、285、290、297、437、500、819、850、1027 及 1252。

所有平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

z/OS

字碼頁:

290
不轉換為字碼頁 932、943、954、5050、33722

IBM i

字碼頁:

290
不轉換為字碼頁 932

AIX

字碼頁:

932
不轉換為字碼頁 290、897

5050
不轉換為字碼頁 290、897

33722
不轉換為字碼頁 290、897

HP-UX

字碼頁:

897
不轉換為字碼頁 932、943、954、5050、33722

Linux

字碼頁:

943
不轉換為字碼頁 290、897

5050
不轉換為字碼頁 290、897

Solaris

字碼頁:

943
不轉換為字碼頁 290、897

5050
不轉換為字碼頁 290、897

HP Integrity NonStop Server

字碼頁:

943
不轉換為字碼頁 290、897

5050
不轉換為字碼頁 290、897

日文漢字/拉丁文混合

日文漢字/拉丁文混合的 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上日文 Kanji/Latin 混合的原生 CCSID:

平台	原生 CCSID
IBM i, z/OS	1399、5035 (見附註 1)
AIX	932、5050、33722 (請參閱附註 2)
HP-UX	932、954、5039 (見附註 3)
Windows	932、943 (見附註 4 和 5)
Linux、 HP Integrity NonStop Server、 Solaris	943, 5050

註:

1. 5035 是與字碼頁 939 相關的 CCSID
2. 5050 和 33722 是與 AIX 上基本字碼頁 954 相關的 CCSID。 作業系統所報告的 CCSID 是 33722。
3. HP-UX 上的字碼集 `japan15` 和 SJIS 以 CCSID 932 表示。 這些 DBCS 字元在 SJIS 中具有不同的表示法，因此如果未在 HP-UX 系統上執行轉換，則 932 可能轉換不正確。 WebSphere MQ for HP-UX 支援 5039 ，這是 HP SJIS 的正確 CCSID。 可以對檔案 `/var/mqm/conv/ccsid.tbl` 進行變更，以將使用的 CCSID 從 932 變更為 5039。
4. Windows NT 使用字碼頁 932 ，但最好以 CCSID 943 表示。 不過，並非所有 WebSphere MQ 平台都支援此 CCSID。

   在 WebSphere MQ for Windows CCSID 932 上，用來代表字碼頁 932 ，但可以變更檔案 `../conv/table/ccsid.tbl` ，將使用的 CCSID 變更為 943。
5. WebSphere MQ 不支援以 JIS X 0213 (JIS2004) 標準為基礎的字碼頁。

所有平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

z/OS

字碼頁:

1399
不轉換為字碼頁 954、5035、5050、33722

5035
不轉換為字碼頁 954、1399、5050、33722

IBM i

字碼頁:

1399
不轉換為字碼頁 5039

5035
不轉換為字碼頁 5039

HP-UX

字碼頁:

932
不轉換為字碼頁 942、943、1399

954
不轉換為字碼頁 942、943、1399

5039
不轉換為字碼頁 942、943、1399

HP Integrity NonStop Server

字碼頁:

943
不轉換為字碼頁 1399

5050
不轉換為字碼頁 1399

日文漢字/片假名混合

日文漢字/片假名混合的 CCSID 及 CCSID 轉換的詳細資料。

表 *582:* 在支援平台上混合日文漢字/片假名的原生 *CCSID*

平台	原生 CCSID
z/OS	1390、5026 (見附註 1)
IBM i	5026 (見附註 1)
AIX	932、5050、33722 (請參閱附註 2)
HP-UX	932、954、5039 (見附註 3)
Windows	932、943 (見附註 4 和 5)
Linux、 HP Integrity NonStop Server、 Solaris	943, 5050

表 *582:* 在支援平台上混合日文漢字/片假名的原生 *CCSID (繼續)*

平台	原生 **CCSID**

註:

1. CCSID 1390 不接受小寫字元。 5026 是與字碼頁 930 相關的 CCSID。 選取日文片假名 (DBCS) 特性時， CCSID 5026 是 IBM i 上所報告的 CCSID。
2. 5050 和 33722 是與 AIX 上基本字碼頁 954 相關的 CCSID。 作業系統所報告的 CCSID 是 33722。
3. HP-UX 上的字碼集 `japan15` 和 SJIS 以 CCSID 932 表示。 這些 DBCS 字元在 SJIS 中具有不同的表示法，因此如果未在 HP-UX 系統上執行轉換，則 932 可能轉換不正確。 WebSphere MQ for HP-UX 支援 5039 ，這是 HP SJIS 的正確 CCSID。 可以對檔案 `/var/mqm/conv/ccsid.tbl` 進行變更，以將使用的 CCSID 從 932 變更為 5039。
4. Windows NT 使用字碼頁 932 ，但最好以 CCSID 943 表示。 不過，並非所有 WebSphere MQ 平台都支援此 CCSID。

   在 WebSphere MQ for Windows 上， CCSID 932 用來代表字碼頁 932 ，但可以對檔案 `../conv/table/ccsid.tbl` 進行變更，將使用的 CCSID 變更為 943。
5. WebSphere MQ 不支援以 JIS X 0213 (JIS2004) 標準為基礎的字碼頁。

所有平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

z/OS

字碼頁:

1390
不轉換為字碼頁 954、5026、5050、33722

不接受小寫字元。

5026
不轉換為字碼頁 954、1390、5050、33722

IBM i

字碼頁:

5026
不轉換為字碼頁 1390、5039

HP-UX

字碼頁:

932
不轉換為字碼頁 942、943、1390

954
不轉換為字碼頁 942、943、1390

5039
不轉換為字碼頁 942、943、1390

HP Integrity NonStop Server

字碼頁:

943
不轉換為字碼頁 1390

5050
不轉換為字碼頁 1390

韓文

韓文 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上韓文的原生 CCSID:

平台	原生 CCSID
z/OS, IBM i	933, 1364
AIX、 HP-UX、 Linux、 HP Integrity NonStop Server、 Solaris	970
Windows	949, 1363

所有平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

z/OS

字碼頁:

933
不轉換為字碼頁 970

1364
不轉換為字碼頁 970

HP-UX

字碼頁:

970
不轉換為字碼頁 949、1363、1364

簡體中文

簡體中文的 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上簡體中文的原生 CCSID:

平台	原生 CCSID
z/OS	935, 1388
IBM i	935, 1388
AIX	1383, 1386
HP-UX	1381 (見附註 1)
Windows	1381、1386 (見附註 2)
Linux、 HP Integrity NonStop Server、 Solaris	1383

平台	原生 CCSID
註: 1. HP-UX 上的字碼集 prc15 和 hp15CN 以 CCSID 1381 表示。 2. Windows 使用字碼頁 936，但最好以 CCSID 1386 來表示。 不過，並非所有 WebSphere MQ 平台都支援此 CCSID。 在 WebSphere MQ for Windows 上， CCSID 1381 是用來代表字碼頁 936，但可以變更檔案 ../conv/table/ccsid.tbl，將所用的 CCSID 變更為 1386。 3. WebSphere MQ 支援中文 GB18030 標準的第一階段。 在 z/OS、 Linux、 Windows 及 Solaris 上，提供 Unicode (UTF-8 及 UCS-2) 與 CCSID 1388 (EBCDIC 含 GB18030 延伸)、Unicode (UTF-8 及 UCS-2) 與 CCSID 5488 (GB18030 階段 1) 之間以及 CCSID 1388 與 CCSID 5488 之間的轉換支援。 **註:** 在 IBM i 上，作業系統支援 Unicode (UTF-8 及 UCS-2) 與 CCSID 1388 (EBCDIC 含 GB18030 延伸) 之間的轉換。 在 HP-UX 上，目前沒有 GB18030 的 HP11 作業系統支援。 在 HP11i 上，修補程式 PHCO_26456 提供 GB18030 (CCSID 5488) 與 Unicode 之間的轉換支援。 不支援 GB18030 與 1388 (EBCDIC) 之間的轉換。	

所有平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

z/OS

字碼頁:

935
不轉換為字碼頁 1383

1388
不轉換為字碼頁 1383

HP-UX

字碼頁:

1381
不轉換為字碼頁 1383、1386、1388

繁體中文

繁體中文 CCSID 及 CCSID 轉換的詳細資料。

下表顯示受支援平台上繁體中文的原生 CCSID:

平台	原生 CCSID
z/OS, IBM i	937
HP-UX	938、950、964 (請參閱附註)
Windows	950
AIX、 HP Integrity NonStop Server、 Solaris、 Linux	950, 964
註: HP-UX 上的字碼集 roc15 以 CCSID 938 表示。	

所有平台都支援其原生 CCSID 與其他平台的原生 CCSID 之間的轉換，但下列例外。

z/OS

字碼頁:

937
不轉換為字碼頁 964

1388
不轉換為字碼頁 1383

HP-UX

字碼頁:

938
不轉換為字碼頁 948

950
不轉換為字碼頁 948

964
不轉換為字碼頁 948

Linux、Solaris

字碼頁:

964
不轉換為字碼頁 938

Unicode 轉換支援

部分平台支援使用者資料與 Unicode 編碼之間的轉換。 支援兩種形式的 Unicode 編碼: UCS-2 (CCSID 1200、13488 和 17584) 和 UTF-8 (CCSID 1208)。

術語 *UCS-2* 通常可交換使用，但不正確地與 *UTF-16* 搭配使用。 UCS-2 是固定寬度編碼，其中每一個字元佔用 2 個位元組。 UTF-16 是 UCS-2 超集的可變寬度編碼。 除了 2 位元組 UCS-2 字元之外， UTF-16 還包含長度為 4 個位元組的字元 (稱為代理配對)。 WebSphere MQ 不支援代理配對。 因此， WebSphere MQ 中對 UTF-16 和 UTF-8 的支援僅限於那些可以在 UCS-2 中編碼的 Unicode 字元。

註: WebSphere MQ 不支援 UCS-2 佇列管理程式 CCSID ，因此無法在 UCS-2 中編碼訊息標頭資料。

WebSphere MQ AIX Unicode 支援

在 WebSphere MQ for AIX 上，下表中的 CCSID 支援來回轉換 Unicode CCSID。

037	273	278	280	284	285
297	423	437	500	813	819
850	852	856	857	858	860
861	865	867	869	875	878
880	901	902	912	915	916
920	923	924	932	933	935
937	938	939	942	943	948
949	950	954	964	970	1026
1046	1089	1129	1130	1131	1132
1133	1140	1141	1142	1143	1144
1145	1146	1147	1148	1149	1200

1153	1156	1157	1208	1250	1251
1253	1254	1258	1280	1281	1282
1283	1284	1285	1363	1364	1381
1383	1386	1388	4899	5026	5035
5050	5346	5347	5348	5349	5350
5351	5352	5353	5354	5488	9044
9048	9449	12712	13488	17584	33722

WebSphere MQ HP-UX 支援 Unicode

在 WebSphere MQ for HP-UX 上，下表列出的 CCSID 支援與 Unicode CCSID 之間的轉換。

437	737	813	819	850	852
855	857	861	864	865	866
869	874	912	915	916	920
932	938	950	954	964	970
1051	1089	1140	1141	1142	1143
1144	1145	1146	1147	1148	1149
1200	1208	1250	1251	1252	1253
1254	1255	1256	1257	1258	1381
5050	5488	13488	33722		

WebSphere MQ for Windows、Solaris 及 Linux Unicode 支援

在 WebSphere MQ for Windows、 WebSphere MQ for Solaris 及 WebSphere MQ for Linux 上，下表中的 CCSID 支援與 Unicode CCSID 之間的來回轉換。

037	277	278	280	284	285
290	297	300	301	420	424
437	500	813	819	833	835
836	837	838	850	852	855
856	857	858	860	861	862
863	864	865	866	867	868
869	870	871	874	875	878
880	891	897	901	902	903
904	912	913 (5)	915	916	918
920	921	922	923	924	927
928	930	931 (1)	932 (2)	933	935
937	938 (3)	939	941	942	943
947	948	949	950	951	954 (4)
964	970	1006	1025	1026	1027
1040	1041	1042	1043	1046	1047
1051	1088	1089	1097	1098	1112

1114	1115	1122	1123	1124	1129
1130	1132	1133	1140	1141	1142
1143	1144	1145	1146	1147	1148
1149	1153	1156	1157	1200	1208
1250	1251	1252	1253	1254	1255
1256	1257	1258	1275	1280	1281
1282	1283	1363	1364	1380	1381
1383	1386	1388	4899	5050	5346
5347	5348	5349	5350	5351	5352
5353	5354	5488 (5)	9044	9048	9449
12712	13488	17584	33722 (4)		

附註:

1. 931 使用 939 進行轉換。
2. 932 使用 942 進行轉換。
3. 938 使用 948 進行轉換。
4. 954 和 33722 使用 5050 進行轉換。
5. 僅在 Windows、 Linux 及 Solaris 上。

IBM i 支援 Unicode

如需 UNICODE 支援的詳細資料，請參閱與作業系統相關的適當 IBM i 出版品。

WebSphere MQ for z/OS Unicode 支援

在 WebSphere MQ for z/OS 上，下列 CCSID 支援來回轉換 Unicode CCSID:

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	834	835	836	837	838
848	849	850	851	852	855
856	857	858	859	860	861
862	863	864	865	866	867
868	869	870	871	872	874
875	878	880	891	895	896
897	901	902	903	904	905
912	914	915	916	918	920
921	922	923	924	927	928
930	932	933	935	937	939
941	942	943	944	946	947

948	949	950	951	1004	1006
1008	1009	1010	1011	1012	1013
1014	1015	1016	1017	1018	1019
1025	1026	1027	1040	1041	1042
1043	1046	1047	1051	1088	1089
1097	1098	1112	1114	1115	1122
1123	1124	1125	1126	1129	1130
1131	1132	1133	1137	1140	1141
1142	1143	1144	1145	1146	1147
1148	1149	1153	1154	1155	1156
1157	1158	1159	1160	1161	1162
1164	1200	1208	1250	1251	1252
1253	1254	1255	1256	1257	1258
1275	1276	1277	1280	1281	1282
1283	1284	1285	1351	1362	1363
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	5347	5348	5349	5350
5351	5352	5353	5354	5488	8482
8612	9027	9030	9044	9048	9049
9056	9061	9066	9238	9449	12712
13121	13218	13488	16684	16804	17248
17584	21427	28709			

64 位元平台上的編碼標準

使用此資訊來瞭解 64 位元平台上的編碼標準以及偏好的資料類型。

偏好的資料類型

這些類型永不變更大小，同時適用於 32 位元和 64 位元 WebSphere MQ 平台:

姓名	長度
MQLONG	4 個位元組
MQULONG	4 個位元組
MQINT32	4 個位元組
MQUINT32	4 個位元組
MQINT64	8 個位元組
MQUINT64	8 個位元組

標準資料類型

瞭解 32 位元 UNIX、64 位元 UNIX 及 64 位元 Windows 應用程式上的標準資料類型。

32 位元 UNIX 應用程式

本節包含用於比較，並以 Solaris 為基礎。 請注意與其他 UNIX 平台的任何差異:

姓名	長度
char	1 個位元組
短	2 個位元組
整數	4 個位元組
長整數	4 個位元組
浮點數	4 個位元組
倍精準數	8 個位元組
長雙	16 個位元組 請注意，在 AIX 和 Linux PPC 上，長倍精準數是 8 位元組。
指標	4 個位元組
ptrdiff_t	4 個位元組
size_t	4 個位元組
time_t	4 個位元組
clock_t	4 個位元組
wchar_t	4 個位元組 請注意，在 AIX 上， wchar_t 是 2 個位元組。

64 位元 UNIX 應用程式

本節以 Solaris 為基礎。 請注意與其他 UNIX 平台的任何差異:

姓名	長度
char	1 個位元組
短	2 個位元組
整數	4 個位元組
長整數	8 個位元組
浮點數	4 個位元組
倍精準數	8 個位元組
長雙	16 個位元組 請注意，在 AIX 和 Linux PPC 上，長倍精準數是 8 位元組。
指標	8 個位元組
ptrdiff_t	8 個位元組
size_t	8 個位元組
time_t	8 個位元組

姓名	長度
clock_t	8 個位元組 請注意，在其他 UNIX 平台上， clock_t 是 4 個位元組。
wchar_t	4 個位元組 請注意，在 AIX 上， wchar_t 是 2 個位元組。

Windows 64 位元應用程式

姓名	長度
char	1 個位元組
短	2 個位元組
整數	4 個位元組
長整數	4 個位元組
浮點數	4 個位元組
倍精準數	8 個位元組
長雙	8 個位元組
指標	8 個位元組 請注意，所有指標都是 8 個位元組。
ptrdiff_t	8 個位元組
size_t	8 個位元組
time_t	8 個位元組
clock_t	4 個位元組
wchar_t	2 個位元組
單字	2 個位元組
DWORD	4 個位元組
控點	8 個位元組
HFILE	4 個位元組

Windows 上的編碼考量

HANDLE hf;

使用

```
hf = CreateFile((LPCTSTR) FileName,
                Access,
                ShareMode,
                xihSecAttsNTRestrict,
                Create,
                AttrAndFlags,
                NULL);
```

不使用

```
HFILE hf;
hf = (HFILE) CreateFile((LPCTSTR) FileName,
                        Access,
```

```
                        ShareMode,
                        xihSecAttsNTRestrict,
                        Create,
                        AttrAndFlags,
                        NULL);
```

因為這會產生錯誤。

size_t len fgets

使用

```
size_t len
while (fgets(string1, (int) len, fp) != NULL)
len = strlen(buffer);
```

不使用

```
int len;

while (fgets(string1, len, fp) != NULL)
len = strlen(buffer);
```

printf

使用

```
printf("My struc pointer: %p", pMyStruc);
```

不使用

```
printf("My struc pointer: %x", pMyStruc);
```

如果您需要十六進位輸出，則必須分別列印大寫及小寫 4 個位元組。

char * ptr

使用

```
char * ptr1;
char * ptr2;
size_t bufLen;

bufLen = ptr2 - ptr1;
```

不使用

```
char *ptr1;
char *ptr2;
UINT32 bufLen;

bufLen = ptr2 - ptr1;
```

alignBytes

使用

```
alignBytes = (unsigned short) ((size_t) address % 16);
```

不使用

```
void *address;
unsigned short alignBytes;

alignBytes = (unsigned short) ((UINT32) address % 16);
```

len

使用

```
len = (UINT32) ((char *) address2 - (char *) address1);
```

不使用

```
void *address1;
void *address2;
UINT32 len;

len = (UINT32) ((char *) address2 - (char *) address1);
```

斯坎夫

使用

```
MQLONG SBCSprt;

sscanf(line, "%d", &SBCSprt);
```

不使用

```
MQLONG SBCSprt;

sscanf(line, "%1d", &SBCSprt);
```

%ld 會嘗試將 8 位元組類型放入 4 位元組類型; 只有在處理實際 long 資料類型時，才會使用 %l 。MQLONG、 UINT32 及 INT32 定義為四個位元組，與所有 WebSphere MQ 平台上的 int 相同:

SOAP 參照

WebSphere MQ SOAP 參照資訊的傳輸依字母順序排列。

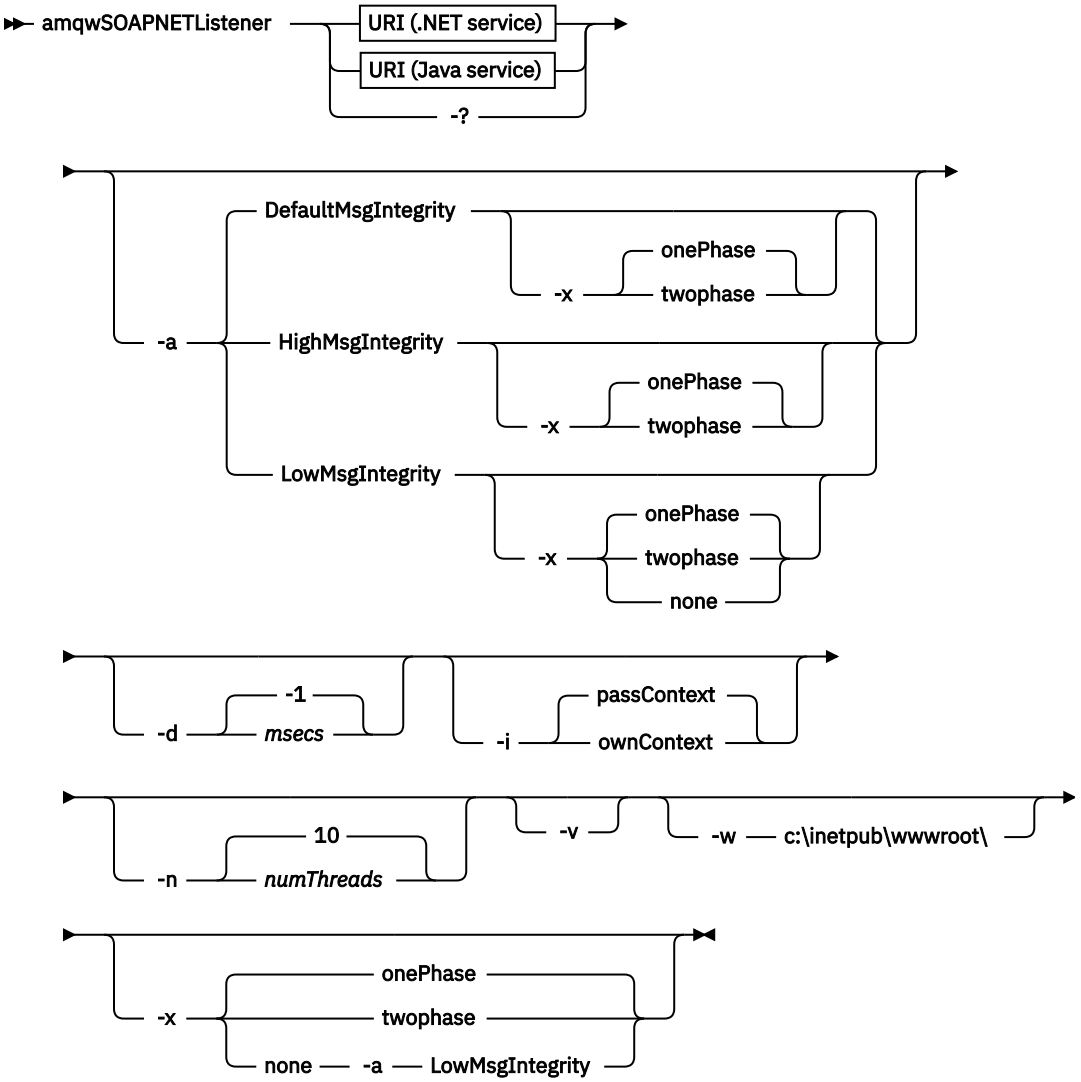
amqwSOAPNETListener: IBM WebSphere MQ .NET Framework 1 或 2 的 SOAP 接聽器

WebSphere MQ SOAP 接聽器 for .NET Framework 1 或 2 的語法和參數。

用途

啟動 IBM WebSphere MQ SOAP listener for .NET Framework 1 或 2。

.NET



必要參數

URI 平台
請參閱 第 885 頁的『Web 服務部署的 URI 語法和參數』。

-?
印出說明如何使用指令的說明文字。

選用參數

-a *integrityOption*
integrityOption 指定 WebSphere MQ SOAP 接聽器的行為 (如果無法將失敗要求訊息放置在無法傳送郵件的佇列上)。 *integrityOption* 可以採用下列其中一個值:

DefaultMsgIntegrity
對於非持續訊息，接聽器會顯示警告訊息，並繼續執行，並捨棄原始訊息。 若為持續訊息，它會顯示一則錯誤訊息，取消要求訊息，讓它保留在要求佇列上並結束。 如果省略 -a 選項，或未指定 *integrityOption* ，則會套用 DefaultMsgIntegrity 。

LowMsgIntegrity
對於持續及非持續訊息，接聽器會顯示警告並繼續執行，捨棄訊息。

HighMsgIntegrity
對於持續及非持續訊息，接聽器會顯示一則錯誤訊息，取消要求訊息，使其保留在要求佇列上並結束。

部署公用程式會檢查 -x 和 -a 旗標的相容性。 如果指定 -x none ，則必須指定 -a LowMsgIntegrity 。 如果旗標不相容，則部署公用程式會結束並顯示錯誤訊息，且未執行任何部署步驟。

-d 毫秒
毫秒 指定在任何執行緒上收到要求訊息時， WebSphere MQ SOAP 接聽器保持作用中的毫秒數。 如果 *msecs* 設為 -1，則接聽器會無限期地保持作用中。

-i 環境定義
環境定義 指定接聽器是否傳遞身分環境定義。 環境定義 採用下列值:

passContext
在回應訊息中設定原始要求訊息的身分環境定義。 SOAP 接聽器會檢查它是否有權從要求佇列儲存環境定義，並將它傳遞至回應佇列。 當開啟要儲存環境定義的要求佇列，以及要傳遞環境定義的回應佇列時，它會在執行時期進行檢查。 如果它沒有必要的權限，或 MQOPEN 呼叫失敗，則不會處理回應訊息。 回應訊息會放置在無法傳送郵件的佇列中，且無法傳送郵件的標頭包含來自失敗 MQOPEN 的回覆碼。 然後，接聽器會繼續正常處理後續的送入訊息。

ownContext
SOAP 接聽器不傳遞環境定義。 傳回的環境定義反映接聽器執行時所使用的使用者 ID ，而不是建立原始要求訊息的使用者 ID。

原始環境定義中的欄位是由佇列管理程式所設定，而不是由 SOAP 接聽器所設定。

-n *numThreads*
numThreads 指定 WebSphere MQ SOAP 接聽器的已產生啟動 Script 中的執行緒數目。 預設值為 10。 如果您有高訊息傳輸量，請考慮增加此數目。

-v
-v 設定外部指令的詳細輸出。 一律會顯示錯誤訊息。 使用 -v 來輸出您可以自訂以建立自訂部署 Script 的指令。

-w *serviceDirectory*
serviceDirectory 是包含 Web 服務的目錄。

-x 交易性
交易方式 指定接聽器的交易式控制項類型。 交易方式 可以設為下列其中一個值:

onePhase
使用 IBM WebSphere MQ 一段式支援。 如果系統在處理期間失敗，則會將要求訊息重新遞送至應用程式。 WebSphere MQ 交易可確保只寫入回應訊息一次。

twoPhase
使用兩階段支援。 如果適當地撰寫服務，則在服務的單一已確定執行內，訊息只會遞送一次 (與其他資源協調)。 這個選項只適用於伺服器連結連線。

none
無交易式支援。 如果系統在處理期間失敗，即使持續保存，也可能會遺失要求訊息。 服務可能已執行，也可能未執行，且可能寫入回應、報告或無法傳送郵件的訊息。

部署公用程式會檢查 -x 和 -a 旗標的相容性。 如需詳細資料，請參閱 -a 旗標的說明。

.NET 範例

```
amqwSOAPNETlistener
-u "jms:/queue?destination=myQ&connectionFactory=()
&targetService=myService&initialContextFactory=com.ibm.mq.jms.Nojndi"
-w C:/wmqsoap/demos
-n 20
```

amqswsdl: 產生 WSDL for .NET Framework 1 或 2 服務

amqswsdl 會採用針對 .NET Framework 1 或 2 撰寫的 Web 服務，並產生類別的 WSDL ，將您為 WebSphere MQ SOAP 傳輸提供的 URI 插入所產生的 WSDL 中。

用途
使用 **amqswsdl** 來產生 WSDL，其中包含部署至 WebSphere MQ 之服務的 URI。 使用 WSDL 來產生用戶端 Proxy。

參數
***escapedUri* (輸入)**
服務的 URI，所有 "&" 都跳出至 "&amp.";。 例如:

```
"jms:/queue?destination=REQUESTDOTNET
&amp.initialContextFactory=com.ibm.mq.jms.Nojndi
&amp.connectionFactory=(connectQueueManager(QM1)binding(server))
&amp.targetService=Quote.asmx"
```

***className*.asmx (輸入)**
服務類別。

***className*.wsdl (輸出)**
服務 WSDL。

說明
如果使用幕後程式碼程式設計模型來實作類別，您必須建置 *className*.dll 並將它儲存在 ./bin 中。

amqwclientconfig: 建立 Axis 1.4 WebSphere MQ SOAP 傳輸的 Web 服務用戶端部署描述子
amqwclientconfig 會建立 client-config.wsdd Axis 1.4 用戶端部署描述子檔案。

用途
它會將 jms:/ 傳輸新增至描述子，並將 java:com.ibm.mq.soap.transport.jms.WMQSender 登錄為類別，以處理 jms: 傳輸的 SOAP 要求。

語法

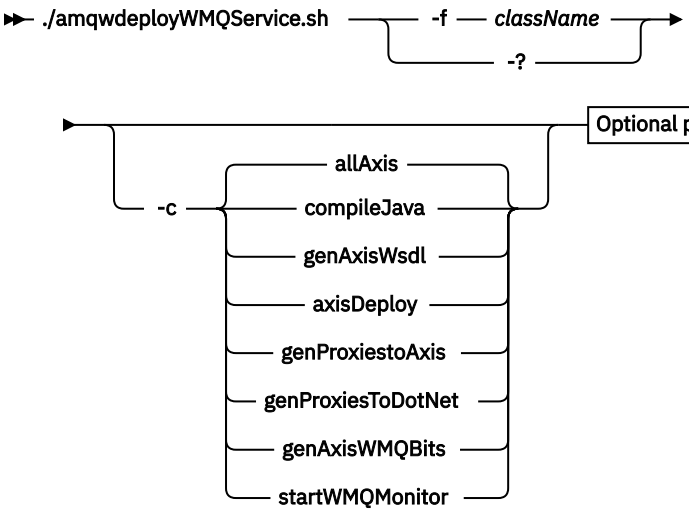
說明
amqwclientconfig 會呼叫 **amqwsetcp** 來設定 CLASSPATH，並執行下列指令:

```
java org.apache.axis.utils.Admin client "%WMQSOAP_HOME%\bin\amqwclientTransport.wsdd"
```

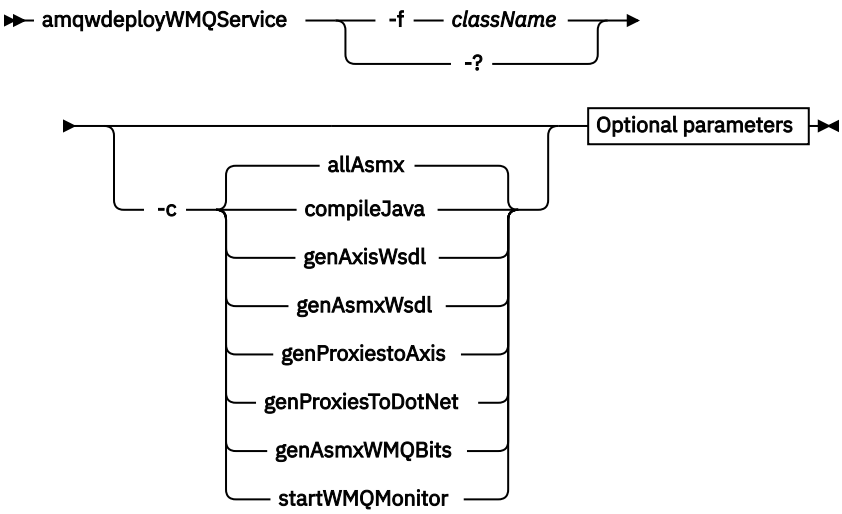
amqwdeployWMQService: 部署 Web 服務公用程式
部署公用程式會準備服務類別，以使用 WebSphere MQ 作為傳輸來作為 Web 服務。

用途
使用部署公用程式來產生部署 Axis 1.4、.NET Framework 1 或 .NET Framework 2 服務所需的檔案。 使用這些檔案來部署 IBM WebSphere MQ 所呼叫的服務。 **amqwdeployWMQService** 所產生的檔案顯示在 第 858 頁的『來自 amqwdeployWMQService 的輸出檔』中。

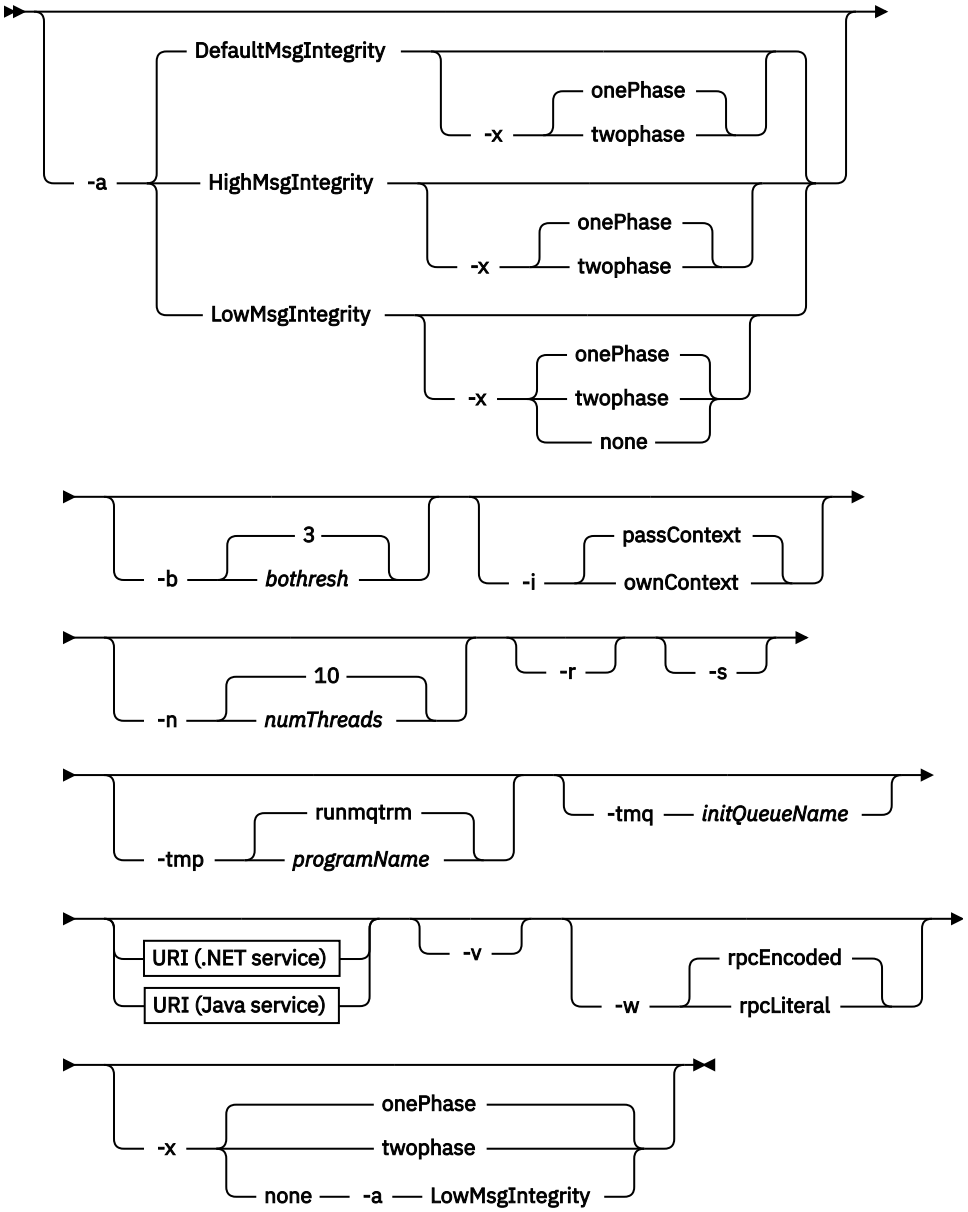
Syntax diagram
UNIX and Linux systems



Windows



Optional parameters



必要參數

-f ***className***

className 是要部署的類別名稱。 若為 Axis 服務 *className* 是 Java 原始檔，若為 .NET 服務，則是 `.asmx` 檔。 第 855 頁的圖 11 說明 Axis 服務和 .NET 服務 第 855 頁的圖 12 的部署。

```
amqwdeployWMQService -f javaDemos/service/StockQuoteAxis.java
```

圖 *11: Axis* 服務的部署範例

```
amqwdeployWMQService -f StockQuoteDotNet.asmx
```

圖 *12: .NET* 服務的部署範例

若為 Java， *className* 必須由套件名稱完整定義。 它可以指定為具有目錄分隔字元的路徑名稱，或指定為具有句點分隔字元的類別名稱。 產生的類別位於 `./generated/client/remote/`*path name* 中。 對於 .NET 服務，雖然可以指定目錄，但產生的 Java Proxy 一律位於 `./generated/client/remote/dotNetService` 中。

如果您使用 -u 選項指定 URI，且在 URI 內指定 *targetService*，則部署公用程式會檢查 *className*。 *className* 必須符合 *targetService*。 如果類別與服務不相符，部署公用程式會顯示錯誤訊息並結束。

-?
印出說明如何使用指令的說明文字。

選用參數

-a ***integrityOption***
integrityOption 指定 WebSphere MQ SOAP 接聽器的行為 (如果無法將失敗要求訊息放置在無法傳送郵件的佇列上)。 *integrityOption* 可以採用下列其中一個值:

DefaultMsgIntegrity
對於非持續訊息，接聽器會顯示警告訊息，並繼續執行，並捨棄原始訊息。 若為持續訊息，它會顯示一則錯誤訊息，取消要求訊息，讓它保留在要求佇列上並結束。 如果省略 -a 選項，或未指定 *integrityOption*，則會套用 DefaultMsgIntegrity。

LowMsgIntegrity
對於持續及非持續訊息，接聽器會顯示警告並繼續執行，捨棄訊息。

HighMsgIntegrity
對於持續及非持續訊息，接聽器會顯示一則錯誤訊息，取消要求訊息，使其保留在要求佇列上並結束。

部署公用程式會檢查 -x 和 -a 旗標的相容性。 如果指定 -x none，則必須指定 -a LowMsgIntegrity。 如果旗標不相容，則部署公用程式會結束並顯示錯誤訊息，且未執行任何部署步驟。

-b ***bothresh***
bothresh 指定要求佇列的取消臨界值設定。 預設是 3。

-c 作業
operation 指定要執行部署程序的哪個部分。 *operation* 是下列其中一個選項:

allAxis
執行 Axis 或 Java 服務的所有編譯及設定步驟 [4].

compileJava
將 Java 服務 `.java` 編譯為 `.class`。

genAxisWsdl
產生 WSDL: `.class` 至 `.wsdl`。

axisDeploy
將類別檔 `.wsdl` 部署至 `.wsdd`，套用 `.wsdd`。

genProxiestoAxis
產生 Proxy: `.wsdl` 至 `.java` 和 `.class`。

genAxisWMQBits
設定 Axis 服務的 IBM WebSphere MQ 佇列、 IBM WebSphere MQ SOAP 接聽器及觸發程式。

allAsmx
執行 .NET 服務的所有設定步驟 [5].

genAsmxWsdl
產生 WSDL: `.asmx` 至 `.wsdl`。

genProxiesToDotNet
產生 Proxy: `.wsdl` 至 `.java`、 `.class`、 `.cs` 和 `.vb`。

genAsmxWMQBits
設定 IBM WebSphere MQ 佇列、 IBM WebSphere MQ SOAP 接聽器及觸發程式

startWMQMonitor
啟動 WebSphere MQ SOAP 服務的觸發監視器。

---

[4] 如果 *className* 具有 .java 副檔名，則為預設值
[5] 如果 *className* 具有 .asmx 副檔名，則為預設值。

註: runmqtrm 以 mqm 使用者 ID 執行。 如果安全是一個問題，您必須確保以適當的使用者 ID 來啟動接聽器。

-i 環境定義
環境定義 指定接聽器是否傳遞身分環境定義。 環境定義 採用下列值:

passContext
在回應訊息中設定原始要求訊息的身分環境定義。 SOAP 接聽器會檢查它是否有權從要求佇列儲存環境定義，並將它傳遞至回應佇列。 當開啟要儲存環境定義的要求佇列，以及要傳遞環境定義的回應佇列時，它會在執行時期進行檢查。 如果它沒有必要的權限，或 MQOPEN 呼叫失敗，則不會處理回應訊息。 回應訊息會放置在無法傳送郵件的佇列中，且無法傳送郵件的標頭包含來自失敗 MQOPEN 的回覆碼。 然後，接聽器會繼續正常處理後續的送入訊息。

ownContext
SOAP 接聽器不傳遞環境定義。 傳回的環境定義反映接聽器執行時所使用的使用者 ID，而不是建立原始要求訊息的使用者 ID。

原始環境定義中的欄位是由佇列管理程式所設定，而不是由 SOAP 接聽器所設定。

-n *numThreads*
numThreads 指定 WebSphere MQ SOAP 接聽器的已產生啟動 Script 中的執行緒數目。 預設值為 10。 如果您有高訊息傳輸量，請考慮增加此數目。

-r
-r 指定取代任何現有的要求或觸發監視器佇列定義。 只有在同時指定 -tmq 時，才會取代觸發監視器佇列。 使用標準預設屬性重建佇列，並刪除佇列上的現有訊息。 如果未使用 -r 選項，則不會變更任何現有的佇列定義，也不會刪除現有的訊息。 不指定 -r，可確保保留任何自訂佇列屬性。

-s
將接聽器配置成以 WebSphere MQ 服務形式執行。 如果同時指定 -s 及 -tmq，則部署公用程式會顯示錯誤訊息並結束。

-tmp *programName*
programName 指定觸發監視器程式的名稱。 在 UNIX 或 Linux 環境中使用 -tmp *programName*，作為使用 **runmqtrm** 的替代方案。 它在 mqm 權限下起始執行的程式。

例如:

```
amqwdeployWMQService -f javaDemos/service/StockQuoteAxis.java
-tmq trigger.monitor.queue -tmp trigmon
```

-tmq *queueName*
queueName 指定觸發監視器佇列名稱。 建立 IBM WebSphere MQ 程序定義以配置自動觸發 WebSphere MQ SOAP 接聽器，以及相關聯的觸發監視器佇列名稱。 如果未指定選項，則部署公用程式不會定義任何觸發配置。 如果同時指定 -s 及 -tmq，則部署公用程式會顯示錯誤訊息並結束。

URI 平台
請參閱 第 885 頁的『Web 服務部署的 URI 語法和參數』。

-v
-v 設定外部指令的詳細輸出。 一律會顯示錯誤訊息。 使用 -v 來輸出您可以自訂以建立自訂部署 Script 的指令。

-w
-w 控制要產生的 WSDL 樣式。 預設值為 rpcEncloded，以與舊版 WebSphere MQ SOAP 傳輸相容。 使用 rpcLiteral 來建立與 Axis2 用戶端 Proxy 產生相容的 WSDL。 rpcEncoded 與 WS-I 建議不相容。

-x 交易性
交易方式 指定接聽器的交易式控制項類型。 交易方式 可以設為下列其中一個值:

onePhase
使用 IBM WebSphere MQ 一段式支援。 如果系統在處理期間失敗，則會將要求訊息重新遞送至應用程式。 WebSphere MQ 交易可確保只寫入回應訊息一次。

twoPhase
使用兩階段支援。 如果適當地撰寫服務，則在服務的單一已確定執行內，訊息只會遞送一次 (與其他資源協調)。 這個選項只適用於伺服器連結連線。

none
無交易式支援。 如果系統在處理期間失敗，即使持續保存，也可能會遺失要求訊息。 服務可能已執行，也可能未執行，且可能寫入回應、報告或無法傳送郵件的訊息。

部署公用程式會檢查 -x 和 -a 旗標的相容性。 如需詳細資料，請參閱 -a 旗標的說明。

錯誤

在 Windows 上，如果從 **amqswsdl** 報告錯誤，請嘗試發出下列指令，將 .asmx 檔案登錄為服務。

```
%windir%/Microsoft.NET/Framework/version number/aspnet_regiis.exe -ir
```

在未安裝 IIS 的系統上，或在 NET 之後安裝 IIS 時，通常會發生問題。 當 **amqswsdl** 產生 .wsdl 檔案時發生問題。

註: 也需要登錄機碼，以允許接聽器呼叫服務。 如果您使用自己的自訂部署程序，則在執行時期之前可能不會發生問題。

來自 amqwdeployWMQService 的輸出檔

amqwdeployWMQService 中的目錄及檔案輸出清單

表 *583:* 來自 ***amqwdeployWMQService*** 的輸出檔

輸出	說明	輸出目錄	檔名
.class	已編譯 Java 原始檔	./generated/server/*server package*	*classname*.class
.wsdl	服務說明	./generated	*classNameAxis*_Wmq.wsdl *classNameDotNet*_Wmq.wsdl
.wsdd	Axis 用戶端和服務部署檔案	./	client-config.wsdd server-config.wsdd
		./generated/server/*server package*	*className*_deploy.wsdd *className*_undeploy.wsdd
用戶端來源 (.vb、.cs、.java)	.Net 用戶端 Stub 至 Axis 服務	./generated/client	*classname*AxisService.cs *classname*AxisService.vb
	.Net 用戶端 Stub 至 .Net 服務	./generated/client	*classname*DotNet.cs *classname*DotNet.vb

表 583: 來自 ***amqwdeployWMQService*** 的輸出檔 (繼續)

輸出	說明	輸出目錄	檔名
用戶端協助程式 (.java 和 .class)	Java 用戶端 Proxy 至 .Net 服務	./generated/server/soap/client/remote/dotnetService	*className*DotNet.class *className*DotNet.java *className*DotNetLocator.class *className*DotNetLocator.java *className*DotNetSoap12Stub.class *className*DotNetSoap12Stub.java *className*DotNetSoap_BindingStub.class *className*DotNetSoap_BindingStub.java *className*DotNetSoap_PortType.class *className*DotNetSoap_PortType.java
	Java 用戶端 Proxy 至 Axis 服務	./generated/server/soap/client/remote/*client package*	SoapServer*className*AxisBindingSoapStub.class SoapServer*className*AxisBindingSoapStub.java *className*Axis.class *className*Axis.java *className*AxisService.class *className*AxisService.java *className*AxisServiceLocator.class *className*AxisServiceLocator.java
Script (.cmd 和 .sh)	接聽器 Script	/generated/server	startWMQJListener.cmd startWMQJListener.sh startWMQNListener.cmd endWMQJListener.cmd endWMQJListener.sh endWMQNListener.cmd

amqwdeployWMQService 使用注意事項

說明 **amqwdeployWMQService** 所執行的作業。

部署公用程式會執行下列動作。

1. 檢查下列檔案的路徑:
   - axis.jar.
   - *WMQSOAP_HOME*/java/lib/com.ibm.mq.soap.jar.
   - 在 Windows 上: csc.exe
2. 在 Windows 上，使用 %SystemRoot%\Microsoft.NET\Framework\v1.1.432 ，或者如果已安裝 C# 編譯器，則使用 csc.exe 的路徑作為 .NET Framework 的路徑。

   註: 如果您已安裝 Microsoft Visual Studio 2008 (第 9 版) ，則 wsdl.exe 不在 csc.exe 的路徑中。 您需要將 .NET Framework 的路徑新增至 Path 變數; 例如:

   ```
   Set Path=C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727;%Path%
   ```
3. 建立 ./generated 目錄及必要的子目錄 (如果它們不存在的話)。
4. 若為 Java 服務，請將原始檔編譯成 *className*.class。
5. 產生 WSDL。
6. 對於 Java 服務，建立部署描述子檔案 *className*_deploy.wsdd 和 *className*_undeploy.wsdd
7. 對於 Java 服務，建立或更新 Axis 部署描述子檔案 server-config.wsdd。

8. 從 WSDL 產生 Java、C# 和 Visual Basic 的用戶端 Proxy。

   註: 在 Windows 上，部署公用程式會產生 Visual Basic 及 C# 的 Proxy，而不論服務撰寫的語言為何。WSDL 和從它產生的 Proxy 包含適當的 URI 來呼叫服務:

   a.
   ```
   jms:/queue?destination=SOAPN.demos@WMQSOAP.DEMO.QM
   &connectionFactory=(connectQueueManager(WMQSOAP.DEMO.QM))
   &initialContextFactory=com.ibm.mq.jms.Nojndi
   &targetService=StockQuoteDotNet.asmx
   &replyDestination=SYSTEM.SOAP.RESPONSE.QUEUE
   ```

   圖 *13:* 產生的 *.NET* 用戶端中用來呼叫 *.NET* 服務的 *URI* 範例

   b.
   ```
   jms:/queue?destination=SOAPJ.demos@WMQSOAP.DEMO.QM
   &connectionFactory=(connectQueueManager(WMQSOAP.DEMO.QM))
   &initialContextFactory=com.ibm.mq.jms.Nojndi
   &targetService=soap.server.StockQuoteAxis.java
   &replyDestination=SYSTEM.SOAP.RESPONSE.QUEUE
   ```

   圖 *14:* 產生的 *.NET* 用戶端中用來呼叫 *Axis 1* 服務的 *URI* 範例

9. 編譯 Java Proxy。
10. 建立 WebSphere MQ 佇列 *requestQueue*，以保留服務的要求。預設佇列名稱的格式為 SOAPJ.*directory*，或者您可以在 -u URI 選項中指定 *requestQueue*。
11. 建立指令及 Shell Script 檔，以啟動處理要求佇列的 WebSphere MQ SOAP 接聽器。
12. 如果已使用 -tmq 選項，則部署公用程式會建立 WebSphere MQ 定義，以自動觸發 WebSphere MQ SOAP 接聽器處理程序。

    - 部署公用程式使用 **runmqsc** DEFINE PROCESS 指令的 APPLICID 屬性來包含啟動接聽器的指令。指令中內嵌了部署目錄的名稱。APPLICID 欄位的長度上限為 256，這會限制部署目錄的長度上限。Java 服務的目錄限制如下:
      - UNIX and Linux 系統: 218
      - Windows: 197 減去要求佇列名稱的長度。

      若為 .NET 服務，目錄限制如下:

      - Windows: 209 減去服務名稱長度，減去 .asmx 副檔名。
    - 部署公用程式會檢查是否超出 APPLICID 的限制。如果超出限制，公用程式不會嘗試定義觸發程序。它會顯示錯誤訊息，且部署程序會失敗，而不執行任何部署步驟。

    下列範例顯示部署公用程式所產生的配置和啟動指令，以啟動 WebSphere MQ SOAP 接聽器。

    ```
    DEFINE PROCESS(requestQueue) APPLICID(applicIDStr) REPLACE
    ALTER  QLOCAL (requestQueue) TRIGTYPE(FIRST) TRIGGER
           PROCESS(requestQueue) INITQ(initQueueName) TRIGMPRI(0)
    ```

    圖 *15: WebSphere MQ* 配置指令，用來觸發 *SOAP* 接聽器。

    ```
    applicIDStr = start "Java WMQSoapListener -requestQueue"
                        /min .\generated\server\startWMQJListener.cmd;
    ```

    圖 *16:* 在 *Windows* 上啟動 *Axis SOAP* 接聽器

    ```
    applicIDStr = start "WMQAsmxListener -className\
                        /min .\generated\server\startWMQNListener.cmd;
    ```

    圖 *17:* 在 *Windows* 上啟動 *.NET SOAP* 接聽器

```
applicIDStr = xterm -iconic -T \"Java WMQSoapListener_requestQueue\"
                    -e ./generated/server/startWMQJListener.sh & #
```

圖 *18:* 在 *UNIX and Linux* 系統上啟動 *Axis SOAP* 接聽器

amqwRegisterdotNet: 將 SOAP 的 IBM WebSphere MQ 傳輸登錄到 .NET

在 .NET 上向廣域組件快取登錄 SOAP 的 IBM WebSphere MQ 傳輸。

用途

amqwRegisterdotNet 會向 .NET Framework 1 或 2 登錄 WebSphere MQ SOAP 傳送端、SOAP 接聽器及 WSDL 處理器。

語法

說明

在安裝期間會自動執行 **amqwRegisterdotNet** 。 如果您使用的 .NET Framework 是在 SOAP 的 WebSphere MQ 傳輸之前安裝，則不需要重新執行它。 您可以執行它任意次數。 請使用它來重新登錄具有不同 .NET Framework 版本之 SOAP 的 WebSphere MQ 傳輸。

註: 在 Windows 2003 Server 上，即使您未部署至網際網路 Information Server (IIS) ，也必須執行 **aspnet_regiis** 公用程式。 `aspnet_regiis.exe` 公用程式的位置可能會因 Microsoft .NET Framework 的不同版本而不同，但它通常位於: `%SystemRoot%/Microsoft.NET/Framework/version number/aspnet_regiis`。 如果安裝多個版本，請針對您使用的 .NET Framework 版本使用 **aspnet_regiis** 。

Apache 軟體授權

Apache 軟體使用權 2.0 版，2004 年 1 月 http://www.apache.org/licenses/

http://www.apache.org/licenses/

Apache 授權
2.0 版， 2004 年 1 月
http://www.apache.org/licenses/

使用、重製及散布之條款

1. 定義。

「授權」係指使用、重製、
及本文件第 1 至 9 節所定義之散布。

「授權人」係指授權之著作權擁有者或實體。
授權的著作權擁有者。

"法律實體" 是指代理實體和所有
控制、控制或共用的其他實體
使用該實體進行控制。 為了這個定義的目的,
"control" 是指 (i) 直接或間接導致
這類實體的指示或管理，不論是透過合約或
或 (ii) 百分之五十 (50%) 或以上的
(iii) 該等實體的實益擁有權。

"您" (或 "您") 係指個人或法律實體
執行此「授權」所授與的許可權。

"來源" 表格係指進行修改的首選表格，
包括但不限於軟體原始碼、說明文件
來源及配置檔。

"物件" 形式是指機械
轉換或轉換「來源」表單，包括但
未限制為已編譯的物件程式碼、產生的文件、
及轉換至其他媒體類型。

"工作" 是指作者的工作，不論是在來源或
物件表單，在「授權」下提供，如
包含在工作中或附加在工作中的著作權聲明
(附錄中提供了一個範例)。

「衍生著作」係指任何著作，不論其為「原始碼」或「物件」
表單，以「工作」為基礎 (或衍生自) ，且
編輯修訂、註釋、詳細闡述或其他修改
作為一個整體，代表著原始的作者作品。 為了這些目的
本「授權」之「衍生著作」不包括下列著作:
可從或僅鏈結 (或依名稱連結) 至的介面，
其著作及其衍生著作。

"貢獻" 是指任何作者的工作，包括
工作的原始版本及任何修改或新增
該著作或其衍生著作的
提交給授權人以由著作權擁有者併入著作中
或由個人或法律實體授權代表
著作權擁有者。 為了這個定義的目的， "submitted"
表示傳送任何形式的電子、口頭或書面通訊
授權或其代表，包括但不限於
電子郵件清單上的通訊，原始碼控制系統，
及問題追蹤系統，這些系統由或代表
用於討論和改進工作的許可人，但
排除已明顯標示或以其他方式標示的通訊
由著作權擁有者以書面方式指定為 "Not a Contribution"。

「貢獻者」係指授權人及任何個人或法律實體
授權人已代表其收到「貢獻」的人，
後來併入工作中。

2. 授與「著作權授權」。 受的條款和條件限制
本授權，每位投稿者在此授予您永久的
全球、非專屬性、免費、免付權利金、不可撤銷
複製、準備下列項目之衍生著作之著作權授權:
公開顯示、公開執行、再授權及配送
以「來源」或「物件」形式之著作及該等衍生著作。

3. 授予專利授權。 受的條款和條件限制
本授權，每位投稿者在此授予您永久的
全球、非專屬性、免費、免付權利金、不可撤銷
(本節另有說明)
使用、提供銷售、銷售、匯入及以其他方式移轉工作，
當該授權僅適用於那些可授權的專利索賠時
這類提供者必須受到他們的
單獨或依其「貢獻」組合的貢獻
提交這類「貢獻」的「工作」。 若您
對任何實體提出專利訴訟 (包括

在訴訟中提出交叉索賠或反索賠)
或納入「工作」中的「貢獻」構成直接
或供款性專利侵權，然後任何專利授權
根據本「授權」授予您的工作將終止
這類訴訟的提交日期。

4. 再分配 貴客戶得重製及散布
在任何介質中的著作或其衍生著作，含或不含
修改，以及在「來源」或「物件」表單中，前提是您
符合下列條件:

(a) 您必須提供工作的任何其他收件人，或
Derivative Works-本授權之複本; 及

(b) 您必須讓任何修改過的檔案攜帶顯著注意事項
說明「您」已變更檔案; 及

(c) 您必須以任何衍生著作的「來源」形式保留
您所發行的所有著作權、專利、商標及
本「工作」之「來源」表單中之歸屬注意事項,
排除不適用於任何部分的那些通知
the Derivative Works; and

(d) If the Work includes a "NOTICE" text file as part of its
配送，則您配送的任何衍生著作必須
包含可讀取的歸屬權注意事項副本
在這類 NOTICE 檔案內，排除那些不符合下列規定的通知
與衍生著作的任何部分相關，至少有一個
下列位置: 在已配送的 NOTICE 文字檔內
作為衍生著作的一部分; 在「原始碼」形式或
說明文件 (如果隨附於衍生著作); 或
在「衍生著作」所產生的顯示畫面內，如果
這類第三方通知通常出現的位置。 內容
NOTICE 檔案的僅供參考之用，且
不修改授權。 您可以新增您自己的歸屬權
「您」所散布之「衍生著作」內的注意事項，以及
或作為「工作」之「通知」文字的附錄,
這些額外的歸屬通知不能被解釋
作為修改授權。

您可以將您自己的著作權聲明新增至您的修改及
可能提供其他或不同的授權條款
用於使用、重製或散佈您的修改，或
作為整體的任何此類衍生著作，但前提是您的使用,
「工作」之重製及散布，以其他方式遵循
「授權」中所規定的條件。

5. 提交會費。 除非你明確地說
有意提交以納入工作中的任何貢獻
「客戶」對「授權人」應遵循下列條款:
本授權，不含任何其他條款。
縱有上述各項，此處之任何內容均不得取代或修改
您可能已執行任何個別授權合約的條款
與授權人的相關資訊。

6. 商標。 此授權未授與使用交易的許可權
「授權人」的名稱、商標、服務標記或產品名稱,
除非在說明中有合理和習慣用途的需求,

「工作」的來源及重新產生「通知」檔案的內容。

7. 保固的免責聲明。 除非適用法律或
以書面同意者，「授權人」提供著作 (及各著作)
貢獻者在 "AS IS" BASIS 上提供其貢獻) ，
不提供任何明示或默示之保證或條件
默示之保證，包括但不限於任何保證或條件
標題、未涉侵權、可售性或符合
特殊目的。 您完全負責決定
使用或重新配送工作的適當性，並假設任何
與您在此授權下行使許可權相關聯的風險。

8. 賠償上限。 在沒有法律理論的情況下
無論是侵權 (包括過失)、合約或其他，
除非適用法律要求 (例如故意及嚴重
或以書面同意的方式，任何貢獻者
對貴客戶負責，包括任何直接、間接、特殊、
所產生的任何性質的附帶或衍生性損害
此授權的結果，或因使用或無法使用
工作 (包括但不限於商譽損失，
工作停止、電腦故障或故障，或任何及所有
其他商業損害或損失) ，即使這類貢獻者
已被告知此類損害的可能性。

9. 接受保固或額外責任。 重新配送時
著作或其衍生著作，貴客戶得選擇提供:
並收取費用、接受支援、保固、賠償、
或其他責任義務及/或與此一致的權利
授權。 不過，在接受這類義務時，您只能採取行動
代表您和您的唯一責任，而不是代表您
任何其他貢獻者，且只有在您同意賠償時，
為每一個貢獻者辯護，並使其免於承擔任何責任
由這類提供者所引起或針對這類提供者所提出的索賠 (依原因)
接受任何此類保證或額外責任。

條款結束

附錄: 如何將 Apache 授權套用至您的工作。

若要將 Apache 授權套用至您的工作，請附加下列
樣板注意事項，以方括弧 "[]" 括住欄位
取代為您自己的識別資訊。 (不包括
方括弧!) 文字應括在適當的
檔案格式的註解語法。 我們也建議您
檔案或類別名稱及目的說明併入
與版權聲明相同的「列印頁面」，以方便
協力廠商保存檔內的識別。

著作權 [yyyy] [著作權擁有者的名稱]

根據 Apache 授權 2.0 (「授權」) 授權;
除非符合「授權」規定，否則不得使用此檔案。
您可以在下列網址取得授權的副本:

http://www.apache.org/licenses/LICENSE-2.0

除非適用法律要求或以書面同意，否則軟體
授權下的分散式是在 "AS IS" BASIS 上配送，

不提供任何明示或默示之保證或條件。
請參閱授權，以取得特定語言控管許可權及
授權的限制。

MQMD SOAP 設定

IBM WebSphere MQ SOAP 傳送端和 IBM WebSphere MQ SOAP 接聽器會建立訊息描述子 (**MQMD**)。 主題說明如果您建立自己的 SOAP 傳送端或接聽器，則必須在 MQMD 中設定的欄位。

用途

MQMD 中設定的值控制 IBM WebSphere MQ SOAP 傳送端、 IBM WebSphere MQ SOAP 接聽器和 SOAP 用戶端程式之間的訊息交換。 如果您建立自己的 SOAP 傳送端或接聽器，請遵循 第 865 頁的表 584 中的規則。

說明

第 865 頁的表 584 說明 IBM WebSphere MQ SOAP 傳送端和 IBM WebSphere MQ SOAP 接聽器如何設定 **MQMD** 欄位。 如果您撰寫自己的寄件者或接聽器，則必須根據交換訊息的規則來設定這些欄位。 IBM WebSphere MQ SOAP 接聽器符合一般 IBM WebSphere MQ 訊息交換通訊協定。 如果您撰寫自己的傳送端來使用 IBM WebSphere MQ SOAP 接聽器，則可以設定不同的 **MQMD** 值。

在 第 865 頁的表 584 中， 設定 直欄中的值組織如下:

要求，單向
IBM WebSphere MQ SOAP 傳送端所做的設定。

回應，報告
IBM WebSphere MQ SOAP 接聽器為回應 IBM WebSphere MQ SOAP 傳送端要求所做的設定。

ALL
IBM WebSphere MQ SOAP 傳送端和 IBM WebSphere MQ SOAP 接聽器兩者所做的設定。

自訂寄件者
您可以撰寫自己的寄件者。 一般而言，自訂寄件者會置換標準報告選項。

表 *584: MQMD SOAP* 設定

欄位名稱	設定	值
StrucId	**ALL** MQMD_STRUC_ID	'MD¬¬'[1]
Version	**ALL** MQMD_VERSION_2	2
Report	**ALL** MQRO_NONE + MQRO_NEW_MSG_ID + MQRO_COPY_MSG_ID_TO_CORREL_ID + MQRO_EXCEPTION + MQRO_EXPIRY + MQRO_DISCARD **自訂寄件者** 請參閱第 869 頁的『自訂報告選項』。	52428800

表 584: MQMD SOAP 設定 (繼續)		
欄位名稱	**設定**	**值**
MsgType	**要求** MQMT_REQUEST **回應** MQMT_REPLY **報告** MQMT_REPORT **單向** MQMT_DATAGRAM	**MQMT_REQUEST** 1 **MQMT_REPLY** 2 **MQMT_REPORT** 4 **MQMT_DATAGRAM** 8
Expiry	**要求，單向** 由 URI 中的 期限 選項指定。 預設為 MQEI_UNLIMITED。 **回應** 要求訊息中 Expiry 的值 **報告** MQEI_UNLIMITED	**MQEI_UNLIMITED** -1
Feedback	**要求、回應、單向** MQFB_NONE. **報告** • 由佇列管理程式產生-根據一般規則設定的值。 • 由「IBM WebSphere MQ SOAP 接聽器」產生: **MQRC_BACKOUT_THRESHOLD_REACHED** 已超出多次嘗試的取消臨界值。 **MQRCCF_MD_FORMAT_ERROR** 訊息無法辨識為具有 MQRFH2 標頭。 **MQRC_RFH_PARM_MISSING** 遺漏 **MQRFH2** 中的必要參數，例如 SoapAction。 **MQRC_RFH_FORMAT_ERROR** **MQRFH2** 的基本完整性檢查失敗，例如，內部長度已毀損。 **MQRC_RFH_ERROR** **MQRFH2** 已通過完整性檢查，但訊息內文未設為 MQFMT_NONE。	**MQFB_NONE** 0 **MQRC_BACKOUT_THRESHOLD_REACHED** 2362 **MQRCCF_MD_FORMAT_ERROR** 3023 **MQRC_RFH_PARM_MISSING** 2339 **MQRC_RFH_FORMAT_ERROR** 2421 **MQRC_RFH_ERROR** 2334
Encoding	**ALL** MQENC_NATIVE	取決於環境
CodedCharSetId	**ALL** 設為 UTF-8	1208

表 584: *MQMD SOAP* 設定 (繼續)

欄位名稱	設定	值
Format	**要求、回應、單向** MQFMT_RF_HEADER_2 **報告** **佇列管理程式報告** 遵循 IBM WebSphere MQ 規則 **IBM WebSphere MQ SOAP 接聽器報告** 原始要求訊息的格式。	**MQFMT_RF_HEADER_2** "MQRFH2 "
Priority	**要求，單向** 由 URI 中的 優先順序 選項指定。 預設為 MQPRI_PRIORITY_AS_Q_DEF。 **回應，報告** 要求訊息中的 **優先順序** 值。	**MQPRI_PRIORITY_AS_Q_DEF** -1
Persistence	**要求，單向** MQPER_PERSISTENCE_AS_Q_DEF. **回應，報告** 要求訊息中的 **持續性** 值。	**MQPER_PERSISTENCE_AS_Q_DEF** 2
MsgId	**要求，單向** 由佇列管理程式產生。 **回應，報告** 即會產生 IBM WebSphere MQ SOAP 傳送端集 MQRO_NEW_MSG_ID 和 *MsgId*	**已產生** 佇列管理程式所產生的唯一值
CorrelId	**要求，單向，報告** MQCI_NONE **回應，報告** IBM WebSphere MQ SOAP 傳送端集 MQRO_COPY_MSG_ID_TO_CORREL_ID 和接聽器會從要求訊息複製 *MsgId* 。	**MQCI_NONE** 0
BackoutCount	**ALL** 未使用	0
ReplyToQ	**要求** 由 URI 中的 replyDestination 選項指定。 預設為 SYSTEM.SOAP.RESPONSE.QUEUE。 **回應、單向、報告** 保留空白	
ReplyToQMgr	**ALL** 欄位保留空白	由佇列管理程式產生; 請參閱 回覆佇列及佇列管理程式 。

表 584: *MQMD SOAP* 設定 (繼續)

欄位名稱	設定	值
UserIdentifier	**要求，單向，報告** 保留空白 **回應** 取決於提供給接聽器的 -i *passContext* 選項，以及接聽器在其下執行的權限。	**要求，單向，報告** 由佇列管理程式產生; 請參閱 第 400 頁的『UserIdentifier (MQCHAR12)』 **回應** 變數
AccountingToken	**ALL** MQACT_NONE	**MQACT_NONE** 空字串或空白 由佇列管理程式設定; 請參閱 第 366 頁的『AccountingToken (MQBYTE32)』
ApplIdentityData	**ALL** 無	空字串或空白[2]
PutApplType	**ALL** MQAT_NO_CONTEXT	**MQAT_NO_CONTEXT** 0 佇列管理程式所產生的值; 請參閱 第 389 頁的『PutAppl 類型 (MQLONG)』。
PutApplName	**ALL** 無	佇列管理程式所產生的值; 請參閱 第 388 頁的『PutAppl 名稱 (MQCHAR28)』。
PutDate	**ALL** 無	佇列管理程式所產生的值; 請參閱 第 390 頁的『PutDate (MQCHAR8)』。
PutTime	**ALL** 無	佇列管理程式所產生的值; 請參閱 第 391 頁的『PutTime (MQCHAR8)』。
ApplOriginData	**ALL** 無	空字串或空白[2]
GroupId	**要求，單向，報告** MQGI_NONE **回應** 從要求訊息複製欄位	空值
MsgSeqNumber	**要求，單向，報告** 未使用 **回應** 從要求訊息複製欄位	由佇列管理程式產生; 請參閱 佇列上的實體訂單。
Offset	**要求，單向，報告** 未使用 **回應** 從要求訊息複製欄位	0

表 584: MQMD SOAP 設定 (繼續)		
欄位名稱	**設定**	**值**
MsgFlags	**要求，單向，報告** MQMF_NONE **回應** 從要求訊息複製欄位	**MQMF_NONE** 0 請參閱第 380 頁的『MsgFlags (MQLONG)』。
OriginalLength	**要求，單向，回應** MQOL_UNDEFINED **報告** 原始要求訊息的長度	**MQOL_UNDEFINED** -1
附註: 1. 符號 ¬ 代表單一空白字元。 2. 值 空字串或空白 表示 C 中的空字串，而其他程式設計語言中的空白字元。		

自訂報告選項

您可以撰寫自己的 SOAP 傳送端，並將它與所提供的接聽器搭配使用。 通常您可以撰寫寄件者來變更報告選項的選項。 IBM WebSphere MQ SOAP 接聽器支援大部分報告選項組合，如下列清單中所述。

- IBM WebSphere MQ SOAP 接聽器支援的報告選項:
  - MQRO_EXCEPTION
  - MQRO_EXCEPTION_WITH_DATA
  - MQRO_EXCEPTION_WITH_FULL_DATA
  - MQRO_DEAD_LETTER_Q
  - MQRO_DISCARD_MSG
  - MQRO_NONE
  - MQRO_NEW_MSG_ID
  - MQRO_PASS_MSG_ID
  - MQRO_COPY_MSG_ID_TO_CORREL_ID
  - MQRO_PASS_CORREL_ID
- 佇列管理程式支援的報告選項:
  - MQRO_COA
  - MQRO_COA_WITH_DATA
  - MQRO_COA_WITH_FULL_DATA
  - MQRO_COD
  - MQRO_COD_WITH_DATA
  - MQRO_COD_WITH_FULL_DATA
  - MQRO_EXPIRATION
  - MQRO_EXPIRATION_WITH_DATA
  - MQRO_EXPIRATION_WITH_FULL_DATA
- IBM WebSphere MQ SOAP 接聽器不支援下列報告選項。
  - MQRO_PAN
  - MQRO_NAN

第 870 頁的表 585 中說明 IBM WebSphere MQ SOAP 接聽器回應 MQRO_EXCEPTION_* 和 MQRO_DISCARD 組合的行為。

表示法 MQRO_EXCEPTION_* 指出使用 MQRO_EXCEPTION、 MQRO_EXCEPTION_WITH_DATA 或 MQRO_EXCEPTION_WITH_FULL_DATA。

表 *585: MQRO_EXCEPTION_** 和 *MQRO_DISCARD* 設定產生的接聽器行為

	MQRO_DISCARD 已啟用	**MQRO_DISCARD 未啟用**
MQRO_EXCEPTION_* 已啟用	預設行為。 必要的話，會自動產生報告訊息，並捨棄原始要求。 如果無法將報告訊息傳回回應佇列，則會將報告訊息傳送至無法傳送的郵件佇列。	必要的話，會自動產生報告訊息，並將原始訊息傳送至無法傳送的郵件佇列。 如果報告訊息無法傳回回應佇列，則也會傳送至無法傳送郵件的佇列。 在此情況下，失敗要求會有兩個無法傳送的郵件佇列項目。
MQRO_EXCEPTION_* 未啟用	當無法辨識送入格式或超出取消嘗試次數時，不會自動產生報告訊息。 訊息不會傳送至無法傳送的郵件佇列。 不會傳回用戶端可以檢查的通知，且會遺失原始要求訊息。	當無法辨識送入格式或超出取消嘗試次數時，不會自動產生報告訊息。 不過，當產生報告時，會將原始要求訊息寫入無法傳送的郵件佇列。

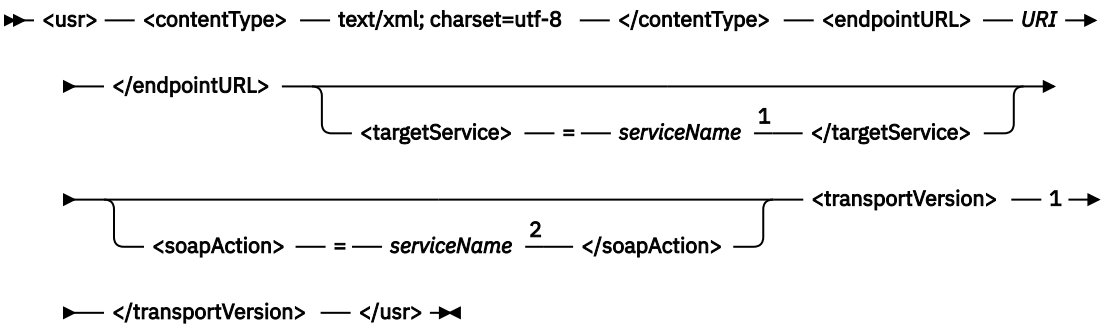
MQRFH2 SOAP 設定

IBM WebSphere MQ SOAP 傳送端和接聽器會建立或預期接收具有下列設定的 MQRFH2 。

用途

WebSphere MQ SOAP 傳送端會將內容新增至 WebSphere MQ JMS 所建立的 <usr> 資料夾。 內容包含目標環境中 SOAP 儲存器所需的資訊。 第 870 頁的『內容語法』 說明內容新增至 MQRFH2 時的語法。 如需 MQRFH2 標頭的說明，請參閱 MQRFH2 -規則和格式化標頭 2。

內容語法



註:

[1] targetService 是 .NET Framework 1 或 2 的必要項目，且未在 Axis 1.4 上使用。
[2] soapAction 是 .NET Framework 1 或 2 的選用項目，且未在 Axis 1.4 上使用。

參數

contentType
contentType 一律包含 text/xml; charset=utf-8 字串。

endpointURL
請參閱 第 885 頁的『Web 服務部署的 URI 語法和參數』。

targetService

[6] 在 Axis 上， *serviceName* 是 Java 服務的完整名稱，例如: `targetService=javaDemos.service.StockQuoteAxis`。 如果未指定 targetService ，則會使用預設 Axis 機制來載入服務。

[7] 在 .NET 上， *serviceName* 是位於部署目錄中的 .NET 服務名稱，例如: `targetService=myService.asmx`。 在 .NET 環境中， targetService 參數可讓單一 WebSphere MQ SOAP 接聽器能夠處理多個服務的要求。 這些服務必須從相同的目錄部署。

soapAction

transportVersion

transportVersion 一律設為 1。

範例

此範例顯示 MQRFH2 及下列 SOAP 訊息。 資料夾的長度以十進位顯示。

註: URI 中的 & 編碼為 &

```
52464820 00000002 000002B0 00000001 RFH 0002 1208 0001
000004B8 20202020 20202020 00000000 1208 0000
000004B8                           1208
32  <mcd>
        <Msd>jms_bytes</Msd>
    </mcd>?
208 <jms>
        <Dst>queue://queue://SOAPJ.demos</Dst>
        <Rto>queue://WMQSOAP.DEMO.QM/SYSTEM.SOAP.RESPONSE.QUEUE</Rto>
        <Tms>1157388516465</Tms>
        <Cid>ID:000000000000000000000000000000000000000000000000</Cid>
        <Dlv>1</Dlv>
    </jms>
400 <usr>
        <contentType>text/xml; charset=utf-8</contentType>
        <transportVersion>1</transportVersion>
        <endpointURL>
            jms:/queue?destination=SOAPJ.demos@WMQSOAP.DEMO.QM
            &amp;connectionFactory=connectQueueManager(WMQSOAP.DEMO.QM)
                clientConnection(localhost%25289414%2529)
                clientChannel(TESTCHANNEL)
            &amp;replyDestination=SYSTEM.SOAP.RESPONSE.QUEUE
            &amp;initialContextFactory=com.ibm.mq.jms.Nojndi
        </endpointURL>
    </usr>
<?xml version="1.0" encoding="UTF-8"?>
    <soapenv:Envelope
            xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="https://www.w3.org/2001/XMLSchema"
            xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance">
        <soapenv:Body>
            <ns1:getQuote
                    soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                    xmlns:ns1="soap.server.StockQuoteAxis_Wmq">
                <in0 xsi:type="xsd:string">XXX</in0>
            </ns1:getQuote>
        </soapenv:Body>
    </soapenv:Envelope>
```

runivt: WebSphere MQ 傳輸以進行 SOAP 安裝驗證測試

IBM WebSphere MQ SOAP 傳輸隨附安裝驗證測試套組 (IVT)。 **runivt** 會執行一些示範應用程式，並確保在安裝之後正確設定環境。

[6] 僅限 Java 服務
[7] 僅限 .NET 服務

用途

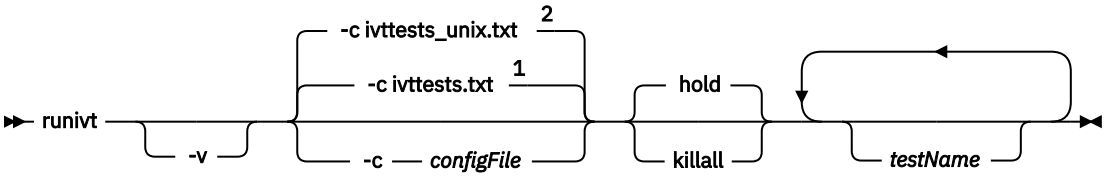
runivt 指令使用隨 WebSphere MQ SOAP 傳輸提供的範例程式，將 Web 服務要求從用戶端傳送至服務。 它會執行 Axis 1.4、.NET Framework 1 和 .NET Framework 2 的測試。 測試配置在測試 Script 檔中。 Windows 的預設測試 Script 檔會在 Java 與 .NET 用戶端及服務之間執行測試組合。

說明

runivt 必須從它自己的目錄執行。

指令會在不同的指令視窗中啟動接聽器。 因此，您必須在 UNIX and Linux 系統上從 X Window 系統階段作業執行指令。

runivt syntax



註:

[1] Default on Windows
[2] Default on UNIX and Linux systems

runivt 參數

-v
詳細模式。 將更完整的錯誤訊息寫入主控台。

-c ***configFile***
定義要執行之測試的配置檔。 依預設會使用 Windows、 UNIX 或 Linux 系統所提供的預設配置檔。

hold
在測試完成之後讓接聽器保持執行

killall
在測試完成時結束接聽器

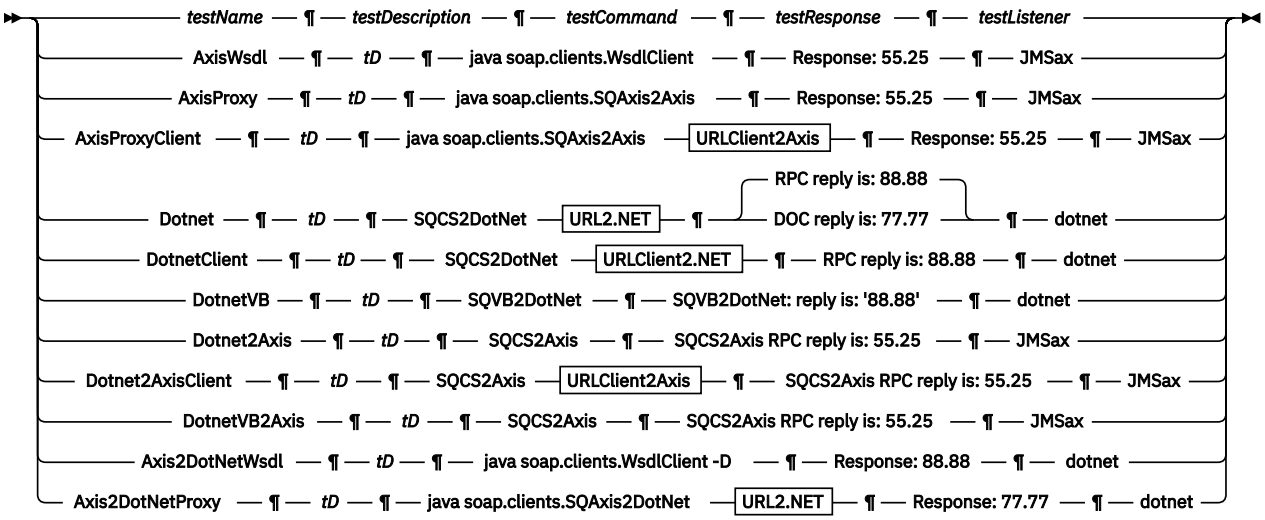
testName
要執行的測試清單 (以空格區隔)。 從配置檔中選取測試名稱。 如果未指定名稱，則會執行配置檔中的所有測試。

Configuration file

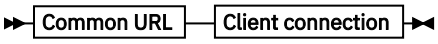
Each configuration file parameter is a separate line of the file. Leave a blank line between each group of parameters.

The parameters in the `ivttests.txt` parameter file are listed.

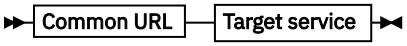
***configFile* syntax**



URLClient2Axis

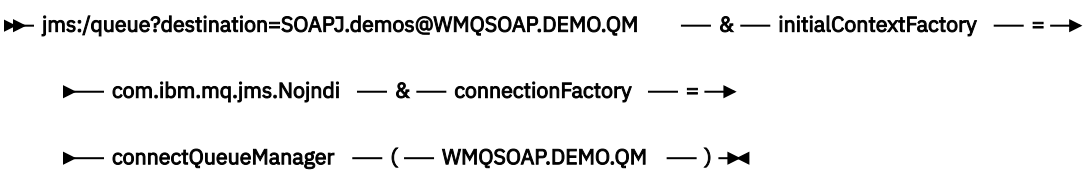


URL2.NET

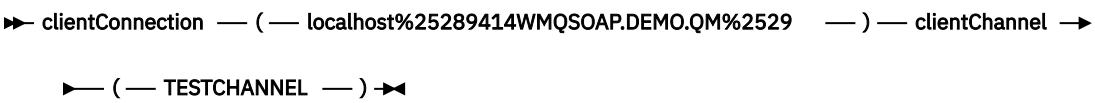


URLClient2.NET

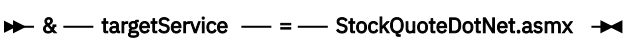
Common URL



Client connection



Target service



configFile 參數

testName
測試的名稱。 在 **runivt** 指令中使用 *testName*

testDescription
關於測試的 Documentation

testCommand
由 **runivt** 指令執行以提出用戶端要求的指令。

testResponse
用戶端要求傳回主控台的確切回應字串。 若要讓測試成功， *testResponse* 必須符合實際回應。

testListener
由 **runivt** 啟動以處理 SOAP 要求的 WebSphere MQ SOAP 接聽器名稱。 dotnet 和 JMSax 是所提供的接聽器 **amqwSOAPNETlistener** 和 **SimpleJavaListener** 的同義字。

範例

```
runivt
```

圖 *19:* 執行所有預設測試

```
runivt dotnet
```

圖 *20:* 從預設測試執行特定測試

```
runivt -c mytests.txt
```

圖 *21:* 執行一組自訂測試

相關資訊
驗證 SOAP 的 WebSphere MQ 傳輸

SOAP 的透過 IBM WebSphere MQ 傳輸的安全 Web 服務

您可以透過兩種方式之一來保護使用 IBM WebSphere MQ SOAP 傳輸的 Web 服務。 請在用戶端與伺服器之間建立 SSL 通道，或使用 Web 服務安全。

SSL 和 WebSphere MQ SOAP 傳輸

SOAP 的 WebSphere MQ 傳輸提供數個 SSL 選項，可指定與配置成以 SSL 模式執行的用戶端通道搭配使用。 .NET 與 Java 環境之間的選項不同。 WebSphere MQ SOAP 傳送端和接聽器只會處理適用於其特定環境的 SSL 選項。 它們會忽略不適用的選項。

是否存在 .NET 用戶端的 `sslCipher` **規格** 選項，以及 Java 用戶端的 `sslCipher` **套組** 選項，會決定是否使用 SSL。 如果 URI 中未指定選項，依預設不會使用 SSL，且會忽略所有其他 SSL 選項。 除了指出的選項之外，所有 SSL 選項都是選用的。

若為 WebSphere MQ 用戶端，請在 URI 或通道定義表中設定 SSL 屬性。 在伺服器上，使用 WebSphere MQ 的機能來設定屬性。

依預設，在通道上啟用 SSL 時，會設定標準 WebSphere MQ SSL 選項 `SSLCAUTH`。 用戶端必須自行鑑別，才能開始 SSL 通訊。 如果未設定 `SSLCAUTH`，則會在沒有用戶端鑑別的情況下建立 SSL 通訊。

若要自行鑑別，用戶端必須在其金鑰儲存庫中指派佇列管理程式可接受的憑證。 為了額外安全，WebSphere MQ 通道可以配置成只接受受限清單中的憑證。 根據通道的同層級名稱屬性來檢查憑證的識別名稱，以限制此清單。

如果您使用 Java，來自 WebSphere MQ SOAP 用戶端的第一個 SSL 連線會導致修正下列 SSL 參數。 在使用相同用戶端處理程序的後續連線中，會使用相同的值:

- `sslKey` **儲存庫**
- `sslKeyStorePassword`
- `sslTrust` **儲存庫**
- `sslTrustStorePassword`
- `sslFipsRequired`
- `sslLDAPCRLservers`

在來自此用戶端的後續連線上改變這些參數的效果未定義。

如果您使用 .NET，來自 WebSphere MQ SOAP 用戶端的第一個 SSL 連線會導致修正下列 SSL 參數。 在使用相同用戶端處理程序的後續連線中，會使用相同的值:

- `SSLKeyRepository`
- `SSLCryptoHardware`
- `sslFipsRequired`

- sslLDAPCRLservers

在來自此用戶端的後續連線上改變這些參數的效果未定義。 如果所有 SSL 連線都變成非作用中，並建立新的 SSL 連線，則會重設這些參數。

下列內容也可以指定為系統內容:

- sslKey 儲存庫
- sslKeyStorePassword
- sslTrust 儲存庫
- sslTrustStorePassword

如果同時將它們指定為系統內容和在 URI 中，且值不同，部署公用程式會顯示警告。 URI 值優先。

相關工作

指定在執行時期於 MQI 用戶端上僅使用 FIPS 認證的 CipherSpecs

相關參考

WebSphere MQ Web 服務 URI 中的 SSL Connection Factory 參數
將 SSL 選項新增至 IBM WebSphere MQ Web 服務 URI 中的 Connection Factory 選項清單。

UNIX、 Linux 和 Windows 的聯邦資訊存取安全標準 (FIPS)

WebSphere MQ Web 服務 URI 中的 SSL Connection Factory 參數

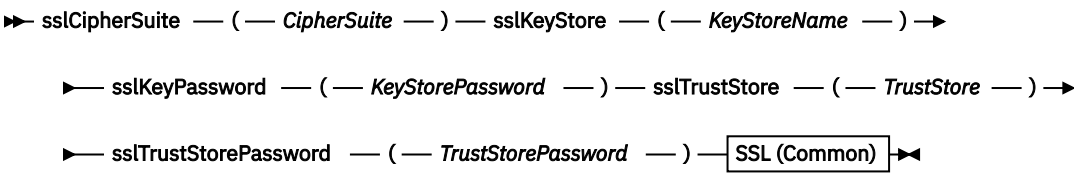
將 SSL 選項新增至 IBM WebSphere MQ Web 服務 URI 中的 Connection Factory 選項清單。

用途

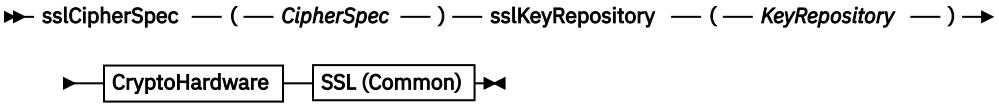
您可以在 IBM WebSphere MQ Web 服務用戶端與管理 Web 服務的佇列管理程式之間使用安全連線。 SSL 選項控制如何在 IBM WebSphere MQ MQI 主從式通道連線上配置 SSL。

Syntax diagram

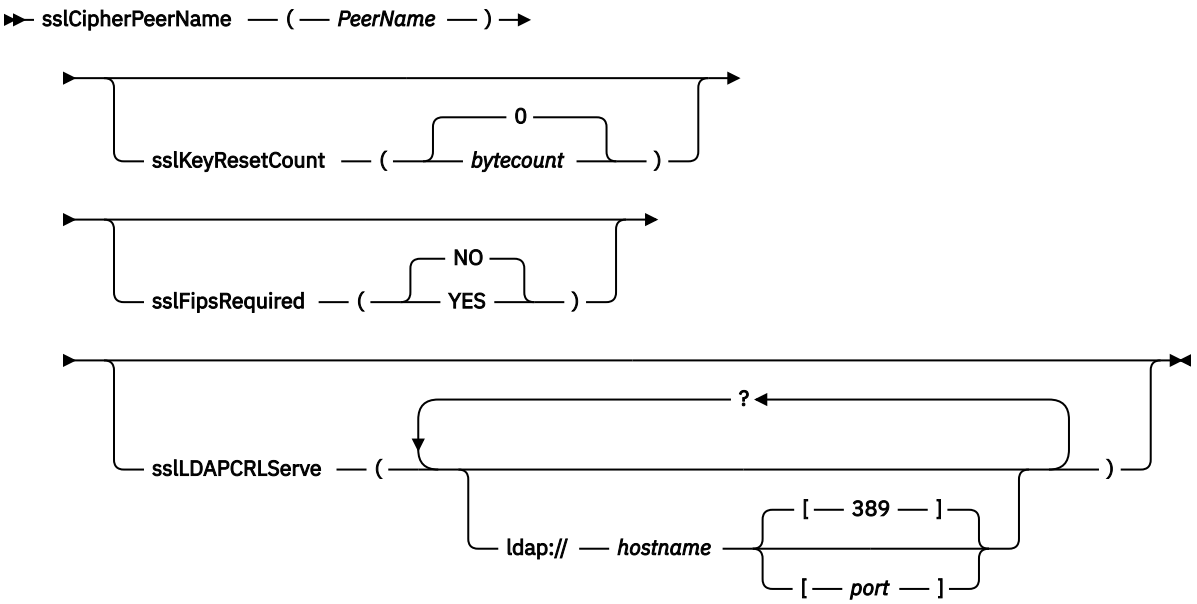
SSL (Java)



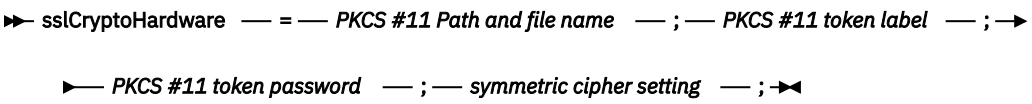
SSL (.NET)



SSL (Common)



CryptoHardware



必要的 SSL 參數 (共用)

sslPeerName(*peerName*)
peerName 指定通道上使用的 sslPeer 名稱 。

必要的 SSL 參數 (Java)

sslCipherSuite(*CipherSuite*)
CipherSuite 指定通道上使用的 sslCipher 套組 。 用戶端指定的 CipherSuite 必須符合伺服器連線通道上指定的 CipherSuite 。

sslKeyStore(*KeyStore* 名稱)
KeyStore 名稱 指定通道上使用的 sslKeyStoreName 。 金鑰儲存庫會保留用來向伺服器鑑別用戶端的用戶端私密金鑰。 如果 SSL 連線配置為接受匿名用戶端連線，則金鑰儲存庫是選用的。

sslKeyStorePassword(*KeyStore* 密碼)
KeyStore 密碼 指定通道上使用的 sslKeyStorePassword 。

sslTrustStore(*TrustStore* 名稱)
TrustStore 名稱 指定通道上使用的 sslTrustStoreName 。 信任儲存庫會保留伺服器或其金鑰鏈的公用憑證，以向用戶端鑑別伺服器。 如果使用憑證管理中心的主要憑證來鑑別伺服器，則信任儲存庫是選用的。 在 Java 中，主要憑證保留在 JRE 憑證儲存庫 `cacerts` 中。

sslTrustStorePassword(*TrustStore* 密碼)
TrustStore 密碼 指定通道上使用的 sslTrustStorePassword 。

必要的 SSL 參數 (.NET)

sslCipherSpec(*CipherSpec*)
CipherSpec 指定通道上使用的 sslCipher 規格 。 如果指定此選項，則會在用戶端通道上使用 SSL。

sslKeyRepository(*KeyRepository*)
KeyRepository 指定在儲存 SSL 金鑰和憑證的通道上使用的 sslCipher 規格 。 *KeyRepository* 是以詞幹格式指定，亦即，具有檔名但省略副檔名的完整路徑。 設定 sslKey 儲存庫 的效果與在 `MQCONNX` 呼叫的 **MQSCO** 結構中設定 `KeyRepository` 欄位相同。

選用 SSL 參數 (.NET)

sslCryptoHardware(*CryptoHardware*)
CryptoHardware 指定通道上使用的 sslCryptoHardware 。 此欄位的可能值及其設定的效果，與 MQCONNX 上 **MQSCO** 結構的 CryptoHardware 欄位相同。

選用 SSL 參數 (共用)

sslKeyResetCount(*bytecount*)
bytecount 指定在必須重新協議 SSL 秘密金鑰之前，透過 SSL 通道傳遞的位元組數。 若要停用 SSL 金鑰的重新協議，請省略欄位或將它設為零。 零是部分環境中唯一支援的值，請參閱 在 WebSphere MQ for Java 類別中重新協議秘密金鑰。 設定 sslKeyResetCount 的效果與在 MQCONNX 呼叫的 **MQSCO** 結構中設定 KeyResetCount 欄位相同。

sslFipsRequired(*fipsCertified*)
fipsCertified 指定 *CipherSpec* 或 *CipherSuite* 是否必須在通道上 IBM WebSphere MQ 中使用 FIPS 認證的加密法。 設定 *fipsCertified* 的效果與在 MQCONNX 呼叫上設定 **MQSCO** 結構的 FipsRequired 欄位相同。

sslLDAPCRLServers(*LDAPServerList*)

LDAPServerList 指定要用於「憑證撤銷清冊」檢查的 LDAP 伺服器清單。

對於已啟用 SSL 的用戶端連線， *LDAPServerList* 是要用於「憑證撤銷清冊 (CRL)」檢查的 LDAP 伺服器清單。 會根據其中一個列出的 LDAP CRL 伺服器來檢查佇列管理程式提供的憑證; 如果找到，連線會失敗。 系統會依序嘗試每一部 LDAP 伺服器，直到建立與其中一部 LDAP 伺服器的連線功能為止。 如果無法連接至任何伺服器，則會拒絕憑證。 順利建立與其中一個的連線之後，會根據該 LDAP 伺服器上存在的 CRL 來接受或拒絕憑證。

如果 *LDAPServerList* 為空白，則不會針對「憑證撤銷清冊」檢查屬於佇列管理程式的憑證。 如果提供的 LDAP URI 清單無效，則會顯示錯誤訊息。 設定此欄位的效果與包括 MQAIR 記錄並從 MQCONNX 上的 **MQSCO** 結構存取這些記錄的效果相同。

相關工作
指定在執行時期於 MQI 用戶端上僅使用 FIPS 認證的 CipherSpecs
相關參考
SSL 和 WebSphere MQ SOAP 傳輸
SOAP 的 WebSphere MQ 傳輸提供數個 SSL 選項，可指定與配置成以 SSL 模式執行的用戶端通道搭配使用。 .NET 與 Java 環境之間的選項不同。 WebSphere MQ SOAP 傳送端和接聽器只會處理適用於其特定環境的 SSL 選項。 它們會忽略不適用的選項。

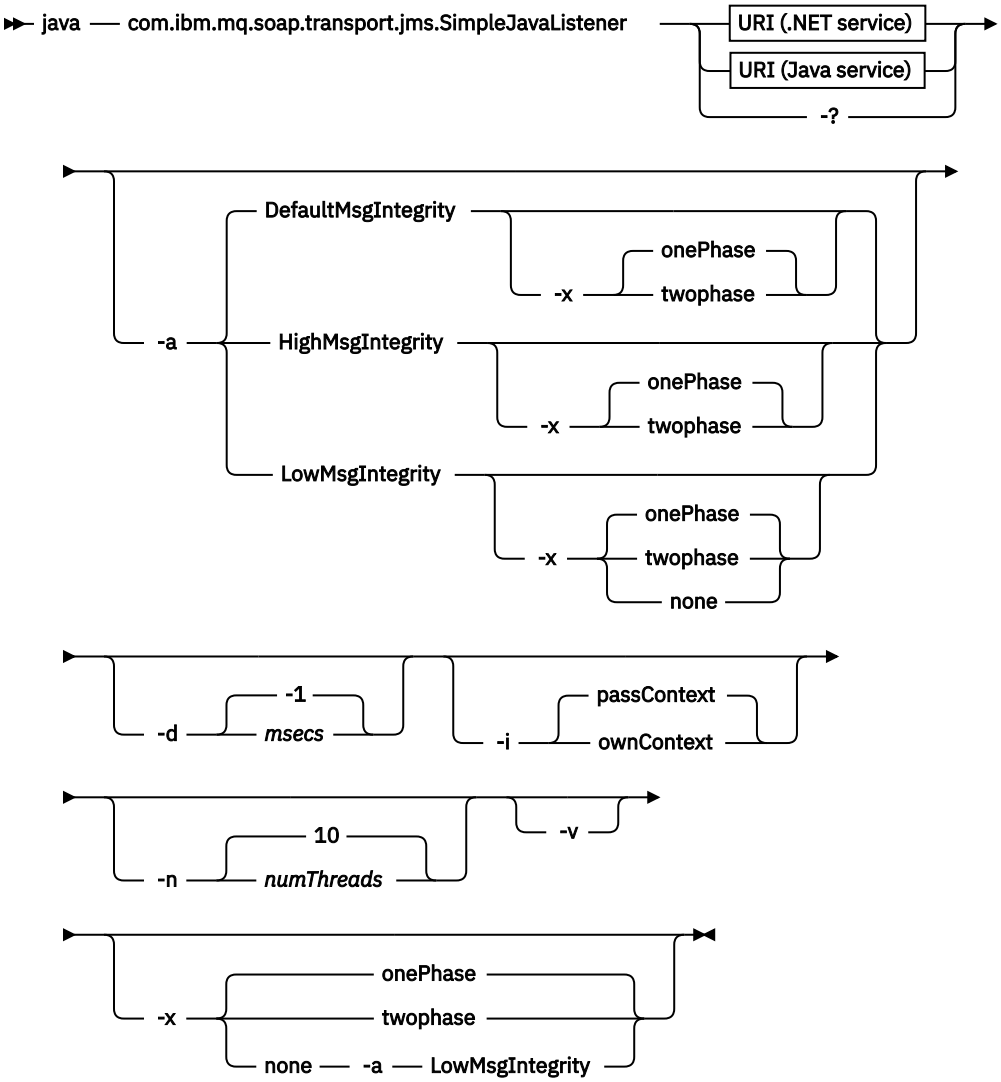
UNIX、 Linux 和 Windows 的聯邦資訊存取安全標準 (FIPS)

SimpleJavaListener: IBM WebSphere MQ Axis 1.4

適用於 Axis 1.4 的 IBM WebSphere MQ SOAP 接聽器的語法和參數。

用途
啟動 IBM WebSphere MQ Axis 1.4 的 SOAP 接聽器。

Java



必要參數

URI 平台
請參閱 第 885 頁的『Web 服務部署的 URI 語法和參數』。

-?
印出說明如何使用指令的說明文字。

選用參數

-a *integrityOption*
integrityOption 指定 WebSphere MQ SOAP 接聽器的行為 (如果無法將失敗要求訊息放置在無法傳送郵件的佇列上)。 *integrityOption* 可以採用下列其中一個值:

DefaultMsgIntegrity
對於非持續訊息，接聽器會顯示警告訊息，並繼續執行，並捨棄原始訊息。 若為持續訊息，它會顯示一則錯誤訊息，取消要求訊息，讓它保留在要求佇列上並結束。 如果省略 -a 選項，或未指定 *integrityOption* ，則會套用 DefaultMsgIntegrity 。

LowMsgIntegrity
對於持續及非持續訊息，接聽器會顯示警告並繼續執行，捨棄訊息。

HighMsgIntegrity
對於持續及非持續訊息，接聽器會顯示一則錯誤訊息，取消要求訊息，使其保留在要求佇列上並結束。

部署公用程式會檢查 -x 和 -a 旗標的相容性。 如果指定 -x none ，則必須指定 -a LowMsgIntegrity 。 如果旗標不相容，則部署公用程式會結束並顯示錯誤訊息，且未執行任何部署步驟。

-d 毫秒
毫秒 指定在任何執行緒上收到要求訊息時， WebSphere MQ SOAP 接聽器保持作用中的毫秒數。 如果 *msecs* 設為 -1，則接聽器會無限期地保持作用中。

-i 環境定義
環境定義 指定接聽器是否傳遞身分環境定義。 環境定義 採用下列值:

passContext
在回應訊息中設定原始要求訊息的身分環境定義。 SOAP 接聽器會檢查它是否有權從要求佇列儲存環境定義，並將它傳遞至回應佇列。 當開啟要儲存環境定義的要求佇列，以及要傳遞環境定義的回應佇列時，它會在執行時期進行檢查。 如果它沒有必要的權限，或 MQOPEN 呼叫失敗，則不會處理回應訊息。 回應訊息會放置在無法傳送郵件的佇列中，且無法傳送郵件的標頭包含來自失敗 MQOPEN 的回覆碼。 然後，接聽器會繼續正常處理後續的送入訊息。

ownContext
SOAP 接聽器不傳遞環境定義。 傳回的環境定義反映接聽器執行時所使用的使用者 ID ，而不是建立原始要求訊息的使用者 ID。

原始環境定義中的欄位是由佇列管理程式所設定，而不是由 SOAP 接聽器所設定。

-n *numThreads*
numThreads 指定 WebSphere MQ SOAP 接聽器的已產生啟動 Script 中的執行緒數目。 預設值為 10。 如果您有高訊息傳輸量，請考慮增加此數目。

-v
-v 設定外部指令的詳細輸出。 一律會顯示錯誤訊息。 使用 -v 來輸出您可以自訂以建立自訂部署 Script 的指令。

-w *serviceDirectory*
serviceDirectory 是包含 Web 服務的目錄。

-x 交易性
交易方式 指定接聽器的交易式控制項類型。 交易方式 可以設為下列其中一個值:

onePhase
使用 IBM WebSphere MQ 一段式支援。 如果系統在處理期間失敗，則會將要求訊息重新遞送至應用程式。 WebSphere MQ 交易可確保只寫入回應訊息一次。

twoPhase
使用兩階段支援。 如果適當地撰寫服務，則在服務的單一已確定執行內，訊息只會遞送一次 (與其他資源協調)。 這個選項只適用於伺服器連結連線。

none
無交易式支援。 如果系統在處理期間失敗，即使持續保存，也可能會遺失要求訊息。 服務可能已執行，也可能未執行，且可能寫入回應、報告或無法傳送郵件的訊息。

部署公用程式會檢查 -x 和 -a 旗標的相容性。 如需詳細資料，請參閱 -a 旗標的說明。

Java 範例

```
java com.ibm.mq.soap.transport.jms.SimpleJavaListener
-u "jms:/queue?destination=myQ&connectionFactory=()
&initialContextFactory=com.ibm.mq.jms.Nojndi"
-n 20
```

WebSphere MQ SOAP 接聽器

WebSphere MQ SOAP 接聽器會從 URI 中指定為目的地的佇列讀取送入的 SOAP 要求。 它會檢查要求訊息的格式，然後使用 Web 服務基礎架構來呼叫 Web 服務。 WebSphere MQ SOAP 接聽器會使用 URI 中的回覆目的地佇列，從 Web 服務傳回任何回應或錯誤。 它會將 WebSphere MQ 報告傳回至回覆佇列。

此術語接聽器在其標準 Web 服務感應中使用。 它不同於 **runmqlsr** 指令所呼叫的標準 WebSphere MQ 接聽器。

說明

Java SOAP 接聽器會實作為 Java 類別，並使用 Axis 1.4 來執行服務。 .NET 接聽器是一個主控台應用程式，並執行 .NET Framework 1 或 .NET Framework 2 服務。 若為 .NET Framework 3 服務，請使用 Microsoft Windows Communication Foundation (WCF) 的 WebSphere MQ 自訂通道。

部署公用程式會建立 Script，以自動啟動 Java 或 .NET SOAP 接聽器。 「SOAP 接聽器」可以使用 **amqSOAPNETListener** 指令或呼叫 `SimpleJavaListener` 類別來手動啟動。 您可以透過在部署公用程式中設定 -s 選項，將 WebSphere MQ SOAP 接聽器配置成啟動為 WebSphere MQ 服務。 或者，使用觸發來啟動接聽器，或使用部署公用程式所產生的啟動及結束接聽器 Script。 您可以手動配置觸發，或使用 -tmq 及 -tmp 部署選項自動配置觸發。 您可以透過將要求佇列設為 `GET(DISABLED)` 來結束接聽器。

表 *586:* 部署公用程式所產生的指令 *Script*

Web 服務基礎架構	UNIX and Linux 系統	Windows Java	Windows .NET
啟動接聽器	**startWMQJListener.sh**	**startWMQJListener.cmd**	**startWMQNListener.cmd**
停止接聽器	**endWMQJListener.sh**	**endWMQJListener.cmd**	**endWMQNListener.cmd**
定義接聽器服務	**defineWMQJListener.sh**	**defineWMQJListener.cmd**	**defineWMQNListener.cmd**

WebSphere MQ SOAP 接聽器會將 `endpointURL` 和 `soapAction` 欄位從 SOAP 訊息傳遞至 SOAP 基礎架構。 接聽器會透過 Web 服務基礎架構呼叫服務，並等待回應。 接聽器不會驗證 `endpointURL` 和 `soapAction`。 這些欄位是由 WebSphere MQ SOAP 傳送端從 SOAP 用戶端所設定的 URI 中提供的資料所設定。

接聽器會建立回應訊息，並將它傳送至要求訊息 URI 中提供的回覆目的地。 此外，接聽器會根據要求訊息中的報告選項，在回應訊息中設定相關性 ID。 它會從要求訊息傳回期限、持續性和優先順序設定。 在某些情況下，接聽器也會將報告訊息傳回給用戶端。

如果 SOAP 要求中有格式錯誤，則接聽器會使用回覆目的地佇列向用戶端傳回報告訊息。 如果已要求報告，佇列管理程式也會使用回覆目的地佇列將報告訊息傳回用戶端。 完整報告訊息會寫入回應佇列，以回應一些事件:

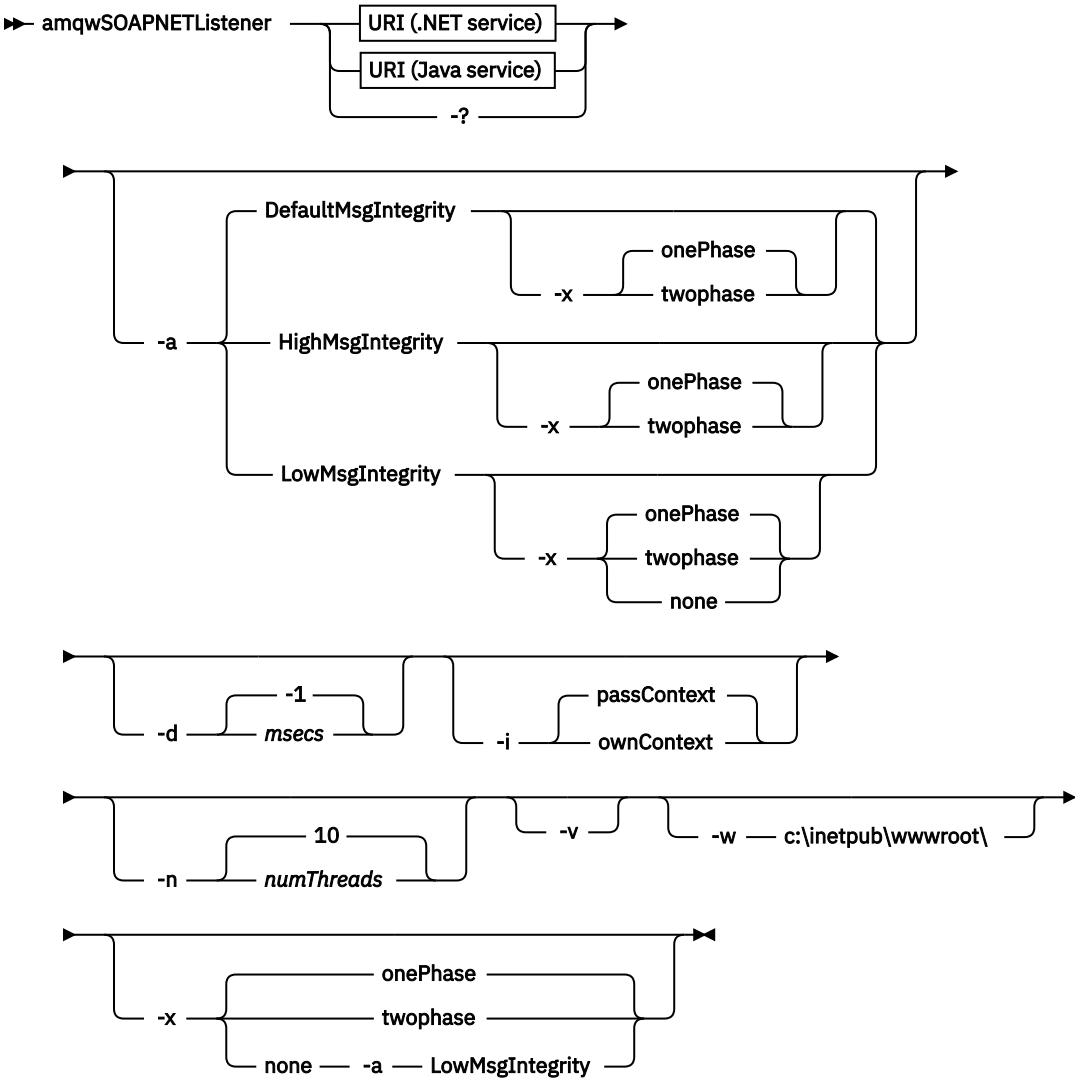
- 異常狀況。
- 訊息期限。
- 無法辨識要求訊息的格式。
- **MQRFH2** 標頭的完整性檢查失敗。
- 主要訊息內文的格式不是 `MQFMT_NONE`。
- 當 WebSphere MQ SOAP 接聽器正在處理要求時，已超出回應/重試臨界值。

WebSphere MQ SOAP 傳送端集 `MQRO_EXCEPTION_WITH_FULL_DATA` 和 `MQRO_EXPIRATION_WITH_FULL_DATA` 報告選項。 由於 WebSphere MQ SOAP 傳送端所設定的報告選項，報告訊息會包含整個原始要求訊息。 WebSphere MQ SOAP 傳送端也會設定 `MQRO_DISCARD` 選項，這會導致在傳回報告訊息之後捨棄訊息。 如果報告選項不符合您的需求，請撰寫您自己的寄件者來使用不同的 `MQRO_EXCEPTION` 和 `MQRO_DISCARD` 報告選項。 如果 SOAP 要求由未設定 `MQRO_DISCARD` 的不同傳送端傳送，則會將失敗訊息寫入無法傳送的郵件佇列 (DLQ)。

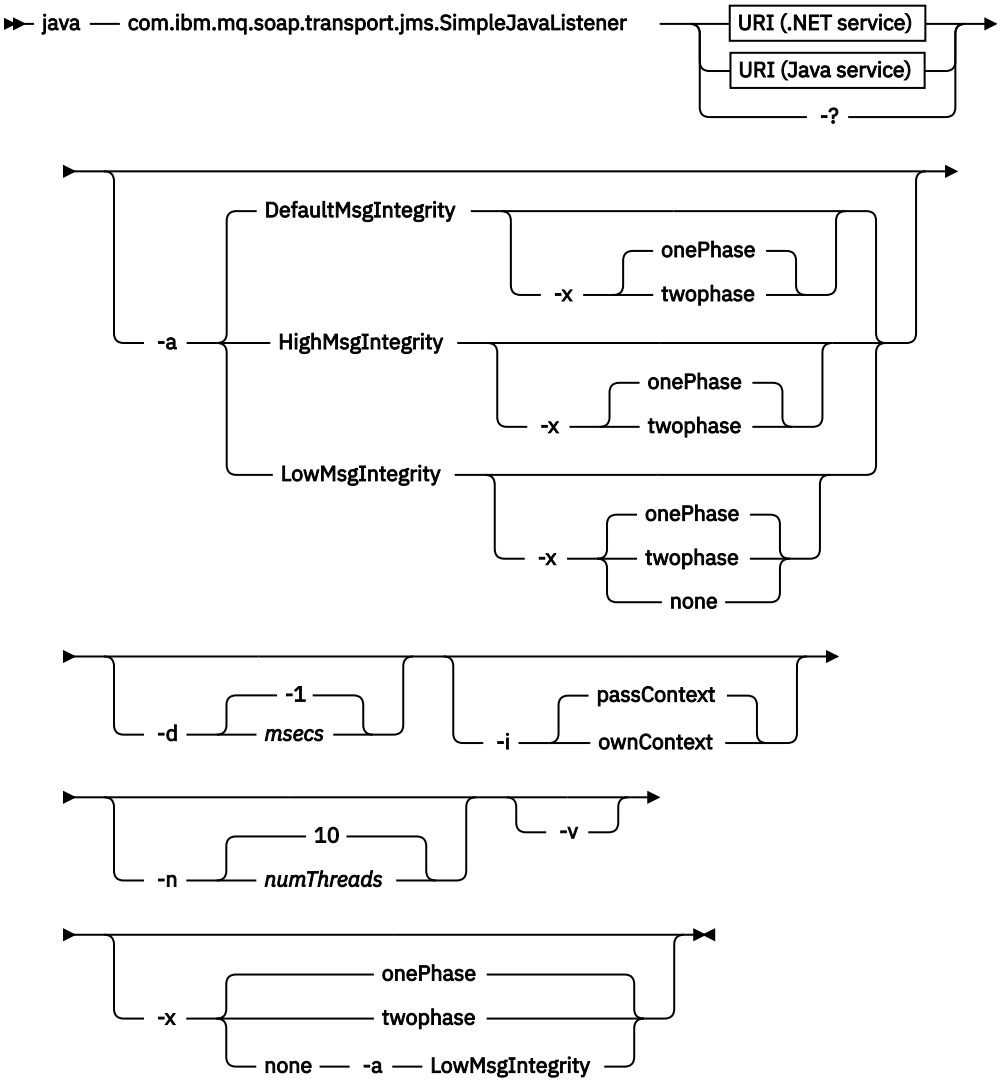
如果接聽器產生報告訊息，但在傳送報告的程序中失敗，則會將報告訊息傳送至 DLQ。 請確定 DLQ 處理程式正確處理這些訊息。

如果嘗試寫入無法傳送郵件的佇列時發生錯誤，則會將訊息寫入 WebSphere MQ 錯誤日誌。 接聽器是否繼續處理更多訊息取決於選取的訊息持續性和交易式選項。 如果接聽器以一段式交易式模式執行，且正在處理非持續要求訊息，則會捨棄原始訊息。 WebSphere MQ SOAP 接聽器繼續執行。 如果要求訊息持續存在，則要求訊息會取消至要求佇列並結束接聽器。 要求佇列設為禁止取得，以防止意外觸發的重新啟動。

Syntax diagram
.NET


amqwSOAPNETListener
URI (.NET service)
URI (Java service)
-?
-a
DefaultMsgIntegrity
-x
onePhase
twophase
HighMsgIntegrity
-x
onePhase
twophase
LowMsgIntegrity
-x
onePhase
twophase
none
-d
-1
msecs
-i
passContext
ownContext
-n
10
numThreads
-v
-w
c:\inetpub\wwwroot\
-x
onePhase
twophase
none
-a
LowMsgIntegrity

Java



必要參數

URI 平台
請參閱 第 885 頁的『Web 服務部署的 URI 語法和參數』。

-?
印出說明如何使用指令的說明文字。

選用參數

-a *integrityOption*
integrityOption 指定 WebSphere MQ SOAP 接聽器的行為 (如果無法將失敗要求訊息放置在無法傳送郵件的佇列上)。 *integrityOption* 可以採用下列其中一個值:

DefaultMsgIntegrity
對於非持續訊息，接聽器會顯示警告訊息，並繼續執行，並捨棄原始訊息。 若為持續訊息，它會顯示一則錯誤訊息，取消要求訊息，讓它保留在要求佇列上並結束。 如果省略 -a 選項，或未指定 *integrityOption* ，則會套用 DefaultMsgIntegrity 。

LowMsgIntegrity
對於持續及非持續訊息，接聽器會顯示警告並繼續執行，捨棄訊息。

HighMsgIntegrity
對於持續及非持續訊息，接聽器會顯示一則錯誤訊息，取消要求訊息，使其保留在要求佇列上並結束。

部署公用程式會檢查 -x 和 -a 旗標的相容性。 如果指定 -x none ，則必須指定 -a LowMsgIntegrity 。 如果旗標不相容，則部署公用程式會結束並顯示錯誤訊息，且未執行任何部署步驟。

-d 毫秒
毫秒 指定在任何執行緒上收到要求訊息時， WebSphere MQ SOAP 接聽器保持作用中的毫秒數。 如果 *msecs* 設為 -1，則接聽器會無限期地保持作用中。

-i 環境定義
環境定義 指定接聽器是否傳遞身分環境定義。 環境定義 採用下列值:

passContext
在回應訊息中設定原始要求訊息的身分環境定義。 SOAP 接聽器會檢查它是否有權從要求佇列儲存環境定義，並將它傳遞至回應佇列。 當開啟要儲存環境定義的要求佇列，以及要傳遞環境定義的回應佇列時，它會在執行時期進行檢查。 如果它沒有必要的權限，或 MQOPEN 呼叫失敗，則不會處理回應訊息。 回應訊息會放置在無法傳送郵件的佇列中，且無法傳送郵件的標頭包含來自失敗 MQOPEN 的回覆碼。 然後，接聽器會繼續正常處理後續的送入訊息。

ownContext
SOAP 接聽器不傳遞環境定義。 傳回的環境定義反映接聽器執行時所使用的使用者 ID ，而不是建立原始要求訊息的使用者 ID。

原始環境定義中的欄位是由佇列管理程式所設定，而不是由 SOAP 接聽器所設定。

-n *numThreads*
numThreads 指定 WebSphere MQ SOAP 接聽器的已產生啟動 Script 中的執行緒數目。 預設值為 10。 如果您有高訊息傳輸量，請考慮增加此數目。

-v
-v 設定外部指令的詳細輸出。 一律會顯示錯誤訊息。 使用 -v 來輸出您可以自訂以建立自訂部署 Script 的指令。

-w *serviceDirectory*
serviceDirectory 是包含 Web 服務的目錄。

-x 交易性
交易方式 指定接聽器的交易式控制項類型。 交易方式 可以設為下列其中一個值:

onePhase
使用 IBM WebSphere MQ 一段式支援。 如果系統在處理期間失敗，則會將要求訊息重新遞送至應用程式。 WebSphere MQ 交易可確保只寫入回應訊息一次。

twoPhase
使用兩階段支援。 如果適當地撰寫服務，則在服務的單一已確定執行內，訊息只會遞送一次 (與其他資源協調)。 這個選項只適用於伺服器連結連線。

none
無交易式支援。 如果系統在處理期間失敗，即使持續保存，也可能會遺失要求訊息。 服務可能已執行，也可能未執行，且可能寫入回應、報告或無法傳送郵件的訊息。

部署公用程式會檢查 -x 和 -a 旗標的相容性。 如需詳細資料，請參閱 -a 旗標的說明。

.NET 範例

```
amqwSOAPNETlistener
-u "jms:/queue?destination=myQ&connectionFactory=()
&targetService=myService&initialContextFactory=com.ibm.mq.jms.Nojndi"
-w C:/wmqsoap/demos
-n 20
```

Java 範例

```
java com.ibm.mq.soap.transport.jms.SimpleJavaListener
-u "jms:/queue?destination=myQ&connectionFactory=()
&initialContextFactory=com.ibm.mq.jms.Nojndi"
-n 20
```

IBM WebSphere MQ SOAP 傳送端的傳輸

提供 Axis 和 .NET Framework 1 及 .NET Framework 2 的傳送端類別。 傳送端建構 SOAP 要求並將它放置在佇列上，然後封鎖直到它從回應佇列讀取回應為止。 您可以從 SOAP 用戶端傳遞不同的 URI 來變更類別的行為。 若為 .NET Framework 3 ， 請使用 Microsoft Windows Communication Foundation (WCF) 的 WebSphere MQ 自訂通道。

用途

WebSphere MQ SOAP 傳送端會將呼叫 Web 服務的 SOAP 要求放到 WebSphere MQ 要求佇列中。 傳送者會根據 URI 中指定的選項，或根據預設值，在 **MQRFH2** 標頭中設定欄位。

如果您需要在可能使用 URI 選項以外變更寄件者的行為，請撰寫您自己的寄件者。 您的傳送端可以使用 SOAP 接聽器的 IBM WebSphere MQ 傳輸，或使用其他 SOAP 環境。 您的傳送端必須以 WebSphere MQ 所定義的格式來建構 SOAP 訊息。 該格式受 IBM WebSphere MQ SOAP 接聽器以及 WebSphere Application Server 和 CICS 提供的 SOAP 接聽器支援。 您的寄件者必須遵循 IBM WebSphere MQ 要求者的規則。 IBM WebSphere MQ SOAP 接聽器會傳回回覆和報告訊息。 如需如何在 **MQMD** 中設定報告選項的詳細資料，請參閱 第 865 頁的『MQMD SOAP 設定』 。 報告選項控制 WebSphere MQ SOAP 接聽器所傳回的報告訊息。

說明

WebSphere MQ SOAP Java 傳送端已向 Axis 主機環境登錄 `jms:` URI 字首。 傳送端在 `com.ibm.mq.soap.transport.jms.WMQSender` 類別中實作，其衍生自 `org.apache.axis.handlers.BasicHandler`。 如果 Axis 主機環境偵測到 `jms:` URI 字首，則會呼叫 `com.ibm.mq.soap.transport.jms.WMQSender` 類別。 在放置訊息之後，類別會封鎖，直到它已從回應佇列讀取回應為止。 如果在逾時間隔內未收到任何回應，傳送端會擲出異常狀況。 如果在逾時間隔內收到回應，則會使用 Axis 架構將回應訊息傳回給用戶端。 您的用戶端應用程式必須能夠處理這些回應訊息。

對於 Microsoft .NET Framework 1 和 .NET Framework 2 服務， WebSphere MQ SOAP 傳送端在類別 `IBM.WMQSOAP.MQWebRequest` 中實作，該類別衍生自 `System.Net.WebRequest` 和 `System.Net.IwebRequestCreate`。 如果 .NET Framework 1 或 .NET Framework 2 偵測到 `jms:` URI 字首，則會呼叫 `IBM.WMQSOAP.MQWebRequest` 類別。 傳送端會建立 `MQWebResponse` 物件，以讀取回應佇列中的回應訊息，並將它傳回給用戶端。

`com.ibm.mq.soap.transport.jms.WMQSender` 是最終類別， `IBM.WMQSOAP.MQWebRequest` 是密封的。 您無法透過建立子類別來修改其行為。

參數

設定 URI ， 以控制 IBM WebSphere MQ SOAP 傳送端在 Web 服務 SOAP 用戶端中的行為。 部署公用程式會建立 Web 服務用戶端 Stub ， 並納入提供給部署公用程式的 URI 選項。

搭配使用通道定義表與 WebSphere MQ SOAP 傳送端的 SOAP 傳輸

用戶端連線通道定義是在 Web 服務 URI 的 ConnectionFactory 屬性中設定連線內容的替代方案。 連線內容為 clientChannel、 clientConnection 及 SSL 參數。

說明

透過定義用戶端連線來建立用戶端通道說明表格。 即使 Web 服務用戶端連接至不同的佇列管理程式，也會在單一佇列管理程式上的連線表格中建立所有連線。 連線表格的預設名稱和位置是 *queue manager directory*`/@ipcc/AMQCLCHL.TAB`。

透過設定 `com.ibm.mq.soap.transport.jms.mqchlurl` 系統內容，將連線表格的位置傳遞至 Java 用戶端。

透過設定 `MQCHLLIB` 及 `MQCHLTAB` 環境變數，將連線表格的位置傳遞至 .NET 用戶端。

您可以在 Web 服務 URI 的 ConnectionFactory 屬性中同時提供通道連線表格和通道連線參數。 ConnectionFactory 中設定的值優先於通道定義表中的值。

在 Java 中使用通道定義表

```
java -Dcom.ibm.msg.client.config.location=file:/C:/mydir/myjms.config MyAppClass
```

圖 *22:* 使用配置檔啟動 *Java* 用戶端

```
com.ibm.mq.soap.transport.jms.mqchlurl=file:/C:/ibm/wmq/qmgrs/QM1/@ipcc/AMQCLCHL.TAB
```

圖 *23: myjms.config*

交易

在啟動接聽器時，請使用 -x 選項，以交易方式執行 Web 服務。 在服務 URI 中設定 **持續性** 選項，以選取訊息的完整性。

Web 服務

在啟動接聽器時，請使用 -x 選項，以交易方式執行 Web 服務。 在 .NET Framework 1 和 2 上， SOAP 接聽器使用 Microsoft Transaction Coordinator (MTS)。 在 Axis 1.4 上， SOAP 接聽器使用佇列管理程式協調的交易。

Web 服務用戶端

SOAP 寄件者不是交易式。

WebSphere MQ 連結

您可以設定 SOAP 傳送端的連結類型。 它可以作為 WebSphere MQ 伺服器應用程式或用戶端應用程式進行連接。 在 .NET 上，您也可以將 SOAP 傳送端連結成 XA 用戶端。

訊息持續性

在 URI 中設定 **持續性** 選項，以選取持續性層次。

Web 服務交易

您可以使用 Web 服務交易，因為 SOAP 傳送端不是交易式。 如果您撰寫自己的 SOAP 傳送端，且想要使用 Web 服務交易，請不要建立交易式 SOAP 傳送端。 您無法在相同交易中傳送要求訊息及接收回覆訊息。 傳送和接收不得由 Web 服務交易來協調。

Web 服務部署的 URI 語法和參數

部署 IBM WebSphere MQ Web 服務的語法和參數定義在 URI 中。 部署公用程式會根據 Web 服務的名稱來產生預設 URI。 您可以透過將自己的 URI 定義為部署公用程式的參數來置換預設值。 部署公用程式會在產生的 Web 服務用戶端 Stub 中納入 URI。

用途

使用「通用資源 ID (URI)」來指定 Web 服務。 語法圖指定 SOAP 的 IBM WebSphere MQ 傳輸中支援的 URI。 URI 會控制 IBM WebSphere MQ 特定的 SOAP 參數，以及用來存取目標服務的選項。 URI 與 .NET、Apache Axis 1、 WebSphere Application Server CICS 所管理的 Web 服務相容。

說明

URI 會納入部署公用程式所產生的 Web 服務用戶端類別中。 用戶端會將 URI 傳遞至 IBM WebSphere MQ 訊息中的 IBM WebSphere MQ SOAP 傳送端。 URI 控制 IBM WebSphere MQ SOAP 傳送端和 IBM WebSphere MQ SOAP 接聽器兩者所執行的處理。

Syntax

The URI syntax is as follows:

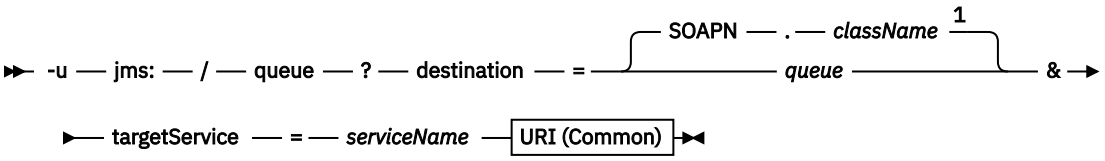
```
jms:/queue?name=value&name=value...
```

where name is a parameter name and *value* is an appropriate value, and the name=*value* element can be repeated any number of times with the second and subsequent occurrences being preceded by an ampersand (&).

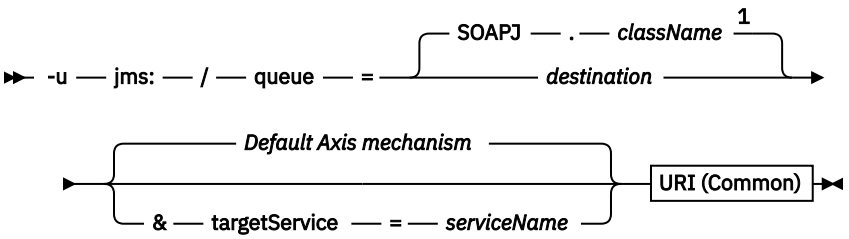
Parameter names are case-sensitive, as are names of IBM WebSphere MQ objects. If any parameter is specified more than once, the final occurrence of the parameter takes effect. Client applications can override a generated parameter by appending another copy of the parameter to the URI. If any additional unrecognized parameters are included, they are ignored.

If you store a URI in an XML string, you must represent the ampersand character as `&`. Similarly, if a URI is coded in a script, take care to escape characters such as & that would otherwise be interpreted by the shell.

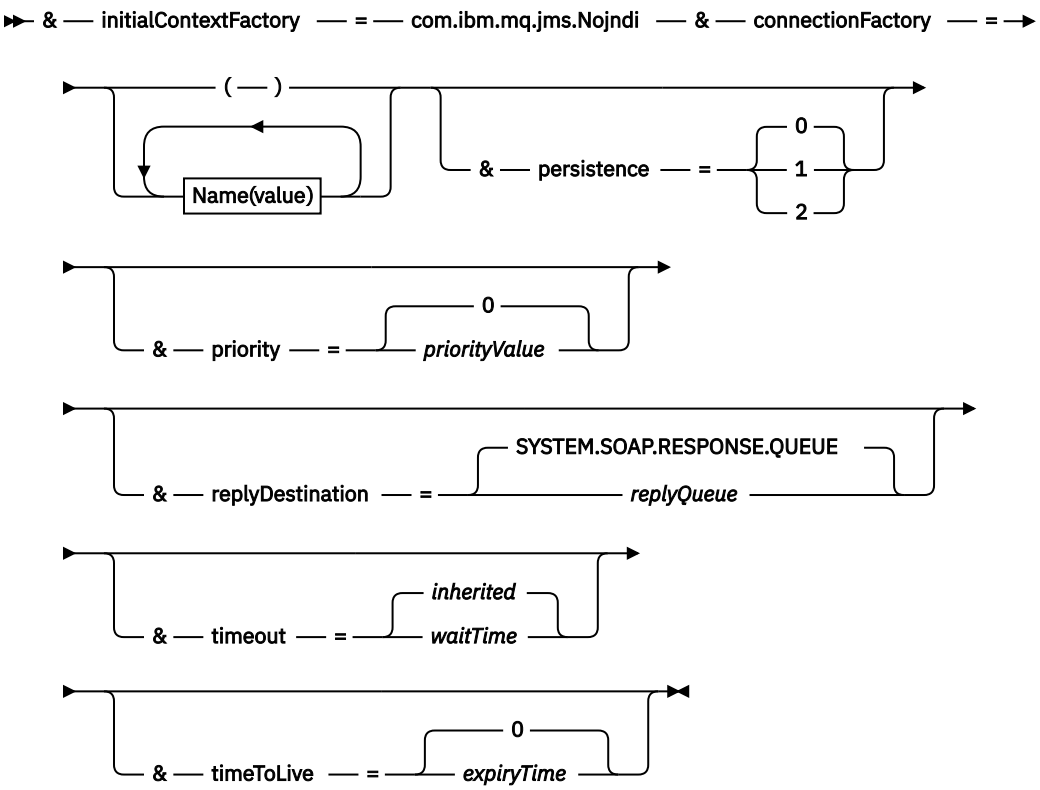
Syntax diagram
URI (.NET service)



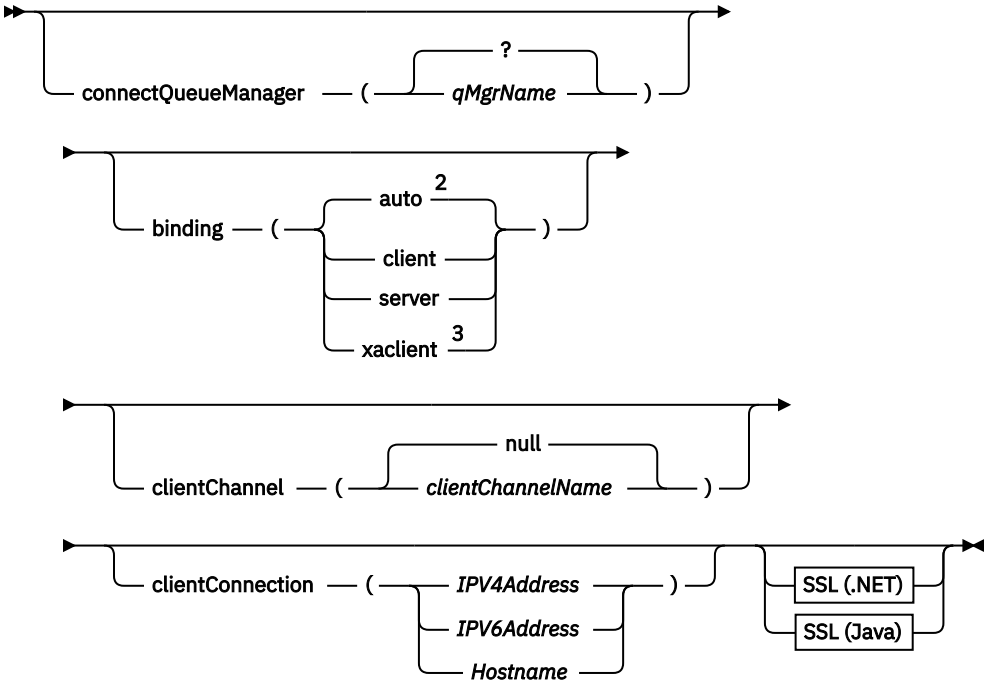
URI (Java service)



URI (Common)



***Name*(*value*)**



註:

[1] The queue manager transforms *className* to a queue name following the steps described in 第 887 頁的『目的地至佇列名稱轉換』

[2] `client` is the default if other options appropriate for a client are specified; for example `clientConnection`.

[3] `xaclient` applies to .NET only

目的地至佇列名稱轉換

1. *className* 以 `SOAPJ.` (適用於 Java 服務) 或 `SOAPN.` (適用於 .NET 服務) 作為字首。
2. 副檔名會從 *className* 參數中給定的完整路徑名稱移除。
3. 產生的字串會截斷為不超過 48 個字元
4. 目錄分隔字元取代為句點字元。
5. 內含的空格會取代為底線字元。
6. 磁碟機字首字母後面的冒號取代為 .NET 服務的句點。

註: 在某些環境中，部署公用程式所產生的佇列名稱可能不是唯一的。 部署公用程式會檢查是否要建立佇列。 您可以選擇透過重組部署目錄階層或自訂提供的部署程序來置換部署公用程式。

必要的 URI 參數

`destination=`佇列

queue 是要求目的地的名稱。 它可以是佇列或佇列別名。 如果它是佇列別名，則別名可以解析為主題。

- 如果省略 -u 參數，則會使用 第 887 頁的『目的地至佇列名稱轉換』中說明的步驟，從 *classname* 產生 佇列 。
- 如果指定-u 參數，則 *queue* 是必要參數，且必須是 URI 在起始 `jms:/queue?` 之後的第一個參數 字串。 指定 IBM WebSphere MQ 佇列名稱，或以 @ 符號連接的佇列名稱和佇列管理程式名稱，例如 `SOAPN.trandemos@WMQSOAP.DEMO.QM`。
- 部署公用程式會檢查所產生或提供的佇列名稱是否符合現有佇列的名稱。 所採取的動作在 第 888 頁的表 587 中說明。

表 587: *queue* 驗證

接聽器 Script 存在嗎?	**接聽器 Script 存在於 ./generated/server 目錄中**		**接聽器 Script 不存在於 ./generated/server 目錄中**
接聽器 Script 中的佇列符合 *queue*?	***queue* 不符合接聽器 Script 中使用的要求佇列**	***queue* 符合接聽器 Script 中使用的要求佇列**	
***queue* 存在**	– 部署結束時發生錯誤。 – 服務已部署在 ./generated/server 中，但使用不同的佇列。	– **正常繼續部署**。 – 服務已部署在 ./generated/server 中	– 部署結束時發生錯誤。 – 在 ./generated/server 中找不到接聽器啟動 Script，但不同的服務或應用程式正在使用 *queue*。
***queue* 不存在**		– **部署繼續進行，但有警告**。 – 前一個部署可能失敗，因為啟動有效，但遺漏 *queue*。	– **正常繼續部署**。 – 尚未從這個目錄部署任何服務。

&connectionFactory=名稱 *(值)*
名稱 是下列其中一個參數:

- connectQueue 管理程式 (qMgr 名稱)
- 連結 (bindingType)
- clientChannel(通道)
- clientConnection(連線)
- 第 876 頁的『必要的 SSL 參數 (Java)』

如需這些參數值的說明，請參閱 第 889 頁的『Connection Factory 參數』。

&targetService=*serviceName*
[8] 在 .NET 上， *serviceName* 是位於部署目錄中的 .NET 服務名稱，例如: `targetService=myService.asmx`。 在 .NET 環境中， targetService 參數可讓單一 WebSphere MQ SOAP 接聽器能夠處理多個服務的要求。 這些服務必須從相同的目錄部署。

選用 URI 參數

&initialContextFactory=*contextFactory*
contextFactory 是必要項目，且必須設為 `com.ibm.mq.jms.Nojndi`。 請確定 `Nojndi.jar` 位於 WebSphere Application Server Web 服務用戶端的類別路徑中。 `Nojndi.jar` 會根據 `connectionFactory` 和 `destination` 參數的內容來傳回 Java 物件，而不是透過參照目錄來傳回。

&targetService=*serviceName*
[9] 在 Axis 上， *serviceName* 是 Java 服務的完整名稱，例如: `targetService=javaDemos.service.StockQuoteAxis`。 如果未指定 targetService，則會使用預設 Axis 機制來載入服務。

&persistence=*messagePersistence*
messagePersistence 採用下列其中一個值:

0
持續性繼承自佇列定義。

[8] 僅限 .NET 服務
[9] 僅限 Java 服務

1
訊息是非持續性。

2
訊息持續

&priority=*priorityValue*
priorityValue 在 0-9 範圍內。 0 是低優先順序。 預設值為環境專用，若為 IBM WebSphere MQ ，則為 0。

&replyDestination=*replyTo* 佇列
用戶端要用於回應訊息的佇列。 預設回覆佇列為 SYSTEM.SOAP.RESPONSE.QUEUE。

- 執行 setupWMQSOAP Script ， 以建立預設 WebSphere MQ SOAP 物件。
- 指定 *replyTo* 佇列 的模型佇列，以建立暫時或永久動態回應佇列。 對於暫時及永久動態回應佇列，會針對每一個要求建立動態佇列的個別實例。 如果發生下列任何事件，則會刪除佇列:
  - 回應到達並已處理。
  - 要求逾時。
  - 要求程式終止。

  若要取得最佳效能，請使用暫時動態佇列，而非永久動態佇列。 請勿將持續要求訊息傳送至具有暫時動態佇列的 URI。 IBM WebSphere MQ 接聽器 SOAP 無法處理訊息並輸出錯誤。 用戶端在等待回覆時逾時。
- setupWMQSOAP Script 會建立稱為 SYSTEM.SOAP.MODEL.RESPONSE.QUEUE 的預設永久動態模型佇列。

&timeout=*waitTime*
用戶端等待回應訊息的時間 (毫秒)。 *waitTime* 會置換基礎架構或用戶端應用程式所設定的值。 如果未指定，則會繼承應用程式值 (如果已指定) 或基礎架構預設值。

註: 在逾時與 timeToLive 之間未施行任何關係。

&timeToLive=*expiryTime*
expiryTime 是訊息到期之前指定的時間 (毫秒)。 預設值為零，表示生命期限無限制。

註: 在逾時與 timeToLive 之間未施行任何關係。

Connection Factory 參數

connectQueueManager(*qMgr* 名稱)
qMgr 名稱 指定用戶端所連接的佇列管理程式。 預設值為空白。

binding(*bindingType*)
bindingType 指定用戶端如何連接至 *qMgrName*。 預設值為 auto。 *bindingType* 採用下列值:

auto
傳送端依序嘗試下列連線類型:

1. 如果指定適用於用戶端連線的其他選項，傳送端會使用用戶端連結。 其他選項為 clientConnection 或 clientChannel。
2. 使用伺服器連線。
3. 使用用戶端連線。

如果 SOAP 用戶端沒有本端佇列管理程式，請在 *URI* 中使用 binding(auto) 。 為 SOAP 用戶端建置用戶端連線。

client
使用 *URI* 中的 binding(**用戶端**) 來建置 SOAP 傳送端的用戶端配置。

server
在 *URI* 中使用 binding(**伺服器**) ，以建置 SOAP 傳送端的伺服器配置。 如果連線具有用戶端類型參數，則連線會失敗，且 IBM WebSphere MQ SOAP 傳送端會顯示錯誤。 用戶端類型參數為 clientConnection、 clientChannel 或 SSL 參數。

xaclient
xaclient 僅適用於 .NET，不適用於 Java 用戶端。使用 XA 用戶端連線。

clientChannel(通道)

SOAP 用戶端使用 通道 來建立 IBM WebSphere MQ 用戶端連線。除非在伺服器上啟用通道自動定義，否則 *channel* 必須符合伺服器連線通道的名稱。除非您已提供「用戶端連線定義」表格 (CCDT)，否則 clientChannel 是必要參數。

設定 `com.ibm.mq.soap.transport.jms.mqchlurl`，以 Java 提供 CCDT。在 .NET 中，設定 `MQCHLLIB` 和 `MQCHLTAB` 環境變數; 請參閱 第 884 頁的『搭配使用通道定義表與 WebSphere MQ SOAP 傳送端的 SOAP 傳輸』。

clientConnection(連線)
SOAP 用戶端使用 連線 來建立 IBM WebSphere MQ 用戶端連線。預設主機名稱是 localhost，預設埠是 1414。如果 連線 是 TCP/IP 位址，則它採用三種格式之一，且可以使用埠號作為字尾。

JMS 用戶端可以使用格式：`hostname:port`，或使用格式 %X 來「跳出」方括弧，其中 X 是代表 URI 字碼頁中方括弧字元的十六進位值。例如，在 ASCII 中，%28 和 %29 分別代表（和）。

.Net 用戶端可以明確使用方括弧：`hostname(port)` 或使用「跳出」格式。

IPv4 位址
例如，`192.0.2.0`。

IPv4 位址
例如，`2001:DB8:0:0:0:0:0:0`。

主機名稱
例如，`www.example.com%281687%29`、`www.example.com:1687` 或 `www.example.com(1687)`。

SSL 平台
請參閱 第 876 頁的『必要的 SSL 參數 (Java)』。

範例 URI

註:

1. URI 中的 & 編碼為 `&`
2. 先前列出的所有參數都適用於用戶端。
3. 只有 **destination**、**connectionFactory** 和 **initialContextFactory** 適用於 WCF 服務。

```
jms:/queue?
destination=myQ&amp;connectionFactory=()&amp;initialContextFactory=com.ibm.mq.jms.Nojndi
```

圖 *24: Axis* 服務的 *URI*，只提供必要的參數

```
jms:/queue?destination=myQ&amp;connectionFactory=()&amp;targetService=MyService.asmx
&amp;initialContextFactory=com.ibm.mq.jms.Nojndi
```

圖 *25: .NET* 服務的 *URI*，只提供必要的參數

```
jms:/queue?destination=myQ@myRQM&amp;connectionFactory=connectQueueManager(myconnQM)
binding(client)clientChannel(myChannel)clientConnection(myConnection)
&amp;initialContextFactory=com.ibm.mq.jms.Nojndi
```

圖 *26: Axis* 服務的 *URI*，提供一些選用 *connectionFactory* 參數

```
jms:/queue?destination=myQ@myRQM&amp;connectionFactory=connectQueueManager(myconnQM)
binding(client)clientChannel(myChannel)clientConnection(myConnection)
sslPeerName(CN=MQ Test 1,O=IBM,S=Hampshire,C=GB)
&amp;initialContextFactory=com.ibm.mq.jms.Nojndi
```

圖 *27: Axis* 服務的 *URI*，提供 *connectionFactory* 參數的 *sslPeer* 名稱 選項

Nojndi 機制

Nojndi 機制可讓使用 JNDI 介面的 JMS 程式使用與不使用 JNDI 的 WebSphere MQ 程式相同的 URI。

您可以使用 SOAP 的 WebSphere MQ 傳輸，在 WebSphere Application Server 上呼叫 Web 服務。 WebSphere Application Server SOAP over JMS 會使用 JNDI 來查閱 JMS 資源。 Web 服務用戶端可能在 .NET 上執行，或使用 Axis 1.4 來呼叫 Web 服務，但不使用 JNDI。 如果要對用戶端和伺服器使用相同的 URL，不論環境是否使用 JNDI，它都必須提供相同的資訊。

Web 服務用戶端傳給 SOAP 的 WebSphere MQ 傳輸的 URI 包含特定的 WebSphere MQ 佇列管理程式和佇列名稱。 這些名稱由 WebSphere MQ SOAP 支援直接剖析及使用。

Nojndi 機制會將 JMS 程式使用的 initialContextFactory 引導至 com.ibm.mq.jms.Nojndi。 com.ibm.mq.jms.Nojndi 類別是 JNDI 介面的實作，它會從 URL 以 ConnectionFactory 和 Queue Java 物件傳回 connectionFactory 和 destination 。 如果 JMS 實作是 WebSphere MQ， MQConnectionFactory 和 MQQueue 會繼承自 ConnectionFactory 和 Queue 類別。

透過使用 Nojndi 機制，您可以使用相同的 URL 來提供與 WebSphere Application Server 和 .NET 的相同連線資訊。

W3C WebSphere MQ Axis 2 用戶端的 SOAP over JMS URI

定義 W3C SOAP over JMS URI，以使用 WebSphere MQ JMS 作為 SOAP 傳輸從 Axis 2 用戶端呼叫 Web 服務。 Web 服務必須由支援 WebSphere MQ JMS 的伺服器提供，以及 SOAP/JMS 連結的 W3C SOAP over JMS 候選建議。

說明

W3C 候選項建議定義 SOAP over JMS 連結； SOAP over Java Message Service 1.0。 也適用於其範例是 URI Scheme for Java（tm） Message Service 1.0[10].

使用語法圖來建立語法正確且 WebSphere MQ Axis 2 用戶端接受的 W3C SOAP over JMS URI。 它只能定義 WebSphere MQ Axis 2 用戶端所接受的 URI。 從兩個方面來看，它是 W3C 建議的子集:

1. 不支援 jms-variant 主題，且不得在傳遞至 WebSphere MQ Axis 2 用戶端的 URI 中指定。
2. 語法圖會省略下列內容，因為它們是 JMS 內容，而不是 URI 的一部分。
    a. bindingVersion
    b. contentType
    c. soapAction
    d. requestURI
    e. isFault

JMS 內容由 Axis 2 用戶端或伺服器設定。

此圖透過定義自訂參數 connectionFactory 來延伸 W3C 建議。 connectionFactory 是 JNDI 的替代方案，用來指定 Axis 2 用戶端如何使用佇列來連接佇列管理程式。

WebSphere MQ Axis 2 用戶端只接受內容作為用戶端應用程式傳遞給用戶端的 URI 的一部分，或作為環境變數。 WebSphere MQ Axis 2 用戶端無法處理 WSDL 文件。 用戶端應用程式或開發工具可能會處理 WSDL，並建立 URI 以傳遞至 Axis 2 用戶端。 WebSphere MQ Axis 2 用戶端應用程式無法直接設定 JMS 訊息內容。

Syntax

In accordance with the W3C recommendation, all the parameters can be obtained from environment variables. The environment variable names are formed by prefacing the parameter name with soapjms_. The syntax is: soapjms_*parameterName*; for example,

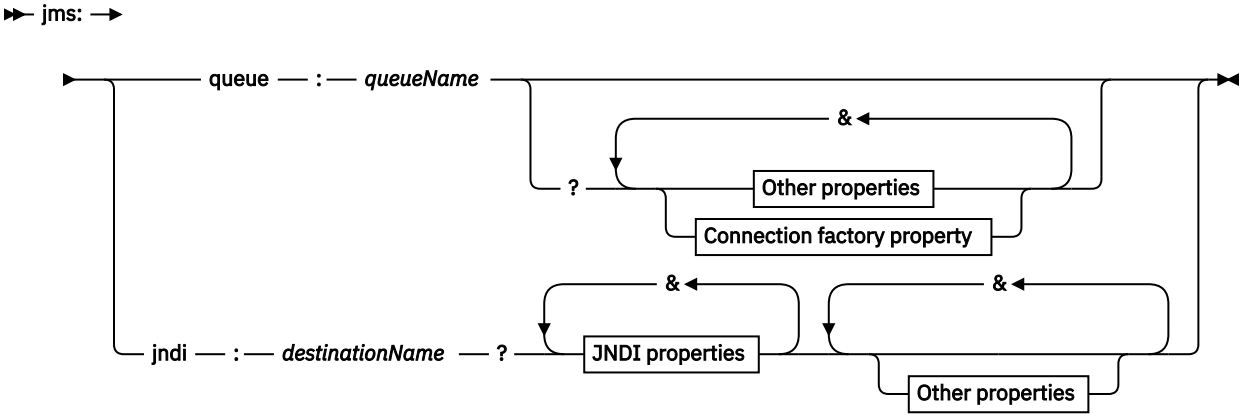
```
set soapjms_targetServer=com.example.org.stockquote
```

10 請在 W3C 規格參照中尋找最新草稿的 *URI Scheme for JMS*。

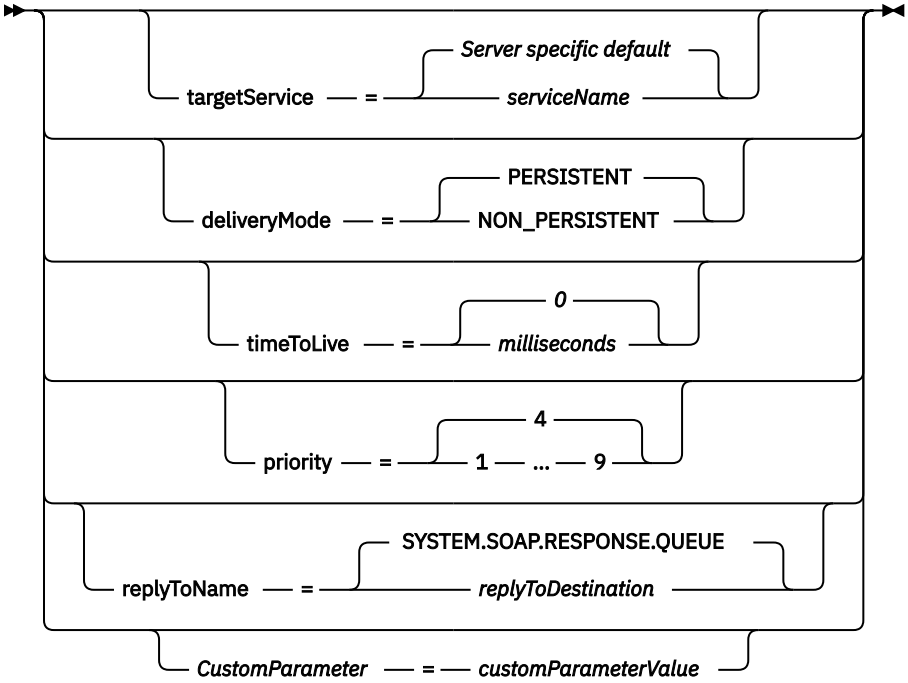
If a parameter is set using an environment variable it overrides the value set in the URI.

In accordance with the W3C recommendation, all the parameters can be repeated. The last instance of a parameter is used, unless overridden by an environment variable.

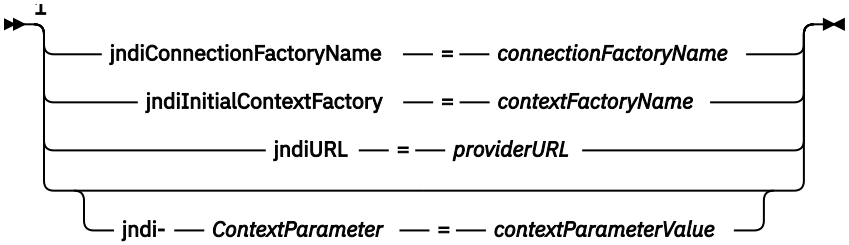
jms-uri



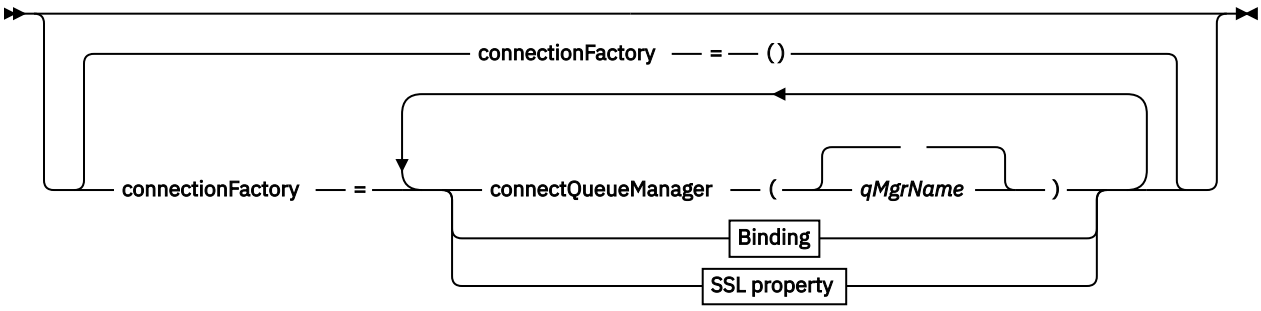
Other properties



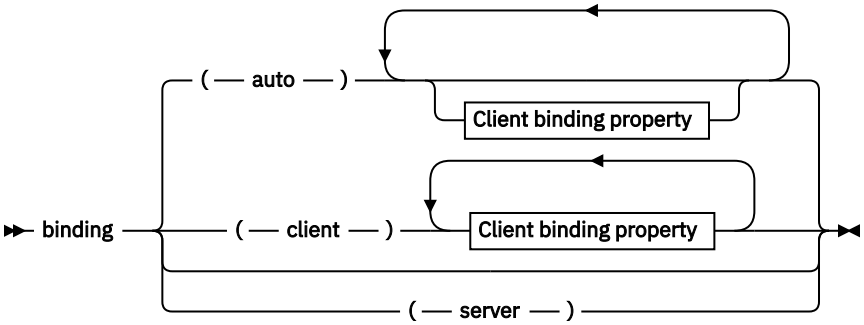
JNDI properties



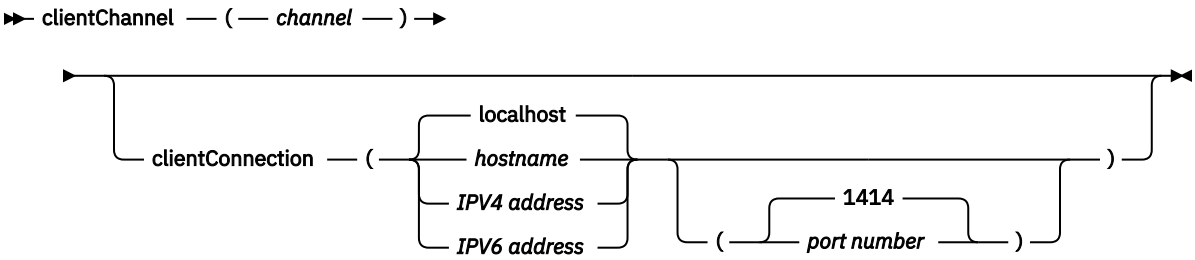
Connection factory property



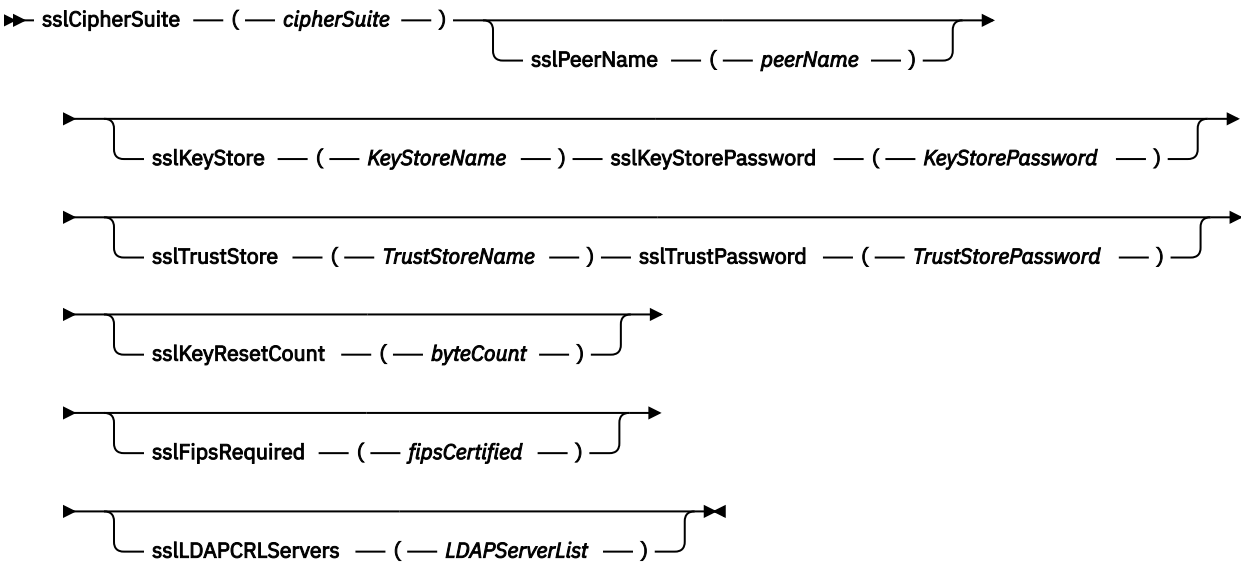
Binding



Client binding property



SSL property



註:

[1] **jndiConnectionFactoryName**, **jndiConnectionFactoryName** and **jndiURL** are all required parameters. **jndi-***ContextParameter* is optional.

參數

connectionFactory=*connectionFactoryParameterList*

connectionFactoryParameterList 是當目的地變式為 queue 時，限定 Axis 2 用戶端如何連接佇列管理程式的參數。

connectionFactory 不得與 jndi 目的地變式一起指定。

不會將參數傳遞給要求 URI 中的伺服器。

如果省略 connectionFactory，則佇列必須屬於與 Axis 2 用戶端在相同伺服器上執行的預設佇列管理程式。

connectionFactoryParameterList:

binding(*bindingType*)

bindingType 指定用戶端如何連接至 *qMgrName*。 預設值為 auto。 *bindingType* 採用下列值:

auto

傳送端依序嘗試下列連線類型:

1. 如果指定適用於用戶端連線的其他選項，傳送端會使用用戶端連結。 其他選項為 clientConnection 或 clientChannel。
2. 使用伺服器連線。
3. 使用用戶端連線。

如果 SOAP 用戶端沒有本端佇列管理程式，請在 *URI* 中使用 `binding(auto)`。 為 SOAP 用戶端建置用戶端連線。

client

使用 *URI* 中的 `binding(用戶端)` 來建置 SOAP 傳送端的用戶端配置。

server

在 *URI* 中使用 `binding(伺服器)`，以建置 SOAP 傳送端的伺服器配置。 如果連線具有用戶端類型參數，則連線會失敗，且 IBM WebSphere MQ SOAP 傳送端會顯示錯誤。 用戶端類型參數為 clientConnection、 clientChannel 或 SSL 參數。

xaclient

xaclient 僅適用於 .NET，不適用於 Java 用戶端。 使用 XA 用戶端連線。

clientChannel(通道)

SOAP 用戶端使用 通道 來建立 IBM WebSphere MQ 用戶端連線。 除非在伺服器上啟用通道自動定義，否則 *channel* 必須符合伺服器連線通道的名稱。 除非您已提供「用戶端連線定義」表格 (CCDT)，否則 clientChannel 是必要參數。

設定 `com.ibm.mq.soap.transport.jms.mqchlurl`，以 Java 提供 CCDT。 在 .NET 中，設定 `MQCHLLIB` 和 `MQCHLTAB` 環境變數; 請參閱 第 884 頁的『搭配使用通道定義表與 WebSphere MQ SOAP 傳送端的 SOAP 傳輸』。

clientConnection(連線)

SOAP 用戶端使用 連線 來建立 IBM WebSphere MQ 用戶端連線。 預設主機名稱是 localhost，預設埠是 1414。 如果 連線 是 TCP/IP 位址，則它採用三種格式之一，且可以使用埠號作為字尾。

JMS 用戶端可以使用格式: `hostname:port`，或使用格式 %X 來「跳出」方括弧，其中 X 是代表 URI 字碼頁中方括弧字元的十六進位值。 例如，在 ASCII 中， %28 和 %29 分別代表 (和)。

.Net 用戶端可以明確使用方括弧: `hostname(port)` 或使用「跳出」格式。

IPv4 位址

例如，`192.0.2.0`。

IPv4 位址

例如，`2001:DB8:0:0:0:0:0:0`。

主機名稱

例如， `www.example.com%281687%29`、 `www.example.com:1687` 或 `www.example.com(1687)`。

sslCipherSuite(*CipherSuite*)
CipherSuite 指定通道上使用的 sslCipher 套組 。 用戶端指定的 CipherSuite 必須符合伺服器連線通道上指定的 CipherSuite 。

sslFipsRequired(*fipsCertified*)
fipsCertified 指定 *CipherSpec* 或 *CipherSuite* 是否必須在通道上 IBM WebSphere MQ 中使用 FIPS 認證的加密法。 設定 *fipsCertified* 的效果與在 MQCONNX 呼叫上設定 **MQSCO** 結構的 FipsRequired 欄位相同。

sslKeyStore(*KeyStore* 名稱)
KeyStore 名稱 指定通道上使用的 sslKeyStoreName 。 金鑰儲存庫會保留用來向伺服器鑑別用戶端的用戶端私密金鑰。 如果 SSL 連線配置為接受匿名用戶端連線，則金鑰儲存庫是選用的。

sslKeyResetCount(*bytecount*)
bytecount 指定在必須重新協議 SSL 秘密金鑰之前，透過 SSL 通道傳遞的位元組數。 若要停用 SSL 金鑰的重新協議，請省略欄位或將它設為零。 零是部分環境中唯一支援的值，請參閱 在 WebSphere MQ for Java 類別中重新協議秘密金鑰。 設定 sslKeyResetCount 的效果與在 MQCONNX 呼叫的 **MQSCO** 結構中設定 KeyResetCount 欄位相同。

sslKeyStorePassword(*KeyStore* 密碼)
KeyStore 密碼 指定通道上使用的 sslKeyStorePassword 。

sslLDAPCRLServers(*LDAPServerList*)

LDAPServerList 指定要用於「憑證撤銷清冊」檢查的 LDAP 伺服器清單。

對於已啟用 SSL 的用戶端連線， *LDAPServerList* 是要用於「憑證撤銷清冊 (CRL)」檢查的 LDAP 伺服器清單。 會根據其中一個列出的 LDAP CRL 伺服器來檢查佇列管理程式提供的憑證; 如果找到，連線會失敗。 系統會依序嘗試每一部 LDAP 伺服器，直到建立與其中一部 LDAP 伺服器的連線功能為止。 如果無法連接至任何伺服器，則會拒絕憑證。 順利建立與其中一個的連線之後，會根據該 LDAP 伺服器上存在的 CRL 來接受或拒絕憑證。

如果 *LDAPServerList* 為空白，則不會針對「憑證撤銷清冊」檢查屬於佇列管理程式的憑證。 如果提供的 LDAP URI 清單無效，則會顯示錯誤訊息。 設定此欄位的效果與包括 MQAIR 記錄並從 MQCONNX 上的 **MQSCO** 結構存取這些記錄的效果相同。

sslPeerName(*peerName*)
peerName 指定通道上使用的 sslPeer 名稱 。

sslTrustStore(*TrustStore* 名稱)
TrustStore 名稱 指定通道上使用的 sslTrustStoreName 。 信任儲存庫會保留伺服器或其金鑰鏈的公用憑證，以向用戶端鑑別伺服器。 如果使用憑證管理中心的主要憑證來鑑別伺服器，則信任儲存庫是選用的。 在 Java 中，主要憑證保留在 JRE 憑證儲存庫 cacerts 中。

sslTrustStorePassword(*TrustStore* 密碼)
TrustStore 密碼 指定通道上使用的 sslTrustStorePassword 。

***CustomParameter=customParameter* 值**

CustomParameter 是自訂參數的使用者定義名稱，而 *customParameterValue* 是參數的值。
Axis 2 用戶端未使用的自訂參數由 Axis 2 用戶端傳送至 SOAP 伺服器。 請參閱伺服器文件。
connectionFactory 是 Axis 2 用戶端使用的自訂參數，不會傳遞給伺服器。
CustomParameter 不得符合現有參數的名稱。
如果 *CustomParameter* 以字串 jndi- 開頭，則是用來查閱 JNDI 目的地; 請參閱 jndi-。

deliveryMode=*deliveryMode*
deliveryMode 設定訊息持續性。 預設值為 PERSISTENT。

jndi:*destinationName*
destinationName 是對映至 JMS 佇列的 JNDI 目的地名稱。 如果指定 jndi 目的地變式，您必須提供 *destinationName*。

jndiConnectionFactoryName=*connectionFactory* 名稱
connectionFactory 名稱 設定 Connection Factory 的 JNDI 名稱。 如果目的地變式是 jndi，則必須提供 *connectionFactory* 名稱 。

jndiInitialContextFactory=*contextFactory* 名稱
contextFactory 名稱 設定起始環境定義 Factory 的 JNDI 名稱。 如果目的地變式是 jndi，則必須提供 *contextFactory* 名稱 。 請參閱 使用 JNDI 來擷取 JMS 應用程式中的受管理物件。

jndiURL=*providerURL*
jndiURL 設定 JNDI 提供者的 URL 名稱。 如果目的地變式是 jndi，則必須指定 *jndiURL* 。

jndi-*ContextParameter*=*contextParameter* 值
jndi-*ContextParameter* 是用來傳遞資訊給 JNDI 提供者之自訂參數的使用者定義名稱。 *contextParameter* 值 是傳遞的資訊。

priority=*priorityValue*
priorityValue 設定 JMS 訊息優先順序。 0 低， 9 高。 預設值為 4。

queue:*queueName*
queueName 是放置 SOAP 要求的 JMS 佇列名稱。 如果指定佇列變式，則必須提供佇列名稱。 如果佇列不屬於與用戶端相同的伺服器上的預設佇列管理程式，請設定 connectionFactory 參數。

replyToName=*replyTo* 目的地
replyTo 目的地 會設定目的地佇列名稱。 如果目的地變式是 jndi，則名稱是必須對映至佇列的 JNDI 名稱。 如果變式是 queue ，則名稱是 JMS 佇列。 預設值為 SYSTEM.SOAP.RESPONSE.QUEUE。

targetService=*serviceName*
SOAP 伺服器用來啟動目標 Web 服務的名稱。

在 Axis 上， *serviceName* 是 Java 服務的完整名稱，例如: `targetService=www.example.org.StockQuote`。 如果未指定 targetService ，則會使用預設 Axis 機制來載入服務。

timeToLive=毫秒
將 毫秒 設為訊息到期之前的時間。 預設值 0 是訊息永不到期。

範例

```
jms:jndi:REQUESTQ
    ?jndiURL=file:/C:/JMSAdmin
    &jndiInitialContextFactory=com.sun.jndi.fscontext.RefFSContextFactory
    &jndiConnectionFactoryName=ConnectionFactory
    &replyToName=RESPONSEQ
    &deliveryMode(NON_PERSISTENT)
```

圖 *28:* 使用 *jms: jndi* 來傳送 *SOAP/JMS* 要求

```
jms:queue:SOAPJ.demos
    ?connectionFactory=connectQueueManager(QM1)
     Bind(Client)
     ClientChannel(SOAPClient)
     ClientConnection(www.example.org(1418))
    &deliveryMode(NON_PERSISTENT)
```

圖 *29:* 使用 *jms: queue* 來傳送 *SOAP/JMS* 要求

支援的 Web 服務

撰寫成以 Web 服務方式執行的程式碼不需要修改，即可使用 SOAP 的 IBM WebSphere MQ 傳輸。 您確實需要以不同方式部署服務，以搭配 SOAP 的 IBM WebSphere MQ 傳輸來執行，而不是使用 HTTP。

說明

SOAP 的 WebSphere MQ 傳輸提供 SOAP 接聽器來執行 .NET Framework 1 和 .NET 2 以及 Axis 1.4 的服務。 Microsoft Windows Communication Foundation 的 WebSphere MQ 自訂通道會執行 .NET Framework 3 的服務。 WebSphere Application Server 和 CICS 支援透過 WebSphere MQ SOAP 傳輸來執行服務。 建立自訂「匯出」以使用 WebSphere Enterprise Service Bus 或 WebSphere Process Server。

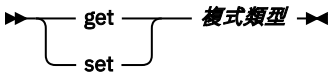
WebSphere MQ SOAP 接聽器可以交易式處理 SOAP 要求。 使用 -x 選項執行 **amqwdeployWMQService** 。 只有使用伺服器連結的接聽器才支援兩階段選項。 其他環境可能提供 WebSphere MQ SOAP 傳輸的交易式支援。 請參閱他們的文件。

WebSphere MQ SOAP 傳輸目前不支援已提交給 W3C 的新興業界標準 SOAP over JMS 通訊協定。 您可以尋找 JMS BindingVersion 內容，來區分寫入新標準的 SOAP/JMS 訊息。 WebSphere SOAP 的 MQ 傳輸未設定 BindingVersion 內容。

軸 1.4

Java 類別通常可以在不修改的情況下使用。 Axis 引擎必須支援 Web 服務中方法的任何引數類型。 如需進一步詳細資料，請參閱 Axis 文件。 如果服務使用複式物件作為引數，或傳回一個引數，則該物件必須符合 Java™ 規格。 請參閱 第 898 頁的圖 32、 第 899 頁的圖 33 及 第 899 頁的圖 34 中的範例:

1. 具有公用無參數建構子。
2. Bean 的任何複式類型必須具有下列格式的公用 getter 和 setter:



使用 **amqwdeployWMQService** 公用程式準備服務以進行部署。 該服務由 WebSphere MQ SOAP 接聽器呼叫，該接聽器使用 `axis.jar` 來執行該服務。

軸 1.4 唯一支援的兩段式交易管理程式是 WebSphere MQ。

所提供的部署公用程式不支援服務將不同套件中的物件傳回至服務本身的情況。 若要使用不同套件中傳回的物件，請撰寫您自己的部署公用程式。 您可以使用 -v 選項，讓部署公用程式以所提供的範例為基礎，或擷取它所產生的指令。 修正指令以產生自訂 Script。

如果服務使用 Axis 基礎架構和 WebSphere MQ SOAP 執行時期環境之外的類別，您必須設定正確的 CLASSPATH。 若要變更 CLASSPATH，請使用下列其中一種方式，修正啟動或定義接聽器的已產生 Script ，以包括所需的服務:

- 在呼叫 **amqwsetcp** 之後，直接在 Script 中修正 CLASSPATH 。
- 建立服務專用 Script 以自訂 CLASSPATH ，並在呼叫 **amqwsetcp** 之後在產生的 Script 中呼叫此 Script。
- 建立自訂的部署程序，以在產生的 Script 中自動自訂 CLASSPATH 。

.NET Framework 1 和 .NET Framework 2

已準備作為 HTTP Web 服務的服務不需要修改，即可用作 WebSphere MQ Web 服務。 需要使用 **amqwdeployWMQService** 公用程式來部署它。

.NET Framework 1 和 .NET 2 唯一支援的兩階段交易管理程式是 Microsoft Transaction Server (MTS)。

如果服務代碼尚未準備成 HTTP Web 服務，您必須將它轉換成 Web 服務。 將類別宣告為 Web 服務，並識別如何格式化每一個方法的參數。 您必須檢查服務方法的任何引數是否與環境相容。 第 898 頁的圖 30 及 第 898 頁的圖 31 顯示已準備成 Web 服務的 .NET 類別。 所做的新增會以粗體類型顯示。

第 898 頁的圖 30 使用 .NET Web 服務的幕後程式碼程式設計模型。 在程式碼背後模型中，服務的來源與 `.asmx` 檔案分開。 `.asmx` 檔案使用 Codear 後 關鍵字宣告相關聯來源檔的名稱。 WebSphere MQ 同時具有 .NET Web 服務的行內及程式碼背後的範例。

在 **amqwdeployWMQService** 部署公用程式部署之前，必須先編譯 .NET Web 服務來源。 服務會編譯成程式庫 (`.dll`)。 程式庫必須放置在部署目錄的 `./bin` 子目錄中。

.NET Framework 3

建立 Microsoft Windows Communication Foundation (WCF) 的 WebSphere MQ 自訂通道，以呼叫部署至 .NET Framework 3 的服務。 如需如何配置 WCF 以使用 WebSphere MQ SOAP 傳輸的說明，請參閱 Microsoft Windows Communication Foundation (WCF) 的 IBM WebSphere MQ 自訂通道 。

WebSphere Application Server

您可以使用 WebSphere MQ Transport for SOAP 來呼叫 WebSphere Application Server 所管理的 Web 服務; 請參閱 使用 SOAP over JMS 來傳輸 Web 服務 (已淘汰)。

您需要修改將 JMS 服務部署至 WebSphere Application Server 所產生的 WSDL，以便產生 Web 服務用戶端。 部署至 WebSphere Application Server 所建立的 WSDL 包含一個 URI，其中含有 JMS `InitialContextFactory` 的 JNDI 參照。 您需要修改 Nojndi 的 JNDI 參照，並依照 第 885 頁的『Web 服務部署的 URI 語法和參數』中的說明來提供連線屬性。

CICS

您可以使用 WebSphere MQ Transport for SOAP 來呼叫 CICS 應用程式; 請參閱 配置 Web 服務的 CICS 系統。

WebSphere Enterprise Service Bus 和 WebSphere Process Server for Multiplatforms

只有在使用預設 WebSphere Application Server 傳訊提供者時， WebSphere ESB 和 WebSphere Process Server for Multiplatforms 才支援 SOAP over JMS (具有備妥建置連結)。 建立 JMS 的自訂連結，以支援 SOAP 的 WebSphere MQ 傳輸。 請參閱 JMS 資料連結。

範例

```
<%@ WebService Language="C#" CodeBehind="Quote.asmx.cs" Class="Quote.QuoteDotNet" %>
```

圖 *30: .NET Framework 2* 的服務定義*: Quote.asmx*

```
<%@ WebService Language="C#" CodeBehind="Quote.asmx.cs" Class="Quote.QuoteDotNet" %>
using System;
using System.Web;
using System.Web.Services;
using System.Web.Services.Protocols;

namespace Quote {
    [WebService(Namespace = "http://www.example.org/")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    public class QuoteDotNet : System.Web.Services.WebService {
        [WebMethod]
        public string getQuote(String symbol){
            return symbol.ToUpper();
        }
    }
}
```

圖 *31: .NET Framework 2* 的服務實作*: Quote.asmx.cs*

```
package org.example.www;
public interface CustomerInfoInterface extends java.rmi.Remote {
    public org.example.www.CustomerRecord
           getCustomerName(org.example.www.CustomerRecord request)
           throws java.rmi.RemoteException, org.example.www.GetCustomerName_faultMsg;
}
```

圖 *32:* 使用複式類型的 *Java JAX-RPC* 服務介面

```
package org.example.www;
public class CustomerInfoPortImpl implements org.example.www.CustomerInfoInterface{
    public org.example.www.CustomerRecord
           getCustomerName(org.example.www.CustomerRecord request)
           throws java.rmi.RemoteException, org.example.www.GetCustomerName_faultMsg {
        request.setName(request.getID().toString());
        return request;
    }
}
```

圖 *33:* 使用複式類型的 *Java JAX-RPC* 服務實作

```
package org.example.www;
public class CustomerRecord  {
    private java.lang.String name;
    private java.lang.Integer ID;
    public CustomerRecord() {}
    public java.lang.String getName() {
        return name; }
    public void setName(java.lang.String name) {
        this.name = name; }
    public java.lang.Integer getID() {
        return ID; }
    public void setID(java.lang.Integer ID) {
        this.ID = ID; }
}
```

圖 *34:* 複式類型的 *Java JAX-RPC* 服務 *Bean* 實作

SOAP Web 服務用戶端的 IBM WebSphere MQ 傳輸

您可以在 SOAP 的 IBM WebSphere MQ 傳輸中重複使用現有的 SOAP over HTTP 用戶端。 您必須對程式碼和建置程序進行一些小修改，以將用戶端轉換成使用 IBM WebSphere MQ SOAP 傳輸。

編碼

JAX-RPC 用戶端必須以 Java 撰寫。 .NET Framework 1 和 2 用戶端可以使用任何使用 Common Language Runtime 的語言來撰寫。 程式碼範例在 C# 和 Visual Basic 中提供。

交易式支援的層次取決於用戶端環境及 SOAP 互動的型樣。 SOAP 要求和 SOAP 回覆不能是相同基本交易的一部分。

您必須在 .NET Framework 1、.NET Framework 2 用戶端中呼叫 `IBM.WMQSOAP.Register.Extension()`。 在 JAX-RPC Java Web 服務用戶端中，呼叫 `com.ibm.mq.soap.Register.extension` 來登錄 WebSphere MQ SOAP 傳送端。 此方法會使用 `jms:` 通訊協定，將 SOAP 傳送端的 WebSphere MQ 傳輸登錄為 SOAP 訊息的處理程式。

若要建立 .NET Framework 3 用戶端，請使用 **svcutil** 工具來產生 Windows Communication Foundation 用戶端 Proxy; 請參閱 使用 svcutil 工具及來自執行中服務的 meta 資料來產生 WCF 用戶端 Proxy 及應用程式配置檔。

建置及執行 .NET Framework 1 和 2 用戶端所需的程式庫

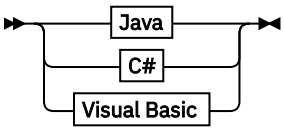
- amqsoap
- 系統
- System.Web.Services
- System.Xml

建置及執行 Axis 1.4 用戶端所需的程式庫

- *MQ_Install*\java\lib\com.ibm.mq.soap.jar;
- *MQ_Install*\java\lib\com.ibm.mq.commonservices.jar;
- *MQ_Install*\java\lib\soap\axis.jar;

- *MQ_Install*\java\lib\soap\jaxrpc.jar
- *MQ_Install*\java\lib\soap\saaj.jar;
- *MQ_Install*\java\lib\soap\commons-logging-1.0.4.jar;
- *MQ_Install*\java\lib\soap\commons-discovery-0.2.jar;
- *MQ_Install*\java\lib\soap\wsdl4j-1.5.1.jar;
- *MQ_Install*\java\jre\lib\xml.jar;
- *MQ_Install*\java\lib\soap\servlet.jar;
- *MQ_Install*\java\lib\com.ibm.mq.jar;
- *MQ_Install*\java\lib\com.ibm.mq.headers.jar;
- *MQ_Install*\java\lib\com.ibm.mq.pcf.jar;
- *MQ_Install*\java\lib\com.ibm.mq.jmqi.jar;
- *MQ_Install*\java\lib\com.ibm.mq.jmqi.remote.jar;
- *MQ_Install*\java\lib\com.ibm.mq.jmqi.local.jar;
- *MQ_Install*\java\lib\connector.jar;
- *MQ_Install*\java\lib\jta.jar;
- *MQ_Install*\java\lib\jndi.jar;
- *MQ_Install*\java\lib\ldap.jar

Register SOAP extension



Java

com.ibm.mq.soap.Register.extension()

C#

IBM.WMQSOAP.Register.Extension();

Visual Basic

IBM.WMQSOAP.Register.Extension

用戶端範例

第 901 頁的圖 35 是使用行內程式設計模型的 .NET Framework 1 或 .NET Framework 2 C# 用戶端範例。**IBM.WMQSOAP.Register.Extension()** 方法會以 .NET 作為 jms: 通訊協定處理程式來登錄 WebSphere MQ SOAP 傳送端。

```
using System;
namespace QuoteClientProgram {
    class QuoteMain {
        static void Main(string[] args) {
            try {
                IBM.WMQSOAP.Register.Extension();
                Quote q = new Quote();
                Console.WriteLine("Response is: " + q.getQuote("ibm"));
             }  catch (Exception e) {
                Console.WriteLine("Exception is: " + e);
             }
        }
    }
}
```

圖 *35: C# Web* 服務用戶端範例

第 901 頁的圖 36 是使用 JAX-RPC 靜態 Proxy 用戶端介面的 Java 用戶端範例。 **com.ibm.mq.soap.Register.extension();** 方法會向服務 Proxy 登錄 WebSphere MQ SOAP 傳送端，以處理 `jms:` 通訊協定。

```
package org.example.www;
import com.ibm.mq.soap.Register;
public class QuoteClient {
   public static void main(String[] args) {
      try {
         Register.extension();
         QuoteSOAPImplServiceLocator locator = new QuoteSOAPImplServiceLocator();
         System.out.println("Response = "
                  + locator.getOrgExampleWwwQuoteSOAPImpl_Wmq().getQuote("IBM"));
      } catch (Exception e) {
         System.out.println("Exception = " + e.getMessage());
      }
   }
}
```

圖 *36: Java Web* 服務用戶端範例

使用者結束程式、API 結束程式及可安裝的服務參照

使用本節中提供的鏈結，可協助您開發使用者結束程式、API 結束程式及可安裝的服務應用程式:

- 第 902 頁的『MQIEP 結構』
- 第 905 頁的『資料轉換結束程式參照』
- 第 908 頁的『MQ_PUBLISH_EXIT-發佈結束程式』
- 第 915 頁的『通道結束程式呼叫和資料結構』
- 第 972 頁的『API 結束程式參照』
- 第 1028 頁的『可安裝的服務介面參照資訊』

相關概念

第 7 頁的『MQI 應用程式參照』
使用本節中提供的鏈結，可協助您開發 MQI 應用程式:

第 1277 頁的『Java 程式庫的 IBM WebSphere MQ 類別』
Java 程式庫的 IBM WebSphere MQ 類別位置會因平台而異。 當您啟動應用程式時，請指定此位置。

相關工作

開發應用程式

相關參考

第 850 頁的『SOAP 參照』
WebSphere MQ SOAP 參照資訊的傳輸依字母順序排列。

第 1087 頁的『HTTP IBM WebSphere MQ 橋接器的參考資料』
HTTP IBM WebSphere MQ 橋接器的參考主題，按字母順序排列

第 1120 頁的『IBM WebSphere MQ .NET 類別和介面』
IBM WebSphere MQ .NET 類別和介面會按字母順序列出。 說明內容、方法和建構子。

第 1176 頁的『IBM WebSphere MQ C++ 類別』
IBM WebSphere MQ C++ 類別會封裝「 IBM WebSphere MQ 訊息佇列介面 (MQI)」。 有一個單一 C++ 標頭檔 **imqi.hpp**，它涵蓋所有這些類別。

WebSphere MQ classes for JMS

MQIEP 結構

MQIEP 結構包含允許結束程式進行的每一個函數呼叫的進入點。

欄位

StrucId
類型: MQCHAR4 -輸入

結構 ID。 值如下所示:

MQIEP_STRUC_ID

版本
類型 :MQLONG-輸入

結構版本號碼。 值如下所示:

MQIEP_VERSION_1
第 1 版結構版本號碼。

MQIEP_CURRENT_VERSION
結構的現行版本。

StrucLength
類型 :MQLONG

MQIEP 結構的大小 (以位元組為單位)。 值如下所示:

MQIEP_LENGTH_1

旗標
類型 :MQLONG

提供函數位址的相關資訊。 指出媒體庫是否具有執行緒作業的旗標可以與旗標一起使用，以指出媒體庫是否為用戶端或伺服器媒體庫。

下列值用來不指定任何檔案庫資訊:

MQIEPF_NONE

下列其中一個值是用來指定共用程式庫是含執行緒作業或不含執行緒作業:

MQIEPF_NON_THREADED_LIBRARY
非執行緒共用程式庫

MQIEPF_THREADED_LIBRARY
執行緒共用程式庫

下列其中一個值用來指定共用程式庫是用戶端或伺服器共用程式庫:

MQIEPF_CLIENT_LIBRARY
用戶端共用程式庫

MQIEPF_LOCAL_LIBRARY
伺服器共用程式庫

已保留
類型 :MQPTR

MQBACK_Call
類型 :PMQ_BACK_CALL

MQBACK 呼叫的位址。

MQBEGIN_Call
類型 :PMQ_BEGIN_CALL

MQBEGIN 呼叫的位址。

MQBUFMH_Call
類型 :PMQ_BUFMH_CALL

MQBUFMH 呼叫的位址。

MQCB_Call
類型 :PMQ_CB_CALL

MQCB 呼叫的位址。

MQCLOSE_Call
類型 :PMQ_CLOSE_CALL

MQCLOSE 呼叫的位址。

MQCMIT_Call
類型 :PMQ_CMIT_CALL

MQCMIT 呼叫的位址。

MQCONN_Call
類型 :PMQ_CONN_CALL

MQCONN 呼叫的位址。

MQCONNX_Call
類型 :PMQ_CONNX_CALL

MQCONNX 呼叫的位址。

MQCRTMH_Call
類型 :PMQ_CRTMH_CALL

MQCRTMH 呼叫的位址。

MQCTL_Call
類型 :PMQ_CTL_CALL

MQCTL 呼叫的位址。

MQDISC_Call
類型 :PMQ_DISC_CALL

MQDISC 呼叫的位址。

MQDLTMH_Call
類型 :PMQ_DLTMH_CALL

MQDLTMH 呼叫的位址。

MQDLTMP_Call
類型 :PMQ_DLTMP_CALL

MQDLTMP 呼叫的位址。

MQGET_Call
類型 :PMQ_GET_CALL

MQGET 呼叫的位址。

MQINQ_Call
類型 :PMQ_INQ_CALL

MQINQ 呼叫的位址。

MQINQMP_Call
類型 :PMQ_INQMP_CALL

MQINQMP 呼叫的位址。

MQMHBUF_Call
類型 :PMQ_MHBUF_CALL

MQMHBUF 呼叫的位址。

MQOPEN_Call
類型 :PMQ_OPEN_CALL

MQOPEN 呼叫的位址。

MQPUT_Call
類型 :PMQ_PUT_CALL

MQPUT 呼叫的位址。

MQPUT1_Call
類型: PMQ_PUT1_CALL

MQPUT1 呼叫的位址。

MQSET_Call
類型 :PMQ_SET_CALL

MQSET 呼叫的位址。

MQSETMP_Call
類型 :PMQ_SETMP_CALL

MQSETMP 呼叫的位址。

MQSTAT_Call
類型 :PMQ_STAT_CALL

MQSTAT 呼叫的位址。

MQSUB_Call
類型 :PMQ_SUB_CALL

MQSUB 呼叫的位址。

MQSUBRQ_Call
類型 :PMQ_SUBRQ_CALL

MQSUBRQ 呼叫的位址。

MQXCNVC_Call
類型 :PMQ_XCNVC_CALL

MQXCNVC 呼叫的位址。

MQXCLWLN_Call
類型 :PMQ_XCLWLN_CALL

MQXCLWLN 呼叫的位址。

MQXDX_Call
類型 :PMQ_XDX_CALL

MQXDX 呼叫的位址。

MQXEP_Call
類型 :PMQ_XEP_CALL

MQXEP 呼叫的位址。

MQZEP_Call
類型 :PMQ_ZEP_CALL

MQZEP 呼叫的位址。

C 宣告

```
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_SUBRQ_CALL   MQSUBRQ_Call;     /* Address of MQSUBRQ */
   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 */
 };
```

資料轉換結束程式參照

若為 z/OS，您必須以組譯語言撰寫資料轉換結束程式。 對於其他平台，建議您使用 C 程式設計語言。

為了協助您建立資料轉換跳出程式，提供下列項目:

- Skeleton 原始檔
- 轉換字元呼叫
- 建立程式碼片段的公用程式，可對資料類型結構執行資料轉換。此公用程式只接受 C 輸入。 在 z/OS 上，它會產生組譯器程式碼。

如需撰寫程式的程序，請參閱:

- 在 UNIX and Linux 系統上撰寫 WebSphere MQ 的資料轉換結束程式
- 撰寫 WebSphere MQ for Windows 的資料轉換結束程式

Skeleton 原始檔

在寫入資料轉換跳出程式時，可以使用這些作為起始點。

提供的檔案列在 第 906 頁的表 588 中。

表 *588: Skeleton* 原始檔	
平台	**檔案**
AIX	amqsvfc0.c
IBM i	QMQMSAMP/QCSRC (AMQSVFC4)
HP-UX	amqsvfc0.c
Linux	amqsvfc0.c
z/OS	CSQ4BAX8 (第 906 頁的『1』) CSQ4BAX9 (第 906 頁的『2』) CSQ4CAX9 (第 906 頁的『3』)
Solaris	amqsvfc0.c
Windows 系統	amqsvfc0.c
附註: 1. 說明 MQXCVNC 呼叫。 2. 公用程式所產生程式碼片段的封套，用於 CICS 以外的所有環境。 3. 公用程式所產生用於 CICS 環境之程式碼片段的封套。	

轉換字元呼叫

從資料轉換結束程式內使用 MQXCNVC (轉換字元) 呼叫，將字元訊息資料從一個字集轉換成另一個字集。 對於某些多位元組字集 (例如， UCS2 字集) ，必須使用適當的選項。

無法從結束程式內進行其他 MQI 呼叫; 嘗試進行此類呼叫會失敗，原因碼為 MQRC_CALL_IN_PROGRESS。

如需 MQXCNVC 呼叫及適當選項的進一步資訊，請參閱 第 802 頁的『MQXCNVC-轉換字元』 。

用於建立轉換-結束碼的公用程式

使用此資訊來進一步瞭解建立轉換-結束碼。

用於建立 conversion-exit 程式碼的指令如下:

IBM i
CVTMQMDTA (轉換 WebSphere MQ 資料類型)

Windows、 UNIX and Linux 系統
`crtmqcvx` (建立 WebSphere MQ 轉換-結束程式)

適用於您平台的指令會產生程式碼片段，可對資料類型結構執行資料轉換，以在資料轉換結束程式中使用。 這個指令會取得包含一或多個 C 語言結構定義的檔案。 。

Windows、 UNIX and Linux 系統中的錯誤訊息

`crtmqcvx` 指令會傳回 AMQ7953 至 AMQ7970 範圍內的訊息。

這些訊息列在 原因碼 *WebSphere MQ* 訊息中。

錯誤有兩種主要類型:

- 處理程序無法繼續時發生主要錯誤，例如語法錯誤。

畫面上會顯示一則訊息，指出輸入檔中錯誤的行號。 輸出檔可能已局部建立。

- 當顯示訊息指出發現問題，但結構的剖析可以繼續時，會發生其他錯誤。

  已建立輸出檔，並包含所發生問題的相關錯誤資訊。 此錯誤資訊會以 `#error` 作為字首，因此任何編譯器都不會接受所產生的程式碼，而不需要人為介入來更正問題。

有效語法

公用程式的輸入檔必須符合 C 語言語法。

如果您不熟悉 C ，請參閱這個主題中的 C 範例 。

此外，請注意下列規則:

- typedef 只能在 struct 關鍵字之前辨識。
- 您的結構宣告需要結構標籤。
- 您可以使用空的方括弧 [] 來表示訊息結尾的可變長度陣列或字串。
- 不支援多維度陣列和字串陣列。
- 可辨識下列其他資料類型:
  - MQBOOL
  - MQBYTE
  - MQCHAR
  - MQFLOAT32
  - MQFLOAT64
  - MQSHORT
  - MQLONG
  - MQINT8
  - MQUINT8
  - MQINT16
  - MQUINT16
  - MQINT32
  - MQUINT32
  - MQINT64
  - MQUINT64

  MQCHAR 欄位已轉換字碼頁，但 MQBYTE、 MQINT8 及 MQUINT8 保持不變。 如果編碼不同，則會相應地轉換 MQSHORT、MQLONG、 MQINT16、 MQUINT16、 MQINT32、 MQUINT32、 MQINT64、 MQUINT64、 MQFLOAT32、 MQFLOAT64 及 MQBOOL。
- 請 不要 使用下列類型的資料:
  - 倍精準數
  - 指標
  - 位元欄位

  這是因為用來建立轉換結束碼的公用程式未提供轉換這些資料類型的機能。 若要克服此問題，您可以撰寫自己的常式，並從結束程式呼叫它們。

要注意的其他要點:

- 請勿在輸入資料集中使用序號。
- 如果有欄位要提供您自己的轉換常式，請將它們宣告為 MQBYTE ，然後將產生的 CMQXCFBA 巨集取代為您自己的轉換碼。

C 範例

```
  struct TEST { MQLONG    SERIAL_NUMBER;
                MQCHAR    ID[5];
                MQINT16   VERSION;
                MQBYTE    CODE[4];
                MQLONG    DIMENSIONS[3];
                MQCHAR    NAME[24];
              } ;
```

這對應於其他程式設計語言中的下列宣告:

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

僅在 z/OS 上受支援

```
 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-發佈結束程式

MQ_PUBLISH_EXIT 呼叫可以檢查並變更遞送至訂閱者的訊息。

用途

使用發佈結束程式來檢查及變更遞送至訂閱者的訊息:

- 檢查發佈至每一個訂閱者之訊息的內容
- 修改發佈至每一個訂閱者的訊息內容
- 變更放置訊息的佇列
- 停止將訊息遞送至訂閱者

語法

MQ_PUBLISH_EXIT(*ExitParms*, *PubContext*, *SubContext*)

參數

***ExitParms* (MQPSXP)- Input/Output**
ExitParms 包含呼叫結束程式的相關資訊。

***PubContext* (MQPBC) - Input**
PubContext 包含發佈資訊發佈者的相關環境定義資訊。

***SubContext* (MQSBC) - Input/Output**
SubContext 包含接收發佈資訊之訂閱者的相關環境定義資訊。

MQPSXP-發佈結束程式資料結構

MQPSXP 結構說明傳遞至發佈結束程式並從發佈結束程式傳回的資訊。

第 909 頁的表 589 彙總結構中的欄位:

表 *589: MQPSXP* 中的欄位

欄位	說明
StrucID	結構 ID
Version	結構版本號碼
ExitId	正在呼叫的結束程式類型
ExitReason	呼叫結束程式的原因
ExitResponse	來自結束程式的回應
ExitResponse2	來自結束程式的次要回應
Feedback	回饋碼
ExitUserArea	結束使用者區域
ExitData	結束程式資料
QMgrName	本端佇列管理程式的名稱
Hconn	連線控點
MsgDescPtr	訊息描述子 (MQMD) 的位址
MsgHandle	處理訊息內容 (MQHMSG)
MsgInPtr	輸入訊息的位址
MsgInLength	輸入訊息的長度
MsgOutPtr	輸出訊息的位址
MsgOutLength	輸出訊息的長度
pEntryPoints	MQIEP 結構的位址

欄位

***StrucID* (MQCHAR4)**
StrucID 是結構 ID。 值如下所示:

MQPSXP_STRUCID
MQPSXP_STRUCID 是發佈結束程式參數結構的 ID。 對於 C 程式設計語言，也會定義常數 MQPSXP_STRUC_ID_ARRAY；其值與 MQPSXP_STRUC_ID 相同，但卻是字元陣列而非字串。

StrucID 是結束程式的輸入欄位。

***Version* (MQLONG)**
Version 是結構版本號碼。 值如下所示:

MQPSXP_VERSION_1
MQPSXP_VERSION_1 是第 1 版發佈結束程式參數結構。 常數 MQPSXP_CURRENT_VERSION 也定義了相同的值。

Version 是結束程式的輸入欄位。

***ExitId* (MQLONG)**
ExitId 是要呼叫的結束程式類型。 值如下所示:

MQXT_PUBLISH_EXIT
發佈結束程式。

ExitId 是結束程式的輸入欄位。

***ExitReason* (MQLONG)**
ExitReason 是呼叫結束程式的原因。 可能值包括:

MQXR_INIT
會呼叫此連線的結束程式以進行起始設定。 結束程式可能會獲得並起始設定它需要的資源; 例如，主儲存體。

MQXR_TERM
因為即將停止結束程式，所以會呼叫此連線的結束程式。 結束程式必須釋放自起始設定以來所獲得的任何資源; 例如，主儲存體。

MQXR_PUBLICATION
結束程式會先由佇列管理程式呼叫，再將發佈資訊放入訂閱者的訊息佇列中。 跳出程式可以變更訊息、不將訊息放置在佇列上，或中止發佈。

ExitReason 是結束程式的輸入欄位。

***ExitResponse* (MQLONG)**
在結束程式中設定 *ExitResponse* ，以指定必須如何繼續處理。 *ExitResponse* 是下列其中一個值:

MQXCC_OK
設定 MQXCC_OK 以正常繼續處理。 設定 MQXCC_OK 以回應 ExitReason 的任何值。

如果 ExitReason 具有值 MQXR_PUBLICATION ，則 MQSBC 結構的 *DestinationQName* 及 *DestinationQMgrName* 欄位會識別訊息傳送至的目的地。

MQXCC_FAILED
設定 MQXCC_FAILED 以停止發佈作業。 完成碼 MQCC_FAILED 及原因碼 2557 (09FD) (RC2557): MQRC_PUBLISH_EXIT_ERROR 是在結束程式傳回時設定。

MQXCC_SUPPRESS_FUNCTION
設定 MQXCC_SUPPRESS_FUNCTION 以停止正常處理訊息。 僅當 *ExitReason* 具有值 MQXR_PUBLICATION 時，才設定 MQXCC_SUPPRESS_FUNCTION 。

佇列管理程式會根據訊息的訊息描述子中 *Report* 欄位的 MQRO_DISCARD_MSG 選項，繼續處理訊息。

- 如果指定 MQRO_DISCARD_MSG 選項，則不會將訊息遞送至訂閱者。
- 如果未指定 MQRO_DISCARD_MSG 選項，則會將訊息放置在無法傳送郵件的佇列上。 如果沒有無法傳送郵件的佇列，或訊息無法順利放置在無法傳送郵件的佇列上，則不會將發佈遞送至訂閱者。發佈至其他訂閱者的遞送取決於 PMSGDLV 和 NPMSGDLV 主題物件屬性的值。 如需這些屬性的說明，請參閱 DEFINE TOPIC 指令的參數說明。

ExitResponse 是結束程式的輸出欄位。

***ExitResponse2* (MQLONG)**
ExitResponse2 保留供未來使用。

***Feedback* (MQLONG)**
Feedback 是當結束程式在 *ExitResponse* 中傳回 MQXCC_SUPPRESS_FUNCTION 時要使用的回饋碼。

在輸入結束程式時， *Feedback* 一律具有值 MQFB_NONE。 如果結束程式傳回 MQXCC_SUPPRESS_FUNCTION，請將 *Feedback* 設為當佇列管理程式將訊息置於無法傳送郵件的佇列時要用於訊息的值。 從結束程式返回時，如果 *Feedback* 具有原始值 MQFB_NONE，則佇列管理程式會將 *Feedback* 設為 MQFB_STOPPED_BY_PUBSUB_EXIT。

Feedback 是結束程式的輸入/輸出欄位。

***ExitUserArea* (MQBYTE16)**
ExitUserArea 是可供結束程式使用的欄位。 每一個連線都有個別的 *ExitUserArea*。 *ExitUserArea* 的長度由 MQ_EXIT_USER_AREA_LENGTH 提供。

ExitReason 欄位在第一次呼叫結束程式時具有值 MQXR_INIT 。 在第一次呼叫連線的結束程式時，*ExitUserArea* 會起始設定為 MQXUA_NONE 。 在呼叫結束程式時，會保留對 *ExitUserArea* 所做的後續變更。

ExitUserArea 是結束程式的輸入/輸出欄位。

***ExitData* (MQCHAR32)**
ExitData 是由佇列管理程式起始設定檔中段落的 *PublishExitData* 參數所定義的固定結束程式資料。 資料會以空白填補欄位的完整長度。 如果起始設定檔中未定義任何固定結束程式資料，則 *ExitData* 為空白。 *ExitData* 的長度由 MQ_EXIT_DATA_LENGTH 提供。

ExitData 是結束程式的輸入欄位。

***QMgrName* (MQCHAR48)**
QMgrName 是本端佇列管理程式的名稱。 名稱會以空白填補欄位的完整長度。 此欄位的長度由 MQ_Q_MGR_NAME_LENGTH 提供。

QMgrName 是結束程式的輸入欄位。

***Hconn* (MQHCONN)**
Hconn 是代表佇列管理程式連線的控點。 僅使用 *Hconn* 作為 MQSETMP、 MQINQMMP 或 MQDLTMP 訊息內容函數呼叫的參數，以使用訊息內容。

Hconn 是結束程式的輸入欄位。

***MsgDescPtr* (PMQMD)**
MsgDescPtr 是所處理訊息的訊息描述子 (MQMD) 位址，並且是 MQPUT 呼叫所傳回 MQMD 的副本。 結束程式可以變更訊息描述子的內容。 對訊息描述子內容的任何變更都必須小心完成。 尤其是在 MQSBC 結構的 *SubType* 欄位值為 MQSUBTYPE_PROXY 的情況下，訊息描述子中的 *CorrelId* 欄位不得變更。

如果 *ExitReason* 是 MQXR_INIT 或 MQXR_TERM ，則不會將任何訊息描述子傳遞至結束程式; 在這些情況下， *MsgDescPtr* 是空值指標。

MsgDescPtr 是結束程式的輸入欄位。

***MsgHandle* (MQHMSG)**
MsgHandle 是訊息內容的控點。 僅搭配使用 *MsgHandle* 與 MQSETMP、 MQINQMMP 或 MQDLTMP 訊息內容函數呼叫，以使用訊息內容。

MsgHandle 是結束程式的輸入欄位。

***MsgInPtr* (PMQVOID)**
MsgInPtr 是輸入訊息資料的位址。 結束程式可以修改 *MsgInPtr* 所處理的緩衝區內容; 請參閱 *MsgOutPtr* 。

MsgInPtr 是結束程式的輸入欄位。

***MsgInLength* (MQLONG)**
MsgInLength 是傳遞至結束程式的訊息資料長度 (以位元組為單位)。 資料的位址由 *MsgInPtr* 提供。

MsgInLength 是結束程式的輸入欄位。

***MsgOutPtr* (PMQVOID)**
MsgOutPtr 是包含從結束程式傳回之訊息資料的緩衝區位址。 進入結束程式時， *MsgOutPtr* 是空值。 從結束返回時，如果值仍然是空值，則佇列管理程式會傳送 *MsgInPtr* 指定的訊息，長度為 *MsgInLength* 。

如果結束程式修改訊息資料，請使用下列其中一個程序:

- 如果資料的長度未變更，則可以在 *MsgInPtr* 所定址的緩衝區中修改資料。 在此情況下，請勿變更 *MsgOutPtr* 和 *MsgOutLength*。
- 如果已修改的資料短於原始資料，則可以在 *MsgInPtr* 所定址的緩衝區中修改資料。 在此情況下，*MsgOutPtr* 必須設為輸入訊息緩衝區的位址，而 *MsgOutLength* 必須設為訊息資料的新長度。
- 如果已修改的資料比原始資料長或可能長，則結束程式必須取得新的訊息緩衝區。 將已修改的資料複製到其中。 將 *MsgOutPtr* 設為新緩衝區的位址，並將 *MsgOutLength* 設為新訊息資料的長度。 當下次呼叫結束程式時，結束程式會負責釋放 *MsgOutPtr* 所處理的緩衝區。

註: *MsgOutPtr* 一律是結束程式輸入上的空值指標，而不是先前所取得訊息緩衝區的位址。 若要釋放先前取得的緩衝區，結束程式必須儲存其位址及長度。 將資訊儲存在 *ExitUserArea* 中，或儲存在其位址儲存在 *ExitUserArea* 中的控制區塊中。

MsgOutPtr 是結束程式的輸入/輸出欄位。

***MsgOutLength* (MQLONG)**
MsgOutLength 是結束程式所傳回訊息資料的長度 (以位元組為單位)。 在結束程式輸入時，此欄位一律為零。 從結束程式返回時，如果 *MsgOutPtr* 是空值，則會忽略此欄位。 如需修改訊息資料的相關資訊，請參閱 *MsgOutPtr* 。

MsgOutLength 是結束程式的輸入/輸出欄位。

***pEntryPoints* (PMQIEP)**
pEntryPoints 是 MQIEP 結構的位址，可透過該結構進行 MQI 及 DCI 呼叫。

C 語言宣告-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-發佈環境定義資料結構

MQPBC 結構包含與發佈的發佈者相關且傳遞至發佈結束程式的環境定義資訊。

第 912 頁的表 590 彙總結構中的欄位:

表 *590: MQPBC* 中的欄位

欄位	說明
StrucID	結構 ID
Version	結構版本號碼
PubTopicString	發佈主題字串
MsgDescPtr	訊息描述子 (MQMD) 的位址

欄位

***StrucID* (MQCHAR4)**
StrucID 是結構 ID。 值如下所示:

MQPBC_STRUCID
MQPBC_STRUCID 是發佈環境定義結構的 ID。 對於 C 程式設計語言，也會定義常數 MQPBC_STRUC_ID_ARRAY；其值與 MQPBC_STRUC_ID 相同，但卻是字元陣列而非字串。

StrucID 是結束程式的輸入欄位。

***Version* (MQLONG)**
Version 是結構版本號碼。 值如下所示:

MQPBC_VERSION_1
MQPBC_VERSION_1 是第 1 版發佈結束程式參數結構。

MQPBC_VERSION_2
MQPBC_VERSION_2 是第 2 版發佈結束程式參數結構。 常數 MQPBC_CURRENT_VERSION 也定義了相同的值。

Version 是結束程式的輸入欄位。

***PubTopicString* (MQCHARV)**
PubTopicString 是要發佈至的主題字串。

PubTopicString 是結束程式的輸入欄位。

***MsgDescPtr* (PMQMD)**
MsgDescPtr 是所處理訊息的訊息描述子 (MQMD) 副本位址。

MsgDescPtr 是結束程式的輸入欄位。

C 語言宣告-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-訂閱環境定義資料結構

MQSBC 結構包含環境定義資訊，與傳遞至發佈結束程式的接收發佈的訂閱者相關。

第 913 頁的表 591 彙總結構中的欄位:

表 *591:* 欄位位於*(F) MQSBC*

欄位	說明
StrucID	結構 ID
Version	結構版本號碼
DestinationQMgrName	目的地佇列管理程式的名稱
DestinationQName	目的地佇列的名稱
SubType	訂閱的類型
SubOptions	訂閱選項
ObjectName	物件名稱
ObjectString	物件字串
SubTopicString	訂閱主題字串

表 591: 欄位位於(F) MQSBC (繼續)	
欄位	**說明**
SubName	訂閱名稱
SubId	訂閱 ID
SelectionString	選取字串的位址
SubLevel	訂閱層次
PSProperties	發佈/訂閱內容

欄位

***StrucID* (MQCHAR4)**
結構 ID。 值如下所示:

MQSBC_STRUCID
MQSBC_STRUCID 是發佈結束程式參數結構的 ID。 對於 C 程式設計語言，也會定義常數 MQSBC_STRUC_ID_ARRAY ; MQSBC_STRUC_ID_ARRAY 的值與 MQSBC_STRUC_ID 相同，但卻是字元陣列而非字串。

StrucID 是結束程式的輸入欄位。

***Version* (MQLONG)**
結構版本號碼。 值如下所示:

MQSBC_VERSION_1
第 1 版發佈結束程式參數結構。 常數 MQSBC_CURRENT_VERSION 也定義了相同的值。

Version 是結束程式的輸入欄位。

***DestinationQMgrName* (MQCHAR48)**
DestinationQMgrName 是訊息傳送至其中的佇列管理程式名稱。 名稱會以空白填補欄位的完整長度。 此名稱可以由結束程式變更。 此欄位的長度由 MQ_Q_MGR_NAME_LENGTH 提供。

DestinationQMgrName 是結束程式的輸入/輸出欄位; 請參閱 附註。

***DestinationQName* (MQCHAR48)**
DestinationQName 是訊息傳送至其中的佇列名稱。 名稱會以空白填補欄位的完整長度。 此名稱可以由結束程式變更。 此欄位的長度由 MQ_Q_NAME_LENGTH 提供。

DestinationQName 是結束程式的輸入/輸出欄位; 請參閱 附註。

***SubType* (MQLONG)**
SubType 指出如何建立訂閱。 有效值為 MQSUBTYPE_API、 MQSUBTYPE_ADMIN 及 MQSUBTYPE_PROXY; 請參閱 查詢訂閱狀態 (回應)。

SubType 是結束程式的輸入欄位。

***SubOptions* (MQLONG)**
SubOptions 是訂閱選項; 如需此欄位可以採用的值說明，請參閱 第 493 頁的『選項 (MQLONG)』 。

SubOptions 是結束程式的輸入欄位。

***ObjectName* (MQCHAR48)**
ObjectName 是本端佇列管理程式上定義的主題物件名稱。 此欄位的長度由 MQ_TOPIC_NAME_LENGTH 提供。 物件名稱是佇列管理程式與主題字串相關聯的管理主題物件名稱。 即使訂閱者提供了主題物件作為訂閱的一部分， *ObjectName* 也可能是不同的主題物件。 主題物件與訂閱的關聯取決於 *SubTopicString* 的完整解決方案。

ObjectName 是結束程式的輸入欄位。

***ObjectString* (MQCHARV)**
ObjectString 是已訂閱之發佈的完整主題字串。 會解析原始訂閱字串中的任何萬用字元。 它不同於 第 493 頁的『ObjectString (MQCHARV)』中說明的 MQSD 訂閱 *ObjectString* 欄位，該欄位可能包含萬用字元，且不包含訂閱者提供的任何物件名稱。

ObjectString 是結束程式的輸入欄位。

***SubTopicString* (MQCHARV)**
SubTopicString 是訂閱者提供的完整主題字串。 *SubTopicString* 是主題物件中定義的主題字串與主題字串的組合。 訂閱者必須提供主題物件、主題字串或兩者。 如果訂閱者提供主題字串，則可能包含萬用字元。

SubTopicString 是結束程式的輸入欄位。

***SubName* (MQCHARV)**
SubName 是訂閱者提供的訂閱名稱，或是產生的名稱。

SubName 是結束程式的輸入欄位。

***SubId* (MQBYTE 24)**
SubId 是唯一內部訂閱 ID。

SubId 是結束程式的輸入欄位。

***SelectionString* (MQCHARV)**
SelectionString 是從主題訂閱訊息時使用的選取準則; 請參閱選取器 。

SelectionString 是結束程式的輸入欄位。

***SubLevel* (MQLONG)**
SubLevel 是與訂閱相關聯的截取層次; 如需進一步詳細資料，請參閱 第 502 頁的『SubLevel (MQLONG)』 。

SubLevel 是結束程式的輸入欄位。

***PSProperties* (MQLONG)**
PSProperties 是發佈/訂閱內容。 它們指定如何將發佈/訂閱相關訊息內容新增至傳送至此訂閱的訊息。 可能的值為 MQPSPROP_NONE、 MQPSPROP_COMPAT、 MQPSPROP_RFH2、 MQPSPROP_MSGPROP。 如需這些值的說明，請參閱 選用參數 (變更、複製及建立訂閱) 。

PSProperties 是結束程式的輸入欄位。

註: 在將授權檢查傳遞至發佈結束程式之前，只會對 *DestinationQMgrName* 及 *DestinationQName* 的原始值執行授權檢查。 當結束程式透過變更 *DestinationQMgrName* 或 *DestinationQName* 來變更目的地佇列時，不會執行新的授權檢查。

C 語言宣告-MQSBC

```
typedef struct tagMQSBC {
  MQCHAR4    StrucId;             /* 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     PSProperties;        /* Publish/subscribe properties */
} MQSBC;
```

通道結束程式呼叫和資料結構

此主題集合提供特殊 WebSphere MQ 呼叫及資料結構的相關參考資訊，當您撰寫通道結束程式時可以使用這些資訊。

此資訊是產品相關程式設計介面資訊。 您可以使用下列程式設計語言來撰寫 WebSphere MQ 使用者結束程式:

平台	程式設計語言
WebSphere MQ for z/OS	組譯器和 C (必須符合系統結束程式的 C 系統程式設計環境，如 *z/OS C/C++* 程式設計手冊中所述。)
WebSphere MQ for IBM i	ILE C、ILE COBOL 及 ILE RPG
所有其他 WebSphere MQ 平台	C

您也可以在 Java 中撰寫使用者結束程式，以便只與 Java 和 JMS 應用程式搭配使用。 如需使用適用於 Java 的 WebSphere MQ 類別來建立及使用通道結束程式的相關資訊，請參閱 在適用於 Java 的 WebSphere MQ 類別中使用通道結束程式 ，以及在適用於 JMS 的 WebSphere MQ 類別中使用通道結束程式，請參閱 在適用於 JMS 的 WebSphere MQ 類別中使用通道結束程式。

您無法在 TAL 或 Visual Basic 中寫入 WebSphere MQ 使用者結束程式。 不過， Visual Basic 中提供 MQCD 結構的宣告，以用於來自 WebSphere MQ MQI 用戶端程式的 MQCONNX 呼叫。

在下列說明中的一些情況下，參數是陣列或具有未修正大小的字串。 對於這些參數，會使用小寫 "n" 來代表數值常數。 當該參數的宣告已編碼時， "n" 必須取代為所需的數值。 如需這些說明中所使用慣例的進一步相關資訊，請參閱 第 214 頁的『基本資料類型』。

資料定義檔

WebSphere MQ 會針對每一種支援的程式設計語言提供資料定義檔。 如需這些檔案的詳細資料，請參閱 複製、標頭、併入及模組檔案。

MQ_CHANNEL_EXIT-通道結束程式

MQ_CHANNEL_EXIT 呼叫說明傳遞至「訊息通道代理程式」所呼叫的每一個通道結束程式的參數。

佇列管理程式未提供稱為 MQ_CHANNEL_EXIT 的進入點; 名稱 MQ_CHANNEL_EXIT 沒有特殊意義，因為通道結束程式的名稱在通道定義 MQCD 中提供。

通道結束程式有五種類型:

- 通道安全結束程式
- 通道訊息結束程式
- 通道傳送結束程式
- 通道接收結束程式
- 通道訊息-重試結束程式

每一種類型的結束程式都有類似的參數，這裡提供的說明適用於所有這些類型，但特別註明的除外。

語法

MQ_CHANNEL_EXIT (***ChannelExitParms, ChannelDefinition, DataLength,***
AgentBufferLength, AgentBuffer, ExitBufferLength, ExitBufferAddr)

參數

MQ_CHANNEL_EXIT 呼叫具有下列參數。

ChannelExitParms (MQCXP)-輸入/輸出

通道結束程式參數區塊。

此結構包含與呼叫結束程式相關的其他資訊。結束程式會設定此結構中的資訊，以指出 MCA 如何繼續進行。

ChannelDefinition (MQCD)-輸入/輸出

通道定義。

此結構包含管理者設定用來控制通道行為的參數。

DataLength (MQLONG)-輸入/輸出

資料的長度。

資料取決於結束程式的類型:

- 對於通道安全結束程式，當呼叫該結束程式時，如果 *ExitReason* 是 MQXR_SEC_MSG，則此參數在 *AgentBuffer* 欄位中包含任何安全訊息的長度。 如果沒有訊息，則為零。 如果將 *ExitResponse* 設為 MQXCC_SEND_SEC_MSG 或 MQXCC_SEND_AND_REQUEST_SEC_MSG，則結束程式必須將此欄位設為要傳送至其友機的任何安全訊息長度。 訊息資料位於 *AgentBuffer* 或 *ExitBufferAddr* 中。

  安全訊息的內容是安全結束程式的唯一責任。
- 對於通道訊息結束程式，當呼叫結束程式時，此參數包含訊息的長度 (包括傳輸佇列標頭)。 結束程式必須將此欄位設為 *AgentBuffer* 或 *ExitBufferAddr* 中要繼續進行的訊息長度。 這必須大於或等於傳輸佇列標頭 (MQXQH) 的長度。
- 對於通道傳送或通道接收結束程式，當呼叫結束程式時，此參數包含傳輸長度。 結束程式必須將此欄位設為 *AgentBuffer* 或 *ExitBufferAddr* 中要繼續進行的傳輸長度。

如果安全結束程式傳送訊息，且通道另一端沒有安全結束程式，或另一端設定 *ExitResponse* 為 MQXCC_OK，則會使用 MQXR_SEC_MSG 及空值回應 (*DataLength*= 0) 重新呼叫起始結束程式。

AgentBuffer 長度 (MQLONG)-輸入

代理程式緩衝區的長度。

呼叫時此參數可以大於 *DataLength*。

對於通道訊息、傳送及接收結束程式，結束程式可以使用呼叫的任何未用空間來就地展開資料。 如果這樣做，結束程式必須適當地設定 *DataLength* 參數。

在 C 程式設計語言中，此參數依位址傳遞。

AgentBuffer (MQBYTE ×AgentBuffer 長度)-輸入/輸出

代理程式緩衝區。

此參數的內容視結束程式類型而定:

- 對於通道安全結束程式，如果 *ExitReason* 是 MQXR_SEC_MSG，則在呼叫結束程式時它會包含安全訊息。 若要傳回安全訊息，結束程式可以使用此緩衝區或它自己的緩衝區 (*ExitBufferAddr*)。
- 對於通道訊息結束程式，在呼叫結束程式時，此參數包含:
  - 傳輸佇列標頭 (MQXQH)，包含訊息描述子 (本身包含訊息的環境定義資訊)，後面緊接著
  - 訊息資料

  如果訊息要繼續，結束程式可以執行下列其中一項:
  - 保持不接觸緩衝區的內容
  - 就地修改內容 (在 *DataLength* 中傳回資料的新長度; 這不得大於 *AgentBufferLength*)
  - 將內容複製到 *ExitBufferAddr*，並進行任何必要的變更

  不會檢查結束程式對傳輸佇列標頭所做的任何變更; 不過，錯誤修改可能表示訊息無法放置在目的地。
- 若為通道傳送或接收結束程式，在呼叫結束程式時，這會包含傳輸資料。 結束程式可以執行下列其中一項:
  - 保持不接觸緩衝區的內容
  - 就地修改內容 (在 *DataLength* 中傳回資料的新長度; 這不得大於 *AgentBufferLength*)
  - 將內容複製到 *ExitBufferAddr*，並進行任何必要的變更

結束程式不得變更資料的前 8 個位元組。

ExitBuffer 長度 (MQLONG)-輸入/輸出

結束緩衝區的長度。

在第一次呼叫結束程式時，此參數會設為零。 之後，每次呼叫時，結束程式所傳回的任何值都會在下次呼叫時呈現給結束程式。 MCA 未使用此值。

註: 以不支援指標資料類型的程式設計語言撰寫的結束程式不得使用此參數。

ExitBufferAddr (MQPTR)-輸入/輸出

結束緩衝區的位址。

此參數是指向結束程式所管理儲存體之緩衝區位址的指標，如果代理程式的緩衝區夠大或可能不夠大，或結束程式更方便執行，則它可以選擇將訊息或傳輸資料 (視結束程式類型而定) 傳回給代理程式。

在第一次呼叫結束程式時，傳給結束程式的位址是空值。 之後，每次呼叫時，結束程式所傳回的任何位址都會在下一次呼叫時呈現給結束程式。

註: 以不支援指標資料類型的程式設計語言撰寫的結束程式不得使用此參數。

C 呼叫

```
exitname (&ChannelExitParms, &ChannelDefinition,
         &DataLength, &AgentBufferLength, AgentBuffer,
         &ExitBufferLength, &ExitBufferAddr);
```

傳遞至結束程式的參數宣告如下:

```
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 呼叫

```
     CALL 'exitname' USING CHANNELEXITPARMS, CHANNELDEFINITION,
                           DATALENGTH, AGENTBUFFERLENGTH, AGENTBUFFER,
                           EXITBUFFERLENGTH, EXITBUFFERADDR.
```

傳遞至結束程式的參數宣告如下:

```
**   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 呼叫 (ILE)

```
C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
C                     CALLP     exitname(MQCXP : MQCD : DATLEN :
C                                        ABUFL : ABUF : EBUFL :
C                                        EBUF)
```

呼叫的原型定義為:

```
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 組譯器呼叫

```
        CALL EXITNAME,(CHANNELEXITPARMS,CHANNELDEFINITION,DATALENGTH, X
              AGENTBUFFERLENGTH,AGENTBUFFER,EXITBUFFERLENGTH,         X
              EXITBUFFERADDR)
```

傳遞至結束程式的參數宣告如下:

```
CHANNELEXITPARMS   CMQCXPA  ,      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
```

使用注意事項

1. 通道結束程式所執行的功能由結束程式的提供者定義。 不過，結束程式必須符合這裡及相關聯控制區塊 (MQCXP) 中所定義的規則。
2. 傳遞至通道結束程式的 *ChannelDefinition* 參數可能是數個版本之一。 如需相關資訊，請參閱 MQCD 結構中的 *Version* 欄位。
3. 如果通道結束程式收到 MQCD 結構，且 *Version* 欄位設為大於 MQCD_VERSION_1 的值，則結束程式必須使用 MQCD 中的 *ConnectionName* 欄位，而不是 *ShortConnectionName* 欄位。
4. 一般而言，容許通道結束程式變更訊息資料的長度。 這可能是因為結束程式將資料新增至訊息，或從訊息中移除資料，或壓縮或加密訊息。 不過，如果訊息是只包含部分邏輯訊息的區段，則會有特殊限制。 特別是，由於傳送及接收結束程式的補充動作，訊息長度不得有任何淨變更。

   例如，容許傳送結束程式壓縮訊息來縮短訊息，但補充接收結束程式必須透過解壓縮訊息來還原訊息的原始長度，以便訊息長度沒有淨變更。

   產生此限制是因為變更區段的長度會導致訊息中後續區段的偏移不正確，而這會禁止佇列管理程式辨識區段形成完整邏輯訊息的能力。

MQ_CHANNEL_AUTO_DEF_EXIT-通道自動定義結束程式

MQ_CHANNEL_AUTO_DEF_EXIT 呼叫說明傳遞至「訊息通道代理程式」所呼叫通道自動定義結束程式的參數。

佇列管理程式未提供稱為 MQ_CHANNEL_AUTO_DEF_EXIT 的進入點; 名稱 MQ_CHANNEL_AUTO_DEF_EXIT 沒有特殊意義，因為佇列管理程式中提供自動定義結束程式的名稱。

語法

MQ _CHANNEL_AUTO_DEF_EXIT *(ChannelExitParms, ChannelDefinition)*

參數

MQ_CHANNEL_AUTO_DEF_EXIT 呼叫具有下列參數。

ChannelExitParms (MQCXP)-輸入/輸出

通道結束程式參數區塊。

此結構包含與呼叫結束程式相關的其他資訊。結束程式會設定此結構中的資訊，以指出 MCA 如何繼續進行。

ChannelDefinition (MQCD)-輸入/輸出

通道定義。

此結構包含管理者設定的參數，用來控制自動建立之通道的行為。結束程式會設定此結構中的資訊，以修改管理者所設定的預設行為。

結束程式不得變更列出的 MQCD 欄位:

- *ChannelName*
- *ChannelType*
- *StrucLength*
- *Version*

如果變更其他欄位，則結束程式所設定的值必須有效。 如果值無效，則會將錯誤訊息寫入錯誤日誌檔或顯示在主控台上 (視環境而定)。

C 呼叫

```
exitname (&ChannelExitParms, &ChannelDefinition);
```

傳遞至結束程式的參數宣告如下:

```
MQCXP  ChannelExitParms;   /* Channel exit parameter block */
MQCD   ChannelDefinition;  /* Channel definition */
```

COBOL 呼叫

```
     CALL 'exitname' USING CHANNELEXITPARMS, CHANNELDEFINITION.
```

傳遞至結束程式的參數宣告如下:

```
**   Channel exit parameter block
 01  CHANNELEXITPARMS.
     COPY CMQCXPV.
**   Channel definition
```

```
 01  CHANNELDEFINITION.
     COPY CMQCDV.
```

RPG 呼叫 (ILE)

```
     C*..1....:....2....:....3....:....4....:....5....:....6....:....7..
     C                     CALLP     exitname(MQCXP : MQCD)
```

呼叫的原型定義為:

```
     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 組譯器呼叫

```
         CALL EXITNAME,(CHANNELEXITPARMS,CHANNELDEFINITION)
```

傳遞至結束程式的參數宣告如下:

```
CHANNELEXITPARMS   CMQCXPA  ,  Channel exit parameter block
CHANNELDEFINITION  CMQCDA   ,  Channel definition
```

使用注意事項

1. 通道結束程式所執行的功能由結束程式的提供者定義。 不過，結束程式必須符合這裡及相關聯控制區塊 (MQCXP) 中所定義的規則。
2. 傳遞至通道自動定義結束程式的 *ChannelExitParms* 參數是 MQCXP 結構。 傳遞的 MQCXP 版本視結束程式執行所在的環境而定; 如需詳細資料，請參閱 第 958 頁的『MQCXP-通道結束程式參數』 中 *Version* 欄位的說明。
3. 傳遞至通道自動定義結束程式的 *ChannelDefinition* 參數是 MQCD 結構。 所傳遞 MQCD 的版本視結束程式執行所在的環境而定; 如需詳細資料，請參閱 第 923 頁的『MQCD-通道定義』 中 *Version* 欄位的說明。

MQXWAIT-結束中等待

MQXWAIT 呼叫會等待事件發生。 它只能從 z/OS 上的通道結束程式使用。

使用 MQXWAIT 有助於避免在通道結束程式執行導致等待的動作時可能發生的效能問題。 事件 MQXWAIT 正在等待由 MVS ECB (事件控制區塊) 發出信號。 歐洲央行在 MQXWD 控制區塊說明中說明。

語法

MQXWAIT *(Hconn, WaitDesc, CompCode, Reason)*

參數

MQXWAIT 呼叫具有下列參數。

Hconn (MQHCONN)-輸入

連線控點。

此控點代表佇列管理程式的連線。在相同或更早的結束程式呼叫中發出的前一個 MQCONN 呼叫已傳回 *Hconn* 值。

WaitDesc (MQXWD)-輸入/輸出

等待描述子。

此參數說明要等待的事件。如需此結構中欄位的詳細資料，請參閱 第 971 頁的『MQXWD-結束程式等待描述子』 。

CompCode (MQLONG)-輸出

完成碼。

它是下列其中一個代碼:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因 (MQLONG)-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

MQRC_ADAPTER_NOT_AVAILABLE
(2204 ， X'89C') 配接卡無法使用。

MQRC_OPTIONS_ERROR
(2046 ， X'7FE') 選項無效或不一致。

已取消 MQRC_XWAIT_CANCELED
(2107 ， X'83B') MQXWAIT 呼叫已取消。

MQRC_XWAIT_ERROR
(2108 ， X'83C') 呼叫 MQXWAIT 呼叫無效。

C 呼叫

```
MQXWAIT (Hconn, &WaitDesc, &CompCode, &Reason);
```

宣告參數如下:

```
MQHCONN  Hconn;     /* Connection handle */
MQXWD    WaitDesc;  /* Wait descriptor */
MQLONG   CompCode;  /* Completion code */
MQLONG   Reason;    /* Reason code qualifying CompCode */
```

System/390 組譯器呼叫

```
         CALL MQXWAIT,(HCONN,WAITDESC,COMPCODE,REASON)
```

宣告參數如下:

```
HCONN     DS       F  Connection handle
WAITDESC  CMQXWDA  ,  Wait descriptor
COMPCODE  DS       F  Completion code
REASON    DS       F  Reason code qualifying COMPCODE
```

MQCD-通道定義

MQCD 結構包含控制通道執行的參數。 它會傳遞至從「訊息通道代理程式 (MCA)」呼叫的每一個通道結束程式。

如需通道結束程式的相關資訊，請參閱 第 916 頁的『MQ_CHANNEL_EXIT-通道結束程式』。 本主題中的說明與訊息通道及 MQI 通道相關。

結束程式名稱欄位

呼叫結束程式時， *SecurityExit*、 *MsgExit*、 *SendExit*、 *ReceiveExit* 及 *MsgRetryExit* 中的相關欄位會包含目前正在呼叫之結束程式的名稱。 這些欄位中名稱的意義取決於 MCA 執行所在的環境。 除非另有說明，否則名稱會在欄位內靠左對齊，不含內嵌空白; 名稱會以空白填補欄位長度。 在下列說明中，方括弧 ([]) 表示選用資訊:

UNIX 系統
結束程式名稱是可動態載入模組或程式庫的名稱，字尾是位於該程式庫中的函數名稱。 函數名稱必須以括弧括住。 程式庫名稱可以選擇性地以目錄路徑作為字首:

```
[path]library(function)
```

名稱限制為最多 128 個字元。

z/OS
結束程式名稱是載入模組的名稱，對 LINK 或 LOAD 巨集的 EP 參數上的規格有效。 名稱限制為最多 8 個字元。

Windows
結束程式名稱是動態鏈結程式庫的名稱，字尾是位於該程式庫中的函數名稱。 函數名稱必須以括弧括住。 媒體庫名稱可以選擇性地以目錄路徑和磁碟機作為字首:

```
[d:][path]library(function)
```

名稱限制為最多 128 個字元。

IBM i
跳出名稱是 10 個位元組的程式名稱，後面接著 10 個位元組的檔案庫名稱。 如果名稱長度小於 10 個位元組，則會以空白填補每一個名稱，使其成為 10 個位元組。 媒體庫名稱可以是 `*LIBL` ，但呼叫通道自動定義結束程式時除外，在此情況下需要完整名稱。

變更通道結束程式中的 MQCD 欄位

通道結束程式可以變更 MQCD 中的欄位。 變更的值會保留在 MQCD 中，並傳遞至結束鏈中任何剩餘的結束程式，以及傳遞至任何共用通道實例的交談。 在通道的持續生命期限期間， MCA 也會使用變更的 MQCD 進行正常處理。

結束程式不得變更下列 MQCD 欄位:

- ChannelName
- ChannelType
- StrucLength
- 版本

相關參考
第 924 頁的『欄位』
本主題列出 MQCD 結構中的所有欄位，並說明每一個欄位。

第 945 頁的『C 宣告』
此宣告是 MQCD 結構的 C 宣告。

第 947 頁的『COBOL 宣告』

此宣告是 MQCD 結構的 COBOL 宣告。

第 950 頁的『RPG 宣告 (ILE)』
此宣告是 MQCD 結構的 RPG 宣告。

第 952 頁的『 System/390 組譯器宣告』
此宣告是 MQCD 結構的 System/390 組譯器宣告。

第 954 頁的『Visual Basic 宣告』
此宣告是 MQCD 結構的 Visual Basic 宣告。

第 955 頁的『變更通道結束程式中的 MQCD 欄位』
通道結束程式可以變更 MQCD 中的欄位。 不過，除了列出的情況之外，通常不會處理這些變更。

欄位

本主題列出 MQCD 結構中的所有欄位，並說明每一個欄位。

BatchHeartbeat (MQLONG)
這個欄位指定用來觸發通道批次活動訊號的時間間隔。

批次活動訊號可讓傳送端通道在不確定之前判斷遠端通道實例是否仍在作用中。 如果傳送端通道未在指定時間間隔內與遠端通道實例通訊，則會發生批次活動訊號。

該值在 0 到 999 999 的範圍內; 單位是毫秒。 零值表示未啟用批次活動訊號。

此欄位僅適用於 *ChannelType* 為 MQCHT_SENDER、MQCHT_SERVER、MQCHT_CLUSSDR 或 MQCHT_CLUSRCVR 的通道。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_7，則此欄位不存在。

BatchInterval (MQLONG)
此欄位指定在現行批次中傳輸的訊息數少於 *BatchSize* 則通道保持批次開啟的大約時間 (毫秒)。

如果 *BatchInterval* 大於零，則會先發生下列任何事件來終止批次:

- 已傳送 *BatchSize* 訊息，或
- 自批次開始以來已經歷 *BatchInterval* 毫秒。

如果 *BatchInterval* 為零，則會先發生下列任一事件來終止批次:

- 已傳送 *BatchSize* 訊息，或
- 傳輸佇列會變成空的。

BatchInterval 必須在 0 到 999 999 999 的範圍內。

此欄位僅適用於 *ChannelType* 為 MQCHT_SENDER、MQCHT_SERVER、MQCHT_CLUSSDR 或 MQCHT_CLUSRCVR 的通道。

這是結束程式的輸入欄位。 當 *Version* 小於 MQCD_VERSION_4 時，此欄位不存在。

BatchSize (MQLONG)
此欄位指定在同步化通道之前可透過通道傳送的訊息數上限。

此欄位與 *ChannelType* 為 MQCHT_SVRCONN 或 MQCHT_CLNTCONN 的通道無關。

ChannelMonitoring (MQLONG)
此欄位指定通道監視資料收集的現行層次。

此欄位與 ChannelType 為 MQCHT_CLNTCONN 的通道無關。

它是下列其中一個值:

- MQMON_OFF
- MQMON_LOW
- MQ mon_MEDIT
- MQMON_HIGH

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_8，則不存在。

ChannelName (MQCHAR20)
此欄位指定通道定義名稱。

遠端機器上必須有同名的通道定義，才能進行通訊。

名稱只能使用下列字元:

- 大寫 A-Z
- 小寫 a-z
- 數字 0-9
- 句點 (.)
- 正斜線 (/)
- 底線 (_)
- 百分比符號 (%)

並在右側填補空白。 不容許前導或內嵌空白。

此欄位的長度由 MQ_CHANNEL_NAME_LENGTH 提供。

ChannelStatistics (MQLONG)
此欄位指定通道統計資料收集的現行層次。

此欄位與 ChannelType 為 MQCHT_CLNTCONN 的通道無關。

它是下列其中一個值:

- MQMON_OFF
- MQMON_LOW
- MQ mon_MEDIT
- MQMON_HIGH

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_8，則不存在。

ChannelType (MQLONG)
此欄位指定通道的類型。

它是下列其中一個值:

MQCHT_SENDER
寄件者。

MQCHT_SERVER
伺服器。

MQCHT_RECEIVER
接收器。

MQCHT_REQUESTER
要求者。

MQCHT_CLNTCONN
用戶端連線。

MQCHT_SVRCONN
伺服器連線 (供用戶端使用)。

MQCHT_CLUSSDR
叢集傳送端。

MQCHT_CLUSRCVR
叢集接收端。

ClientChannel 加權 *(MQLONG)*
此欄位指定加權，以影響使用的用戶端連線通道定義。

使用 ClientChannelWeight 屬性，以便在有多個適合的定義可用時，可以根據用戶端通道定義的加權來隨機選取用戶端通道定義。 當用戶端透過指定以星號開頭的佇列管理程式名稱來發出 MQCONN 要求連線至佇列管理程式群組時，如果用戶端通道定義表 (CCDT) 中有多個合適的通道定義可用，則會根據加權隨機選取要使用的定義，並按字母順序先選取任何適用的「 ClientChannel 加權 (0)」定義。

請指定範圍在 0 - 99 的值。 預設值是 0。

0 的值指出未執行負載平衡，並按字母順序選取適用的定義。 若要啟用負載平衡，請選擇範圍在 1 - 99 的值，其中 1 是最低加權，而 99 是最高加權。 在具有非零加權的兩個以上通道之間的訊息分佈，與這些加權的比例成正比。 例如，選取時間大約 10%、20% 及 70% 具有 ClientChannel 加權值 2、4 及 14 的三個通道。 無法保證此配送。

此屬性僅適用於用戶端連線通道類型。

這是結束程式的輸入欄位。 如果 版小於 MQCD_VERSION_9，則此欄位不存在。

ClusterPtr (MQPTR)
此欄位指定叢集名稱的位址清單。

如果 *ClustersDefined* 大於零，則此位址是叢集名稱清單的位址。通道屬於列出的每一個叢集。

此欄位僅適用於具有 MQCHT_CLUSSDR 或 MQCHT_CLUSRCVR 的 *ChannelType* 通道。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_5，則此欄位不存在。

ClustersDefined (MQLONG)
此欄位指定通道所屬的叢集數目。

此欄位是 *ClusterPtr* 所指向的叢集名稱數目。它是零或更大。

此欄位僅適用於具有 MQCHT_CLUSSDR 或 MQCHT_CLUSRCVR 的 *ChannelType* 通道。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_5，則此欄位不存在。

CLWLChannelPriority (MQLONG)
此欄位指定叢集工作量通道優先順序。

工作量管理程式選擇演算法會根據等級，從選取的目的地集中選取優先順序最高的目的地。 如果有兩個可能的目的地佇列管理程式，此屬性可用來讓一個佇列管理程式失效接手至另一個佇列管理程式。 所有訊息都會移至具有最高優先順序的佇列管理程式，直到該佇列管理程式結束為止，然後這些訊息會移至具有下一個最高優先順序的佇列管理程式。

該值在 0 到 9 的範圍內。 預設值是 0。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_8，則此欄位不存在。

如需進一步資訊，請參閱 配置佇列管理程式叢集。

CLWLChannelRank (MQLONG)
此欄位指定叢集工作量通道等級。

工作量管理程式選擇演算法會選取等級最高的目的地。 當最終目的地是不同叢集上的佇列管理程式時，您可以設定中間閘道佇列管理程式的等級 (在鄰接叢集的交集處) ，以便選擇演算法正確地選擇更接近最終目的地的目的地佇列管理程式。

該值在 0 到 9 的範圍內。 預設值是 0。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_8，則此欄位不存在。

如需進一步資訊，請參閱 配置佇列管理程式叢集 。

CLWLChannelWeight (MQLONG)
此欄位指定叢集工作量通道加權。

叢集工作量通道加權。

工作量管理程式選擇演算法會使用通道的「加權」屬性來扭曲目的地選項，以便將更多訊息傳送至特定機器。 例如，您可以為大型 UNIX 伺服器上的通道提供比小型桌面 PC 上另一個通道更大的「加權」，而選擇演算法會比 PC 更頻繁地選擇 UNIX 伺服器。

該值在 1 到 99 的範圍內。 預設值為 50。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_8，則此欄位不存在。

如需進一步資訊，請參閱 配置佇列管理程式叢集。

ConnectionAffinity (MQLONG)
此欄位指定使用相同佇列管理程式名稱多次連接的用戶端應用程式是否使用相同的用戶端通道。

當有多個適用的通道定義可供使用時，請使用這個屬性。

此值是下列其中一個:

MQCAFTY_PREFERRED
在讀取用戶端通道定義表 (CCDT) 的處理程序中，第一個連線會根據加權，以任何適用的 CLNTWGHT (0) 定義優先且按字母順序來建立適用定義的清單。 程序中的每一個連線都會嘗試使用清單中的第一個定義來連接。 如果連線不成功，則會使用下一個定義。 CLNTWGHT 值不是 0 的失敗定義會移至清單結尾。 CLNTWGHT (0) 定義會保留在清單開頭，且會先針對每一個連線選取。

每一個具有相同主機名稱的用戶端處理程序一律會建立相同的清單。

對於以 C、C++ 或 .NET 程式設計架構 (包括完全受管理的 .NET) 撰寫的用戶端應用程式，如果自建立清單以來已修改 CCDT，則會更新清單。

此值為預設值。

MQCAFTY_NONE
在程序中讀取 CCDT 的第一個連線，會建立適用定義的清單。 程序中的所有連線都會根據加權來選取適用的定義，並按字母順序先選取任何適用的 CLNTWGHT (0) 定義。

對於以 C、C++ 或 .NET 程式設計架構 (包括完全受管理的 .NET) 撰寫的用戶端應用程式，如果自建立清單以來已修改 CCDT，則會更新清單。

此屬性僅適用於用戶端連線通道類型。

這是結束程式的輸入欄位。 如果 版小於 MQCD_VERSION_9，則此欄位不存在。

ConnectionName (MQCHAR264)
此欄位指定通道的連線名稱。

若為叢集接收端通道 (指定時)， CONNAME 會與本端佇列管理程式相關，若為其他通道，則會與目標佇列管理程式相關。 您指定的值取決於要使用的傳輸通訊協定 (*TransportType*):

- 對於 MQXPT_LU62，它是夥伴「邏輯單元」的完整名稱。
- 對於 MQXPT_NETBIOS，它是在遠端機器上定義的 NetBIOS 名稱。
- 若為 MQXPT_TCP，它是主機名稱、以 IPv4 帶點十進位或 IPv6 十六進位格式指定之遠端機器的網址，或叢集接收端通道的本端機器。
- 對於 MQXPT_SPX，它是 SPX 樣式位址，包含 4 位元組網址、6 位元組節點位址及 2 位元組 socket 號碼。

定義通道時，此欄位與 *ChannelType* 為 MQCHT_SVRCONN 或 MQCHT_RECEIPT 的通道無關。 不過，當通道定義傳遞至結束程式時，不論通道類型為何，這個欄位都會包含夥伴的位址。

此欄位的長度由 MQ_CONN_NAME_LENGTH 提供。 如果 *Version* 小於 MQCD_VERSION_2，則此欄位不存在。

DataConversion (MQLONG)
此欄位指定如果接收訊息通道代理程式無法執行此轉換，傳送訊息通道代理程式是否嘗試轉換應用程式訊息資料。

此欄位僅適用於不是邏輯訊息區段的訊息; MCA 絕不會嘗試轉換作為區段的訊息。

此欄位僅適用於 *ChannelType* 為 MQCHT_SENDER、MQCHT_SERVER、MQCHT_CLUSSDR 或 MQCHT_CLUSRCVR 的通道。 它是下列其中一項:

MQCDC_SENDER_CONVERSION
由傳送端進行轉換。

MQCDC_NO_SENDER_CONVERSION
寄件者沒有轉換。

DefReconnect (MQLONG)
DefReconnect 通道屬性會設定用戶端連線通道的預設重新連線屬性值。

預設自動用戶端重新連線選項。 您可以配置 IBM WebSphere MQ MQI client 以自動重新連接用戶端應用程式。 在連線失敗之後，IBM WebSphere MQ MQI client 會嘗試重新連接到佇列管理程式。 它會在沒有應用程式用戶端發出 MQCONN 或 MQCONNX MQI 呼叫的情況下，嘗試重新連接。

重新連線是 MQCONNX 選項。 透過使用 DefReconnect 通道屬性，您可以將重新連線行為新增至使用 MQCONN 的現有應用程式。 您也可以變更使用 MQCONNX 之應用程式的重新連線行為。

您也可以從 mqclient.ini 檔設定 DefRecon 值，以設定或修改重新連線行為。 mqclient.ini 檔案中的 DefRecon 值優先於 DefReconnect 通道屬性。

Syntax

```
DefReconnect( MQRCN_NO|MQRCN_YES|MQRCN_Q_MGR|
MQRCN_DISABLED)
```

參數

MQRCN_NO
MQRCN_NO 是預設值。

除非以 MQCONNX 置換，否則不會自動重新連接用戶端。

MQRCN_YES
除非被 MQCONNX 置換，否則用戶端會自動重新連接。

MQRCN_Q_MGR
除非以 MQCONNX 置換，否則用戶端會自動重新連接，但只會重新連接至相同的佇列管理程式。 QMGR 選項具有與 MQCNO_RECONNECT_Q_MGR 相同的效果。

MQRCN_DISABLED
即使用戶端程式使用 MQCONNX MQI 呼叫來要求，也會停用重新連線。

適用於 Java 的 IBM WebSphere MQ 類別不支援自動用戶端重新連線。

表 *592:* 自動重新連線取決於在應用程式中及通道定義中設定的值

DefReconnect	**應用程式中設定的重新連線選項**			
	MQCNO_RECONNECT	MQCNO_RECONNECT_Q_MGR	MQCNO_RECONNECT_AS_DEF	MQCNO_RECONNECT_DISABLED
MQRCN_NO	YES	QMGR	NO	NO
MQRCN_YES	YES	QMGR	YES	NO
MQRCN_Q_MGR	YES	QMGR	QMGR	NO
MQRCN_DISABLED	NO	NO	NO	NO

相關概念
自動用戶端重新連線
通道及用戶端重新連線
用戶端配置檔的 CHANNELS 段落

相關參考
連線選項
控制 MQCONNX 動作的選項。

說明 *(MQCHAR64)*
此欄位可用於敘述性註解。

欄位的內容對「訊息通道代理程式」不重要。 不過，它必須只包含可顯示的字元。 它不能包含任何空值字元; 必要的話，會以空白填補右邊。 在 DBCS 安裝中，欄位可以包含 DBCS 字元 (欄位長度上限為 64 個位元組)。

註: 如果此欄位包含不在佇列管理程式字集中的字元 (如 *CodedCharSetId* 佇列管理程式屬性所定義) ，則當此欄位傳送至另一個佇列管理程式時，這些字元可能會不正確地轉換。

此欄位的長度由 MQ_CHANNEL_DESC_LENGTH 提供。

DiscInterval (MQLONG)
此欄位指定在終止通道之前，通道等待訊息到達傳輸佇列的時間上限 (以秒為單位)。

換句話說，它會指定斷線間隔。

A 值零會導致 MCA 無限期等待。

對於使用 TCP 通訊協定的伺服器連線通道，間隔代表用戶端閒置斷線值 (以秒為單位指定)。 如果伺服器連線在此期間未收到來自其友機用戶端的通訊，則會終止連線。 伺服器連線閒置間隔僅適用於來自用戶端的 WebSphere MQ API 呼叫之間，因此在具有等待呼叫的長時間執行 MQGET 期間不會中斷任何用戶端的連線。

此屬性不適用於使用 TCP 以外通訊協定的伺服器連線通道。

此欄位僅適用於 *ChannelType* 為 MQCHT_SENDER、MQCHT_SERVER、MQCHT_CLUSSDR、MQCHT_CLUSRCVR 或 MQCHT_SVRCONN 的通道。

ExitData 長度 *(MQLONG)*
此欄位指定 *MsgUserDataPtr*、 *SendUserDataPtr* 及 *ReceiveUserDataPtr* 欄位所定址之結束程式使用者資料項目清單中每一個使用者資料項目的長度 (以位元組為單位)。

此長度不一定與 MQ_EXIT_DATA_LENGTH 相同。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_4，則此欄位不存在。

ExitName 長度 *(MQLONG)*
此欄位指定 *MsgExitPtr*、 *SendExitPtr* 及 *ReceiveExitPtr* 欄位所定址之結束程式名稱清單中每一個名稱的長度 (以位元組為單位)。

此長度不一定與 MQ_EXIT_NAME_LENGTH 相同。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_4，則此欄位不存在。

HdrComp 清單 *[2] (MQLONG)*
此欄位指定通道支援的標頭資料壓縮技術清單。

清單包含下列一或多個值:

MQCOMPRESS_NONE
不執行標頭資料壓縮。

MQCOMPRESS_SYSTEM
執行標頭資料壓縮。

陣列中未用的值設為 MQCOMPRESS_NOT_AVAILABLE。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_8，則此欄位不存在。

HeartbeatInterval (MQLONG)
此欄位指定活動訊號流程之間的時間 (以秒為單位)。

此欄位的解譯取決於通道類型，如下所示:

- 若通道類型為 MQCHT_SENDER、MQCHT_SERVER、MQCHT_RECEIVER MQCHT_REQUESTER、MQCHT_CLUSSDR 或 MQCHT_CLUSRCVR，此欄位是當傳輸佇列上沒有訊息時，從傳送端 MCA 傳遞活動訊號的間隔時間 (以秒為單位)。 這可讓接收端 MCA 有機會靜止通道。 若要有用， *HeartbeatInterval* 必須小於 *DiscInterval*。
- 對於 MQCD「共用交談」欄位設為零的 MQCHT_CLNTCONN 或 MQCHT_SVRCONN 通道類型，此欄位是當 MCA 代表用戶端應用程式發出 MQGMO_WAIT 選項的 MQGET 呼叫時，從伺服器 MCA 傳遞活動訊號的間隔時間 (以秒為單位)。 這可讓伺服器 MCA 處理在 MQGMO_WAIT 的 MQGET 期間用戶端連線失敗的狀況。
- 對於 MQCD 共用交談欄位設為非零值的 MQCHT_CLNTCONN 或 MQCHT_SVRCONN 通道類型，此欄位是沒有傳送或接收資料流程時活動訊號流程之間的時間 (以秒為單位)。 這可讓通道有效地靜止。

此值在 0 到 999 999 的範圍內。 除非在任一端指定 0 值 (在此情況下不會發生活動訊號交換)，否則所使用的值是在傳送端和接收端指定的較大值。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_4，則此欄位不存在。

KeepAlive 間隔 *(MQLONG)*
此欄位指定傳遞至通訊堆疊以通道保持作用中計時的值。

此值適用於 TCP/IP 及 SPX 通訊協定，但並非所有實作都支援此參數。

該值在 0 到 99 999 的範圍內; 單位是秒。 零值表示未啟用通道保留作用中，但如果啟用 TCP/IP 保留作用中 (而非通道保留作用中)，則仍可能發生保留作用中。 下列特殊值也有效:

MQ 開 _ 自動
自動。

此值指出從協議的活動訊號間隔計算保留作用中間隔，如下所示:

- 如果協議的活動訊號間隔大於零，則使用的保持作用中間隔是活動訊號間隔加 60 秒。
- 如果協議的活動訊號間隔為零，則所使用的保持作用中間隔為零。

- 在 z/OS 上，當在佇列管理程式物件上指定 TCPKEEP (YES) 時，會發生 TCP/IP 保持作用中。
- 在其他環境中，當在分散式佇列配置檔的 TCP 段落中指定 KEEPALIVE=YES 參數時，即會發生 TCP/IP 保持作用中。

此欄位僅適用於 *TransportType* 為 MQXPT_TCP 或 MQXPT_SPX 的通道。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_7，則此欄位不存在。

LocalAddress (MQCHAR48)
此欄位指定為出埠通訊通道定義的本端 TCP/IP 位址。

如果未定義出埠通訊的特定位址，則此欄位為空白。位址可以選擇性地包含埠號或埠號範圍。 此位址的格式為:

```
[ip-addr][(low-port[,high-port])]
```

其中方括弧 ([]) 表示選用資訊， `ip-addr` 以 IPv4 帶點十進位、 IPv6 十六進位或英數格式指定， `low-port` 及 `high-port` 是以括弧括住的埠號。 全部都是選用項目。

出埠通訊的特定 IP 位址、埠或埠範圍適用於在不同 TCP/IP 堆疊上重新啟動通道的回復實務範例。

LocalAddress 在形式上類似於 *ConnectionName*，但不得與它混淆。 *LocalAddress* 指定本端通訊的性質，而 *ConnectionName* 指定如何呼叫到遠端佇列管理程式。

此欄位僅適用於 *TransportType* 為 MQXPT_TCP 且 *ChannelType* 為 MQCHT_SENDER、MQCHT_SERVER、MQCHT_REQUESTER、MQCHT_CLNTCONN、MQCHT_CLUSSDR 或 MQCHT_CLUSRCVR 的通道。

此欄位的長度由 MQ_LOCAL_ADDRESS_LENGTH 指定。 如果 *Version* 小於 MQCD_VERSION_7，則此欄位不存在。

LongMCAUserIdLength (MQLONG)
此欄位指定 *LongMCAUserIdPtr* 所指向完整 MCA 使用者 ID 的長度 (以位元組為單位)。

此欄位與 *ChannelType* 為 MQCHT_CLNTCONN 的通道無關。

這是結束程式的輸入/輸出欄位。 如果 *Version* 小於 MQCD_VERSION_6，則此欄位不存在。

LongMCAUserIdPtr (MQPTR)
此欄位指定長 MCA 使用者 ID 的位址。

如果 *LongMCAUserIdLength* 大於零，則此欄位是完整 MCA 使用者 ID 的位址。完整 ID 的長度由 *LongMCAUserIdLength* 提供。 MCA 使用者 ID 的前 12 個位元組也包含在欄位 *MCAUserIdentifier* 中。

如需 MCA 使用者 ID 的詳細資料，請參閱 *MCAUserIdentifier* 欄位的說明。

此欄位與 *ChannelType* 為 MQCHT_SDR、MQCHT_SVR、MQCHT_CLNTCONN 或 MQCHT_CLUSSDR 的通道無關。

這是結束程式的輸入/輸出欄位。 如果 *Version* 小於 MQCD_VERSION_6，則此欄位不存在。

LongRemoteUserId 長度 (MQLONG)
此欄位指定 *LongRemoteUserIdPtr* 所指向完整遠端使用者 ID 的長度 (以位元組為單位)。

此欄位僅適用於 *ChannelType* 為 MQCHT_CLNTCONN 或 MQCHT_SVRCONN 的通道。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_6，則此欄位不存在。

LongRemoteUserIdPtr (MQPTR)
此欄位指定長遠端使用者 ID 的位址。

如果 *LongRemoteUserIdLength* 大於零，則此旗標是完整遠端使用者 ID 的位址。完整 ID 的長度由 *LongRemoteUserIdLength* 提供。 遠端使用者 ID 的前 12 個位元組也包含在欄位 *RemoteUserIdentifier* 中。

如需遠端使用者 ID 的詳細資料，請參閱 *RemoteUserIdentifier* 欄位的說明。

此欄位僅適用於 *ChannelType* 為 MQCHT_CLNTCONN 或 MQCHT_SVRCONN 的通道。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_6，則此欄位不存在。

LongRetry 計數 (MQLONG)
此欄位指定在 *ShortRetryCount* 指定的計數已用盡之後使用的計數。

它指定在將錯誤記載至操作員之前，以 *LongRetryInterval* 指定的間隔進一步嘗試連接至遠端機器的次數上限。

此欄位僅適用於 *ChannelType* 為 MQCHT_SENDER、MQCHT_SERVER、MQCHT_CLUSSDR 或 MQCHT_CLUSRCVR 的通道。

LongRetry 間隔 (MQLONG)
此欄位指定在重新嘗試連接遠端機器之前等待的秒數上限。

如果通道必須等待變成作用中，則可以延長重試之間的間隔。

此欄位僅適用於 *ChannelType* 為 MQCHT_SENDER、MQCHT_SERVER、MQCHT_CLUSSDR 或 MQCHT_CLUSRCVR 的通道。

MaxInstances (MQLONG)
這個欄位指定個別伺服器連線通道可同時啟動的實例數上限。

此欄位僅在伺服器連線通道上使用。

欄位可以有 0-999 999 999 範圍內的值。 零值會防止所有用戶端存取。

此欄位的預設值為 999 999 999 999。

如果此欄位的值減少到小於目前執行中的伺服器連線通道實例數的數字，則不會影響那些執行中的實例。 不過，除非已停止執行足夠的現有實例，否則新實例無法啟動，因此目前執行中的實例數會小於欄位值。

MaxInstancesPerClient (MQLONG)
這個欄位指定可從單一用戶端啟動之個別伺服器連線通道的同時實例數上限。

在此環境定義中，源自相同遠端網址的連線視為來自相同用戶端。

此欄位僅在伺服器連線通道上使用。

欄位可以有 0-999 999 999 範圍內的值。 零值會防止所有用戶端存取。

此欄位的預設值為 999 999 999 999。

如果此欄位的值減少至小於目前從個別用戶端執行的伺服器連線通道實例數，則不會影響那些執行中實例。 不過，任何那些用戶端的新實例都無法啟動，除非有足夠的現有實例停止執行，使得目前執行中的實例數目 (源自嘗試啟動新實例的用戶端) 小於欄位值。

MaxMsg 長度 (MQLONG)
此欄位指定可在通道上傳輸的訊息長度上限。

這會與遠端通道的值相互比較，而實際最大值是兩個值中的較低值。

MCAName (MQCHAR20)
此欄位是保留欄位。

此欄位的值為空白。

此欄位的長度由 MQ_MCA_NAME_LENGTH 提供。

MCASecurityId (MQBYTE40)
此欄位指定 MCA 的安全 ID。

此欄位與 *ChannelType* 為 MQCHT_CLNTCONN 的通道無關。

下列特殊值指出沒有安全 ID:

MQSID_NONE
未指定安全 ID。

欄位長度的值為二進位零。

對於 C 程式設計語言，也會定義常數 MQSID_NONE_ARRAY; 此常數具有與 MQSID_NONE 相同的值，但它是字元陣列而非字串。

這是結束程式的輸入/輸出欄位。 此欄位的長度由 MQ_SECURITY_ID_LENGTH 提供。 如果 *Version* 小於 MQCD_VERSION_6，則此欄位不存在。

MCAType (MQLONG)
此欄位指定訊息通道代理程式的類型。

此欄位僅適用於 *ChannelType* 為 MQCHT_SENDER、MQCHT_SERVER、MQCHT_REQUESTER、MQCHT_CLUSSDR 或 MQCHT_CLUSRCVR 的通道。

此值是下列其中一個:

MQMCAT_PROCESS
程序。

訊息通道代理程式會以個別處理程序執行。

MQMCAT_THREAD
執行緒 (IBM i、 UNIX 及 Windows)。

訊息通道代理程式會作為個別執行緒來執行。

當 版小於 MQCD_VERSION_2 時，此欄位不存在。

MCAUserIdentifier (MQCHAR12)
此欄位指定訊息通道代理程式 (MCA) 的使用者 ID。

此欄位使用 MCA 使用者 ID 的前 12 個位元組，且可以由安全代理程式設定。

有兩個欄位包含 MCA 使用者 ID:

- *MCAUserIdentifier* 包含 MCA 使用者 ID 的前 12 個位元組，如果 ID 短於 12 個位元組，則會以空白填補。 *MCAUserIdentifier* 可以為空白。
- *LongMCAUserIdPtr* 指向完整 MCA 使用者 ID，其長度可以超過 12 個位元組。 其長度由 *LongMCAUserIdLength* 提供。 完整 ID 不包含尾端空白，且不是以空值結尾。 如果 ID 為空白，則 *LongMCAUserIdLength* 為零，且未定義 *LongMCAUserIdPtr* 的值。

  註: 如果 *Version* 小於 MQCD_VERSION_6，則 *LongMCAUserIdPtr* 不存在。

如果 MCA 使用者 ID 不是空白，它會指定訊息通道代理程式用來授權存取 WebSphere MQ 資源的使用者 ID。 對於通道類型 MQCHT_REQUESTER、MQCHT_RECEIVER 及 MQCHT_CLUSRCVR，如果 PutAuthority 是 MQPA_DEFAULT，則這是用於對目的地佇列的放置作業進行授權檢查的使用者 ID。

如果 MCA 使用者 ID 為空白，則訊息通道代理程式會使用其預設使用者 ID。

安全結束程式可以設定 MCA 使用者 ID，以指出訊息通道代理程式必須使用的使用者 ID。 結束程式可以變更 *MCAUserIdentifier* 或 *LongMCAUserIdPtr* 所指向的字串。 如果兩者都已變更，但彼此不同，則 MCA 會優先使用 *LongMCAUserIdPtr*，而不是 *MCAUserIdentifier*。 如果結束程式變更 *LongMCAUserIdPtr* 所指出的字串長度，則必須相應地設定 *LongMCAUserIdLength*。 如果結束程式增加 ID 的長度，則結束程式必須配置所需長度的儲存體，將該儲存體設為所需 ID，並將該儲存體的位址放置在 *LongMCAUserIdPtr* 中。 稍後以 MQXR_TERM 原因呼叫結束程式時，結束程式會負責釋放該儲存體。

對於 *ChannelType* 為 MQCHT_SVRCONN 的通道，如果通道定義中的 *MCAUserIdentifier* 為空白，則會將從用戶端傳送的任何使用者 ID 複製到其中。 這個使用者 ID (在伺服器上的安全結束程式進行任何修改之後) 是用戶端應用程式用來執行的使用者 ID。

MCA 使用者 ID 與 *ChannelType* 為 MQCHT_SDR、MQCHT_SVR、MQCHT_CLNTCONN、MQCHT_CLUSSDR 的通道無關。

這是結束程式的輸入/輸出欄位。 此欄位的長度由 MQ_USER_ID_LENGTH 提供。 當 *Version* 小於 MQCD_VERSION_2 時，此欄位不存在。

ModeName (MQCHAR8)
此欄位指定 LU 6.2 模式名稱。

僅當傳輸通訊協定 (*TransportType*) 為 MQXPT_LU62 且 *ChannelType* 不是 MQCHT_SVRCONN 或 MQCHT_RECEIPT 時，此欄位才相關。

此欄位一律為空白。 資訊會改為包含在通訊端物件中。

此欄位的長度由 MQ_MODE_NAME_LENGTH 提供。

MsgComp 清單 *[16] (MQLONG)*
此欄位指定通道支援的訊息資料壓縮技術清單。

清單包含下列一或多個值:

MQCOMPRESS_NONE
不執行訊息資料壓縮。

MQCOMPRESS_RLE
使用執行長度編碼來執行訊息資料壓縮。

MQCOMPRESS_ZLIBFAST
訊息資料壓縮是使用 zlib 壓縮技術來執行。 建議使用快速壓縮時間。

MQCOMPRESS_ZLIBHIGH
訊息資料壓縮是使用 zlib 壓縮技術來執行。 建議使用高階壓縮。

陣列中未用的值設為 MQCOMPRESS_NOT_AVAILABLE。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_8，則此欄位不存在。

MsgExit (MQCHARn)
此欄位指定通道訊息結束程式名稱。

如果此名稱非空白，則會在下列時間呼叫結束程式:

- 在從傳輸佇列 (傳送端或伺服器) 擷取訊息之後立即，或在將訊息放入目的地佇列 (接收端或要求端) 之前立即。

  該結束程式會提供整個應用程式訊息及傳輸佇列標頭，以進行修改。
- 在通道起始設定及終止時。

此欄位不適用於 *ChannelType* 為 MQCHT_SVRCONN 或 MQCHT_CLNTCONN; 此類通道絕不會呼叫訊息結束程式。

如需各種環境中此欄位內容的說明，請參閱 第 923 頁的『MQCD-通道定義』 。

此欄位的長度由 MQ_EXIT_NAME_LENGTH 提供。

註: 此常數的值是環境特定的。

MsgExitPtr (MQPTR)
此欄位指定第一個 *MsgExit* 欄位的位址。

如果 *MsgExitsDefined* 大於零，則此位址是鏈結中每一個通道訊息結束程式的名稱清單位址。

每一個名稱都在長度為 *ExitNameLength* 的欄位中，右側以空白填補。 有 *MsgExitsDefined* 欄位彼此相鄰-每一個結束程式各一個。

會保留結束程式對這些名稱所做的任何變更，但訊息通道結束程式不會採取明確動作-它不會變更呼叫的結束程式。

如果 *MsgExitsDefined* 為零，則此欄位是空值指標。

在程式設計語言不支援指標資料類型的平台上，此欄位宣告為適當長度的位元組字串。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_4，則此欄位不存在。

MsgExits 已定義 (MQLONG)
此欄位指定鏈中定義的通道訊息結束程式數目。

它大於或等於零。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_4，則此欄位不存在。

MsgRetry 計數 (MQLONG)
此欄位指定在第一次嘗試失敗之後， MCA 嘗試放置訊息的次數。

此欄位指出在第一個 MQOPEN 或 MQPUT 失敗且完成碼為 MQCC_FAILED 的情況下， MCA 嘗試開啟或放置作業的次數。此屬性的效果取決於 *MsgRetryExit* 是空白還是非空白:

- 如果 *MsgRetryExit* 空白， *MsgRetryCount* 屬性會控制 MCA 是否會嘗試重試。 如果屬性值為零，則不會嘗試重試。 如果屬性值大於零，則會在 *MsgRetryInterval* 屬性給定的間隔內嘗試重試。

  只會嘗試重試下列原因碼:

  - MQRC_PAGESET_FULL
  - MQRC_PUT_INHIBITED
  - MQRC_Q_FULL

  對於其他原因碼， MCA 會立即繼續其正常失敗處理程序，而不重試失敗訊息。
- 如果 *MsgRetryExit* 不是空白，則 *MsgRetryCount* 屬性不會影響 MCA; 相反地，它是訊息重試結束程式，可決定重試的次數及間隔; 即使 *MsgRetryCount* 屬性為零，也會呼叫結束程式。

  MsgRetryCount 屬性可供 MQCD 結構中的結束程式使用，但不需要允許它的結束程式-會無限期地繼續重試，直到結束程式在 MQCXP 的 *ExitResponse* 欄位中傳回 MQXCC_SUPPRESS_FUNCTION 為止。

此欄位僅適用於 *ChannelType* 為 MQCHT_REQUESTER、MQCHT_RECEIVER 或 MQCHT_CLUSRCVR 的通道。

當 *Version* 小於 MQCD_VERSION_3 時，此欄位不存在。

MsgRetry 結束 *(MQCHARn)*
此欄位指定通道訊息重試結束程式名稱。

訊息重試結束程式是當 MCA 從 MQOPEN 或 MQPUT 呼叫收到完成碼 MQCC_FAILED 時， MCA 所呼叫的結束程式。結束程式的目的是指定 MCA 在重試 MQOPEN 或 MQPUT 作業之前等待的時間間隔。 或者，可以將結束程式設為不重試作業。

會針對完成碼為 MQCC_FAILED 的所有原因碼來呼叫結束程式-結束程式的設定會決定它希望 MCA 重試的原因碼、嘗試次數及時間間隔。

當不再嘗試作業時， MCA 會執行其正常失敗處理程序; 此處理程序包括產生異常狀況報告訊息 (如果由寄件者指定的話) ， 以及將原始訊息放置在無法傳送郵件的佇列上或捨棄訊息 (根據寄件者是否指定 MQRO_DEAD_LETTER_Q 或 MQRO_D_DISCARD_MSG)。 涉及無法傳送郵件的佇列 (例如，無法傳送郵件的佇列已滿) 的失敗不會導致呼叫訊息重試結束程式。

如果結束程式名稱非空白，則會在下列時間呼叫結束程式:

- 在再次嘗試遞送訊息之前立即執行等待
- 在通道起始設定及終止時

如需各種環境中此欄位內容的說明，請參閱 第 923 頁的『MQCD-通道定義』 。

此欄位僅適用於 *ChannelType* 為 MQCHT_REQUESTER、MQCHT_RECEIVER 或 MQCHT_CLUSRCVR 的通道。

此欄位的長度由 MQ_EXIT_NAME_LENGTH 提供。

註: 此常數的值是環境特定的。

當 *Version* 小於 MQCD_VERSION_3 時，此欄位不存在。

MsgRetry 間隔 *(MQLONG)*
此欄位指定重試開啟或放置作業之前的間隔下限 (毫秒)。

此屬性的效果取決於 *MsgRetryExit* 是空白還是非空白:

- 如果 *MsgRetryExit* 為空白，則 *MsgRetryInterval* 屬性指定在第一個 MQOPEN 或 MQPUT 失敗且完成碼為 MQCC_FAILED 時， MCA 在重試訊息之前等待的最短期間。 零值表示在前一次嘗試之後，將盡快執行重試。 只有在 *MsgRetryCount* 大於零時，才會執行重試。

  如果訊息重試結束程式在 MQCXP 的 *MsgRetryInterval* 欄位中傳回無效值，則此屬性也會用作等待時間。
- 如果 *MsgRetryExit* 不是空白，則 *MsgRetryInterval* 屬性不會影響 MCA; 相反地，它是決定 MCA 等待時間的訊息重試結束程式。 *MsgRetryInterval* 屬性可供 MQCD 結構中的結束程式使用，但結束程式不需要允許使用它。

該值在 0 到 999 999 999 的範圍內。

此欄位僅適用於 *ChannelType* 為 MQCHT_REQUESTER、MQCHT_RECEIVER 或 MQCHT_CLUSRCVR 的通道。

當 *Version* 小於 MQCD_VERSION_3 時，此欄位不存在。

如果 *Version* 小於 MQCD_VERSION_4，則此結構中不存在下列欄位。

MsgRetryUserData (MQCHAR32)
此欄位指定通道訊息重試結束程式使用者資料。

此資料會傳遞至 *ChannelExitParms* 參數的 *ExitData* 欄位中的通道訊息重試結束程式 (請參閱 MQ_CHANNEL_EXIT)。

此欄位最初包含在通道定義中設定的資料。 不過，在這個 MCA 實例的生命期限內， MCA 會保留任何類型的結束程式對此欄位內容所做的任何變更，且對於這個 MCA 實例的後續結束程式呼叫 (不論類型為何) ， 都會使其可見。 這類變更不會影響其他 MCA 實例所使用的通道定義。 可以使用任何字元 (包括二進位資料)。

此欄位僅適用於 *ChannelType* 為 MQCHT_REQUESTER、MQCHT_RECEIVER 或 MQCHT_CLUSRCVR 的通道。

此欄位的長度由 MQ_EXIT_DATA_LENGTH 提供。 當 *Version* 小於 MQCD_VERSION_3 時，此欄位不存在。

此欄位在 WebSphere MQ for IBM i 中不相關。

MsgUser 資料 *(MQCHAR32)*
此欄位指定通道訊息結束程式使用者資料。

此資料會傳遞至 *ChannelExitParms* 參數的 *ExitData* 欄位中的通道訊息結束程式 (請參閱 MQ_CHANNEL_EXIT)。

此欄位最初包含在通道定義中設定的資料。 不過，在這個 MCA 實例的生命期限內， MCA 會保留任何類型的結束程式對此欄位內容所做的任何變更，且對於這個 MCA 實例的後續結束程式呼叫 (不論類型為何) ，都會使其可見。 這類變更不會影響其他 MCA 實例所使用的通道定義。 可以使用任何字元 (包括二進位資料)。

此欄位的長度由 MQ_EXIT_DATA_LENGTH 提供。

此欄位在 WebSphere MQ for IBM i 中不相關。

MsgUserDataPtr (MQPTR)
此欄位指定第一個 *MsgUserData* 欄位的位址。

如果 *MsgExitsDefined* 大於零，則此位址是鏈中每一個通道訊息結束程式的使用者資料項目清單位址。

每一個使用者資料項目都在長度為 *ExitDataLength* 的欄位中，右側以空白填補。 有 *MsgExitsDefined* 欄位彼此相鄰-每一個結束程式各一個。 如果定義的使用者資料項目數目小於結束程式名稱數目，則未定義的使用者資料項目會設為空白。 相反地，如果定義的使用者資料項目數大於結束程式名稱數目，則會忽略多餘的使用者資料項目，且不會呈現給結束程式。

會保留結束程式對這些值所做的任何變更。 這可讓一個結束程式將資訊傳遞至另一個結束程式。 不會對任何變更執行任何驗證，例如，二進位資料可以在必要時寫入這些欄位。

如果 *MsgExitsDefined* 為零，則此欄位是空值指標。

在程式設計語言不支援指標資料類型的平台上，此欄位宣告為適當長度的位元組字串。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_4，則此欄位不存在。

NetworkPriority (MQLONG)
此欄位指定通道的網路連線優先順序。

當特定目的地有多條路徑可用時，會選擇優先順序最高的路徑。 值在 0 到 9 的範圍內; 0 是最低優先順序。

此欄位僅適用於具有 MQCHT_CLUSSDR 或 MQCHT_CLUSRCVR 的 *ChannelType* 通道。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_5，則此欄位不存在。

如果 *Version* 小於 MQCD_VERSION_6，則此結構中不存在下列欄位。

NonPersistentMsgSpeed (MQLONG)
此欄位指定非持續訊息通過通道的速度。

此欄位僅適用於 *ChannelType* 為 MQCHT_SENDER、MQCHT_SERVER、MQCHT_RECEIVER、MQCHT_REQUESTER、MQCHT_CLUSSDR 或 MQCHT_CLUSRCVR 的通道。

此值是下列其中一個:

MQNPMS_NORMAL
正常速度

如果通道定義為 MQNPMS_NORMAL ，則非持續訊息會以正常速度透過通道傳送。 這樣做的優點是如果通道失敗，這些訊息不會遺失。 此外，相同傳輸佇列上的持續及非持續訊息也會維持彼此的相對順序。

MQNPMS_FAST
快速。

如果通道定義為 MQNPMS_FAST，則非持續訊息會以快速的速度透過通道傳送。 這可改善通道的傳輸量，但表示如果通道失敗，則會遺失非持續訊息。 此外，非持續訊息也可以跳先於在相同傳輸佇列上等待的持續訊息，也就是說，相對於持續訊息，不會維護非持續訊息的順序。 不過，會維護非持續訊息相對於彼此的順序。 同樣地，會維護持續訊息相對於彼此的順序。

密碼 *(MQCHAR12)*
此欄位指定當嘗試起始與遠端訊息通道代理程式的安全 SNA 階段作業時，訊息通道代理程式所使用的密碼。

此欄位只能在 UNIX 系統及 Windows 上為非空白，且僅適用於 *ChannelType* 為 MQCHT_SENDER、MQCHT_SERVER、MQCHT_REQUESTER 或 MQCHT_CLNTCONN 的通道。 在 z/OS 上，此欄位不相關。

此欄位的長度由 MQ_PASSWORD_LENGTH 提供。 不過，只會使用前 10 個字元。

如果 *Version* 小於 MQCD_VERSION_2，則此欄位不存在。

PropertyControl (MQLONG)
此欄位指定當訊息即將傳送至 V6 或之前的佇列管理程式 (不瞭解內容描述子概念的佇列管理程式) 時，訊息內容會發生什麼情況。

值可以為:

MQPROP_COMPATIBILITY
如果訊息包含字首為 **mcd.**、 **jms.**、 **usr.**或 **mqext.**的內容，則會將所有訊息內容遞送至 MQRFH2 標頭中的應用程式。 否則，訊息的所有內容 (訊息描述子或延伸中包含的內容除外) 都會被捨棄，且不再可供應用程式存取。

此值是預設值; 它容許預期 JMS 相關內容位於訊息資料中 MQRFH2 標頭的應用程式繼續運作而不修改。

MQPROP_NONE
在訊息傳送至遠端佇列管理程式之前，訊息的所有內容 (訊息描述子或延伸中的內容除外) 都會從訊息中移除。

MQPROP_ALL
當訊息傳送至遠端佇列管理程式時，訊息的所有內容都會包含在訊息中。 這些內容（訊息描述子或延伸中的除外）會放在訊息資料內的一個以上 MQRFH2 標頭中。

此屬性適用於「傳送端」、「伺服器」、「叢集傳送端」及「叢集接收端」通道。

第 112 頁的『MQIA_ * (整數屬性選取器)』
第 149 頁的『MQPROP_ * (佇列和通道內容控制值及內容長度上限)』

PutAuthority (MQLONG)
此欄位指定是否使用與訊息相關聯的環境定義資訊中的使用者 ID 來建立將訊息放入目的地佇列的權限。

此欄位僅適用於 *ChannelType* 為 MQCHT_REQUESTER、MQCHT_RECEIVER 或 MQCHT_CLUSRCVR 的通道。 它是下列其中一項:

MQPA_DEFAULT
使用預設使用者 ID。

MQPA_CONTEXT
使用環境定義使用者 ID。

MQPA_ALTERNATE_OR_MCA
使用訊息描述子的 UserIdentifier 欄位中的使用者 ID。 不使用從網路收到的任何使用者 ID。 此值僅在 z/OS 上受支援。

MQPA_ONLY_MCA
使用預設使用者 ID。 不使用從網路收到的任何使用者 ID。 此值僅在 z/OS 上受支援。

QMgrName (MQCHAR48)
此欄位指定結束程式可以連接的佇列管理程式名稱。

對於 *ChannelType* 不是 MQCHT_CLNTCONN 的通道，此欄位是結束程式可以連接至的佇列管理程式名稱，在 UNIX、 Linux 及 Windows 系統上一律為非空白。

此欄位的長度由 MQ_Q_MGR_NAME_LENGTH 提供。

ReceiveExit (MQCHARn)
此欄位指定通道接收結束程式名稱。

如果此名稱非空白，則會在下列時間呼叫結束程式:

- 在處理收到的網路資料之前。

  結束程式會收到完整的傳輸緩衝區。 可以視需要修改緩衝區的內容。
- 在通道起始設定及終止時。

如需各種環境中此欄位內容的說明，請參閱 第 923 頁的『MQCD-通道定義』 。

此欄位的長度由 MQ_EXIT_NAME_LENGTH 提供。

註: 此常數的值是環境特定的。

ReceiveExitPtr (MQPTR)
此欄位指定第一個 *ReceiveExit* 欄位的位址。

如果 *ReceiveExitsDefined* 大於零，則此位址是鏈中每一個通道接收結束程式的名稱清單位址。

每一個名稱都在長度為 *ExitNameLength* 的欄位中，右側以空白填補。 有 *ReceiveExitsDefined* 欄位彼此相鄰-每一個結束程式各一個。

會保留結束程式對這些名稱所做的任何變更，但訊息通道結束程式不會採取明確動作-它不會變更呼叫的結束程式。

如果 *ReceiveExitsDefined* 為零，則此欄位是空值指標。

在程式設計語言不支援指標資料類型的平台上，此欄位宣告為適當長度的位元組字串。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_4，則此欄位不存在。

ReceiveExits 已定義 (MQLONG)
此欄位指定鏈中定義的通道接收結束程式數目。

它大於或等於零。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_4，則此欄位不存在。

ReceiveUser 資料 (MQCHAR32)
此通道指定通道接收結束程式使用者資料。

此資料會傳遞至 *ChannelExitParms* 參數的 *ExitData* 欄位中的通道接收結束程式 (請參閱 MQ_CHANNEL_EXIT)。

此欄位最初包含在通道定義中設定的資料。 不過，在這個 MCA 實例的生命期限內， MCA 會保留任何類型的結束程式對此欄位內容所做的任何變更，且對於這個 MCA 實例的後續結束程式呼叫 (不論類型為何) ，都會使其可見。 這適用於不同交談上的結束程式。 這類變更不會影響其他 MCA 實例所使用的通道定義。 可以使用任何字元 (包括二進位資料)。

此欄位的長度由 MQ_EXIT_DATA_LENGTH 提供。

此欄位在 WebSphere MQ for IBM i 中不相關。

如果 *Version* 小於 MQCD_VERSION_2，則此結構中不存在下列欄位。

ReceiveUserDataPtr (MQPTR)
此欄位指定第一個 *ReceiveUserData* 欄位的位址。

如果 *ReceiveExitsDefined* 大於零，則此位址是鏈中每一個通道接收結束程式的使用者資料項目清單位址。

每一個使用者資料項目都在長度為 *ExitDataLength* 的欄位中，右側以空白填補。 有 *ReceiveExitsDefined* 欄位彼此相鄰-每一個結束程式各一個。 如果定義的使用者資料項目數目小於結束程式名稱數目，則未定義的使用者資料項目會設為空白。 相反地，如果定義的使用者資料項目數大於結束程式名稱數目，則會忽略多餘的使用者資料項目，且不會呈現給結束程式。

會保留結束程式對這些值所做的任何變更。 這可讓一個結束程式將資訊傳遞至另一個結束程式。 不會對任何變更執行任何驗證，例如，二進位資料可以在必要時寫入這些欄位。

如果 *ReceiveExitsDefined* 為零，則此欄位是空值指標。

在程式設計語言不支援指標資料類型的平台上，此欄位宣告為適當長度的位元組字串。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_4，則此欄位不存在。

如果 *Version* 小於 MQCD_VERSION_5，則此結構中不存在下列欄位。

RemotePassword (MQCHAR12)
此欄位指定來自夥伴的密碼。

只有在 *ChannelType* 是 MQCHT_CLNTCONN 或 MQCHT_SVRCONN 時，此欄位才會包含有效的資訊。

- 對於 MQCHT_CLNTCONN 通道的安全結束程式，此密碼是從環境取得的密碼。 結束程式可以選擇將它傳送至伺服器上的安全結束程式。
- 對於 MQCHT_SVRCONN 通道的安全結束程式，如果沒有用戶端安全結束程式，此欄位可能包含從用戶端環境取得的密碼。 結束程式可以使用此密碼來驗證 *RemoteUserIdentifier* 中的使用者 ID。

  如果用戶端有安全結束程式，則可以從用戶端的安全流程中取得此資訊。

此欄位的長度由 MQ_PASSWORD_LENGTH 提供。 如果 *Version* 小於 MQCD_VERSION_2，則此欄位不存在。

RemoteSecurityID (MQBYTE40)
此欄位指定遠端使用者的安全 ID。

此欄位僅適用於 *ChannelType* 為 MQCHT_CLNTCONN 或 MQCHT_SVRCONN 的通道。

下列特殊值指出沒有安全 ID:

MQSID_NONE
未指定安全 ID。

欄位長度的值為二進位零。

對於 C 程式設計語言，也會定義常數 MQSID_NONE_ARRAY; 此常數具有與 MQSID_NONE 相同的值，但它是字元陣列而非字串。

這是結束程式的輸入欄位。 此欄位的長度由 MQ_SECURITY_ID_LENGTH 提供。 如果 *Version* 小於 MQCD_VERSION_6，則此欄位不存在。

如果 *Version* 小於 MQCD_VERSION_7，則此結構中不存在下列欄位。

RemoteUserID (MQCHAR12)
此欄位指定來自友機之使用者 ID 的前 12 個位元組。

有兩個欄位包含遠端使用者 ID:

- *RemoteUserIdentifier* 包含遠端使用者 ID 的前 12 個位元組，如果 ID 短於 12 個位元組，則會以空白填補。 *RemoteUserIdentifier* 可以為空白。
- *LongRemoteUserIdPtr* 指向完整遠端使用者 ID ，其長度可以超過 12 個位元組。 其長度由 *LongRemoteUserIdLength* 提供。 完整 ID 不包含尾端空白，且不是以空值結尾。 如果 ID 為空白，則 *LongRemoteUserIdLength* 為零，且未定義 *LongRemoteUserIdPtr* 的值。

  如果 *Version* 小於 MQCD_VERSION_6，則 *LongRemoteUserIdPtr* 不存在。

遠端使用者 ID 僅適用於 *ChannelType* 為 MQCHT_CLNTCONN 或 MQCHT_SVRCONN 的通道。

- 對於 MQCHT_CLNTCONN 通道上的安全結束程式，此值是從環境取得的使用者 ID。 結束程式可以選擇將它傳送至伺服器上的安全結束程式。
- 對於 MQCHT_SVRCONN 通道上的安全結束程式，如果沒有用戶端安全結束程式，則此欄位可能包含從用戶端環境取得的使用者 ID。 結束程式可能會驗證此使用者 ID (可能使用 *RemotePassword* 中的密碼) ，並更新 *MCAUserIdentifier* 中的值。

  如果用戶端有安全結束程式，則可以從用戶端的安全流程中取得此資訊。

此欄位的長度由 MQ_USER_ID_LENGTH 提供。 如果 *Version* 小於 MQCD_VERSION_2，則此欄位不存在。

SecurityExit (MQCHARn)
此欄位指定通道安全結束程式名稱。

如果此名稱非空白，則會在下列時間呼叫結束程式:

- 在建立通道之後立即進行。

  在傳送任何訊息之前，會讓結束程式有機會啟動安全流程來驗證連線授權。
- 在收到安全訊息流程的回應時。

  從遠端機器上的遠端處理器收到的任何安全訊息流程都會提供給結束程式。
- 在通道起始設定及終止時。

如需各種環境中此欄位內容的說明，請參閱第 923 頁的『MQCD-通道定義』。

此欄位的長度由 MQ_EXIT_NAME_LENGTH 提供。

註: 此常數的值是環境特定的。

SecurityUser 資料 (MQCHAR32)
此通道指定通道安全結束程式使用者資料。

此資料會傳遞至 *ChannelExitParms* 參數的 *ExitData* 欄位中的通道安全結束程式 (請參閱 MQ_CHANNEL_EXIT)。

此欄位最初包含在通道定義中設定的資料。 不過，在這個 MCA 實例的生命期限內， MCA 會保留任何類型的結束程式對此欄位內容所做的任何變更，且對於這個 MCA 實例的後續結束程式呼叫 (不論類型為何)，都會使其可見。 這適用於不同交談上的結束程式。 這類變更不會影響其他 MCA 實例所使用的通道定義。 可以使用任何字元 (包括二進位資料)。

此欄位的長度由 MQ_EXIT_DATA_LENGTH 提供。

此欄位在 WebSphere MQ for IBM i 中不相關。

SendExit (MQCHARn)
此欄位指定通道傳送結束程式名稱。

如果此名稱非空白，則會在下列時間呼叫結束程式:

- 在網路上送出資料之前立即。

  在傳輸結束程式之前，會為該結束程式提供完整的傳輸緩衝區。 可以視需要修改緩衝區的內容。
- 在通道起始設定及終止時。

如需各種環境中此欄位內容的說明，請參閱第 923 頁的『MQCD-通道定義』。

此欄位的長度由 MQ_EXIT_NAME_LENGTH 提供。

註: 此常數的值是環境特定的。

SendExitPtr (MQPTR)
此欄位指定第一個 *SendExit* 欄位的位址。

如果 *SendExitsDefined* 大於零，則此位址是鏈結中每一個通道傳送結束程式的名稱清單位址。

每一個名稱都在長度為 *ExitNameLength* 的欄位中，右側以空白填補。 有 *SendExitsDefined* 欄位彼此相鄰-每一個結束程式各一個。

會保留結束程式對這些名稱所做的任何變更，但訊息傳送結束程式不會採取明確動作-它不會變更呼叫的結束程式。

如果 *SendExitsDefined* 為零，則此欄位是空值指標。

在程式設計語言不支援指標資料類型的平台上，此欄位宣告為適當長度的位元組字串。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_4，則此欄位不存在。

SendExits 已定義 *(MQLONG)*
此欄位指定鏈中定義的通道傳送結束程式數目。

它大於或等於零。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_4，則此欄位不存在。

SendUser 資料 *(MQCHAR32)*
此欄位指定通道傳送結束程式使用者資料。

此資料會傳遞至 *ChannelExitParms* 參數的 *ExitData* 欄位中的通道傳送結束程式 (請參閱 MQ_CHANNEL_EXIT)。

此欄位最初包含在通道定義中設定的資料。 不過，在這個 MCA 實例的生命期限內， MCA 會保留任何類型的結束程式對此欄位內容所做的任何變更，且對於這個 MCA 實例的後續結束程式呼叫 (不論類型為何) ，都會使其可見。 這適用於不同交談上的結束程式。 這類變更不會影響其他 MCA 實例所使用的通道定義。 可以使用任何字元 (包括二進位資料)。

此欄位的長度由 MQ_EXIT_DATA_LENGTH 提供。

此欄位在 WebSphere MQ for IBM i 中不相關。

SendUserDataPtr (MQPTR)
此欄位指定 *SendUserData* 欄位的位址。

如果 *SendExitsDefined* 大於零，則此位址是鏈中每一個通道訊息結束程式的使用者資料項目清單位址。

每一個使用者資料項目都在長度為 *ExitDataLength* 的欄位中，右側以空白填補。 有 *MsgExitsDefined* 欄位彼此相鄰-每一個結束程式各一個。 如果定義的使用者資料項目數目小於結束程式名稱數目，則未定義的使用者資料項目會設為空白。 相反地，如果定義的使用者資料項目數大於結束程式名稱數目，則會忽略多餘的使用者資料項目，且不會呈現給結束程式。

會保留結束程式對這些值所做的任何變更。 這可讓一個結束程式將資訊傳遞至另一個結束程式。 不會對任何變更執行任何驗證，例如，二進位資料可以在必要時寫入這些欄位。

如果 *SendExitsDefined* 為零，則此欄位是空值指標。

在程式設計語言不支援指標資料類型的平台上，此欄位宣告為適當長度的位元組字串。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_4，則此欄位不存在。

SeqNumber 折返 *(MQLONG)*
此欄位指定可容許的最高訊息序號。

當達到此值時，序號會折返以從 1 重新開始。

此值不可協議，且必須同時符合本端及遠端通道定義。

此欄位與 *ChannelType* 為 MQCHT_SVRCONN 或 MQCHT_CLNTCONN 的通道無關。

SharingConversations (MQLONG)
此欄位指定可以共用與此通道相關聯之通道實例的交談數上限。

此欄位用於用戶端連線及伺服器連線通道。

值 0 表示通道在 WebSphere MQ 7.0 之前的版本中運作的方式與下列屬性相同:

- 交談共用
- 先讀
- STOP CHANNEL(<channelname>) MODE(QUIESCE)
- 活動訊號中
- 用戶端非同步取用

值 1 是 WebSphere MQ V7.0 行為的最小值。 雖然在通道實例上只容許一個交談，但可以使用先讀、非同步使用，以及 CLNTCONN-SVRCONN 活動訊號及靜止通道停止的第 7 版行為。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_9，則它不存在。

此欄位的預設值為 10。

註: *MaxInstances* 及 *MaxInstancesPerClient* 適用於通道的限制會限制通道實例數，而不是可能共用那些實例的交談數。

ShortConnection 名稱 *(MQCHAR20)*
此欄位指定連線名稱的前 20 個位元組。

如果 *Version* 欄位是 MQCD_VERSION_1， *ShortConnectionName* 會包含完整連線名稱。

如果 *Version* 欄位為 MQCD_VERSION_2 或更高版本， *ShortConnectionName* 會包含連線名稱的前 20 個字元。 完整連線名稱由 *ConnectionName* 欄位提供; *ShortConnectionName* 與 *ConnectionName* 的前 20 個字元相同。

如需此欄位內容的詳細資料，請參閱 *ConnectionName* 。

註: MQCD_VERSION_2 及 MQCD 的後續版本已變更此欄位的名稱; 該欄位先前稱為 *ConnectionName*。

此欄位的長度由 MQ_SHORT_CONN_NAME_LENGTH 提供。

ShortRetry 計數 *(MQLONG)*
此欄位指定嘗試連接遠端機器的次數上限。

此欄位是在使用 (通常更長) *LongRetryCount* 及 *LongRetryInterval* 之前，以 *ShortRetryInterval* 指定的間隔連接至遠端機器的嘗試次數上限。

此欄位僅適用於 *ChannelType* 為 MQCHT_SENDER、MQCHT_SERVER、MQCHT_CLUSSDR 或 MQCHT_CLUSRCVR 的通道。

ShortRetry 間隔 *(MQLONG)*
此欄位指定在重新嘗試連接遠端機器之前等待的秒數上限。

如果通道必須等待變成作用中，則重試之間的間隔可能會延長。

此欄位僅適用於 *ChannelType* 為 MQCHT_SENDER、MQCHT_SERVER、MQCHT_CLUSSDR 或 MQCHT_CLUSRCVR 的通道。

SSLCipherSpec (MQCHAR32)
此欄位指定使用 SSL 時正在使用的「密碼規格」。

如果 SSLCipherSpec 空白，則通道不會使用 SSL。 如果不是空白，則此欄位包含指定使用中 CipherSpec 的字串。

此參數適用於所有通道類型。 它在 AIX、 HP-UX、 Linux、 IBM i、Solaris、 Windows 及 z/OS 上受支援。 它僅適用於 TCP 傳輸類型 (TRPTYPE) 的通道類型。

這是結束程式的輸入欄位。 此欄位的長度由 MQ_SSL_CIPHER_SPEC_LENGTH 提供。 如果 *Version* 小於 MQCD_VERSION_7，則此欄位不存在。

SSLClientAuth (MQLONG)
此欄位指定是否需要 SSL 用戶端鑑別。

此欄位僅與 SVRCONN 通道定義相關。

它是下列其中一個值:

需要 MQSCA_REQUIRED
需要用戶端鑑別。

MQSCA_OPTIONAL
用戶端鑑別選用。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_7，則此欄位不存在。

SSLPeerName 長度 *(MQLONG)*
此欄位指定 *SSLPeerNamePtr* 所指向 SSL 同層級名稱的長度 (以位元組為單位)。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_7，則此欄位不存在。

SSLPeerNamePtr (MQPTR)
此欄位指定 SSL 同層級名稱的位址。

在順利完成 SSL 信號交換期間收到憑證時，憑證主旨的「識別名稱」會複製到接收憑證之通道尾端的 SSLPeerNamePtr 所存取的 MQCD 欄位中。 它會改寫通道的 SSLPeerName 值 (如果此值存在於本端使用者的通道定義中)。 如果在通道的這一端指定安全結束程式，則會從 MQCD 中的同層級憑證接收「識別名稱」。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCD_VERSION_7，則此欄位不存在。

註: 在 WebSphere MQ v7.1 之前所建構的安全結束程式應用程式可能需要更新。 如需相關資訊，請參閱 通道安全結束程式 。

StrucLength (MQLONG)
此欄位指定 MQCD 結構的長度 (以位元組為單位)。

此長度不包括由結構內包含的指標欄位所定址的任何字串。 此值是下列其中一個:

MQCD_LENGTH_4
version-4 通道定義結構的長度。

MQCD_LENGTH_5
version-5 通道定義結構的長度。

MQCD_LENGTH_6
version-6 通道定義結構的長度。

MQCD_LENGTH_7
version-7 通道定義結構的長度。

MQCD_LENGTH_8
version-8 通道定義結構的長度。

MQCD_LENGTH_9
version-9 通道定義結構的長度。

下列常數指定現行版本的長度:

MQCD_CURRENT_LENGTH
通道定義結構現行版本的長度。

註: 這些常數具有環境特定的值。

如果 *Version* 小於 MQCD_VERSION_4，則此欄位不存在。

TpName (MQCHAR64)
此欄位指定 LU 6.2 交易程式名稱。

僅當傳輸通訊協定 (*TransportType*) 為 MQXPT_LU62 且 *ChannelType* 不是 MQCHT_SVRCONN 或 MQCHT_RECEIPT 時，此欄位才相關。

在通訊端物件中包含資訊的平台上，此欄位一律為空白。

此欄位的長度由 MQ_TP_NAME_LENGTH 提供。

TransportType (MQLONG)
此欄位指定要使用的傳輸通訊協定。

如果通道是從另一端起始，則不會檢查此值。

它是下列其中一個值:

MQXPT_LU62
LU 6.2 傳輸通訊協定。

MQXPT_TCP
TCP/IP 傳輸通訊協定。

MQXPT_NETBIOS
NetBIOS 傳輸通訊協定。

下列環境支援此值: Windows。

MQXPT_SPX
SPX 傳輸通訊協定。

下列環境支援此值: Windows，加上連接至這些系統的 WebSphere MQ 用戶端。

UseDLQ (MQLONG)
此欄位指定當通道無法遞送訊息時，是否使用無法傳送郵件的佇列 (或無法遞送的訊息佇列)。

它可以包含下列其中一個值:

MQUSEDLQ_NO
通道無法遞送的訊息會被視為失敗。 根據 NPMSPEED 設定，通道會捨棄訊息或通道結束。

MQUSEDLQ_YES
當 DEADQ 佇列管理程式屬性提供無法傳送郵件的佇列名稱時，會使用它，否則行為會如「否」。 YES 是預設值。

UserIdentifier (MQCHAR12)
此欄位指定當嘗試起始與遠端訊息通道代理程式的安全 SNA 階段作業時，訊息通道代理程式所使用的使用者 ID。

此欄位只能在 UNIX 系統及 Windows 上為非空白，且僅與 *ChannelType* 為 MQCHT_SENDER、MQCHT_SERVER、MQCHT_REQUESTER 或 MQCHT_CLNTCONN 的通道相關。 在 z/OS 上，此欄位不相關。

此欄位的長度由 MQ_USER_ID_LENGTH 提供。 不過，只會使用前 10 個字元。

當 *Version* 小於 MQCD_VERSION_2 時，此欄位不存在。

版本 *(MQLONG)*
Version 欄位指定您可以為結構設定的最高版本號碼。

此值視環境而定:

MQCD_VERSION_1
第 1 版通道定義結構。

MQCD_VERSION_2
第 2 版通道定義結構。

第 2 版未由任何現行 IBM WebSphere MQ 產品使用。

MQCD_VERSION_3
第 3 版通道定義結構。

在下列環境中，第 3 版是您可以在 MQSeries 第 2 版上設定欄位的最高版本: HP Integrity NonStop Server，以及未列在其他位置的 UNIX and Linux 系統。

MQCD_VERSION_4
第 4 版通道定義結構。

第 4 版未由任何現行 IBM WebSphere MQ 產品使用。

MQCD_VERSION_5
第 5 版通道定義結構。

第 5 版是您可以在 MQSeries for OS/390 5.2 版上設定此欄位的最高版本。

MQCD_VERSION_6
第 6 版通道定義結構。

第 6 版不是任何現有 IBM WebSphere MQ 產品的現行 MQCD 結構版本。 不過，可以使用 MQCNO 結構的 ClientConnOffset 或 ClientConnPtr 欄位，將第 6 版 MQCD 結構傳遞至 MQCONNX 。

在分散式平台上，第 6 版是 MQCD_DEFAULT 和 MQCD_CLIENT_CONN_DEFAULT 起始設定程式中的預設版本。 如果您要參照 MQCD 的 MQCD_VERSION_7、 MQCD_VERSION_8 或 MQCD_VERSION_9 欄位，請視情況將 MQCD **Version** 欄位明確起始設定為 MQCD_VERSION_7、 MQCD_VERSION_8 或 MQCD_VERSION_9 。

在 z/OS 上， MQCD_VERSION_7 是預設值。

MQCD_VERSION_7
第 7 版通道定義結構。

在下列環境中，第 7 版是您可以在 IBM WebSphere MQ Version 5.3 上設定欄位的最高版本: AIX、 HP-UX、 Solaris、 Windows，以及在 IBM WebSphere MQ for z/OS Version 5.3 和 Version 5.3.1 上。 MQCD_VERSION_7 是 IBM WebSphere MQ for z/OS 版本的預設值。

MQCD_VERSION_8
第 8 版通道定義結構。

第 8 版是您可以在所有平台上的 IBM WebSphere MQ Version 6.0 上將欄位設為的最高版本。

MQCD_VERSION_9
第 9 版通道定義結構。

第 9 版是您可以在所有平台上的 IBM WebSphere MQ Version 7.0 及 IBM WebSphere MQ Version 7.0.1 上設定此欄位的最高版本。

MQCD_VERSION_10
第 10 版通道定義結構。

第 10 版是您可以在所有平台上的 IBM WebSphere MQ Version 7.1 及 IBM WebSphere MQ Version 7.5 上設定欄位的最高版本。

僅存在於較新結構版本中的欄位會在欄位說明中如此識別。 下列常數指定現行版本的版本號碼:

MQCD_CURRENT_VERSION
MQCD_CURRENT_VERSION 中設定的值是所使用通道定義結構的現行版本。

MQCD_CURRENT_VERSION 的值視環境而定。 它包含平台支援的最高值。

MQCD_CURRENT_VERSION 不是用來起始設定標頭、副本及併入檔中提供給不同程式設計語言的預設結構。 Version 的預設起始設定視平台及版次而定。

對於 IBM WebSphere MQ Version 7.0 以及更新版本，標頭、副本及併入檔中的 MQCD 宣告會起始設定為 MQCD_VERSION_6。 若要使用其他 MQCD 欄位，應用程式必須將版本號碼設為 MQCD_CURRENT_VERSION。 如果您在數個環境之間撰寫可攜的應用程式，則必須選擇所有環境中支援的版本。

提示: 引入新版本 MQCD 結構時，現有組件的佈置不會變更。 結束程式必須檢查版本號碼。 它必須等於或大於包含結束程式需要使用之欄位的最低版本。

XmitQName (MQCHAR48)
此欄位指定從中擷取訊息的傳輸佇列名稱。

此欄位僅適用於 *ChannelType* 為 MQCHT_SENDER 或 MQCHT_SERVER 的通道。

此欄位的長度由 MQ_Q_NAME_LENGTH 指定。

C 宣告
此宣告是 MQCD 結構的 C 宣告。

```
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 */
   MQLONG    NetworkPriority;           /* Network priority */
   /* 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];         /* SSL CipherSpec */
   MQPTR     SSLPeerNamePtr;            /* Address of SSL peer name */
   MQLONG    SSLPeerNameLength;         /* Length of SSL peer name */
   MQLONG    SSLClientAuth;             /* Whether SSL 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 */
 };
```

COBOL 宣告

此宣告是 MQCD 結構的 COBOL 宣告。

```
** 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-MAXMSGLENGTH 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-NONPERSISTENTMSGSPEED PIC S9(9) BINARY.
** Length of MQCD structure
 15  MQCD-STRUCLENGTH 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 **
       ** SSL CipherSpec
        15  MQCD-SSLCIPHERSPEC PIC X(32).
       ** Address of SSL peer name
        15  MQCD-SSLPEERNAMEPTR POINTER.
       ** Length of SSL peer name
        15  MQCD-SSLPEERNAMELENGTH PIC S9(9) BINARY.
       ** Whether SSL 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 **
```

RPG 宣告 (ILE)

此宣告是 MQCD 結構的 RPG 宣告。

```
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  CDDES                 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  CDRCD                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  CDLRL               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* SSL CipherSpec
     D  CDSCS               1217   1248
     D* Address of SSL peer name
     D  CDSPN               1249   1264*
     D* Length of SSL peer name
     D  CDSPL               1265   1268I 0
     D* Whether SSL 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 組譯器宣告

此宣告是 MQCD 結構的 System/390 組譯器宣告。

```
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_MAXMSGLENGTH            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_NONPERSISTENTMSGSPEED   DS   F      Speed at which nonpersistent
*                                        messages are sent
MQCD_STRUCLENGTH             DS   F      Length of MQCD structure
MQCD_EXITNAMELENGTH          DS   F      Length of exit name
MQCD_EXITDATALENGTH          DS   F      Length of exit user data
MQCD_MSGEXITSDEFINED         DS   F      Number of message exits defined
MQCD_SENDEXITSDEFINED        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   SSL CipherSpec
MQCD_SSLPEERNAMEPTR          DS   F      Address of SSL peer name
MQCD_SSLPEERNAMELENGTH       DS   F      Length of SSL peer name
MQCD_SSLCLIENTAUTH           DS   F      Whether SSL client
*                                        authentication is required
MQCD_KEEPALIVEINTERVAL       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 Affinty
MQCD_BATCHDATALIMIT          DS   F      Batch data limit
MQCD_USEDLQ                  DS   F      Use dead-letter queue
MQCD_DEFRECONNECT            DS   F      Default client reconnect option
MQCD_LENGTH                  EQU  *-MQCD
                             ORG  MQCD
MQCD_AREA                    DS   CL(MQCD_LENGTH)
```

Visual Basic 宣告

此宣告是 MQCD 結構的 Visual Basic 宣告。

在 Visual Basic 中， MQCD 結構可以與 MQCONNX 呼叫中的 MQCNO 結構搭配使用。

```
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  'SSL CipherSpec'
  SSLPeerNamePtr         As MQPTR      'Address of SSL peer name'
  SSLPeerNameLength      As Long       'Length of SSL peer name'
  SSLClientAuth          As Long       'Whether SSL 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
```

變更通道結束程式中的 *MQCD* 欄位

通道結束程式可以變更 MQCD 中的欄位。 不過，除了列出的情況之外，通常不會處理這些變更。

如果通道結束程式變更 MQCD 資料結構中的欄位， WebSphere MQ 通道處理程序通常會忽略新值。 不過，新值會保留在 MQCD 中，並傳遞至結束鏈中的任何剩餘結束程式，以及傳遞至共用通道實例的任何交談。

在 MQCXP 結構中，如果 SharingConversations 設為 FALSE ，則可以根據結束程式類型、通道類型及結束原因碼來處理特定欄位的變更。 下表顯示可以變更並影響通道行為的欄位，以及在何種情況下。 如果跳出程式在任何其他情況下變更其中一個欄位，或未列出任何欄位，則通道處理程序會忽略新值。 新值會保留在 MQCD 中，並傳遞至結束鏈中的任何剩餘結束程式，以及傳遞至共用通道實例的任何交談。

當呼叫起始設定 (MQXR_INIT) 時，只要 MQCXP SharingConverstions 設為 FALSE ，任何類型的跳出程式都可以變更任何通道類型的 ChannelName 欄位。 不論 MQCXP SharingConverstions 的值為何，只有安全結束程式才能變更 MCAUserIdentifier 欄位。

欄位	結束原因碼	結束程式類型	通道類型
ChannelName	MQXR_INIT	全部	全部
TransportType	MQXR_INIT	全部	全部
XmitQName	MQXR_INIT	全部	SDR、RCVR

欄位	結束原因碼	結束程式類型	通道類型
ModeName	MQXR_INIT	全部	全部
TpName	MQXR_INIT	全部	全部
BatchSize	MQXR_INIT	全部	SDR、 SVR、 RCVR、 RQSTR、 CLUSSDR、 CLUSRCVR
DiscInterval	MQXR_INIT	全部	SDR、 SVR、 RCVR、 RQSTR、 CLUSSDR、 CLUSRCVR
ShortRetry 計數	MQXR_INIT	全部	SDR、 SVR、 RCVR、 RQSTR、 CLUSSDR、 CLUSRCVR
ShortRetry 間隔	MQXR_INIT	全部	SDR、 SVR、 RCVR、 RQSTR、 CLUSSDR、 CLUSRCVR
LongRetry 計數	MQXR_INIT	全部	SDR、 SVR、 RCVR、 RQSTR、 CLUSSDR、 CLUSRCVR
LongRetry 間隔	MQXR_INIT	全部	SDR、 SVR、 RCVR、 RQSTR、 CLUSSDR、 CLUSRCVR
SeqNumber 折返	MQXR_INIT	全部	SDR、 SVR、 RCVR、 RQSTR、 CLUSSDR、 CLUSRCVR
MaxMsgLength	MQXR_INIT	全部	全部

欄位	結束原因碼	結束程式類型	通道類型
PutAuthority	MQXR_INIT	全部	SDR、 SVR、 RCVR、 RQSTR、 CLUSSDR、 CLUSRCVR
DataConversion	MQXR_INIT	全部	全部
MCAUserIdentifier	MQXR_INIT、MQXR_INIT_SEC、 MQXR_SEC_MSG、MQXR_SEC_PARMS	安全	RCVR、 RQSTR、 SVRCONN、 CLUSRCVR
ConnectionName	MQXR_INIT	全部	SDR、 SVR、 RQSTR、 CLNTCONN 、 CLUSSDR、 CLUSRCVR
MsgRetryUserData	MQXR_INIT	全部	RCVR、 RQSTR、 CLUSRCVR
MsgRetry 計數	MQXR_INIT	全部	RCVR、 RQSTR、 CLUSRCVR
MsgRetry 間隔	MQXR_INIT	全部	RCVR、 RQSTR、 CLUSRCVR
HeartbeatInterval	MQXR_INIT	全部	全部
BatchInterval	MQXR_INIT	全部	SDR、 SVR、 CLUSSDR、 CLUSRCVR
NonPersistentMsgSpeed	MQXR_INIT	全部	SDR、 SVR、 RCVR、 RQSTR、 CLUSSDR、 CLUSRCVR
MCASecurityId	MQXR_INIT、MQXR_INIT_SEC、 MQXR_SEC_MSG、MQXR_SEC_PARMS	安全	SDR、 SVR、 RCVR、 RQSTR、 SVRCONN、 CLUSSDR、 CLUSRCVR
SSLCipherSpec	MQXR_INIT	全部	全部
SSLPeerNamePtr	MQXR_INIT	全部	全部
SSLPeerName 長度	MQXR_INIT	全部	全部

欄位	結束原因碼	結束程式類型	通道類型
SSLClientAuth	MQXR_INIT	全部	SVR、RCVR、RQSTR、SVRCONN、CLUSRCVR
KeepAliveInterval	MQXR_INIT	全部	全部
LocalAddress	MQXR_INIT	全部	SDR、SVR、RQSTR、CLNTCONN、CLUSSDR、CLUSRCVR
BatchHeartbeat	MQXR_INIT	全部	SDR、SVR、CLUSSDR、CLUSRCVR
HdrComp 清單	MQXR_INIT	全部	全部
MsgComp 清單	MQXR_INIT	全部	全部
ChannelMonitoring	MQXR_INIT	全部	SDR、SVR、RCVR、RQSTR、SVRCONN、CLUSSDR、CLUSRCVR
ChannelStatistics	MQXR_INIT	全部	SDR、SVR、RCVR、RQSTR、CLUSSDR、CLUSRCVR
SharingConversations	MQXR_INIT	全部	SVRCONN、CLNTCONN
PropertyControl	MQXR_INIT	全部	SDR、SVR、CLUSSDR、CLUSRCVR

MQCXP-通道結束程式參數

MQCXP 結構會傳遞至「訊息通道代理程式 (MCA)」、用戶端連線通道或伺服器連線通道所呼叫的每一種結束程式類型。

請參閱 MQ_CHANNEL_EXIT。

當結束程式將控制權交還給通道時，通道會忽略下列說明中說明為「結束程式的輸入」的欄位。 通道結束參數區塊中結束程式所變更的任何輸入欄位，在下次呼叫時將不會保留。 對輸入/輸出欄位 (例如 *ExitUserArea* 欄位) 所做的變更，只會針對該結束程式實例的呼叫而保留。 這類變更無法用來在相同通道上定義的不同結束程式之間，或在不同通道上定義的相同結束程式之間傳遞資料。

相關參考

第 959 頁的『欄位』
本主題列出 MQCXP 結構中的所有欄位，並說明每一個欄位。

第 968 頁的『C 宣告』
此宣告是 MQCXP 結構的 C 宣告。

第 968 頁的『COBOL 宣告』
此宣告是 MQCXP 結構的 COBOL 宣告。

第 969 頁的『RPG 宣告 (ILE)』
此宣告是 MQCXP 結構的 RPG 宣告。

第 970 頁的『 System/390 組譯器宣告』
此宣告是 MQCXP 結構的 System/390 組譯器宣告。

欄位

本主題列出 MQCXP 結構中的所有欄位，並說明每一個欄位。

StrucId (MQCHAR4)
此欄位指定結構 ID。

值必須為:

MQCXP_STRUC_ID
通道結束程式參數結構的 ID。

對於 C 程式設計語言，也會定義常數 MQCXP_STRUC_ID_ARRAY; 此常數與 MQCXP_STRUC_ID 具有相同的值，但它是字元陣列而非字串。

這是結束程式的輸入欄位。

版本 *(MQLONG)*
此欄位指定結構版本號碼。

此值視環境而定:

MQCXP_VERSION_1
Version-1 通道結束程式參數結構。

MQCXP_VERSION_2
Version-2 通道結束程式參數結構。

在下列環境中，欄位具有此值: HP Integrity NonStop Server。

MQCXP_VERSION_3
Version-3 通道結束程式參數結構。

此欄位在下列環境中具有此值: UNIX 系統未列在其他位置。

MQCXP_VERSION_4
Version-4 通道結束程式參數結構。

MQCXP_VERSION_5
Version-5 通道結束程式參數結構。

MQCXP_VERSION_6
Version-6 通道結束程式參數結構。

MQCXP_VERSION_8
Version-8 通道結束程式參數結構。

在下列環境中，此欄位具有此值: z/OS、 AIX、 HP-UX、 Linux、 IBM i、Solaris、 Windows。

僅存在於結構最新版本中的欄位會在欄位說明中如此識別。 下列常數指定現行版本的版本號碼:

MQCXP_CURRENT_VERSION
通道結束程式參數結構的現行版本。

此值視環境而定。

註: 引進新版 MQCXP 結構時，現有組件的佈置不會變更。 因此，結束程式必須檢查版本號碼是否等於或大於包含結束程式需要使用之欄位的最低版本。

這是結束程式的輸入欄位。

ExitId (MQLONG)
此欄位指定要呼叫的結束程式類型，並在進入結束常式時設定。

下列為可能的值:

MQXT_CHANNEL_SEC_EXIT
通道安全結束程式。

MQXT_CHANNEL_MSG_EXIT
通道訊息結束程式。

MQXT_CHANNEL_SEND_EXIT
通道傳送結束程式。

MQXT_CHANNEL_RCV_EXIT
通道接收結束程式。

MQXT_CHANNEL_MSG_RETRY_EXIT
通道訊息-重試結束程式。

MQXT_CHANNEL_AUTO_DEF_EXIT
通道自動定義結束程式。

在 z/OS 上，只有 MQCHT_CLUSSDR 及 MQCHT_CLUSRCVR 類型的通道才支援這種類型的結束程式。

這是結束程式的輸入欄位。

ExitReason (MQLONG)
此欄位指定呼叫結束程式的原因，並在進入結束常式時設定。

自動定義結束程式不會使用它。 下列為可能的值:

MQXR_INIT
結束起始設定。

此值指出第一次呼叫結束程式。 它容許結束程式獲得並起始設定它需要的任何資源 (例如: 記憶體)。

MQXR_TERM
結束終止。

此值指出即將終止結束程式。 結束程式應該會釋放自起始設定以來所獲得的任何資源 (例如: 記憶體)。

MQXR_MSG
處理訊息。

此值指出正在呼叫結束程式來處理訊息。 只有通道訊息結束程式才會出現這個值。

MQXR_XMIT
處理傳輸。

只有通道傳送及接收結束程式才會出現此值。

MQXR_SEC_MSG
收到安全訊息。

只有通道安全結束程式才會出現這個值。

MQXR_INIT_SEC
啟動安全交換。

只有通道安全結束程式才會出現這個值。

在使用 MQXR_INIT 呼叫之後，一律會立即以這個原因來呼叫接收端的安全結束程式，讓它有機會起始安全交換。 如果它拒絕商機 (透過傳回 MQXCC_SEND_SEC_MSG 或 MQXCC_SEND_AND_REQUEST_SEC_MSG)，則會以 MQXR_INIT_SEC 來呼叫傳送端的安全結束程式。

如果接收端的安全結束程式確實起始安全交換 (傳回 MQXCC_SEND_SEC_MSG 或 MQXCC_SEND_AND_REQUEST_SEC_MSG)，則傳送端的安全結束程式絕不會以 MQXR_INIT_SEC; 而是以 MQXR_SEC_MSG 來呼叫它，以處理接收端的訊息。 (在任一情況下，會先使用 MQXR_INIT 來呼叫它。)

除非其中一個安全結束程式要求終止通道 (透過將 *ExitResponse* 設為 MQXCC_SUPPRESS_FUNCTION 或 MQXCC_CLOSE_CHANNEL)，否則安全交換必須在起始交換的一端完成。 因此，如果使用 MQXR_INIT_SEC 呼叫安全結束程式，且它確實起始交換，則下次呼叫該結束程式時，它會使用 MQXR_SEC_MSG。 不論是否有結束程式要處理的安全訊息，都會發生這種情況。 如果友機傳回 MQXCC_SEND_SEC_MSG 或 MQXCC_SEND_AND_REQUEST_SEC_MSG，但如果友機傳回 MQXCC_OK 或友機沒有安全結束程式，則會出現安全訊息。 如果沒有要處理的安全訊息，則會以零的 *DataLength* 重新呼叫起始端的安全結束程式。

MQXR_RETRY
重試訊息。

只有 message-retry 結束程式才會出現這個值。

MQXR_AUTO_CLUSSDR
自動定義叢集傳送端通道。

只有通道自動定義結束程式才會出現此值。

MQXR_AUTO_RECEIVER
自動定義接收端通道。

只有通道自動定義結束程式才會出現此值。

MQXR_AUTO_SVRCONN
自動定義伺服器連線通道。

只有通道自動定義結束程式才會出現此值。

MQXR_AUTO_CLUSRCVR
自動定義叢集接收端通道。

只有通道自動定義結束程式才會出現此值。

MQXR_SEC_PARMS
安全參數

此值僅適用於安全結束程式，並指出正在將 MQCSP 結構傳遞至結束程式。 如需相關資訊，請參閱第 295 頁的『MQCSP-安全參數』。

註:

1. 如果您為通道定義了多個結束程式，當 MCA 起始設定時，會以 MQXR_INIT 來呼叫它們。 此外，當 MCA 終止時，也會以 MQXR_TERM 來呼叫它們。
2. 對於通道自動定義結束程式，如果 *Version* 小於 MQCXP_VERSION_4，則不會設定 *ExitReason*。 在此情況下，值 MQXR_AUTO_SVRCONN 是隱含的。

這是結束程式的輸入欄位。

ExitResponse (MQLONG)
此欄位指定結束程式的回應。

此欄位由結束程式設定，以與 MCA 進行通訊。它必須是下列其中一個值:

MQXCC_OK
已順利完成結束。

- 對於通道安全結束程式，此值指出訊息傳送現在可以正常進行。
- 對於通道訊息重試結束程式，此值指出 MCA 必須等待結束程式在 MQCXP 的 *MsgRetryInterval* 欄位中傳回的時間間隔，然後重試訊息。

ExitResponse2 欄位可能包含其他資訊。

MQXCC_SUPPRESS_FUNCTION
抑制函數。

- 對於通道安全結束程式，此值指出必須終止通道。
- 對於通道訊息結束程式，此值指出訊息不會進一步朝向其目的地繼續進行。 相反地， MCA 會產生異常狀況報告訊息 (如果原始訊息的傳送者要求的話)，並將原始緩衝區中包含的訊息置於無法傳送郵件的佇列 (如果傳送者指定 MQRO_DEAD_LETTER_Q)，或捨棄它 (如果傳送者指定 MQRO_DISCARD_MSG)。

  對於持續訊息，如果傳送端指定 MQRO_DEAD_LETTER_Q，但放置至無法傳送郵件的佇列失敗，或沒有無法傳送郵件的佇列，則原始訊息會留在傳輸佇列中，且不會產生報告訊息。 如果無法順利產生報告訊息，則原始訊息也會留在傳輸佇列中。

  無法傳送郵件的佇列上訊息開頭的 MQDLH 結構中的 *Feedback* 欄位指出訊息放置在無法傳送郵件的佇列上的原因; 此回饋碼也用於異常狀況報告訊息的訊息描述子中 (如果傳送端有要求的話)。
- 對於通道訊息重試結束程式，此值指出 MCA 不等待並重試訊息; 相反地， MCA 會立即繼續其正常失敗處理程序 (訊息會放置在無法傳送郵件的佇列上或被捨棄，如訊息傳送端所指定)。
- 對於通道自動定義結束程式，必須指定 MQXCC_OK 或 MQXCC_SUPPRESS_FUNCTION。 如果這兩個值都未指定，依預設會採用 MQXCC_SUPPRESS_FUNCTION，且會放棄自動定義。

通道傳送及接收結束程式不支援此回應。

MQXCC_SEND_SEC_MSG
傳送安全訊息。

此值只能由通道安全結束程式設定。 它指出結束程式已提供必須傳輸至夥伴的安全訊息。

MQXCC_SEND_AND_REQUEST_SEC_MSG
傳送需要回覆的安全訊息。

此值只能由通道安全結束程式設定。 它指出

- 該結束程式已提供可傳輸至夥伴的安全訊息，以及
- 結束程式需要夥伴的回應。 如果未收到任何回應，則必須終止通道，因為結束程式尚未決定是否可以繼續進行通訊。

MQXCC_SUPPRESS_EXIT
暫停結束。

- 此值可以由安全結束程式或自動定義結束程式以外的所有通道結束程式類型設定。 它會暫停該結束程式的任何進一步呼叫 (如同其名稱在通道定義中是空白的一樣)，直到通道終止為止，當再次以 MQXR_TERM 的 *ExitReason* 呼叫該結束程式時。
- 如果訊息重試結束程式傳回此值，則後續訊息的訊息重試正常由 *MsgRetryCount* 及 *MsgRetryInterval* 通道屬性控制。 對於目前訊息， MCA 會依 *MsgRetryInterval* 通道屬性所提供的間隔，執行未完成的重試次數，但只有在原因碼是 MCA 通常會重試的原因碼時 (請參閱 第 923 頁的『MQCD-通道定義』中說明的 *MsgRetryCount* 欄位)。 未完成的重試次數是 *MsgRetryCount* 屬性的值，減去結束程式針對現行訊息傳回 MQXCC_OK 的次數; 如果此數目為負數，則 MCA 不會對現行訊息執行進一步重試。

MQXCC_CLOSE_CHANNEL
關閉通道。

此值可以由任何類型的通道結束程式設定，但自動定義結束程式除外。

如果未啟用共用交談，則此值會關閉通道。

如果啟用共用交談，則此值會結束交談。 如果此交談是頻道上的唯一交談，則頻道也會關閉。

此欄位是來自結束程式的輸入/輸出欄位。

ExitResponse2 (MQLONG)
此欄位指定來自結束程式的次要回應。

此欄位在進入結束常式時設為零。結束程式可以設定它，以提供進一步資訊給 WebSphere MQ 通道功能。自動定義結束程式不會使用它。

結束程式可以設定下列一或多個值。 如果需要多個值，則會新增這些值。 會記錄無效的組合; 容許其他組合。

MQXR2_PUT_WITH_DEF_ACTION
放置預設動作。

此值是由接收端的通道訊息結束程式所設定。 它指出以 MCA 的預設動作 (即 MCA 的預設使用者 ID) 或訊息 MQMD (訊息描述子) 中的環境定義 *UserIdentifier* 來放置訊息。

值為零，對應於呼叫結束程式時所設定的起始值。 常數是為了文件目的而提供。

MQXR2_PUT_WITH_DEF_USERID
放置預設使用者 ID。

此值只能由接收端的通道訊息結束程式設定。 它指出要以 MCA 的預設使用者 ID 來放置訊息。

MQXR2_PUT_WITH_MSG_USERID
放置訊息的使用者 ID。

此值只能由接收端的通道訊息結束程式設定。 它指出將訊息與環境定義 *UserIdentifier* 一起放置在訊息的 MQMD (訊息描述子) 中 (結束程式可能已修改此訊息)。

只應該設定 MQXR2_PUT_WITH_DEF_ACTION 其中之一。

MQXR2_USE_AGENT_BUFFER
使用代理程式緩衝區。

此值指出要傳遞的任何資料都是在 *AgentBuffer* 中，而不是在 *ExitBufferAddr* 中。

值為零，對應於呼叫結束程式時所設定的起始值。 常數是為了文件目的而提供。

MQXR2_USE_EXIT_BUFFER
使用結束緩衝區。

此值指出要傳遞的任何資料都是在 *ExitBufferAddr* 中，而不是在 *AgentBuffer* 中。

只應該設定 MQXR2_USE_AGENT_BUFFER 和 MQXR2_USE_EXIT_BUFFER 其中之一。

MQXR2_DEFAULT_CONTINUATION
預設接續。

與鏈中的下一個結束程式的接續取決於所呼叫的最後一個結束程式的回應:

- 如果傳回 MQXCC_SUPPRESS_FUNCTION 或 MQXCC_CLOSE_CHANNEL，則不會呼叫鏈中的進一步結束程式。
- 否則，會呼叫鏈中的下一個結束程式。

MQXR2_CONTINUE_CHAIN
繼續下一個結束程式。

MQXR2_SUPPRESS_CHAIN
跳過鏈中的剩餘結束程式。

這是結束程式的輸入/輸出欄位。

回饋 *(MQLONG)*
此欄位指定回饋碼。

進入結束常式時，此欄位會設為 MQFB_NONE。

如果通道訊息結束程式將 *ExitResponse* 欄位設為 MQXCC_SUPPRESS_FUNCTION，則 *Feedback* 欄位會指定回饋碼，以識別訊息放置在無法傳送郵件 (無法遞送的訊息) 佇列的原因，並在已要求時用來傳送異常狀況報告。 在此情況下，如果 *Feedback* 欄位是 MQFB_NONE，則會使用下列回饋碼:

MQFB_STOPPED_BY_MSG_EXIT
通道訊息結束程式已停止訊息。

MCA 不會使用通道安全、傳送、接收及訊息重試結束程式在此欄位中傳回的值。

如果 *ExitResponse* 是 MQXCC_OK，則不會使用自動定義結束程式在此欄位中傳回的值，否則會用於事件訊息中的 *AuxErrorDataInt1* 參數。

這是來自結束程式的輸入/輸出欄位。

MaxSegment 長度 *(MQLONG)*
此欄位指定可在單一傳輸中傳送的長度上限 (以位元組為單位)。

自動定義結束程式不會使用它。 通道傳送結束程式感興趣，因為此結束程式必須確保不會將傳輸區段大小增加到大於 *MaxSegmentLength* 的值。 長度包括結束程式不得變更的起始 8 個位元組。 起始通道時，會在 WebSphere MQ 通道功能之間協議此值。 如需區段長度的相關資訊，請參閱 撰寫通道結束程式 。

如果 *ExitReason* 是 MQXR_INIT，則此欄位中的值沒有意義。

這是結束程式的輸入欄位。

ExitUser 區域 *(MQBYTE16)*
此欄位指定結束程式使用者區域-可供結束程式使用的欄位。

在第一次呼叫結束程式 (將 *ExitReason* 設為 MQXR_INIT) 之前，它會起始設定為二進位零，之後結束程式對這個欄位所做的任何變更都會在呼叫結束程式時保留。

下列是已定義的值:

MQXUA_NONE
沒有使用者資訊。

欄位長度的值為二進位零。

對於 C 程式設計語言，也會定義常數 MQXUA_NONE_ARRAY; 此常數具有與 MQXUA_NONE 相同的值，但它是字元陣列而非字串。

此欄位的長度由 MQ_EXIT_USER_AREA_LENGTH 指定。 這是結束程式的輸入/輸出欄位。

ExitData (MQCHAR32)
此欄位指定結束程式資料。

進入結束常式時，會將這個欄位設為 WebSphere MQ 通道函數從通道定義中取得的資訊。如果沒有這類資訊可用，則此欄位全為空白。

此欄位的長度由 MQ_EXIT_DATA_LENGTH 提供。

這是結束程式的輸入欄位。

如果 *Version* 小於 MQCXP_VERSION_2，則此結構中不存在下列欄位。

MsgRetry 計數 *(MQLONG)*
此欄位指定已重試訊息的次數。

第一次針對特定訊息呼叫結束程式時，此欄位的值為零 (尚未嘗試重試)。每次後續呼叫該訊息的結束程式時， MCA 都會將值加 1。

這是結束程式的輸入欄位。 如果 *ExitReason* 是 MQXR_INIT，則此欄位中的值沒有意義。 如果 *Version* 小於 MQCXP_VERSION_2，則此欄位不存在。

MsgRetry 間隔 *(MQLONG)*
此欄位指定重試放置作業之前的間隔下限 (毫秒)。

第一次針對特定訊息呼叫結束程式時，此欄位包含 *MsgRetryInterval* 通道屬性的值。結束程式可以保留值不變，或修改它以指定不同的時間間隔 (毫秒)。 如果結束程式在 *ExitResponse* 中傳回 MQXCC_OK，則 MCA 會等待至少此時間間隔，然後重試 MQOPEN 或 MQPUT 作業。 指定的時間間隔必須為零或大於零。

針對該訊息呼叫結束程式的第二次及後續時間，此欄位包含先前呼叫結束程式所傳回的值。

如果 *MsgRetryInterval* 欄位中傳回的值小於零或大於 999 999 999 999，且 *ExitResponse* 是 MQXCC_OK，則 MCA 會忽略 MQCXP 中的 *MsgRetryInterval* 欄位，並改為等待 *MsgRetryInterval* 通道屬性指定的間隔。

這是結束程式的輸入/輸出欄位。 如果 *ExitReason* 是 MQXR_INIT，則此欄位中的值沒有意義。 如果 *Version* 小於 MQCXP_VERSION_2，則此欄位不存在。

MsgRetry 原因 *(MQLONG)*
此欄位指定前一次嘗試放置訊息的原因碼。

此欄位是前次嘗試放置訊息的原因碼; 它是其中一個 MQRC_ * 值。

這是結束程式的輸入欄位。 如果 *ExitReason* 是 MQXR_INIT，則此欄位中的值沒有意義。 如果 *Version* 小於 MQCXP_VERSION_2，則此欄位不存在。

如果 *Version* 小於 MQCXP_VERSION_3，則此結構中不存在下列欄位。

HeaderLength (MQLONG)
此欄位指定標頭資訊的長度。

此欄位僅與訊息結束程式及訊息重試結束程式相關。此值是訊息資料開頭的遞送標頭結構長度; 這些是 MQXQH 結構、MQMDE (訊息說明延伸標頭)，以及 (針對配送清單訊息) MQDH 結構及 MQOR 及 MQPMR 記錄陣列遵循 MQXQH 結構。

訊息結束程式可以檢查此標頭資訊，並在必要時修改它，但結束程式傳回的資料仍必須是正確的格式。 例如，即使接收端的訊息結束程式進行補償變更，也不能在傳送端加密或壓縮標頭資料。

如果訊息結束程式以變更其長度的方式修改標頭資訊 (例如，將另一個目的地新增至配送清單訊息)，則必須在傳回之前相應地變更 *HeaderLength* 的值。

這是結束程式的輸入/輸出欄位。 如果 *ExitReason* 是 MQXR_INIT，則此欄位中的值沒有意義。 如果 *Version* 小於 MQCXP_VERSION_3，則此欄位不存在。

PartnerName (MQCHAR48)
此欄位指定夥伴的名稱。

夥伴的名稱，如下所示:

- 對於 SVRCONN 通道，它是在用戶端登入的使用者 ID。
- 對於所有其他類型的通道，它是友機的佇列管理程式名稱。

當起始設定結束程式時，這個欄位會空白，因為在起始協議完成之後，佇列管理程式才會知道友機的名稱。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCXP_VERSION_3，則此欄位不存在。

FAPLevel (MQLONG)
協議格式及通訊協定層次。

這是結束程式的輸入欄位。 對此欄位的變更僅應在 IBM 服務的指示下進行。如果 *Version* 小於 MQCXP_VERSION_3，則此欄位不存在。

CapabilityFlags (MQLONG)
此欄位指定功能旗標。

下列是已定義的:

MQCF_NONE
沒有旗標。

MQCF_DIST_LISTS
支援的配送清單。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCXP_VERSION_3，則此欄位不存在。

ExitNumber (MQLONG)
此欄位指定結束程式的序數。

結束程式在 *ExitId* 中定義的類型內的序數。例如，如果所呼叫的結束程式是所定義的第三個訊息結束程式，則此欄位包含值 3。 如果結束程式類型是無法定義結束程式清單的類型 (例如，安全結束程式)，則此欄位具有值 1。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCXP_VERSION_3，則此欄位不存在。

如果 *Version* 小於 MQCXP_VERSION_5，則此結構中不存在下列欄位。

ExitSpace (MQLONG)
此欄位指定傳輸緩衝區中保留供結束程式使用的位元組數。

此欄位僅與傳送結束程式相關。它指定 WebSphere MQ 通道功能在傳輸緩衝區中保留給結束程式使用的空間量 (以位元組為單位)。 此欄位可讓結束程式將少量資料 (通常不超過數百個位元組) 新增至傳輸緩衝區，以供另一端的互補接收結束程式使用。 傳送結束程式所新增的資料必須由接收結束程式移除。

在 z/OS 上，此值一律為零。

註: 此機能不得用來傳送大量資料，因為它可能會降低效能，甚至會禁止通道作業。

透過設定 *ExitSpace* ，可保證在傳輸緩衝區中一律至少有該位元組數可供結束程式使用。 不過，如果傳輸緩衝區中有可用空間，則結束程式可以使用小於保留的數量，或大於保留的數量。 緩衝區中的結束空間是在現有資料之後提供。

只有在 *ExitReason* 具有值 MQXR_INIT; 在所有其他情況下，會忽略結束程式所傳回的值時，結束程式才能設定 *ExitSpace* 。 在結束程式的輸入上， MQXR_INIT 呼叫的 *ExitSpace* 是零，在其他情況下是 MQXR_INIT 呼叫所傳回的值。

如果 MQXR_INIT 呼叫所傳回的值是負數，或在為鏈中的所有傳送結束程式保留所要求的結束空間之後，在訊息資料的傳輸緩衝區中可用的位元組數少於 1024 ，則 MCA 會輸出錯誤訊息並關閉通道。 同樣地，如果在資料傳送期間，傳送結束鏈中的結束程式所配置的使用者空間比它們所保留的還多，因此在訊息資料的傳輸緩衝區中保留的位元組數少於 1024 個，則 MCA 會輸出錯誤訊息並關閉通道。 1024 的限制容許由傳送結束程式鏈處理通道的控制及管理流程，而不需要分段流程。

如果 *ExitReason* 是 MQXR_INIT ，則這是結束程式的輸入/輸出欄位，在所有其他情況下則是輸入欄位。 如果 *Version* 小於 MQCXP_VERSION_5，則此欄位不存在。

SSLCertUserID (MQCHAR12)
此欄位指定與遠端憑證相關聯的 UserId 。

在 z/OS 以外的所有平台上都是空白

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCXP_VERSION_6，則此欄位不存在。

SSLRemCertIssName 長度 *(MQLONG)*
此欄位指定 SSLCertRemoteIssuerNamePtr 所指向之遠端憑證發證者的完整「識別名稱」(以位元組為單位) 長度。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCXP_VERSION_6，則此欄位不存在。 如果不是 SSL 通道，則此值為零。

SSLRemCertIssNamePtr (PMQVOID)
此欄位指定遠端憑證發證者的完整「識別名稱」位址。

如果它不是 SSL 通道，則其值是空值指標。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCXP_VERSION_6，則此欄位不存在。

註: 在 WebSphere MQ v7.1 版本中，通道安全結束程式在判斷「主旨識別名稱」及「發證者識別名稱」時的行為已變更。 如需相關資訊，請參閱 通道安全結束程式。

SecurityParms (PMQCSP)
此欄位指定用來指定使用者 ID 及密碼的 MQSCP 結構位址。

此欄位的起始值是空值指標。

這是結束程式的輸入/輸出欄位。 如果 *Version* 小於 MQCXP_VERSION_6，則此欄位不存在。

CurHdr 壓縮 *(MQLONG)*
此欄位指定目前用來壓縮標頭資料的技術。

它設為下列其中一項:

MQCOMPRESS_NONE
不執行標頭資料壓縮。

MQCOMPRESS_SYSTEM
執行標頭資料壓縮。

此值可以透過傳送通道的訊息結束程式變更為從 MQCD 的 HdrComp 清單欄位存取的其中一個協議支援值。這會根據訊息的內容，啟用用來壓縮要為每一個訊息選擇之標頭資料的技術。 變更的值僅用於現行訊息。如果屬性變更為不受支援的值，則通道會結束。 如果在傳送通道的訊息結束程式之外變更此值，則會忽略此值。

這是結束程式的輸入/輸出欄位。 如果 *Version* 小於 MQCXP_VERSION_6，則此欄位不存在。

CurMsg 壓縮 (MQLONG)
此欄位指定目前用來壓縮訊息資料的技術。

它設為下列其中一項:

MQCOMPRESS_NONE
不執行標頭資料壓縮。

MQCOMPRESS_RLE
使用執行長度編碼來執行訊息資料壓縮。

MQCOMPRESS_ZLIBFAST
訊息資料壓縮是使用 zlib 壓縮技術來執行。 建議使用快速壓縮時間。

MQCOMPRESS_ZLIBHIGH
訊息資料壓縮是使用 zlib 壓縮技術來執行。 建議使用高階壓縮。

此值可以透過傳送通道的訊息結束程式變更為從 MQCD 的 MsgComp 清單欄位存取的其中一個協議支援值。這可讓您根據訊息的內容來決定每一個訊息的訊息資料壓縮技術。 變更的值僅用於現行訊息。 如果屬性變更為不受支援的值，則通道會結束。 如果在傳送通道的訊息結束程式之外變更此值，則會忽略此值。

這是結束程式的輸入/輸出欄位。 如果 *Version* 小於 MQCXP_VERSION_6，則此欄位不存在。

Hconn (MQHCONN)
此欄位指定結束程式需要在結束程式內進行任何 MQI 呼叫時，所使用的連線控點。

此欄位與在用戶端連線通道上執行的結束程式無關，其中包含值 MQHC_UNUSABLE_HCONN (-1)。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCXP_VERSION_7，則此欄位不存在。

SharingConversations (MQBOOL)
此欄位指定交談是否是目前可以在此通道實例上執行的唯一交談，或目前是否可以在此通道實例上執行多個交談。

它也指出結束程式是否受到同時執行的另一個結束程式變更 MQCD 的風險。

此欄位僅與在用戶端連線或伺服器連線通道上執行的結束程式相關。

它設為下列其中一項:

FALSE
結束實例是目前可以在此通道實例上執行的唯一結束實例。 這可讓結束程式安全地更新 MQCD 欄位，而不會與其他通道實例上執行的其他結束程式競爭。 通道是否處理 MQCD 欄位的變更，是由 第 955 頁的『變更通道結束程式中的 MQCD 欄位』中 MQCD 欄位的表格所定義。

TRUE
結束程式實例不是目前可以在此通道實例上執行的唯一結束程式實例。 通道不會處理對 MQCD 所做的任何變更，但 第 955 頁的『變更通道結束程式中的 MQCD 欄位』 中 MQCD 欄位的表格所列出的變更除外，因為 MQXR_INIT 以外的「結束原因」。 如果此結束程式更新 MQCD 欄位，請在此通道實例上執行的結束程式之間提供序列化，以確保在其他交談上同時執行的其他結束程式沒有競爭。

這是結束程式的輸入欄位。 如果 *Version* 小於 MQCXP_VERSION_7，則此欄位不存在。

MCAUserSource (MQLONG)
此欄位指定所提供 MCA 使用者 ID 的來源。

它可以包含下列其中一個值:

MQUSRC_MAP
使用者 ID 指定在 MCAUSER 屬性中。

MQUSRC_CHANNEL
使用者 ID 從入埠友機傳送，或在通道物件中定義的 MCAUSER 欄位中指定。

這是結束程式的輸入欄位。 如果版本小於 MQCXP_VERSION_8，則此欄位不存在。

pEntry 點 *(PMQIEP)*
此欄位指定 MQI 或 DCI 呼叫的介面進入點位址。

如果 *Version* 小於 MQCXP_VERSION_8，則此欄位不存在。

C 宣告

此宣告是 MQCXP 結構的 C 宣告。

```
typedef struct tagMQCXP MQCXP;
struct tagMQCXP {
  MQCHAR4   StrucId;           /* Structure identifier */
  MQLONG    Version;           /* Structure version number */
  MQLONG    ExitId;            /* Type of exit */
  MQLONG    ExitReason;        /* Reason for invoking exit */
  MQLONG    ExitResponse;      /* Response from exit */
  MQLONG    ExitResponse2;     /* Secondary response from exit */
  MQLONG    Feedback;          /* Feedback code */
  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 SSL certificate */
  MQLONG    SSLRemCertIssNameLength;   /* Length of
                                  distinguished name of issuer
                                  of remote SSL certificate */
  MQPTR     SSLRemCertIssNamePtr;      /* Address of
                                  distinguished name of issuer
                                  of remote SSL certificate */
  PMQVOID   SecurityParms;     /* Security parameters */
  MQLONG    CurHdrCompression; /* Header data compression
                                  used for current message */
  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 */
;
```

COBOL 宣告

此宣告是 MQCXP 結構的 COBOL 宣告。

```
**   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 SSL
** certificate
   15  MQCXP-SSLREMCERTISSNAMELENGTH PIC S9(9) BINARY.
** Address of distinguished name of issuer of remote SSL
** certificate
   15  MQCXP-SSLREMCERTISSNAMEPTR    POINTER.
** Security parameters
   15  MQCXP-SECURITYPARMS           PIC S9(18) BINARY.
** Header data compression used for current message
   15  MQCXP-CURHDRCOMPRESSION       PIC S9(9) BINARY.
** Message data compression used for current message
   15  MQCXP-CURMSGCOMPRESSION       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.
```

RPG 宣告 (ILE)

此宣告是 MQCXP 結構的 RPG 宣告。

```
     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 SSL certificate
     D  CXSSLCU              161    172
     D* Length of distinguished name of issuer of remote SSL certificate
     D  CXSRCINL             173    176I 0
     D* Address of distinguished name of issuer of remote SSL 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
```

System/390 組譯器宣告

此宣告是 MQCXP 結構的 System/390 組譯器宣告。

```
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_EXITSPACE         DS   F     Number of bytes in transmission
*                                  buffer reserved for exit to use
MQCXP_SSLCERTUSERID     DS   CL12  User identifier associated with
*                                  remote SSL certificate
MQCXP_SSLREMCERTISSNAMELENGTH   DS   F     Length of distinguished name
*                                  of issuer of remote SSL certificate
MQCXP_SSLREMCERTISSNAMEPTR      DS   F     Address of distinguished name
*                                  of issuer of remote SSL 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_LENGTH            EQU  *-MQCXP
                        ORG  MQCXP
MQCXP_AREA              DS   CL(MQCXP_LENGTH)
```

MQXWD-結束程式等待描述子

MQXWD 結構是 MQXWAIT 呼叫上的輸入/輸出參數。

此結構僅在 z/OS 上受支援。

相關參考

第 971 頁的『欄位』
本主題列出 MQXWD 結構中的所有欄位，並說明每一個欄位。

第 972 頁的『C 宣告』
此宣告是 MQXWD 結構的 C 宣告。

第 972 頁的『 System/390 組譯器宣告』
此宣告是 MQXWD 結構的 System/390 組譯器宣告。

欄位

本主題列出 MQXWD 結構中的所有欄位，並說明每一個欄位。

StrucId (MQCHAR4)
此欄位指定結構 ID。

值必須為:

MQXWD_STRUC_ID
結束等待描述子結構的 ID。

對於 C 程式設計語言，也會定義常數 MQXWD_STRUC_ARRAY; 此常數與 MQXWD_STRUC_ID 具有相同的值，但它是字元陣列而非字串。

此欄位的起始值是 MQXWD_STRUC_ID。

版本 *(MQLONG)*
此欄位指定結構版本號碼。

值必須為:

MQXWD_VERSION_1
結束程式等待描述子結構的版本號碼。

此欄位的起始值為 MQXWD_VERSION_1。

Reserved1 (MQLONG)
此欄位已保留。 其值必須為零。

這是輸入欄位。

Reserved2 (MQLONG)
此欄位已保留。 其值必須為零。

這是輸入欄位。

Reserved3 (MQLONG)
此欄位已保留。 其值必須為零。

這是輸入欄位。

ECB (MQLONG)
此欄位指定要等待的事件控制區塊。

此欄位是要等待的事件控制區塊 (ECB)。在發出 MQXWAIT 呼叫之前，必須將它設為零; 順利完成時，它會包含郵遞區號。

此欄位是輸入/輸出欄位。

C 宣告
此宣告是 MQXWD 結構的 C 宣告。

```
typedef struct tagMQXWD MQXWD;
struct tagMQXWD {
  MQCHAR4  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 組譯器宣告
此宣告是 MQXWD 結構的 System/390 組譯器宣告。

```
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)
```

API 結束程式參照
本節提供撰寫 API 結束程式的程式設計師主要感興趣的參考資訊。

一般使用注意事項
附註:

1. 所有結束程式函數都可以發出 MQXEP 呼叫; 此呼叫專門設計用來從 API 結束程式函數使用。
2. MQ_INIT_EXIT 函數無法發出 MQXEP 以外的任何 MQ 呼叫。
3. 您無法對現行連線發出 MQDISC 呼叫。
4. 如果結束函數發出 MQCONN 呼叫，或具有 MQCNO_HANDLE_SHARE_NONE 選項的 MQCONNX 呼叫，則呼叫會完成，原因碼為 MQRC_ALREADY_CONNECTED ，且傳回的控點與作為參數傳遞至結束程式的控點相同。
5. 一般而言，當 API 結束程式函數發出 MQI 呼叫時，不會遞迴地呼叫 API 結束程式。 不過，如果結束函數使用 MQCNO_HANDLE_SHARE_BLOCK 或 MQCNO_HANDLE_SHARE_NO_BLOCK 選項發出 MQCONNX 呼叫，則該呼叫會傳回新的共用控點。 這會為結束套組提供其自己的連線控點，以及獨立於

應用程式工作單元的工作單元。 結束套組可以使用此控點在其自己的工作單元內放置及取得訊息，並確定或取消該工作單元; 所有這些都可以完成，而不會以任何方式影響應用程式的工作單元。

因為結束函數所使用的連線控點不同於應用程式所使用的控點，所以結束函數發出的 MQ 呼叫會導致呼叫相關的 API 結束函數。 因此，可以遞迴地呼叫結束函數。 請注意， MQAXP 及結束鏈結區域中的 *ExitUserArea* 欄位都具有連線控點範圍。 因此，結束程式函數無法使用那些區域來向遞迴呼叫它本身的另一個實例發出信號，表示它已在作用中。

6. 結束函數也可以在應用程式的工作單元內放置及取得訊息。 當應用程式確定或取消工作單元時，工作單元內的所有訊息會一起確定或取消，而不管誰將它們放在工作單元中 (應用程式或結束功能)。 不過，結束程式可能會導致應用程式比其他情況更早超出系統限制 (例如，超出工作單元中未確定的訊息數目上限)。

   當結束函數以此方式使用應用程式的工作單元時，結束函數通常應該避免發出 MQCMIT 呼叫，因為這會確定應用程式的工作單元，並可能損害應用程式的正確運作。 不過，如果結束函數發生嚴重錯誤，導致無法確定工作單元 (例如，在應用程式工作單元中放置訊息時發生錯誤) ，則結束函數有時可能需要發出 MQBACK 呼叫。 呼叫 MQBACK 時，請小心確保不會變更應用程式工作單元界限。 在此情況下， exit 函數必須設定適當的值，以確保完成碼 MQCC_WARNING 及原因碼 MQRC_BACKED_OUT 會傳回應用程式，以便應用程式可以偵測工作單元已取消的事實。

   如果結束函數使用應用程式的連線控點來發出 MQ 呼叫，則這些呼叫本身不會導致進一步呼叫 API 結束函數。

7. 如果 MQXR_BEFORE 結束函數異常終止，佇列管理程式可能可以從失敗中回復。 如果可以，則佇列管理程式會繼續處理，如同結束程式函數已傳回 MQXCC_FAILED 一樣。 如果佇列管理程式無法回復，則會終止應用程式。
8. 如果 MQXR_AFTER 結束函數異常終止，佇列管理程式可能可以從失敗中回復。 如果可以，則佇列管理程式會繼續處理，如同結束程式函數已傳回 MQXCC_FAILED 一樣。 如果佇列管理程式無法回復，則會終止應用程式。 請注意，在後一種情況下，在工作單元外部擷取的訊息會遺失 (與從佇列中移除訊息之後應用程式立即失敗的狀況相同)。
9. MCA 處理程序會執行兩階段確定。

   如果 API 結束程式從備妥的 MCA 處理程序截取 MQCMIT ，並嘗試在工作單元內執行動作，則動作會失敗，原因碼為 MQRC_UOW_NOT_AVAILABLE。

10. 若為多重安裝環境，結束程式與 Websphere MQ 7.0 版及 7.1 版搭配使用的唯一方法是在應用程式啟動之前，以鏈結 7.0 版與 mqm.Lib 的方式撰寫結束程式，若為非主要或重新定位的結束程式，則確保應用程式針對佇列管理程式目前與之相關聯的安裝找到正確的 mqm.Lib 。 (例如，在啟動應用程式之前執行 **setmqenv -m QM** 指令，即使佇列管理程式是由 7.0 版安裝所擁有。)
11. 如果有多個 IBM WebSphere MQ 安裝可供使用，請使用針對舊版 IBM WebSphere MQ 所撰寫的結束程式，因為新版中新增的新功能可能無法與舊版搭配使用。 如需版本之間變更的相關資訊，請參閱 WebSphere MQ 7.5 中的變更內容。

IBM WebSphere MQ API 結束程式參數結構 (MQAXP)

MQAXP 結構是一個外部控制區塊，用來作為 API 結束程式的輸入或輸出參數。 本主題也提供佇列管理程式如何處理結束程式功能的相關資訊。

MQAXP 具有下列 C 宣告:

```
typedef struct tagMQAXP {
  MQCHAR4   StrucId;           /* 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 */
};
```

當呼叫 API 結束程式中的函數時，會傳遞下列參數清單:

StrucId (MQCHAR4)-輸入
結束參數結構 ID，值為:

```
MQAXP_STRUC_ID.
```

結束處理程式會在進入每一個結束函數時設定此欄位。

版本 (MQLONG)-輸入
結構版本號碼，值為:

MQAXP_VERSION_1
第 1 版 API 結束程式參數結構。

MQAXP_VERSION_2
第 2 版 API 結束程式參數結構。

MQAXP_CURRENT_VERSION
API 結束程式參數結構的現行版本號碼。

結束處理程式會在進入每一個結束函數時設定此欄位。

ExitId (MQLONG)-輸入
結束程式 ID，在進入結束常式時設定，指出結束程式的類型:

MQXT_API_EXIT
API 結束程式。

ExitReason (MQLONG)-輸入
呼叫結束程式的原因，在進入每一個結束程式函數時設定:

MQXR_CONNECTION
呼叫結束程式以在 MQCONN 或 MQCONNX 呼叫之前自行起始設定，或在 MQDISC 呼叫之後自行結束。

MQXR_before
在執行 API 呼叫之前，或在 MQGET 上轉換資料之前，正在呼叫結束程式。

MQXR_AFTER
在執行 API 呼叫之後呼叫結束程式。

ExitResponse (MQLONG)-輸出
結束程式的回應，在進入每一個結束程式函數時起始設定為:

MQXCC_OK
正常繼續。

此欄位必須由結束函數設定，才能與佇列管理程式通訊執行結束函數的結果。 此值必須是下列其中一個:

MQXCC_OK
已順利完成結束功能。 正常繼續。

此值可以由所有 MQXR_ * 結束程式函數設定。 ExitResponse2 是用來決定是否稍後在鏈結中呼叫結束程式函數。

MQXCC_FAILED
exit 函數失敗，因為發生錯誤。

此值可以由所有 MQXR_ * 結束程式函數設定。 佇列管理程式會將 CompCode 設為 MQCC_FAILED，並將原因設為:

- 如果函數為 MQ_INIT_EXIT，則為 MQRC_API_EXIT_INIT_ERROR

- 如果函數為 MQ_TERM_EXIT，則為 MQRC_API_EXIT_TERM_ERROR
- 所有其他結束函數的 MQRC_API_EXIT_ERROR

您可以稍後在鏈結中透過結束函數來變更值集。

ExitResponse2 會被忽略; 佇列管理程式會繼續處理，就好像已傳回 MQXR2_SUPPRESS_CHAIN 一樣。

MQXCC_SUPPRESS_FUNCTION
暫停 WebSphere MQ API 函數。

此值只能由 MQXR_BEFORE 結束函數設定。 它會略過 API 呼叫。 如果由 MQ_DATA_CONV_ON_GET_EXIT 傳回，則會略過資料轉換。 佇列管理程式會將 CompCode 設為 MQCC_FAILED，並將「原因」設為 MQRC_SUPPRESSED_BY_EXIT，但鏈中稍後的結束函數可以變更所設定的值。 當結束程式離開它們時，仍會保留呼叫的其他參數。 ExitResponse2 是用來決定是否稍後在鏈結中呼叫結束程式函數。

如果此值是由 MQXR_AFTER 或 MQXR_CONNECTION 結束程式函數所設定，則佇列管理程式會繼續處理，如同已傳回 MQXCC_FAILED 一樣。

MQXCC_SKIP_FUNCTION
跳過 WebSphere MQ API 函數。

此值只能由 MQXR_BEFORE 結束函數設定。 它會略過 API 呼叫。 如果由 MQ_DATA_CONV_ON_GET_EXIT 傳回，則會略過資料轉換。 exit 函數必須將 CompCode 及 Reason 設為要傳回給應用程式的值，但該值集稍後可以由鏈中的 exit 函數變更。 當結束程式離開它們時，仍會保留呼叫的其他參數。 ExitResponse2 是用來決定是否稍後在鏈結中呼叫結束程式函數。

如果此值是由 MQXR_AFTER 或 MQXR_CONNECTION 結束程式函數所設定，則佇列管理程式會繼續處理，如同已傳回 MQXCC_FAILED 一樣。

MQXCC_SUPPRESS_EXIT
暫停屬於該結束程式集的所有結束程式函數。

此值只能由 MQXR_BEFORE 及 MQXR_AFTER 結束程式函數設定。 它會略過 所有 後續呼叫屬於此邏輯連線的這組結束程式的結束程式函數。 當以 MQXR_CONNECTION ExitReason 呼叫 MQ_TERM_EXIT 函數時，此略過會繼續進行，直到發生邏輯斷線要求為止。

exit 函數必須將 CompCode 及 Reason 設為要傳回給應用程式的值，但該值集稍後可以由鏈中的 exit 函數變更。 當結束程式離開它們時，仍會保留呼叫的其他參數。 會忽略 ExitResponse2 。

如果此值由 MQXR_CONNECTION 結束函數設定，則佇列管理程式會繼續處理，就好像已傳回 MQXCC_FAILED 一樣。

如需 ExitResponse 與 ExitResponse2 之間的互動及其對結束處理的影響的相關資訊，請參閱 第 977 頁的『佇列管理程式處理結束程式功能的方式』。

ExitResponse2 (MQLONG)-輸出
這是限定 MQXR_BEFORE 結束函數之主要結束回應碼的次要結束回應碼。 它已起始設定為:

```
MQXR2_DEFAULT_CONTINUATION
```

進入 WebSphere MQ API 呼叫結束程式函數時。 然後可以將它設為下列其中一個值:

MQXR2_DEFAULT_CONTINUATION
視 ExitResponse 的值而定，是否要繼續鏈中的下一個結束程式。

如果 ExitResponse 為 MQXCC_SUPPRESS_FUNCTION 或 MQXCC_SKIP_FUNCTION，則稍後會略過 MQXR_BEFORE 鏈中的結束函數及 MQXR_AFTER 鏈中的相符結束函數。 呼叫 MQXR_AFTER 鏈中符合 MQXR_BEFORE 鏈結中先前的結束函數的結束函數。

否則，請呼叫鏈中的下一個結束程式。

MQXR2_SUPPRESS_CHAIN
抑制鏈結。

稍後在 MQXR_BEFORE 鏈中略過結束函數，並在 MQXR_AFTER 鏈中略過此 API 呼叫呼叫的相符結束函數。 呼叫 MQXR_AFTER 鏈中符合 MQXR_BEFORE 鏈結中先前的結束函數的結束函數。

MQXR2_CONTINUE_CHAIN
繼續鏈中的下一個結束程式。

如需 ExitResponse 與 ExitResponse2 之間的互動及其對結束處理的影響的相關資訊，請參閱 第 977 頁的『佇列管理程式處理結束程式功能的方式』。

回饋 (MQLONG)-輸入/輸出
在結束函數呼叫之間傳達回饋碼。 這已起始設定為:

```
MQFB_NONE (0)
```

在鏈中呼叫第一個結束程式的第一個函數之前。

結束程式可以將此欄位設為任何值，包括任何有效的 MQFB_ * 或 MQRC_ * 值。 結束程式也可以將此欄位設為 MQFB_APPL_FIRST 至 MQFB_APPL_LAST 範圍內的使用者定義回饋值。

APICallerType (MQLONG)-輸入
API 呼叫端類型，指出 WebSphere MQ API 呼叫端是佇列管理程式的外部或內部 :MQXACT_EXTERNAL 或 MQXACT_INTERNAL。

ExitUser 區域 (MQBYTE16)-輸入/輸出
使用者區域，可供與特定 ExitInfo 物件相關聯的所有結束程式使用。 在呼叫 hconn 的第一個結束程式函數 (MQ_INIT_EXIT) 之前，它會起始設定為 MQXUA_NONE (ExitUser 區域長度的二進位零)。 從那時開始，結束程式函數對此欄位所做的任何變更都會在相同結束程式的函數呼叫之間保留。

此欄位與 4 個 MQLONGs 的倍數對齊。

結束程式也可以錨定從這個區域配置的任何儲存體。

對於每一個 hconn ，結束程式鏈中的每一個結束程式都有不同的 ExitUser 區域。 鏈中的結束程式無法共用 ExitUser 區域，且一個結束程式的 ExitUser 區域內容無法供鏈中的另一個結束程式使用。

若為 C 程式，常數 MQXUA_NONE_ARRAY 也定義為與 MQXUA_NONE 相同的值，但定義為字元陣列而非字串。

此欄位的長度由 MQ_EXIT_USER_AREA_LENGTH 指定。

ExitData (MQCHAR32)-輸入
結束程式資料，在每一個結束程式函數的輸入上設定為結束程式中提供之結束程式特定資料的 32 個字元。 如果您在結束程式中未定義任何值，則此欄位全為空白。

此欄位的長度由 MQ_EXIT_DATA_LENGTH 提供。

ExitInfo 名稱 (MQCHAR48)-輸入
結束程式資訊名稱，在每一個結束程式函數的輸入上設定給段落中結束程式定義所指定的 ApiExit_name。

ExitPDArea (MQBYTE48)-輸入/輸出
針對每次呼叫結束函數，起始設定為 MQXPDA_NONE (欄位長度為二進位零) 的問題判斷區域。

對於 C 程式，常數 MQXPDA_NONE_ARRAY 也定義為與 MQXPDA_NONE 相同的值，但定義為字元陣列而非字串。

即使函數成功，結束程式處理程式一律會在結束程式結束時將此區域寫入 WebSphere MQ 追蹤。

此欄位的長度由 MQ_EXIT_PD_AREA_LENGTH 提供。

QMgrName (MQCHAR48)-輸入
應用程式所連接的佇列管理程式名稱，因為處理 WebSphere MQ API 呼叫而呼叫結束程式。

如果 MQCONN 或 MQCONNX 呼叫中提供的佇列管理程式名稱空白，不論應用程式是伺服器或用戶端，這個欄位仍會設為應用程式所連接的佇列管理程式名稱。

結束處理程式會在進入每一個結束函數時設定此欄位。

此欄位的長度由 MQ_Q_MGR_NAME_LENGTH 提供。

ExitChainAreaPtr (PMQACH)-輸入/輸出
這用來在鏈中不同結束程式的呼叫之間傳送資料。 在呼叫結束程式鏈中第一個結束程式的第一個函數 (MQ_INIT_EXIT with ExitReason MQXR_CONNECTION) 之前，它會設為 NULL 指標。 結束程式在一次呼叫時所傳回的值會傳遞至下一次呼叫。

如需如何使用結束鏈區域的詳細資料，請參閱 第 980 頁的『結束鏈區域和結束鏈區域標頭 (MQACH)』 。

Hconfig (MQHCONFIG)-輸入
配置控點，代表正在起始設定的函數集。 此值由佇列管理程式在 MQ_INIT_EXIT 函數上產生，稍後會傳遞至 API 結束程式函數。 它在進入每一個結束函數時設定。

您可以使用 Hconfig 作為 MQIEP 結構的指標，以進行 MQI 及 DCI 呼叫。 在使用 HConfig 參數作為 MQIEP 結構的指標之前，您必須先檢查 HConfig 的前 4 個位元組是否符合 MQIEP 結構的 StrucId 。

函數 (MQLONG)-輸入
函數 ID ，其有效值為 第 982 頁的『外部常數』中說明的 MQXF_ * 常數。

根據導致呼叫結束程式的 WebSphere MQ API 呼叫而定，結束程式處理程式會在進入每一個結束函數時將此欄位設為正確值。

ExitMsg 控點 (MQHMSG)-輸入/輸出
當函數為 MQXF_GET 且 ExitReason 為 MQXR_AFTER 時，會在此欄位中傳回有效的訊息控點，容許 API 結束程式存取訊息描述子欄位，以及在登錄 API 結束程式時符合 MQXEPO 結構中所指定 ExitProperties 字串的任何其他內容。

任何在 ExitMsgHandle 中傳回的非訊息描述子內容都無法從 MQGMO 結構中的 MsgHandle (如果已指定的話) 或在訊息資料中使用。

當函數為 MQXF_GET 且 ExitReason 為 MQXR_BEFORE 時，如果結束程式將此欄位設為 MQHM_NONE ，則會暫停移入 ExitMsgHandle 內容。

如果版本小於 MQAXP_VERSION_2，則不會設定此欄位。

佇列管理程式處理結束程式功能的方式

佇列管理程式從結束函數傳回時所執行的處理，取決於 ExitResponse 和 ExitResponse2 兩者。

第 978 頁的表 593 彙總 MQXR_BEFORE 結束函數的可能組合及其效果，顯示:

- 誰設定 API 呼叫的 CompCode 及 Reason 參數
- 是否呼叫 MQXR_BEFORE 鏈中剩餘的結束函數及 MQXR_AFTER 鏈中相符的結束函數
- 是否呼叫 API 呼叫

若為 MQXR_AFTER 結束程式函數:

- CompCode 和「原因」的設定方式與 MQXR_BEFORE 相同
- 會忽略 ExitResponse2 (一律會呼叫 MQXR_AFTER 鏈結中的其餘結束程式函數)
- MQXCC_SUPPRESS_FUNCTION 和 MQXCC_SKIP_FUNCTION 無效

若為 MQXR_CONNECTION 結束程式函數:

- CompCode 和「原因」的設定方式與 MQXR_BEFORE 相同
- 已忽略 ExitResponse2
- MQXCC_SUPPRESS_FUNCTION、MQXCC_SKIP_FUNCTION、MQXCC_SUPPRESS_EXIT 無效

在所有情況下，如果結束程式或佇列管理程式設定 CompCode 及「原因」，則可以透過稍後呼叫的結束程式或 API 呼叫 (如果稍後呼叫 API 呼叫) 來變更值集。

表 593: 在結束處理之前 *MQXR_BEFORE*			
ExitResponse 的值	**CompCode 及「原因」設定者**	**ExitResponse2 (預設接續) 鏈的值**	**ExitResponse2 (預設接續) API 的值**
MQXCC_OK	exit	Y	Y
MQXCC_SUPPRESS_EXIT	exit	Y	Y
MQXCC_SUPPRESS_FUNCTION	佇列管理程式 (queue manager)	N	N
MQXCC_SKIP 函數	exit	N	N
MQXCC_FAILED	佇列管理程式 (queue manager)	N	N

用戶端處理結束程式功能的方式

一般而言，用戶端處理結束程式函數的方式與伺服器應用程式相同，不論該函數位於伺服器或用戶端上，此結構中的 *QMgrName* 屬性都適用。

不過，用戶端沒有 *mqs.ini* 檔案的概念，因此 *ApiExitCommon* 及 *APIExitTemplate* 段落不適用。 只有 *ApiExitLocal* 段落適用，且此段落在 *mqclient.ini* 檔案中配置。

IBM WebSphere MQ API 結束程式環境定義結構 (MQAXC)

MQAXC 結構 (外部控制區塊) 用來作為 API 結束程式的輸入參數。

MQAXC 具有下列 C 宣告:

```
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 */
 };
```

MQAXC 的參數如下:

StrucId (MQCHAR4)-輸入
結束程式環境定義結構 ID，值為 MQAXC_STRUC_ID。 對於 C 程式，也會定義常數 MQAXC_STRUC_ID_ARRAY，其值與 MQAXC_STRUCT_ID 相同，但會作為字元陣列而非字串。

結束處理程式會在進入每一個結束函數時設定此欄位。

版本 (MQLONG)-輸入
結構版本號碼，值為:

MQAXC_VERSION_2
結束環境定義結構的版本號碼。

MQAXC_CURRENT_VERSION
結束環境定義結構的現行版本號碼。

結束處理程式會在進入每一個結束函數時設定此欄位。

環境 (MQLONG)-輸入
發出 WebSphere MQ API 呼叫的環境，導致驅動結束函數。 此欄位的有效值如下:

MQXE_OTHER
此值與從伺服器應用程式呼叫 API 結束程式時所看到的呼叫一致。 這表示 API 結束程式在用戶端上執行時不會變更，且不會看到任何不同的情況。

如果結束程式確實需要判斷它是否在用戶端上執行，則結束程式可以透過查看 *ChannelName* 和 *ChannelDefinition* 欄位來執行此動作。

MQXE_MCA
訊息通道代理程式

MQXE_MCA_SVRCONN
代表用戶端執行的訊息通道代理程式

MQXE_COMMAND_SERVER
指令伺服器

MQXE_MQSC
runmqsc 指令直譯器

結束處理程式會在進入每一個結束函數時設定此欄位。

UserId (MQCHAR12)-輸入
與應用程式相關聯的使用者 ID。 尤其是在用戶端連線的情況下，此欄位包含所採用使用者的使用者 ID，而不是執行通道碼的使用者 ID。 如果空白使用者 ID 來自用戶端，則不會對已使用的使用者 ID 進行任何變更。 也就是說，不會採用新的使用者 ID。

結束處理程式會在進入每一個結束函數時設定此欄位。 此欄位的長度由 MQ_USER_ID_LENGTH 提供。

如果是用戶端，這是從用戶端傳送至伺服器的使用者 ID。 請注意，這可能不是用戶端在佇列管理程式中執行所針對的有效使用者 ID，因為可能有變更使用者 ID 的 MCAUser 或 CHLAUTH 配置。

SecurityId (MQBYTE40)-輸入
執行應用程式之使用者 ID 的延伸。 其長度由 MQ_SECURITY_ID_LENGTH 提供。

如果是用戶端，這是從用戶端傳送至伺服器的使用者 ID。 請注意，這可能不是用戶端在佇列管理程式中執行所針對的有效使用者 ID，因為可能有變更使用者 ID 的 MCAUser 或 CHLAUTH 配置。

ConnectionName (MQCHAR264)-輸入
連線名稱欄位，設為用戶端的位址。 例如，對於 TCP/IP，它會是用戶端 IP 位址。

此欄位的長度由 MQ_CONN_NAME_LENGTH 提供。

如果是用戶端，這是佇列管理程式的友機位址。

LongMCAUserIdLength (MQLONG)-輸入
長 MCA 使用者 ID 的長度。

當 MCA 連接至佇列管理程式時，此欄位會設為長 MCA 使用者 ID 的長度 (如果沒有此類 ID，則為零)。

如果是用戶端，這是用戶端長使用者 ID。

LongRemoteUserId 長度 (MQLONG)-輸入
長遠端使用者 ID 的長度。

當 MCA 連接至佇列管理程式時，此欄位會設為長遠端使用者 ID 的長度。 否則此欄位將設為零

如果是用戶端，請將此欄位設為零。

LongMCAUserIdPtr (MQPTR)-輸入
長 MCA 使用者 ID 的位址。

當 MCA 連接至佇列管理程式時，此欄位會設為長 MCA 使用者 ID 的位址 (如果沒有空值指標，則會設為空值指標)。

如果是用戶端，這是用戶端長使用者 ID。

LongRemoteUserIdPtr (MQPTR)-輸入
長遠端使用者 ID 的位址。

當 MCA 連接至佇列管理程式時，此欄位會設為長遠端使用者 ID 的位址 (如果沒有這類 ID，則會設為空值指標)。

如果是用戶端，請將此欄位設為零。

ApplName (MQCHAR28)-輸入
發出 WebSphere MQ API 呼叫的應用程式或元件名稱。

產生 ApplName 的規則與產生 MQPUT 的預設名稱的規則相同。

查詢作業系統中的程式名稱，即可找到此欄位的值。 其長度由 MQ_APPL_NAME_LENGTH 提供。

ApplType (MQLONG)-輸入
發出 WebSphere MQ API 呼叫的應用程式或元件類型。

對於應用程式編譯所在的平台，此值為 MQAT_DEFAULT，或等於其中一個已定義的 MQAT_ * 值。

結束處理程式會在進入每一個結束函數時設定此欄位。

ProcessId (MQPID)-輸入
作業系統處理程序 ID。

如果適用的話，結束處理程式會在進入每一個結束函數時設定此欄位。

ThreadId (MQTID)-輸入
MQ 執行緒 ID。 此 ID 與 MQ 追蹤及 FFST 傾出中使用的 ID 相同，但可能與作業系統執行緒 ID 不同。

如果適用的話，結束處理程式會在進入每一個結束函數時設定此欄位。

ChannelName (MQCHAR)-輸入
通道的名稱，以空白填補 (如果適用且已知的話)。

如果不適用，則此欄位設為空值字元。

Reserved1 (MQBYTE4)-輸入
此欄位已保留。

ChanneDefinition (PMQCD)-輸入
指向所使用通道定義的指標 (如果適用且已知的話)。

如果不適用，則此欄位設為空值字元。

請注意，只有在代表 WebSphere MQ 通道處理連線且已讀取通道定義時，才會完成指標。

尤其，當針對通道發出第一個 MQCONN 呼叫時，不會在伺服器上提供通道定義。 此外，如果填入指標，則指標所指向的結構 (及任何子結構) 必須視為唯讀; 結構的任何更新都會導致無法預期的結果，且不受支援。

在用戶端的情況下，除了為用戶端指定值的欄位以外，其他欄位包含適用於用戶端應用程式的值。

結束鏈區域和結束鏈區域標頭 (MQACH)

必要的話，結束函數可以獲得結束鏈區域的儲存體，並在 MQAXP 中將 ExitChainAreaPtr 設定為指向此儲存體。

結束程式 (相同或不同的結束程式函數) 可以獲得多個結束程式鏈結區域並將它們鏈結在一起。 只有在從結束處理程式呼叫時，才必須在此清單中新增或移除結束鏈區域。 這可確保在清單中同時新增或移除區域的不同執行緒不會造成序列化問題。

結束鏈區域必須以 MQACH 標頭結構開頭，其 C 宣告為:

```
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 */
};
```

結束鏈區域標頭中的欄位如下:

StrucId (MQCHAR4)-輸入
MQACH_STRUC_ID 的結束鏈結區域結構 ID，具有 MQACH_DEFAULT 所定義的起始值。

若為 C 程式，也會定義常數 MQACH_STRUC_ID_ARRAY; 此值與 MQACH_STRUC_ID 相同，但作為字元陣列而非字串。

版本 (MQLONG)-輸入
結構版本號碼，如下所示:

MQACH_VERSION_1
結束參數結構的版本號碼。

MQACH_CURRENT_VERSION
結束程式環境定義結構的現行版本號碼。

此欄位的起始值 (由 MQACH_DEFAULT 所定義) 是 MQACH_CURRENT_VERSION。

註: 如果您建立此結構的新版本，則現有組件的佈置不會變更。 結束函數必須檢查版本號碼是否等於或大於包含結束函數需要使用之欄位的最低版本。

StrucLength (MQLONG)-輸入
MQACH 結構的長度。 結束程式可以使用此欄位來判定結束程式資料的開始，並將它設為結束程式所建立結構的長度。

此欄位的起始值 (由 MQACH_DEFAULT 所定義) 是 MQACH_CURRENT_LENGTH。

ChainArea 長度 (MQLONG)-輸入
結束鏈區域長度，設為現行結束鏈區域的整體長度，包括 MQACH 標頭。

此欄位的起始值 (由 MQACH_DEFAULT 定義) 為零。

ExitInfo 名稱 (MQCHAR48)-輸入
結束程式資訊名稱。

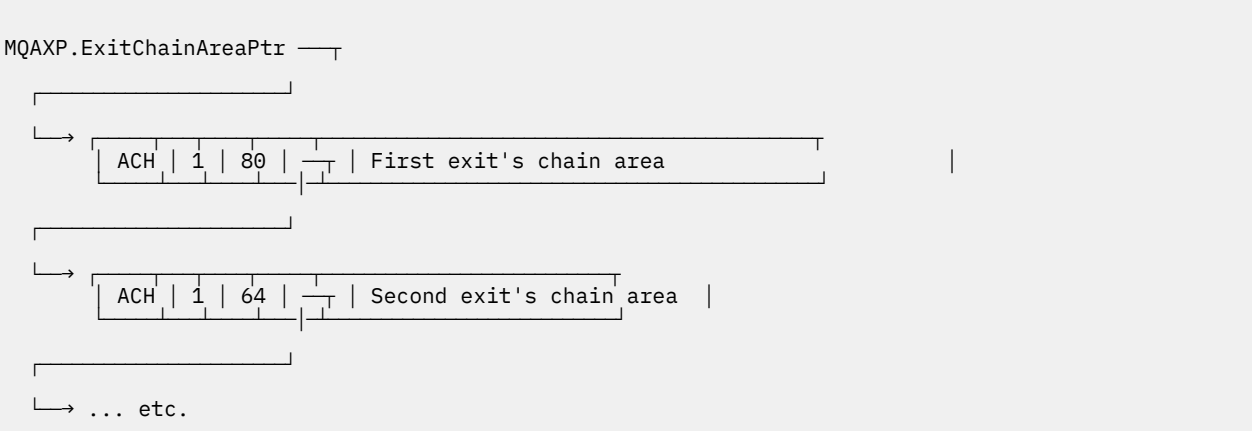
當結束程式建立 MQACH 結構時，它必須以自己的 ExitInfo 名稱來起始設定此欄位，以便稍後可以透過此結束程式的另一個實例或協同作業結束程式找到此 MQACH 結構。

此欄位的起始值 (由 MQACH_DEFAULT 定義) 是長度為零的字串 ({""})。

NextChainAreaPtr (PMQACH)-輸入
指向下一個結束鏈結區域的指標，其起始值由 MQACH_DEFAULT 所定義，為空值指標 (NULL)。

結束函數必須釋放其獲得之任何結束鏈區域的儲存體，並操作鏈指標以從清單中移除其結束鏈區域。

出口鏈區域可以如下所示建構:



外部常數

請使用本主題作為 API 可用的外部常數的參照資訊。

下列外部常數可用於 API 結束程式:

MQXF_ * (結束程式函數 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_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'
```

MQXR_ * (結束原因)

```
MQXR_BEFORE                     1       X'00000001'
MQXR_AFTER                      2       X'00000002'
MQXR_CONNECTION                 3       X'00000003'
```

MQXE_ * (環境)

```
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* _ * (其他常數)

```
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* _ * (空常數)

```
MQXPDA_NONE                     X'00...00' (48 nulls)
MQXPDA_NONE_ARRAY               '\0','\0',...,'\0','\0'
```

MQXCC_ * (完成碼)

```
MQXCC_FAILED                   -8
```

MQRC_ * (原因碼)

MQRC_API_EXIT_ERROR 2374 X'00000946'
結束函數呼叫已傳回無效的回應碼，或以某種方式失敗，且佇列管理程式無法決定要採取的下一個動作。

請檢查 MQAXP 的 ExitResponse 和 ExitResponse2 欄位，以判斷不正確的回應碼，並變更結束程式以傳回有效的回應碼。

MQRC_API_EXIT_INIT_ERROR 2375 X'00000947'
起始設定 API 結束程式函數的執行環境時，佇列管理程式發現錯誤。

MQRC_API_EXIT_TERM_ERROR 2376 X'00000948'
佇列管理程式在關閉 API 結束程式函數的執行環境時發生錯誤。

MQRC_EXIT_REASON_ERROR 2377 X'00000949'
在結束進入點登錄呼叫 (MQXEP) 呼叫中提供的 ExitReason 欄位值發生錯誤。

請檢查 ExitReason 欄位的值，以判斷並更正不正確的結束原因值。

MQRC_RESERVED_VALUE_ERROR 2378 X'0000094A'
「保留」欄位的值發生錯誤。

請檢查「保留」欄位的值，以判斷並更正「保留」值。

C 語言 typedefs

本主題提供與以 C 語言提供之 API 結束程式相關聯的 typedefs 相關資訊。

以下是與 API 結束程式相關聯的 C 語言 typedefs:

```
typedef PMQLONG  MQPOINTER PPMQLONG;
typedef PMQBYTE  MQPOINTER PPMQBYTE;
typedef PMQHOBJ  MQPOINTER PPMQHOBJ;
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;
```

結束程式進入點登錄呼叫 (MQXEP)

使用此資訊來瞭解 MQXEP、MQXEP C 語言呼叫及 MQXEP C 函數原型。

使用 MQXEP 呼叫:

1. 在 WebSphere MQ API 結束程式呼叫點之前及之後登錄，以呼叫結束程式函數
2. 指定結束函數進入點
3. 取消登錄結束函數進入點

您通常會在 MQ_INIT_EXIT 結束函數中撰寫 MQXEP 呼叫的程式碼，但您可以在任何後續的結束函數中指定它們。

如果您使用 MQXEP 呼叫來登錄已登錄的結束函數，則第二個 MQXEP 呼叫會順利完成，並取代已登錄的結束函數。

如果您使用 MQXEP 呼叫來登錄 NULL 結束函數， MQXEP 呼叫會順利完成，且會取消登錄結束函數。

如果在連線要求有效期間使用 MQXEP 呼叫來登錄、取消登錄及重新登錄特定結束程式函數，則會重新啟動先前登錄的結束程式函數。 仍然配置且與此結束程式函數實例相關聯的任何儲存體都可供結束程式的函數使用。 (此儲存體通常在呼叫終止結束程式函數期間釋放。)

MQXEP 的介面如下:

```
MQXEP (Hconfig, ExitReason, Function, EntryPoint, &ExitOpts, &CompCode, &Reason)
```

其中:

Hconfig (MQHCONFIG)-輸入
配置控點，代表包含正在起始設定之函數集的 API 結束程式。 此值由佇列管理程式在呼叫 MQ_INIT_EXIT 函數之前立即產生，並在 MQAXP 中傳遞至每一個 API 結束程式函數。

ExitReason (MQLONG)-輸入
登錄進入點的原因，來自下列原因:

- 連線層次起始設定或終止 (MQXR_CONNECTION)
- 在 WebSphere MQ API 呼叫之前 (MQXR_BEFORE)
- 在 WebSphere MQ API 呼叫 (MQXR_AFTER) 之後

函數 (MQLONG)-輸入
函數 ID ，有效值為 MQXF_ * 常數 (請參閱 第 982 頁的『外部常數』)。

EntryPoint (PMQFUNC)-輸入
要登錄之結束函數的進入點位址。 值 NULL 指出尚未提供結束函數，或正在取消登錄先前的結束函數登錄。

ExitOpts(MQXEPO)
API 結束程式可以指定選項來控制如何登錄 API 結束程式。 如果對此欄位指定空值指標，則會採用 MQXEPO 結構的預設值。

CompCode (MQLONG)-輸出
完成碼，其有效值為:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因 (MQLONG)-輸出
限定完成碼的原因碼。

如果完成碼是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果完成碼是 MQCC_FAILED:

MQRC_HCONFIG_ERROR
(2280 , X'8E8') 提供的配置控點無效。 使用 MQAXP 中的配置控點。

MQRC_EXIT_REASON_ERROR
(2377 , X' 949 ') 提供的結束函數呼叫原因無效或對提供的結束函數 ID 無效。

請使用其中一個有效的結束函數呼叫原因 (MQXR_ * 值) , 或使用有效的函數 ID 與結束原因組合。 (請參閱 第 985 頁的表 594。)

MQRC_FUNCTION_ERROR
(2281 , X'8E9') 針對 API 結束程式原因提供的函數 ID 無效。 下表顯示函數 ID 與 ExitReasons 的有效組合。

表 *594:* 有效的函數 *ID* 與 *ExitReasons* 組合

函數	**ExitReason**
MQXF_INIT MQXF_TERM	MQXR_CONNECTION
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 ') 嘗試登錄或取消登錄跳出函數失敗，因為資源問題。

MQRC_UNEXPECTED_ERROR
(2195 , X'893 ') 嘗試登錄或取消登錄跳出函數非預期地失敗。

MQRC_PROPERTY_NAME_ERROR
(2442 , X'098A') 無效的 ExitProperties 名稱。

MQRC_XEPO_ERROR
(2507 , X'09CB') 結束選項結構無效。

MQXEP C 語言呼叫

```
MQXEP (Hconfig, ExitReason, Function, EntryPoint, &ExitOpts, &CompCode, &Reason);
```

參數清單的宣告:

```
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 函數原型

```
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 */
```

結束函數

本節提供一些一般資訊，以協助您使用函數呼叫，並說明如何呼叫個別結束程式函數。

使用此資訊來瞭解 API 結束常式的一般規則，以及設定和清除結束執行環境。

API 結束常式的一般規則

呼叫 API 結束常式時，會套用下列一般規則。

- 在所有情況下，在驗證 API 呼叫參數之前，以及在任何安全檢查之前 (在 MQCONN、MQCONNX 或 MQOPEN 的情況下) ，都會驅動 API 結束程式函數。
- 從結束常式輸入及輸出的欄位值如下:
  - 在 WebSphere MQ API 結束程式函數之前的輸入中，欄位值可以由應用程式或先前的結束函數呼叫來設定。
  - 在 *(* WebSphere MQ API 結束程式函數之前) 的輸出上，欄位的值可以保持不變，或由結束程式函數設定為某個其他值。
  - 在輸入 之後 WebSphere MQ API 結束函數時，欄位的值可以是佇列管理程式在處理 WebSphere MQ API 呼叫之後所設定的值，也可以設為結束函數鏈中先前的結束函數呼叫所設定的值。
  - 在來自 之後 WebSphere MQ API 呼叫結束函數的輸出上，欄位的值可以保持不變，或由結束函數設為其他值。
- 結束程式函數必須使用 ExitResponse 及 ExitResponse2 欄位與佇列管理程式進行通訊。
- CompCode 和「原因碼」欄位會與應用程式進行通訊。 佇列管理程式及結束函數可以設定 CompCode 及「原因碼」欄位。
- MQXEP 呼叫會將新的原因碼傳回給呼叫 MQXEP 的 exit 函數。 不過，結束函數可以將這些新的原因碼轉換為現有及新應用程式可以瞭解的任何現有原因碼。
- 每一個結束函數原型都具有與 API 函數類似的參數，除了 CompCode 及「原因」之外，還有額外的間接層次。
- API 結束程式可以發出 MQI 呼叫 (MQDISC 除外) ，但這些 MQI 呼叫本身不會呼叫 API 結束程式。

請注意，不論應用程式是在伺服器或用戶端上，您都無法預測 API 結束程式呼叫的排序。 API 結束程式 BEFORE 呼叫之後可能不會立即出現 AFTER 呼叫。

BEFORE 呼叫後面可以接著另一個 BEFORE 呼叫。 例如:

在 MQCTL 之前
前置回呼

在 MQPAD 之前
在 MQPUT 之後
在回呼之後
在 MQCTL 之後

or

在 XAOpen 之前
在 MQCONNX 之前
在 MQCONNX 之後
在 XAOPEN 之後

在用戶端上，有一個結束程式可以修改 MQCONN 或 MQCONNX 呼叫的行為，稱為 PreConnect 結束程式。 PreConnect 結束程式可以修改 MQCONN 或 MQCONNX 呼叫中的任何參數，包括佇列管理程式名稱。 用戶端會先呼叫此結束程式，然後呼叫 MQCONN 或 MQCONNX 呼叫。 請注意，只有起始 MQCONN 或 MQCONNX 呼叫會呼叫 API 結束程式; 任何後續的重新連接呼叫都沒有作用。

執行環境

一般而言，會使用 MQAXP 中的 ExitResponse 及 ExitResponse2 欄位，將結束函數中的所有錯誤傳回結束處理程式。

這些錯誤會依序轉換成 MQCC_ * 和 MQRC_ * 值，並在 CompCode 和「原因」欄位中傳回給應用程式。 不過，在結束處理程式邏輯中發現的任何錯誤，都會以 CompCode 和「原因」欄位中的 MQCC_ * 和 MQRC_ * 值形式傳回給應用程式。

如果 MQ_TERM_EXIT 函數傳回錯誤:

- 已進行 MQDISC 呼叫
- 沒有其他機會驅動 之後 MQ_TERM_EXIT 結束函數 (因此執行結束程式執行環境清理)
- 未 執行結束執行環境清理

無法卸載結束程式，因為它可能仍在使用中。 此外，在結束鏈中更向下的其他已登錄結束程式，其 之前 結束程式已順利完成，將以相反順序驅動。

設定結束程式執行環境

在處理明確 MQCONN 或 MQCONNX 呼叫時，結束程式處理邏輯會在呼叫結束程式起始設定函數 (MQ_INIT_EXIT) 之前設定結束程式執行環境。 結束程式執行環境設定包括載入結束程式、獲得的儲存體，以及起始設定結束程式參數結構。 也會配置結束程式配置控點。

如果在此階段期間發生錯誤， MQCONN 或 MQCONNX 呼叫會失敗，並產生 CompCode MQCC_FAILED 及下列其中一個原因碼:

MQRC_API_EXIT_LOAD_ERROR
嘗試載入 API 結束程式模組失敗。

找不到 MQRC_API_EXIT_NOT_FOUND
在 API 結束程式模組中找不到 API 結束程式函數。

MQRC_STORAGE_NOT_AVAILABLE
嘗試起始設定 API 結束程式函數的執行環境失敗，因為可用的儲存體不足。

MQRC_API_EXIT_INIT_ERROR
起始設定 API 結束程式函數的執行環境時發生錯誤。

清除結束程式執行環境

在處理明確 MQDISC 呼叫時，或因應用程式結束而產生隱含斷線要求時，結束程式處理邏輯可能需要在呼叫結束程式終止函數 (MQ_TERM_EXIT) 之後清除結束程式執行環境 (如果已登錄的話)。

清除結束執行環境涉及釋放結束參數結構的儲存體，可能刪除先前載入記憶體的任何模組。

如果在此階段期間發生錯誤，則明確 MQDISC 呼叫會失敗，並顯示 CompCode MQCC_FAILED 及下列原因碼 (在隱含斷線要求上不會強調顯示錯誤):

MQRC_API_EXIT_TERM_ERROR
關閉 API 結束程式函數的執行環境時發生錯誤。 在 MQ_TERM* API 結束程式函數呼叫之前或之後，結束程式 不應 從 MQDISC 傳回任何失敗。

用戶端上的 *API* 結束程式

用戶端使用 PreConnect 結束程式來修改 MQCONN 和 MQCONNX 呼叫的行為，且不支援 API 結束程式內容。

PreConnect 結束程式

在用戶端上， PreConnect 結束程式可用來從中央儲存庫 (例如 LDAP 伺服器) 查閱通道定義。

PreConnect 結束程式也可以修改 MQCONN 或 MQCONNX 呼叫本身的任何參數或所有參數，例如佇列管理程式名稱。

如果是用戶端應用程式，則必須在 API 結束程式之前呼叫 PreConnect 結束程式，因為只有在已知佇列管理程式名稱且 PreConnect 結束程式可以變更此名稱時，才會呼叫 MQCONN 或 MQCONNX API 結束程式。

請注意，只有起始 MQCONN 或 MQCONNX 呼叫會呼叫結束程式。

API 結束程式內容

在伺服器上， API 結束程式可以在起始設定時登錄 MQXEPO 結構。 MQXEPO 結構包含 ExitProperties 欄位，其詳述結束程式感興趣的內容群組。 這會產生個別訊息內容控點，結束程式可以與任何應用程式訊息內容控點分開操作。

在用戶端上，不支援 API 結束程式內容。 如果嘗試在用戶端上登錄內容群組名稱，則函數會失敗，原因碼為 MQRC_EXIT_PROPS_NOT_SUPPORTED。

取消-*MQ_BACK_EXIT*

MQ_BACK_EXIT 提供取消結束程式函數來執行 之前 及 之後 取消處理。 搭配使用函數 ID MQXF_BACK 與結束原因 MQXR_BEFORE 及 MQXR_AFTER ， 以在取消呼叫結束函數 之前 及 之後 登錄。

此功能的介面為:

```
MQ_BACK_EXIT (&ExitParms, &ExitContext, &Hconn, &CompCode, &Reason)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

CompCode (MQLONG)-輸入/輸出
完成碼，有效值為:

MQCC_OK
順利完成。

MQCC_WARNING
局部完成。

MQCC_FAILED
通話失敗

原因 (MQLONG)-輸入/輸出
定義完成碼的原因碼。

如果完成碼為 MQCC_OK，則唯一有效值為:

MQRC_NONE
(0， x '000') 沒有報告的原因。

如果完成碼是 MQCC_FAILED 或 MQCC_WARNING，則結束函數可以將原因碼欄位設為任何有效的 MQRC_ * 值。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
MQ_BACK_EXIT (&ExitParms, &ExitContext, &Hconn, &CompCode, &Reason);
```

您的結束程式必須符合下列 C 函數原型:

```
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 */
```

開始-*MQ_BEGIN_EXIT*

MQ_BEGIN_EXIT 提供開始結束程式函數來執行 之前 及 之後 MQBEGIN 呼叫處理。 搭配使用函數 ID MQXF_BEGIN 與結束原因 MQXR_BEFORE 及 MQXR_AFTER，以登錄 之前 及 之後 MQBEGIN 呼叫結束程式函數。

此功能的介面為:

```
MQ_BEGIN_EXIT (&ExitParms, &ExitContext, &Hconn, &pBeginOptions, &CompCode,
               &Reason)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

pBegin 選項 (PMQBO)-輸入/輸出
開始選項的指標。

CompCode (MQLONG)-輸入/輸出
完成碼，有效值為:

MQCC_OK
順利完成。

MQCC_WARNING
局部完成。

MQCC_FAILED
通話失敗

原因 (MQLONG)-輸入/輸出
定義完成碼的原因碼。

如果完成碼為 MQCC_OK ， 則唯一有效值為:

MQRC_NONE
(0 ， x '000') 沒有報告的原因。

如果完成碼是 MQCC_FAILED 或 MQCC_WARNING ， 則結束函數可以將原因碼欄位設為任何有效的 MQRC_ * 值。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
MQ_BEGIN_EXIT (&ExitParms, &ExitContext, &Hconn, &pBeginOptions, &CompCode,
               &Reason);
```

您的結束程式必須符合下列 C 函數原型:

```
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 */
```

回呼-*MQ_CALLBACK_EXIT*

MQ_CALLBACK_EXIT 提供結束函數來執行 之前 及 之後 回呼處理。 搭配使用函數 ID MQXF_CALLBACK 與結束原因 MQXR_BEFORE 及 MQXR_AFTER ， 以在 之前及 之後 回呼呼叫結束程式函數。

此功能的介面為:

```
MQ_CALLBACK_EXIT (&ExitParms, &ExitContext, &Hconn, &pMsgDesc, &pGetMsgOpts,
    &pBuffer, &pMQCBContext)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構

Hconn (MQHCONN)-輸入/輸出
連線控點

pMsg 說明
訊息描述子

pGetMsgOpts
控制 MQGET 動作的選項

pBuffer
包含訊息資料的區域

pMQCBContext
回呼的環境定義資料

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
        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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
MQ_SUBRQ_EXIT (&ExitParms, &ExitContext, &Hconn, &pMsgDesc, &pGetMsgOpts, &pBuffer,
 &pContext);
```

您的結束程式必須符合下列 C 函數原型:

```
void MQENTRY MQ_CALLBACK_EXIT (
PMQAXP    pExitParms;    /* Exit parameter structure */
PMQAXC    pExitContext;  /* Exit context structure */
PMQHCONN  pHconn;        /* Connection handle */
PPMQMD    ppMsgDesc;     /* Message descriptor */
PPMQGMO   ppGetMsgOpts;  /* Options that define the operation of the consumer */
PPMQVOID  ppBuffer;      /* Area to conatin the message data */
PPMQCBC   ppContext;)     /* Context data for the callback */
```

使用注意事項

1. 在呼叫消費者之前，以及在完成消費者的消費者函數之後，會呼叫回呼結束程式。 雖然 MQMD 和 MQGMO 結構是可變的，但變更 before exit 中的值並不會重新驅動從佇列中擷取訊息，因為訊息已從要遞送至消費者函數的佇列中移除

管理回呼函數-*MQ_CB_EXIT*

MQ_CB_EXIT 提供結束函數，以在 MQCB 呼叫 之前 及 之後 執行。 使用具有結束原因 MQXR_BEFORE 及 MQXR_AFTER 的函數 ID MQXF_CB 來登錄 之前 及 之後 MQCB 呼叫結束程式函數。

此功能的介面為:

```
MQ_CB_EXIT (&ExitParms, &ExitContext, &Hconn, &Operation, &pCallbackDesc,
            &Hobj, &pMsgDesc, &pGetMsgOpts, &CompCode, &Reason)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構

Hconn (MQHCONN)-輸入/輸出
連線控點

作業 (MQLONG)-輸入/輸出
作業值

pCallback 說明 (PMQCBD)-輸入/輸出
回呼描述子

Hobj (MQHOBJ)-輸入/輸出
物件控點

pMsg 說明 (PMQMD)-輸入/輸出
訊息描述子

pGetMsgOpts (PMQGMO)-輸入/輸出
控制 MQCB 動作的選項

CompCode (MQLONG)-輸入/輸出
完成碼

原因 (MQLONG)-輸入/輸出
定義 CompCode 的原因碼

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
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 */
PMQLONG  CompCode;       /* Completion code.
PMQLONG) Reason;         /* Reason code qualifying CompCode.
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
MQ_CB_EXIT (&ExitParms, &ExitContext, &Hconn, &Operation, &Hobj, &pMsgDesc,
            &pGetMsgOpts, &CompCode, &Reason);
```

您的結束程式必須符合下列 C 函數原型:

```
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 */
```

關閉-*MQ_CLOSE_EXIT*

MQ_CLOSE_EXIT 提供關閉結束程式函數以執行 之前 和 之後 MQCLOSE 呼叫處理。 將函數 ID MQXF_CLOSE 與結束原因 MQXR_BEFORE 及 MQXR_AFTER 搭配使用，以在 MQCLOSE 呼叫結束函數 之前 及 之後 登錄。

此功能的介面為:

```
MQ_CLOSE_EXIT (&ExitParms, &ExitContext, &Hconn, &pHobj,
               &Options, &CompCode, &Reason)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

pHobj (PMQHOBJ)-輸入
指向物件控點的指標。

選項 (MQLONG)-輸入/輸出
關閉選項。

CompCode (MQLONG)-輸入/輸出
完成碼，有效值為:

MQCC_OK
順利完成。

MQCC_FAILED
通話失敗

原因 (MQLONG)-輸入/輸出
定義完成碼的原因碼。

如果完成碼為 MQCC_OK，則唯一有效值為:

MQRC_NONE
(0， x '000') 沒有報告的原因。

如果完成碼是 MQCC_FAILED，則結束函數可以將原因碼欄位設為任何有效的 MQRC_ * 值。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
MQAXP           ExitParms;      /* Exit parameter structure */
MQAXC           ExitContext;    /* Exit context structure */
MQHCONN         Hconn;          /* Connection handle */
PMQHOBJ         pHobj;          /* Ptr to object handle */
MQLONG          Options;        /* Close options */
MQLONG          CompCode;       /* Completion code */
MQLONG          Reason;         /* Reason code */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
MQ_CLOSE_EXIT (&ExitParms, &ExitContext,&Hconn, &pHobj, &Options,
               &CompCode, &Reason);
```

您的結束程式必須符合下列 C 函數原型:

```
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 */
PPMQHOBJ        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 */
```

確定-*MQ_CMIT_EXIT*

MQ_CMIT_EXIT 提供確定結束程式函數，以執行 之前 及 之後 確定處理。 將函數 ID MQXF_CMIT 與結束原因 MQXR_BEFORE 及 MQXR_AFTER 搭配使用，以在確定呼叫結束程式函數 之前 及 之後 登錄。

如果確定作業失敗，且交易已取消，則 MQCMIT 呼叫會失敗，並出現 MQCC_WARNING 和 MQRC_BACKED_OUT。 這些回覆碼及原因碼會傳遞至任何 之後 MQCMIT 結束程式函數，以讓結束程式函數指出工作單元已取消。

此功能的介面為:

```
MQ_CMIT_EXIT (&ExitParms, &ExitContext, &Hconn, &CompCode, &Reason)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

CompCode (MQLONG)-輸入/輸出
完成碼，有效值為:

MQCC_OK
順利完成。

MQCC_WARNING
局部完成。

MQCC_FAILED
通話失敗

原因 (MQLONG)-輸入/輸出
定義完成碼的原因碼。

如果完成碼為 MQCC_OK，則唯一有效值為:

MQRC_NONE
(0， x '000') 沒有報告的原因。

如果完成碼是 MQCC_FAILED 或 MQCC_WARNING，則結束函數可以將原因碼欄位設為任何有效的 MQRC_ * 值。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
MQ_CMIT_EXIT (&ExitParms, &ExitContext,&Hconn, &CompCode, &Reason);
```

您的結束程式必須符合下列 C 函數原型:

```
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 */
```

使用注意事項

1. 這裡說明的 MQ_GET_EXIT 函數介面同時用於 MQXF_GET 結束函數及 第 1000 頁的『MQXF_DATA_CONV_ON_GET』 結束函數。

   這兩個結束函數會定義個別進入點，因此若要截取 兩者，必須使用 MQXEP 呼叫兩次; 對於此呼叫，請使用函數 ID MQXF_GET。

   因為 MQXF_GET_EXIT 介面與 MQXF_GET 及 MQXF_DATA_CONV_ON_GET 相同，所以單一結束函數可用於兩者; MQAXP 結構中的 *Function* 欄位指出已呼叫哪個結束函數。 或者， MQXEP 呼叫可用來針對這兩種情況登錄不同的結束函數。

連接及連接延伸-*MQ_CONNX_EXIT*

MQ_CONNX_EXIT 提供:

- 執行 之前 及 之後 MQCONN 處理的連線結束程式函數
- 執行 之前 及 之後 MQCONNX 處理的連線延伸結束程式函數

針對 MQCONN 和 MQCONNX 呼叫結束程式函數都會呼叫這裡說明的相同介面。

當訊息通道代理程式 (MCA) 回應入埠用戶端連線時， MCA 可以在用戶端狀態完全已知之前連接並發出一些 WebSphere MQ API 呼叫。 這些 API 呼叫會根據 MCA 程式本身 (例如，在 MQAXC 的 UserId 和 ConnectionName 欄位中)，以 MQAXC 來呼叫 API 結束程式函數。

當 MCA 回應後續的入埠用戶端 API 呼叫時， MQAXC 結構會以入埠用戶端為基礎，並適當地設定 UserId 和 ConnectionName 欄位。

應用程式在 MQCONN 或 MQCONNX 呼叫上設定的佇列管理程式名稱會傳遞至基礎連接呼叫。 在 MQ_CONNX_EXIT 之前嘗試變更佇列管理程式的名稱沒有任何作用。

使用函數 ID MQXF_CONN 及 MQXF_CONNX 與結束程式原因 MQXR_BEFORE 及 MQXR_AFTER 來登錄 之前及 之後 MQCONN 及 MQCONNX 呼叫結束程式函數。

MQ_CONNX_EXIT 結束程式呼叫的原因是 MQXR_BEFORE 不得 發出任何 WebSphere MQ API 呼叫，因為此時尚未設定正確的環境。

MQ_CONNX_EXIT 無法從正在呼叫其連線的 API 結束程式呼叫來呼叫 MQDISC。 此限制同時適用於用戶端和伺服器 API 結束程式。

MQCONN 與 MQCONNX 的介面相同:

```
MQ_CONNX_EXIT (&ExitParms, &ExitContext, &pQMgrName, &pConnectOpts,
               &pHconn, &CompCode, &Reason);
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

pQMgr 名稱 (PMQCHAR)-輸入
MQCONNX 呼叫所提供佇列管理程式名稱的指標。 結束程式不得在 MQCONN 或 MQCONNX 呼叫中變更此名稱。

pConnect 選項 (PMQCNO)-輸入/輸出
指向控制 MQCONNX 呼叫動作的選項的指標。

請參閱第 282 頁的『MQCNO-連接選項』，以取得詳細資料。

對於結束函數 MQXF_CONN， pConnectOpts 會指向預設連接選項結構 (MQCNO_DEFAULT)。

pHconn (PMQHCONN)-輸入
指向連線控點的指標。

CompCode (MQLONG)-輸入/輸出
完成碼，有效值為:

MQCC_OK
順利完成。

MQCC_WARNING
警告 (局部完成)

MQCC_FAILED
通話失敗

原因 (MQLONG)-輸入/輸出
定義完成碼的原因碼。

如果完成碼為 MQCC_OK，則唯一有效值為:

MQRC_NONE
(0， x '000') 沒有報告的原因。

如果完成碼是 MQCC_FAILED 或 MQCC_WARNING，則結束函數可以將原因碼欄位設為任何有效的 MQRC_ * 值。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
        MQAXP           ExitParms;      /* Exit parameter structure */
        MQAXC           ExitContext;    /* Exit context structure */
        PMQCHAR         pQMgrName;      /* Ptr to Queue manager name */
        PMQCNO          pConnectOpts;   /* Ptr to Connection options */
        PMQHCONN        pHconn;         /* Ptr to Connection handle */
        MQLONG          CompCode;       /* Completion code */
        MQLONG          Reason;         /* Reason code */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
MQ_CONNX_EXIT (&ExitParms, &ExitContext, &pQMgrName, &pConnectOps,
               &pHconn, &CompCode, &Reason);
```

您的結束程式必須符合下列 C 函數原型:

```
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 */
PPMQCNO         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 */
```

使用注意事項

1. 這裡說明的 MQ_CONNX_EXIT 函數介面同時用於 MQCONN 呼叫和 MQCONNX 呼叫。 不過，這兩個呼叫會定義個別進入點。 若要截取 兩個 呼叫， MQXEP 呼叫必須至少使用兩次-一次與函數 ID MQXF_CONN 搭配使用，一次與 MQXF_CONNX 搭配使用。

   因為 MQCONN 和 MQCONNX 的 MQ_CONNX_EXIT 介面相同，所以單一結束程式函數可以用於這兩個呼叫; MQAXP 結構中的 *Function* 欄位會指出進行中的呼叫。 或者， MQXEP 呼叫可用來登錄兩個呼叫的不同結束程式函數。
2. 當訊息通道代理程式 (MCA) 回應入埠用戶端連線時， MCA 可以在用戶端狀態完全已知之前發出許多 MQ 呼叫。 這些 MQ 呼叫會導致使用 MQAXC 結構來呼叫 API 結束程式函數，該結構包含與 MCA 相關的資料，而不是與用戶端 (例如，使用者 ID 及連線名稱) 相關的資料。 不過，一旦完全知道用戶端狀態，後續的 MQ 呼叫會導致以 MQAXC 結構中的適當用戶端資料來呼叫 API 結束程式函數。
3. 在佇列管理程式執行任何參數驗證之前，會先呼叫所有 MQXR_BEFORE 結束函數。 因此參數可能無效 (包括參數位址的無效指標)。

   在佇列管理程式執行任何授權檢查之前，會先呼叫 MQ_CONNX_EXIT 函數。
4. exit 函數不得變更 MQCONN 或 MQCONNX 呼叫中指定的佇列管理程式名稱。 如果跳出函數變更名稱，則結果未定義。
5. MQ_CONNX_EXIT 的 MQXR_BEFORE 結束函數無法發出 MQXEP 以外的 MQ 呼叫。

控制回呼-*MQ_CTL_EXIT*

MQ_CTL_EXIT 提供訂閱要求結束程式函數，以在控制回呼處理 之前 及 之後 執行。 將函數 ID MQXF_CTL 與結束原因 MQXR_BEFORE 及 MQXR_AFTER 搭配使用，以在控制回呼結束程式函數 之前 及 之後 登錄。

此功能的介面為:

```
MQ_CTL_EXIT (&Hconn, &Operation, &ControlOpts, &CompCode, &Reason)
```

其中參數為:

Hconn (MQHCONN)-輸入/輸出
連線控點。

作業 (MQLONG) 輸入/輸出
在針對指定物件控點定義的回呼上正在處理的作業

ControlOpts (MQCTLO) 輸入/輸出
控制 MQCTL 動作的選項

CompCode (MQLONG)-輸入/輸出
完成碼，有效值為:

MQCC_OK
順利完成。

MQCC_WARNING
局部完成。

MQCC_FAILED
通話失敗

原因 (MQLONG)-輸入/輸出
定義完成碼的原因碼。

如果完成碼為 MQCC_OK，則唯一有效值為:

MQRC_NONE
(0， x '000') 沒有報告的原因。

如果完成碼是 MQCC_FAILED 或 MQCC_WARNING，則結束函數可以將原因碼欄位設為任何有效的 MQRC_ * 值。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
MQ_CTL_EXIT (&Hconn, &Operation, &ControlOpts, &CompCode, &Reason);
```

您的結束程式必須符合下列 C 函數原型:

```
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 */
```

斷線-*MQ_DISC_EXIT*

MQ_DISK_EXIT 提供中斷連線結束程式功能，以執行 之前 及 之後 MQDISC 結束程式處理。 使用函數 ID MQXF_DISC 與結束原因 MQXR_BEFORE 及 MQXR_AFTER，以在 MQDISC 呼叫結束函數 之前 及 之後 登錄。

此函數的介面是

```
MQ_DISC_EXIT (&ExitParms, &ExitContext, &pHconn,
              &CompCode, &Reason);
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

pHconn (PMQHCONN)-輸入
指向連線控點的指標。

對於 *MQDISC* 呼叫之前，此欄位的值為下列其中一項:

- MQCONN 或 MQCONNX 呼叫傳回的連線控點
- 零，適用於環境特定配接器已連接至佇列管理程式的環境
- 前一個結束函數呼叫所設定的值

對於 *MQDISC* 呼叫之後，此欄位的值是零或前一個結束函數呼叫所設定的值。

CompCode (MQLONG)-輸入/輸出
完成碼，有效值為:

MQCC_OK
順利完成。

MQCC_WARNING
局部完成

MQCC_FAILED
通話失敗

原因 (MQLONG)-輸入/輸出
定義完成碼的原因碼。

如果完成碼為 MQCC_OK ，則唯一有效值為:

MQRC_NONE
(0 ， x '000') 沒有報告的原因。

如果完成碼是 MQCC_FAILED 或 MQCC_WARNING ，則結束函數可以將原因碼欄位設為任何有效的 MQRC_ * 值。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
MQAXP           ExitParms;      /* Exit parameter structure */
MQAXC           ExitContext;    /* Exit context structure */
PMQHCONN        pHconn;         /* Ptr to Connection handle */
MQLONG          CompCode;       /* Completion code */
MQLONG          Reason;         /* Reason code */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
MQ_DISC_EXIT (&ExitParms, &ExitContext, &pHconn,
              &CompCode, &Reason);
```

您的結束程式必須符合下列 C 函數原型:

```
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 */
```

取得-*MQ_GET_EXIT*

MQ_GET_EXIT 提供 get exit 函數來執行 之前 及 之後 MQGET 呼叫處理。

有兩個函數 ID:

1. 搭配使用 MQXF_GET 與結束原因 MQXR_BEFORE 及 MQXR_AFTER ， 以登錄 之前 及 之後 MQGET 呼叫結束程式函數。
2. 如需使用 MQXF_DATA_CONV_ON_GET 函數 ID 的相關資訊，請參閱 第 1000 頁的『MQXF_DATA_CONV_ON_GET』 。

此功能的介面為:

```
MQ_GET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
             &pGetMsgOpts, &BufferLength, &pBuffer, &pDataLength,
             &CompCode, &Reason)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

Hobj (MQHOBJ)-輸入/輸出
物件控點。

pMsg 說明 (PMQMD)-輸入/輸出
訊息描述子的指標。

pGetMsgOpts (PMQGMO)-輸入/輸出
取得訊息選項的指標。

BufferLength (MQLONG)-輸入/輸出
訊息緩衝區長度。

pBuffer (PMQBYTE)-輸入/輸出
訊息緩衝區的指標。

pData 長度 (PMQLONG)-輸入/輸出
資料長度欄位的指標。

CompCode (MQLONG)-輸入/輸出
完成碼，有效值為:

MQCC_OK
順利完成。

MQCC_WARNING
局部完成。

MQCC_FAILED
通話失敗

原因 (MQLONG)-輸入/輸出
定義完成碼的原因碼。

如果完成碼為 MQCC_OK，則唯一有效值為:

MQRC_NONE
(0， x '000') 沒有報告的原因。

如果完成碼是 MQCC_FAILED 或 MQCC_WARNING，則結束函數可以將原因碼欄位設為任何有效的 MQRC_ * 值。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
MQ_GET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
             &pGetMsgOpts, &BufferLength, &pBuffer, &pDataLength,
             &CompCode, &Reason)
```

您的結束程式必須符合下列 C 函數原型:

```
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 */
PPMQLONG        ppDataLength,   /* Address of ptr to data length field */
PMQLONG         pCompCode,      /* Address of completion code */
PMQLONG         pReason);       /* Address of reason code qualifying
                                   completion code */
```

使用注意事項

1. 這裡說明的 MQ_GET_EXIT 函數介面同時用於 MQXF_GET 結束函數及 第 1000 頁的『MQXF_DATA_CONV_ON_GET』 結束函數。

   這兩個結束函數會定義個別進入點，因此若要截取 兩者 ，必須使用 MQXEP 呼叫兩次; 對於此呼叫，請使用函數 ID MQXF_GET。

   因為 MQXF_GET_EXIT 介面與 MQXF_GET 及 MQXF_DATA_CONV_ON_GET 相同，所以單一結束函數可用於兩者; MQAXP 結構中的 *Function* 欄位指出已呼叫哪個結束函數。 或者， MQXEP 呼叫可用來針對這兩種情況登錄不同的結束函數。

MQXF_DATA_CONV_ON_GET

如需此呼叫的介面及範例 C 語言宣告的相關資訊，請參閱 MQ_GET_EXIT 。

使用注意事項

如果已登錄，則在訊息到達應用程式時，但在發生任何資料轉換之前，會呼叫此進入點。 如果 API 結束程式需要在將訊息傳遞至資料轉換之前執行處理 (例如解密或解壓縮) ，則這可能很有用。 必要的話，該結束程式會傳回 MQXCC_SUPPRESS_FUNCTION，導致略過資料轉換; 如需相關資訊，請參閱 MQAXP 結構。

在用戶端上登錄此進入點會導致在本端用戶端機器上執行資料轉換。 因此，為了正確作業，可能需要在用戶端上安裝應用程式轉換結束程式。 請注意， MQXF_DATA_CONV_ON_GET 也用於非同步使用。

使用 MQ_GET_EXIT 呼叫時，請使用 MQXF_DATA_CONV_ON_GET (結束原因為 MQXR_BEFORE) 來登錄 *before* MQGET 資料轉換結束程式函數。

MQXF_DATA_CONV_ON_GET 沒有 MQXR_AFTER 結束函數 ;MQXF_GET 的 MQXR_AFTER 結束函數提供在資料轉換之後結束處理所需的功能。

針對 MQ_GET_EXIT 呼叫定義個別進入點，因此若要截取 這兩個 結束函數， MQXEP 呼叫必須使用兩次; 針對此呼叫，請使用函數 ID MQXF_DATA_CONV_ON_GET。

因為 MQXF_GET_EXIT 介面與 MQXF_GET 及 MQXF_DATA_CONV_ON_GET 相同，所以單一結束函數可用於兩者; MQAXP 結構中的 *Function* 欄位指出已呼叫哪個結束函數。 或者， MQXEP 呼叫可用來針對這兩種情況登錄不同的結束函數。

起始設定-*MQ_INIT_EXIT*

MQ_INIT_EXIT 提供連線層次起始設定，方法是將 MQAXP 中的 ExitReason 設為 MQXR_CONNECTION。

在起始設定期間，請注意下列事項:

- MQ_INIT_EXIT 函數會呼叫 MQXEP 來登錄 WebSphere MQ API 動詞及其感興趣的 ENTRY 和 EXIT 點。
- 結束程式不需要截取所有 WebSphere MQ API 動詞。 只有在已登錄興趣時，才會呼叫結束函數。
- 在起始設定結束程式時，可以獲得結束程式要使用的儲存體。
- 如果此函數的呼叫失敗，則呼叫它的 MQCONN 或 MQCONNX 呼叫也會失敗，並具有 CompCode 及「原因」，取決於 MQAXP 中 ExitResponse 欄位的值。
- MQ_INIT_EXIT 結束程式不得發出 WebSphere MQ API 呼叫，因為此時尚未設定正確的環境。
- 如果 MQ_INIT_EXIT 因 MQXCC_FAILED 而失敗，則佇列管理程式會從 MQCONN 或 MQCONNX 呼叫它時傳回 MQCC_FAILED 及 MQRC_API_EXIT_ERROR。
- 在呼叫第一個 MQ_INIT_EXIT 之前，如果佇列管理程式在起始設定 API 結束程式函數執行環境時發生錯誤，則會從呼叫 MQCC_FAILED 及 MQRC_API_EXIT_INIT_ERROR 的 MQCONN 或 MQCONNX 呼叫中傳回佇列管理程式。

MQ_INIT_EXIT 的介面為:

```
MQ_INIT_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

CompCode (MQLONG)-輸入/輸出
完成碼的指標，有效值為:

MQCC_OK
順利完成。

MQCC_WARNING
局部完成。

MQCC_FAILED
通話失敗

原因 (MQLONG)-輸入/輸出
定義完成碼的原因碼指標。

如果完成碼為 MQCC_OK ，則唯一有效值為:

MQRC_NONE
(0 ， x '000') 沒有報告的原因。

如果完成碼是 MQCC_FAILED 或 MQCC_WARNING ，則結束函數可以將原因碼欄位設為任何有效的 MQRC_ * 值。

傳回給應用程式的 CompCode 和「原因」取決於 MQAXP 中 ExitResponse 欄位的值。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
MQAXP           ExitParms;      /* Exit parameter structure */
MQAXC           ExitContext;    /* Exit context structure */
MQLONG          CompCode;       /* Completion code */
MQLONG          Reason;         /* Reason code */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
MQ_INIT_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

您的結束程式必須符合下列 C 函數原型:

```
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 */
```

使用注意事項

1. MQ_INIT_EXIT 函數可以發出 MQXEP 呼叫，以登錄要截取之特定 MQ 呼叫的結束函數位址。 不需要截取所有 MQ 呼叫，或截取 MQXR_BEFORE 及 MQXR_AFTER 呼叫。 例如，結束程式套組可以選擇僅截取 MQPUT 的 MQXR_before 呼叫。
2. 由結束套組中結束函數使用的儲存體可以由 MQ_INIT_EXIT 函數獲得。 或者，結束程式函數可以在必要時呼叫它們時獲得儲存體。 不過，在結束套組終止之前，應該先釋放所有儲存體 ;MQ_TERM_EXIT 函數可以釋放儲存體，或先前呼叫的結束函數。
3. 如果 MQAXP 的 *ExitResponse* 欄位中 MQ_INIT_EXIT 傳回 MQXCC_FAILED ，或以其他方式失敗，則導致呼叫 MQ_INIT_EXIT 的 MQCONN 或 MQCONNX 呼叫也會失敗，並將 *CompCode* 及 *Reason* 參數設為適當的值。
4. MQ_INIT_EXIT 函數無法發出 MQXEP 以外的 MQ 呼叫。

查詢-*MQ_INQ_EXIT*

MQ_INQ_EXIT 提供查詢結束函數，以執行 之前 及 之後 MQINQ 呼叫處理。 將函數 ID MQXF_INQ 與結束原因 MQXR_BEFORE 及 MQXR_AFTER 搭配使用，以在 MQINQ 呼叫結束函數 之前 及 之後 登錄。

此功能的介面為:

```
MQ_INQ_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
             &pCharAttrs, &CompCode, &Reason)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

Hobj (MQHOBJ)-輸入
物件控點。

SelectorCount (MQLONG)-輸入
選取元計數

pSelectors (PMQLONG)-輸入/輸出
指向選取元值陣列的指標。

IntAttr 計數 (MQLONG)-輸入
整數屬性計數。

pInt 屬性 (PMQLONG)-輸入/輸出
整數屬性值陣列的指標。

CharAttr 長度 (MQLONG)-輸入/輸出
字元屬性陣列長度。

pChar 屬性 (PMQCHAR)-輸入/輸出
字元屬性陣列的指標。

CompCode (MQLONG)-輸入/輸出
完成碼，有效值為:

MQCC_OK
順利完成。

MQCC_WARNING
局部完成。

MQCC_FAILED
通話失敗

原因 (MQLONG)-輸入/輸出
定義完成碼的原因碼。

如果完成碼為 MQCC_OK ， 則唯一有效值為:

MQRC_NONE
(0 ， x '000') 沒有報告的原因。

如果完成碼是 MQCC_FAILED 或 MQCC_WARNING ， 則結束函數可以將原因碼欄位設為任何有效的 MQRC_ * 值。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
MQ_INQ_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
             &pCharAttrs, &CompCode, &Reason)
```

您的結束程式必須符合下列 C 函數原型:

```
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 */
```

開啟-*MQ_OPEN_EXIT*

MQ_OPEN_EXIT 提供可執行 之前 及 之後 MQOPEN 呼叫處理的開啟結束程式函數。 使用具有結束原因 MQXR_BEFORE 及 MQXR_AFTER 的函數 ID MQXF_OPEN 來登錄 之前 及 之後 MQOPEN 呼叫結束程式函數。

此函數的介面是

```
MQ_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &Options,
              &pHobj, &CompCode, &Reason)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

pObj 說明 (PMQOD)-輸入/輸出
指向物件描述子的指標。

選項 (MQLONG)-輸入/輸出
開啟選項。

pHobj (PMQHOBJ)-輸入
指向物件控點的指標。

CompCode (MQLONG)-輸入/輸出
完成碼，有效值為:

MQCC_OK
順利完成。

MQCC_WARNING
局部完成

MQCC_FAILED
通話失敗

原因 (MQLONG)-輸入/輸出
定義完成碼的原因碼。

如果完成碼為 MQCC_OK，則唯一有效值為:

MQRC_NONE
(0， x '000') 沒有報告的原因。

如果完成碼是 MQCC_FAILED 或 MQCC_WARNING，則結束函數可以將原因碼欄位設為任何有效的 MQRC_ * 值。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
MQAXP           ExitParms;      /* Exit parameter structure */
MQAXC           ExitContext;    /* Exit context structure */
MQHCONN         Hconn;          /* Connection handle */
PMQOD           pObjDesc;       /* Ptr to object descriptor */
MQLONG          Options;        /* Open options */
PMQHOBJ         pHobj;          /* Ptr to object handle */
MQLONG          CompCode;       /* Completion code */
MQLONG          Reason;         /* Reason code */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
MQ_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &Options,
              &pHobj, &CompCode, &Reason);
```

您的結束程式必須符合下列 C 函數原型:

```
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 */
PPMQHOBJ        ppHobj,         /* Address of ptr to object handle */
PMQLONG         pCompCode,      /* Address of completion code */
```

```
PMQLONG         pReason);       /* Address of reason code qualifying
                                   completion code */
```

放置-*MQ_PUT_EXIT*

MQ_PUT_EXIT 提供放置結束函數，以執行 之前 及 之後 MQPUT 呼叫處理。 搭配使用函數 ID MQXF_PUT 與結束原因 MQXR_BEFORE 及 MQXR_AFTER ， 以登錄 之前 及 之後 MQPUT 呼叫結束程式函數。

此功能的介面為:

```
MQ_PUT_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
             &pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

Hobj (MQHOBJ)-輸入/輸出
物件控點。

pMsg 說明 (PMQMD)-輸入/輸出
訊息描述子的指標。

pPutMsgOpts (PMQPMO)-輸入/輸出
放置訊息選項的指標。

BufferLength (MQLONG)-輸入/輸出
訊息緩衝區長度。

pBuffer (PMQBYTE)-輸入/輸出
訊息緩衝區的指標。

CompCode (MQLONG)-輸入/輸出
完成碼，有效值為:

MQCC_OK
順利完成。

MQCC_WARNING
局部完成。

MQCC_FAILED
通話失敗

原因 (MQLONG)-輸入/輸出
定義完成碼的原因碼。

如果完成碼為 MQCC_OK ，則唯一有效值為:

MQRC_NONE
(0 ， x '000') 沒有報告的原因。

如果完成碼是 MQCC_FAILED 或 MQCC_WARNING ，則結束函數可以將原因碼欄位設為任何有效的 MQRC_ * 值。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
        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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
MQ_PUT_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
             &pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)
```

您的結束程式必須符合下列 C 函數原型:

```
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 */
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 */
```

使用注意事項

- 佇列管理程式所產生的報告訊息會跳過一般呼叫處理。 因此， MQ_PUT_EXIT 函數或 MQPUT1 函數無法截取這類訊息。 不過，訊息通道代理程式所產生的報告訊息會正常處理，因此可能會被 MQ_PUT_EXIT 函數或 MQ_PUT1_EXIT 函數截取。 若要確保截取 MCA 所產生的所有報告訊息，應該同時使用 MQ_PUT_EXIT 及 MQ_PUT1_EXIT 。

Put1 - MQ_PUT1_EXIT

MQ_PUT1_EXIT 提供 只放置一則訊息 結束函數來執行 之前 和 之後 MQPUT1 呼叫處理。 搭配使用函數 ID MQXF_PUT1 與結束原因 MQXR_BEFORE 及 MQXR_AFTER ， 以登錄 之前 及 之後 MQPUT1 呼叫結束程式函數。

此功能的介面為:

```
MQ_PUT1_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &pMsgDesc,
              &pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

pObj 說明 (PMQOD)-輸入/輸出
指向物件描述子的指標。

pMsg 說明 (PMQMD)-輸入/輸出
訊息描述子的指標。

pPutMsgOpts (PMQPMO)-輸入/輸出
放置訊息選項的指標。

BufferLength (MQLONG)-輸入/輸出
訊息緩衝區長度。

pBuffer (PMQBYTE)-輸入/輸出
訊息緩衝區的指標。

CompCode (MQLONG)-輸入/輸出
完成碼，有效值為:

MQCC_OK
順利完成。

MQCC_WARNING
局部完成。

MQCC_FAILED
通話失敗

原因 (MQLONG)-輸入/輸出
定義完成碼的原因碼。

如果完成碼為 MQCC_OK ，則唯一有效值為:

MQRC_NONE
(0 ， x '000') 沒有報告的原因。

如果完成碼是 MQCC_FAILED 或 MQCC_WARNING ，則結束函數可以將原因碼欄位設為任何有效的 MQRC_ * 值。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
MQ_PUT1_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &pMsgDesc,
              &pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)
```

您的結束程式必須符合下列 C 函數原型:

```
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 */
```

設定-*MQ_SET_EXIT*

MQ_SET_EXIT 提供設定結束函數，以執行 之前 及 之後 MQSET 呼叫處理。 使用函數 ID MQXF_SET 與結束原因 MQXR_BEFORE 及 MQXR_AFTER 來登錄 之前 及 之後 MQSET 呼叫結束函數。

此功能的介面為:

```
MQ_SET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
             &pCharAttr, &CompCode, &Reason)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

Hobj (MQHOBJ)-輸入
物件控點。

SelectorCount (MQLONG)-輸入
選取元計數

pSelectors (PMQLONG)-輸入/輸出
指向選取元值陣列的指標。

IntAttr 計數 (MQLONG)-輸入
整數屬性計數。

pInt 屬性 (PMQLONG)-輸入/輸出
整數屬性值陣列的指標。

CharAttr 長度 (MQLONG)-輸入/輸出
字元屬性陣列長度。

pChar 屬性 (PMQCHAR)-輸入/輸出
字元屬性值的指標。

CompCode (MQLONG)-輸入/輸出
完成碼，有效值為:

MQCC_OK
順利完成。

MQCC_WARNING
局部完成。

MQCC_FAILED
通話失敗

原因 (MQLONG)-輸入/輸出
定義完成碼的原因碼。

如果完成碼為 MQCC_OK，則唯一有效值為:

MQRC_NONE
(0， x '000') 沒有報告的原因。

如果完成碼是 MQCC_FAILED 或 MQCC_WARNING，則結束函數可以將原因碼欄位設為任何有效的 MQRC_ * 值。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
MQ_SET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
             &pCharAttrs, &CompCode, &Reason)
```

您的結束程式必須符合下列 C 函數原型:

```
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 */
```

狀態-*MQ_STAT_EXIT*

MQ_STAT_EXIT 提供狀態結束函數來執行 之前 及 之後 MQSTAT 呼叫處理。 使用函數 ID MQXF_STAT 與結束原因 MQXR_BEFORE 及 MQXR_AFTER 來登錄 之前 及 之後 MQSTAT 呼叫結束函數。

此功能的介面為:

```
MQ_STAT_EXIT (&ExitParms, &ExitContext, &Hconn, &Type, &pStatus
              &CompCode, &Reason)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

類型 (MQLONG)-輸入
要擷取的狀態資訊類型。

pStatus (PMQSTS)-輸出
狀態緩衝區的指標。

CompCode (MQLONG)-輸入/輸出
完成碼，有效值為:

MQCC_OK
順利完成。

MQCC_WARNING
局部完成。

MQCC_FAILED
通話失敗

原因 (MQLONG)-輸入/輸出
定義完成碼的原因碼。

如果完成碼為 MQCC_OK ，則唯一有效值為:

MQRC_NONE
(0 ， x '000') 沒有報告的原因。

如果完成碼是 MQCC_FAILED 或 MQCC_WARNING ，則結束函數可以將原因碼欄位設為任何有效的 MQRC_ * 值。

C 語言呼叫

您的結束程式必須符合下列 C 函數原型:

```
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 */
```

終止-*MQ_TERM_EXIT*

MQ_TERM_EXIT 提供以函數 ID MQXF_TERM 及 ExitReason MQXR_CONNECTION 登錄的連線層次終止。如果已登錄，則會針對每個斷線要求呼叫一次 MQ_TERM_EXIT。

作為終止的一部分，可以釋放結束程式不再需要的儲存體，並且可以執行任何必要的清除。

如果 MQ_TERM_EXIT 因 MQXCC_FAILED 而失敗，則佇列管理程式會從以 MQCC_FAILED 及 MQRC_API_EXIT_ERROR 呼叫它的 MQDISC 返回。

在呼叫最後一個 MQ_TERM_EXIT 之後，如果佇列管理程式在終止 API 結束程式函數執行環境時發生錯誤，則佇列管理程式會從呼叫 MQ_TERM_EXIT 且 MQCC_FAILED 及 MQRC_API_EXIT_TERM_ERROR 的 MQDISC 呼叫傳回。

此功能的介面為:

```
MQ_TERM_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

CompCode (MQLONG)-輸入/輸出
完成碼，有效值為:

MQCC_OK
順利完成。

MQCC_FAILED
通話失敗

原因 (MQLONG)-輸入/輸出
定義完成碼的原因碼。

如果完成碼為 MQCC_OK，則唯一有效值為:

MQRC_NONE
(0， x '000') 沒有報告的原因。

如果完成碼是 MQCC_FAILED，則結束函數可以將原因碼欄位設為任何有效的 MQRC_ * 值。

傳回給應用程式的 CompCode 和「原因」取決於 MQAXP 中 ExitResponse 欄位的值。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
MQAXP           ExitParms;      /* Exit parameter structure */
MQAXC           ExitContext;    /* Exit context structure */
MQLONG          CompCode;       /* Completion code */
MQLONG          Reason;         /* Reason code */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
MQ_TERM_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

您的結束程式必須符合下列 C 函數原型:

```
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 */
```

使用注意事項

1. MQ_TERM_EXIT 函數是選用的。 如果沒有要執行的終止處理程序，則結束套組不需要登錄終止結束程式。

   如果屬於結束套組的函數在連線期間獲得資源，則 MQ_TERM_EXIT 函數是釋放這些資源 (例如，釋放動態取得的儲存體) 的方便點。
2. 如果在發出 MQDISC 呼叫時登錄 MQ_TERM_EXIT 函數，則會在呼叫所有 MQDISC 結束函數之後呼叫結束函數。
3. 如果 MQ_TERM_EXIT 在 MQAXP 的 *ExitResponse* 欄位中傳回 MQXCC_FAILED ，或以其他方式失敗，則導致呼叫 MQ_TERM_EXIT 的 MQDISC 呼叫也會失敗，並將 *CompCode* 及 *Reason* 參數設為適當的值。

登錄訂閱-*MQ_SUB_EXIT*

MQ_SUB_EXIT 提供結束函數來執行 之前 及 之後 訂閱重新登錄處理程序。 搭配使用函數 ID MQXF_SUB 與結束原因 MQXR_BEFORE 及 MQXR_AFTER ， 以在 之前及 之後 登錄訂閱登錄呼叫結束程式函數。

此功能的介面為:

```
MQ_SUB_EXIT (&ExitParms, &ExitContext, &Hconn, &pSubDesc, &pHobj, &pHsub, &CompCode, &Reason)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入/輸出
連線控點。

pSub 說明輸入/輸出
屬性選取元的陣列。

pHobj -輸入/輸出
物件控點

pHsub (MQHOBJ) 輸入/輸出
訂閱控點

CompCode (MQLONG)-輸入/輸出
完成碼，有效值為:

MQCC_OK
順利完成。

MQCC_WARNING
局部完成。

MQCC_FAILED
通話失敗

原因 (MQLONG)-輸入/輸出
定義完成碼的原因碼。

如果完成碼為 MQCC_OK ， 則唯一有效值為:

MQRC_NONE
(0 ， x '000') 沒有報告的原因。

如果完成碼是 MQCC_FAILED 或 MQCC_WARNING ， 則結束函數可以將原因碼欄位設為任何有效的 MQRC_ * 值。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
MQAXP    ExitParms;      /* Exit parameter structure */
MQAXC    ExitContext;    /* Exit context structure */
MQHCONN  Hconn;          /* Connection handle */
PMQSD    pSubDesc;       /* Subscription descriptor */
PMQHOBJ  pHobj;          /* Object Handle */
PMQHOBJ  pHsub;          /* Subscription handle */
MQLONG   CompCode;       /* Completion code */
MQLONG   Reason;         /* Reason code qualifying completion code */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
MQ_SUB_EXIT (&ExitParms, &ExitContext, &Hconn, &pSubDesc, &pHobj, &pHsub,
 &CompCode, &Reason);
```

您的結束程式必須符合下列 C 函數原型:

```
PMQAXP    pExitParms;      /* Exit parameter structure */
PMQAXC    pExitContext;    /* Exit context structure */
PMQHCONN  pHconn;          /* Connection handle */
PPMQSD    ppSubDesc;       /* Subscription descriptor */
PPMQHOBJ  ppHobj;          /* Object Handle */
PPMQHOBJ  ppHsub;          /* Subscription handle */
PMQLONG   pCompCode;       /* Completion code */
PMQLONG   pReason;         /* Reason code qualifying completion code */
```

訂閱要求-*MQ_SUBRQ_EXIT*

MQ_SUBRQ_EXIT 提供訂閱要求結束程式函數，以執行 之前 及 之後 訂閱要求處理。 搭配使用函數 ID MQXF_SUBRQ 與結束原因 MQXR_BEFORE 及 MQXR_AFTER ， 以在 之前及 之後 訂閱要求呼叫結束程式函數。

此功能的介面為:

```
MQ_SUBRQ_EXIT (&ExitParms, &ExitContext, &Hconn, &pHsub, &Action, &pSubRqOpts,
    &CompCode, &Reason)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入/輸出
連線控點。

pHsub (MQHOBJ) 輸入/輸出
訂閱控點

動作 (MQLONG) 輸入/輸出
動作

pSubRqOpts (MQSRO) 輸入/輸出

CompCode (MQLONG)-輸入/輸出
完成碼，有效值為:

MQCC_OK
順利完成。

MQCC_WARNING
局部完成。

MQCC_FAILED
通話失敗

原因 (MQLONG)-輸入/輸出
定義完成碼的原因碼。

如果完成碼為 MQCC_OK ，則唯一有效值為:

MQRC_NONE
(0 ， x '000') 沒有報告的原因。

如果完成碼是 MQCC_FAILED 或 MQCC_WARNING ，則結束函數可以將原因碼欄位設為任何有效的 MQRC_ * 值。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
MQ_SUBRQ_EXIT (&ExitParms, &ExitContext, &Hconn, &pHsub, &Action, &pSubRqOpts,
 &CompCode, &Reason);
```

您的結束程式必須符合下列 C 函數原型:

```
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_EXIT 提供 xa_close 結束函數，以在 xa_close 處理之前及之後執行。 使用具有結束原因 MQXR_BEFORE 及 MQXR_AFTER 的函數 ID MQXF_XACLOSE 來登錄之前及之後 xa_close 呼叫結束函數。

此功能的介面為:

```
XA_CLOSE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

pXa_info (PMQCHAR)-輸入/輸出
實例特定的資源管理程式資訊。

Rmid (MQLONG)-輸入/輸出
資源管理程式 ID。

旗標 (MQLONG)-輸入/輸出
資源管理程式選項。

XARetCode (MQLONG)-輸入/輸出
XA 呼叫的回應。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
     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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
XA_CLOSE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode);
```

您的結束程式必須符合下列 C 函數原型:

```
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_EXIT 提供 xa_commit 結束函數，以在 xa_commit 處理之前及之後執行。 搭配使用函數 ID MQXF_XACOMMIT 與結束原因 MQXR_BEFORE 及 MQXR_AFTER ， 以登錄之前及之後 xa_commit 呼叫結束函數。

此功能的介面為:

```
XA_COMMIT_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

pXID (MQPTR)-輸入/輸出
交易分支 ID。

Rmid (MQLONG)-輸入/輸出
資源管理程式 ID。

旗標 (MQLONG)-輸入/輸出
資源管理程式選項。

XARetCode (MQLONG)-輸入/輸出
XA 呼叫的回應。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
     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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
XA_COMMIT_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

您的結束程式必須符合下列 C 函數原型:

```
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_EXIT 提供 xa_complete 結束函數，以在 xa_complete 處理之前及之後執行。 使用具有結束原因 MQXR_BEFORE 及 MQXR_AFTER 的函數 ID MQXF_XACOMPLETE 來登錄之前及之後 xa_complete 呼叫結束函數。

此功能的介面為:

```
XA_COMPLETE_EXIT (&ExitParms, &ExitContext, &Hconn, &pHandle, &pRetval, &Rmid, &Flags,
&XARetCode)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

pHandle (PMQLONG)-輸入/輸出
指向非同步作業的指標。

pRetVal (PMQLONG)-輸入/輸出
非同步作業的回覆值。

Rmid (MQLONG)-輸入/輸出
資源管理程式 ID。

旗標 (MQLONG)-輸入/輸出
資源管理程式選項。

XARetCode (MQLONG)-輸入/輸出
XA 呼叫的回應。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
     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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
XA_COMPLETE_EXIT (&ExitParms, &ExitContext, &Hconn, &pHandle, &pRetval, &Rmid, &Flags,
&XARetCode);
```

您的結束程式必須符合下列 C 函數原型:

```
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_EXIT 提供 xa_end 結束函數，以在 xa_end 處理之前及之後執行。 使用函數 ID MQXF_XAEND 與結束原因 MQXR_BEFORE 及 MQXR_AFTER 來登錄之前及之後 xa_end 呼叫結束函數。

此功能的介面為:

```

XA_END_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

pXID (MQPTR)-輸入/輸出
交易分支 ID。

Rmid (MQLONG)-輸入/輸出
資源管理程式 ID。

旗標 (MQLONG)-輸入/輸出
資源管理程式選項。

XARetCode (MQLONG)-輸入/輸出
XA 呼叫的回應。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
     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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
XA_END_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

您的結束程式必須符合下列 C 函數原型:

```
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_EXIT 提供 xa_forget 結束函數，以在 xa_forget 處理之前及之後執行。 使用具有結束原因 MQXR_BEFORE 及 MQXR_AFTER 的函數 ID MQXF_XAFORGET ，以登錄之前及之後 xa_forget 呼叫結束函數。

此功能的介面為:

```

XA_FORGET_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

pXID (MQPTR)-輸入/輸出
交易分支 ID。

Rmid (MQLONG)-輸入/輸出
資源管理程式 ID。

旗標 (MQLONG)-輸入/輸出
資源管理程式選項。

XARetCode (MQLONG)-輸入/輸出
XA 呼叫的回應。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
     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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
XA_FORGET_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

您的結束程式必須符合下列 C 函數原型:

```
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_EXIT 提供 xa_open 結束函數，以在 xa_open 處理之前及之後執行。 搭配使用函數 ID MQXF_XAOPEN 與結束原因 MQXR_BEFORE 及 MQXR_AFTER ， 以登錄之前及之後 xa_open 呼叫結束程式函數。

此功能的介面為:

```
XA_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

pXa_info (PMQCHAR)-輸入/輸出
實例特定的資源管理程式資訊。

Rmid (MQLONG)-輸入/輸出
資源管理程式 ID。

旗標 (MQLONG)-輸入/輸出
資源管理程式選項。

XARetCode (MQLONG)-輸入/輸出
XA 呼叫的回應。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
     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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
XA_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode);
```

您的結束程式必須符合下列 C 函數原型:

```
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_EXIT 提供 xa_prepare 結束函數，以在 xa_prepare 處理之前及之後執行。 使用函數 ID MQXF_XAPREPARE 與結束原因 MQXR_BEFORE 及 MQXR_AFTER 來登錄之前及之後 xa_prepare 呼叫結束函數。

此功能的介面為:

```
XA_PREPARE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

pXID (MQPTR)-輸入/輸出
交易分支 ID。

Rmid (MQLONG)-輸入/輸出
資源管理程式 ID。

旗標 (MQLONG)-輸入/輸出
資源管理程式選項。

XARetCode (MQLONG)-輸入/輸出
XA 呼叫的回應。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
     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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
XA_PREPARE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

您的結束程式必須符合下列 C 函數原型:

```
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_EXIT 提供 xa_recover 結束函數，以在 xa_recover 處理之前及之後執行。 搭配使用函數 ID MQXF_XARECOVER 與結束原因 MQXR_BEFORE 及 MQXR_AFTER ， 以登錄之前及之後 xa_recover 呼叫結束函數。

此功能的介面為:

```
XA_RECOVER_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Count, &Rmid, &Flags, &XARetCode)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

pXID (MQPTR)-輸入/輸出
交易分支 ID。

計數 (MQLONG)-輸入/輸出
XID 陣列中的 XID 數上限

Rmid (MQLONG)-輸入/輸出
資源管理程式 ID。

旗標 (MQLONG)-輸入/輸出
資源管理程式選項。

XARetCode (MQLONG)-輸入/輸出
XA 呼叫的回應。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
     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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
XA_RECOVER_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Count, &Rmid, &Flags, &XARetCode);
```

您的結束程式必須符合下列 C 函數原型:

```
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_EXIT 提供 xa_rollback 結束函數，以在 xa_rollback 處理之前及之後執行。 使用函數 ID MQXF_XAROLLBACK 與結束原因 MQXR_BEFORE 及 MQXR_AFTER ， 以登錄之前及之後 xa_rollback 呼叫結束函數。

此功能的介面為:

```
XA_ROLLBACK_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

pXID (MQPTR)-輸入/輸出
交易分支 ID。

Rmid (MQLONG)-輸入/輸出
資源管理程式 ID。

旗標 (MQLONG)-輸入/輸出
資源管理程式選項。

XARetCode (MQLONG)-輸入/輸出
XA 呼叫的回應。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
     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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
XA_ROLLBACK_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

您的結束程式必須符合下列 C 函數原型:

```
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_EXIT 提供 xa_start 結束函數，以在 xa_start 處理之前及之後執行。 使用具有結束原因 MQXR_BEFORE 及 MQXR_AFTER 的函數 ID MQXF_XASTART ， 以登錄之前及之後 xa_start 呼叫結束程式函數。

此功能的介面為:

```

XA_START_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

pXID (MQPTR)-輸入/輸出
交易分支 ID。

Rmid (MQLONG)-輸入/輸出
資源管理程式 ID。

旗標 (MQLONG)-輸入/輸出
資源管理程式選項。

XARetCode (MQLONG)-輸入/輸出
XA 呼叫的回應。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
     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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
XA_START_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

您的結束程式必須符合下列 C 函數原型:

```
typedef void MQENTRY XA_START_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 */
```

ax_reg-AX_REG_EXIT

AX_REG_EXIT 提供 ax_reg 結束函數，以在 ax_reg 處理之前及之後執行。 搭配使用函數 ID MQXF_AXREG 與結束原因 MQXR_BEFORE 及 MQXR_AFTER ， 以登錄 before 及 after ax_reg 呼叫結束函數。

此功能的介面為:

```
AX_REG_EXIT (&ExitParms, &ExitContext, &pXID, &Rmid, &Flags, &XARetCode)
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Hconn (MQHCONN)-輸入
連線控點。

pXID (MQPTR)-輸入/輸出
交易分支 ID。

Rmid (MQLONG)-輸入/輸出
資源管理程式 ID。

旗標 (MQLONG)-輸入/輸出
資源管理程式選項。

XARetCode (MQLONG)-輸入/輸出
XA 呼叫的回應。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
     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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
AX_REG_EXIT (&ExitParms, &ExitContext, &pXID, &Rmid, &Flags, &XARetCode);
```

您的結束程式必須符合下列 C 函數原型:

```
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_EXIT 提供 ax_unreg 結束函數，以在 ax_unreg 處理之前及之後執行。 使用函數 ID MQXF_AXUNREG 及結束原因 MQXR_BEFORE 和 MQXR_AFTER 來登錄 ax_unreg 呼叫結束函數之前及之後。

此功能的介面為:

```

AX_UNREG_EXIT (&ExitParms, &ExitContext, &Rmid, &Flags, &XARetCode);
```

其中參數為:

ExitParms (MQAXP)-輸入/輸出
結束參數結構。

ExitContext (MQAXC)-輸入/輸出
結束環境定義結構。

Rmid (MQLONG)-輸入/輸出
資源管理程式 ID。

旗標 (MQLONG)-輸入/輸出
資源管理程式選項。

XARetCode (MQLONG)-輸入/輸出
XA 呼叫的回應。

C 語言呼叫

佇列管理程式在邏輯上定義下列變數:

```
     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 */
```

然後，佇列管理程式會邏輯地呼叫結束程式，如下所示:

```
AX_UNREG_EXIT (&ExitParms, &ExitContext, &Rmid, &Flags, &XARetCode);
```

您的結束程式必須符合下列 C 函數原型:

```
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 */
```

呼叫結束程式函數的一般資訊

本主題提供一些一般指引，以協助您規劃結束程式，特別是與處理錯誤及非預期事件相關。

結束失敗

如果結束函數在破壞性 (不同步點) MQGET 呼叫之後，但在將訊息傳遞至應用程式之前，異常終止，則結束處理程式可以從失敗中回復，並將控制權傳遞至應用程式。

在此情況下，訊息可能會遺失。 這就像應用程式在接收來自佇列的訊息之後立即失敗時所發生的情況。

MQGET 呼叫可能以 MQCC_FAILED 和 MQRC_API_EXIT_ERROR 完成。

如果 之前 API 呼叫結束函數異常終止，則結束處理程式可以從失敗中回復，並將控制權傳遞給應用程式，而不處理 API 呼叫。 在此情況下，結束程式功能必須回復它所擁有的任何資源。

如果鏈結結束程式正在使用中，則可以自行驅動任何 之前 已順利驅動的 API 呼叫結束程式的 *after* API 呼叫結束程式。 API 呼叫可能會失敗，並出現 MQCC_FAILED 和 MQRC_API_EXIT_ERROR。

結束函數的錯誤處理範例

下圖顯示可能發生錯誤的點 (e*N*)。 它只是一個範例，顯示結束程式的行為方式，應該與下表一起讀取。 在此範例中，在每一個 API 呼叫之前及之後都會呼叫兩個結束函數，以顯示鏈結結束程式的行為。

```
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  -->
                  before MQ_OPEN_EXIT  1
            e8
                  before MQ_OPEN_EXIT  2
            e9
                                          -->  MQOPEN
            e10
                  after  MQ_OPEN_EXIT  2
            e11
                  after  MQ_OPEN_EXIT  1
            e12
        <--
MQPUT   -->
                  before MQ_PUT_EXIT   1
            e13
                  before MQ_PUT_EXIT   2
            e14
                                          -->  MQPUT
            e15
                  after  MQ_PUT_EXIT   2
            e16
                  after  MQ_PUT_EXIT   1
            e17
```

```
        <--
MQCLOSE -->
                    before MQ_CLOSE_EXIT 1
             e18
                    before MQ_CLOSE_EXIT 2
             e19
                                              -->  MQCLOSE
             e20
                    after  MQ_CLOSE_EXIT 2
             e21
                    after  MQ_CLOSE_EXIT 1
             e22
        <--
MQDISC  -->
                    before MQ_DISC_EXIT  1
             e23
                    before MQ_DISC_EXIT  2
             e24
                                              -->  MQDISC
             e25
                    after  MQ_DISC_EXIT  2
             e26
                    after  MQ_DISC_EXIT  1
             e27

        <--

end
```

下表列出每一個錯誤點要採取的動作。 僅涵蓋部分錯誤點，因為這裡顯示的規則可以套用至所有其他規則。它是在每一種情況下指定預期行為的動作。

表 *595: API* 結束程式錯誤及要採取的適當動作

ErrPt	**說明**	**動作**
e1	設定環境設定時發生錯誤。	1. 視需要復原環境設定 2. 磁碟機無結束程式函數 3. MQCONN 失敗與 MQCC_FAILED、MQRC_API_EXIT_LOAD_ERROR
e2	MQ_INIT_EXIT 函數完成，具有: • MQXCC_FAILED • MQXCC_ *	• 若為 MQXCC_FAILED: 1. 清除環境 2. MQCONN 失敗， MQCC_FAILED , MQRC_API_EXIT_INIT_ERROR • 適用於 MQXCC_ * 1. 作為 MQXCC_ * 和 MQXR2_ *[1] 的值 2. 清除環境
e3	在 MQ_CONNX_EXIT 1 函數完成之前: • MQXCC_FAILED • MQXCC_ *	• 若為 MQXCC_FAILED: 1. 磁碟機 MQ_TERM_EXIT 函數 2. 清除環境 3. MQCC_FAILED、MQRC_API_EXIT_ERROR 的 MQCONN 呼叫失敗 • 適用於 MQXCC_ * 1. 作為 MQXCC_ * 和 MQXR2_ *[1] 的值 2. 磁帶機 MQ_TERM_EXIT 函數 (必要的話) 3. 必要時清除環境

表 595: API 結束程式錯誤及要採取的適當動作 (繼續)

Err Pt	說明	動作
e4	在 MQ_CONNX_EXIT 2 函數完成之前: • MQXCC_FAILED • MQXCC_ *	• 若為 MQXCC_FAILED: 1. 磁碟機 之後 MQ_CONNX_EXIT 1 函數 2. 磁碟機 MQ_TERM_EXIT 函數 3. 清除環境 4. MQCC_FAILED、MQRC_API_EXIT_ERROR 的 MQCONN 呼叫失敗 • 適用於 MQXCC_ * 1. 作為 MQXCC_ * 和 MQXR2_ *[1] 的值 2. 如果未抑制結束程式，則驅動 之後 MQ_CONNX_EXIT 1 函數 3. 磁帶機 MQ_TERM_EXIT 函數 (必要的話) 4. 必要時清除環境
e5	MQCONN 呼叫失敗。	1. 傳遞 MQCONN CompCode 及原因 2. 如果 之前 MQ_CONNX_EXIT 2 成功且未抑制結束程式，則磁碟機 之後 MQ_CONNX_EXIT 2 函數 3. 如果 *before* MQ_CONNX_EXIT 1 成功且未抑制結束程式，則磁碟機 在 MQ_CONNX_EXIT 1 函數之後 4. 磁碟機 MQ_TERM_EXIT 函數 5. 清除環境
e6	在 MQ_CONNX_EXIT 2 函數完成之後: • MQXCC_FAILED • MQXCC_ *	• 若為 MQXCC_FAILED: 1. 磁碟機 之後 MQ_CONNX_EXIT 1 函數 2. 使用 MQCC_FAILED 完成 MQCONN 呼叫, MQRC_API_EXIT_ERROR • 適用於 MQXCC_ * 1. 作為 MQXCC_ * 和 MQXR2_ *[1] 的值 2. 磁帶機 之後 MQ_CONNX_EXIT 1 函數 (必要的話)
e7	之後 MQ_CONNX_EXIT 1 函數完成時: • MQXCC_FAILED • MQXCC_ *	• 若為 MQXCC_FAILED，請以 MQCC_FAILED、MQRC_API_EXIT_ERROR 完成 MQCONN 呼叫 • 若為 MQXCC_ *，請充當 MQXCC_ * 及 MQXR2_ *[1] 的值
e8	在 MQ_OPEN_EXIT 1 函數完成之前: • MQXCC_FAILED • MQXCC_ *	• 若為 MQXCC_FAILED，請完成 MQCC_FAILED、MQRC_API_EXIT_ERROR 的 MQOPEN 呼叫 • 若為 MQXCC_ *，請充當 MQXCC_ * 及 MQXR2_ *[1] 的值
e9	在 MQ_OPEN_EXIT 2 函數完成之前: • MQXCC_FAILED • MQXCC_ *	• 若為 MQXCC_FAILED: 1. 磁碟機 之後 MQ_OPEN_EXIT 1 函數 2. 使用 MQCC_FAILED 完成 MQOPEN 呼叫, MQRC_API_EXIT_ERROR • 若為 MQXCC_ *，請充當 MQXCC_ * 及 MQXR2_ *[1] 的值

表 595: API 結束程式錯誤及要採取的適當動作 (繼續)		
ErrPt	**說明**	**動作**
e10	MQOPEN 呼叫失敗	1. 傳遞 MQOPEN CompCode 和原因 2. 磁帶機 之後 MQ_OPEN_EXIT 2 函數 (如果未抑制結束程式) 3. 磁碟機 之後 MQ_OPEN_EXIT 1 函數 (如果結束程式未暫停，且如果鏈結結束程式未暫停)
e11	在 MQ_OPEN_EXIT 2 函數完成之後: • MQXCC_FAILED • MQXCC_ *	• 若為 MQXCC_FAILED: 1. 磁碟機 之後 MQ_OPEN_EXIT 1 函數 2. 使用 MQCC_FAILED 完成 MQOPEN 呼叫, MQRC_API_EXIT_ERROR • 適用於 MQXCC_ * 1. 作為 MQXCC_ * 和 MQXR2_ *[1] 的值 2. 磁碟機 之後 MQ_OPEN_EXIT 1 函數 (如果未抑制結束程式)
e25	在 MQ_DISC_EXIT 2 函數完成之後: • MQXCC_FAILED • MQXCC_ *	• 若為 MQXCC_FAILED: 1. 磁碟機 之後 MQ_DISC_EXIT 1 函數 2. 磁碟機 MQ_TERM_EXIT 函數 3. 清除結束執行環境 4. 使用 MQCC_FAILED、MQRC_API_EXIT_ERROR 完成 MQDISC 呼叫 • 適用於 MQXCC_ * 1. 作為 MQXCC_ * 和 MQXR2_ *[1] 的值 2. 磁碟機 MQ_TERM_EXIT 函數 3. 清除結束執行環境

註:

1. MQXCC_ * 和 MQXR2_* 及其對應動作的值定義在 佇列管理程式如何處理結束程式函數中。

ExitResponse 欄位設定不正確

本主題提供當 ExitResponse 欄位設為支援值以外的任何值時將會發生之情況的相關資訊。

如果 ExitResponse 欄位設為其中一個受支援值以外的值，則適用下列動作:

- 若為 之前的 MQCONN 或 MQDISC API 結束程式函數:
  - 會忽略 ExitResponse2 值。
  - 在呼叫結束鏈 (如果有的話) 中的結束函數 之前 沒有進一步的呼叫; 不會發出 API 呼叫本身。
  - 對於已順利呼叫的任何 之前 結束程式，會以相反順序呼叫 之後 結束程式。
  - 如果已登錄，則會驅動鏈中已順利呼叫的那些 (在 MQCONN 或 MQDISC 結束程式函數之前) 的終止結束程式函數，以在這些結束程式函數之後清除。
  - MQCONN 或 MQDISC 呼叫失敗，並產生 MQRC_API_EXIT_ERROR。
- 若為非 MQCONN 或 MQDISC 的 之前 WebSphere MQ API 結束程式函數:
  - 會忽略 ExitResponse2 值。
  - 不會呼叫結束鏈中的進一步 之前 或 之後 資料轉換函數 (如果有的話)。
  - 對於已順利呼叫的任何 之前 結束程式，會以相反順序呼叫 之後 結束程式。

- 不會發出 WebSphere MQ API 呼叫本身。
- WebSphere MQ API 呼叫失敗，並產生 MQRC_API_EXIT_ERROR。

- 若為 之後 MQCONN 或 MQDISC API 結束程式函數:
  - 會忽略 ExitResponse2 值。
  - 在 API 呼叫之前順利呼叫的其餘結束函數會以相反順序呼叫。
  - 如果已登錄，則會驅動鏈中已順利呼叫之 之前 或 之後 MQCONN 或 MQDISC 結束程式函數的終止結束程式函數，以在結束程式之後清除。
  - MQCC_WARNING 較嚴重的 CompCode 和結束程式所傳回的 CompCode 會傳回給應用程式。
  - MQRC_API_EXIT_ERROR 的原因會傳回給應用程式。
  - 已順利發出 WebSphere MQ API 呼叫。
- 對於非 MQCONN 或 MQDISC 的 *after* WebSphere MQ API 呼叫結束程式函數:
  - 會忽略 ExitResponse2 值。
  - 在 API 呼叫之前順利呼叫的其餘結束函數會以相反順序呼叫。
  - MQCC_WARNING 較嚴重的 CompCode 和結束程式所傳回的 CompCode 會傳回給應用程式。
  - MQRC_API_EXIT_ERROR 的原因會傳回給應用程式。
  - 已順利發出 WebSphere MQ API 呼叫。
- 對於 get exit 函數上的 *before* 資料轉換:
  - 會忽略 ExitResponse2 值。
  - 在 API 呼叫之前順利呼叫的其餘結束函數會以相反順序呼叫。
  - 訊息不會轉換，且未轉換的訊息會傳回給應用程式。
  - MQCC_WARNING 較嚴重的 CompCode 和結束程式所傳回的 CompCode 會傳回給應用程式。
  - MQRC_API_EXIT_ERROR 的原因會傳回給應用程式。
  - 已順利發出 WebSphere MQ API 呼叫。

  註: 因為結束程式發生錯誤，所以最好傳回 MQRC_API_EXIT_ERROR ，而不是傳回 MQRC_NOT_CONVERTED。

如果結束函數將 ExitResponse2 欄位設為其中一個受支援值以外的值，則會改為採用 MQXR2_DEFAULT_CONTINUATION 值。

可安裝的服務介面參照資訊

這個主題集合提供可安裝服務的參照資訊。

在每一種服務類型的群組內，會按字母順序列出函數及資料類型。

如何顯示函數

如何記載可安裝服務功能。

每一個函數都有說明，包括函數 ID (適用於 MQZEP)。

參數 會以它們必須出現的順序列出。 他們必須都在場

每一個參數名稱後接其資料類型。 這些是 第 214 頁的『基本資料類型』中說明的基本資料類型。

在參數說明之後，也會提供 C 語言呼叫。

MQZ_AUTHENTICATE_USER-鑑別使用者

此函數由 MQZAS_VERSION_5 授權服務元件提供，並由佇列管理程式呼叫以鑑別使用者或設定身分環境定義欄位。 當建立 WebSphere MQ 的使用者應用程式環境定義時，會呼叫它。

在連接呼叫期間，會在應用程式使用者環境定義起始設定的點，以及在應用程式使用者環境定義變更的每一個點，建立應用程式環境定義。 每次進行連接呼叫時，都會在 *IdentityContext* 欄位中重新獲得應用程式的使用者環境定義資訊。

此函數 (適用於 MQZEP) 的函數 ID 是 MQZID_AUTHENTICATE_USER。

語法

MQZ_AUTHENTICATE_USER (*QMgrName*， *SecurityParms*， *ApplicationContext*， *IdentityContext*， *CorrelationPtr*， *ComponentData*， *Contination*， *CompCode*， 原因)

參數

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何定義的方式 使用 它。

SecurityParms
類型 :MQCSP-輸入

安全參數。 與使用者 ID、密碼及鑑別類型相關的資料。 如果 MQCSP 結構的 AuthenticationType 屬性指定為 MQCSP_AUTH_USER_ID_AND_PWD ，則使用者 ID 和密碼會與 IdentityContext (MQZIC) 參數中的對等欄位進行比較，以判定它們是否符合。 如需相關資訊，請參閱 第 295 頁的『MQCSP-安全參數』。

在 MQCONN MQI 呼叫期間，此參數會包含空值或預設值。

ApplicationContext
類型 :MQZAC-輸入

應用程式環境定義。 與呼叫端應用程式相關的資料。 如需詳細資料，請參閱 MQZAC-應用程式環境定義 。

在每個 MQCONN 或 MQCONNX MQI 呼叫期間，會重新獲得 MQZAC 結構中的使用者環境定義資訊。

IdentityContext
類型 :MQZIC-輸入/輸出

身分環境定義。 在鑑別使用者功能的 輸入 上， 這個 會識別現行身分環境定義。 鑑別使用者功能可以變更 這個，此時佇列管理程式會採用新的身分環境定義。 如需 MQZIC 結構的詳細資料，請參閱 MQZIC-身分環境定義 。

CorrelationPtr
類型 :MQPTR-輸出

相關性指標。 指定任何相關性資料的位址。 此指標 隨後 會傳遞至其他 OAM 呼叫。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此 資料由佇列管理程式代表此特定元件保留; 此元件提供的任何功能對其進行的任何變更都會保留，並在下次呼叫此 元件的的 功能時呈現。

此資料區的長度由佇列管理程式在 MQZ_INIT_AUTHORITY 呼叫的 ComponentData 長度參數中傳遞。

接續
類型 :MQLONG-輸出

接續旗標。 您可以指定下列值:

MQZCI_DEFAULT
繼續相依於其他元件。

停止 MQZCI_STOP
請勿繼續處理下一個元件。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_SERVICE_ERROR
(2289 , X'8F1') 存取服務時發生非預期的錯誤。

如需 這些原因碼的相關 資訊 , 請參閱 原因碼。

C 呼叫

```
MQZ_AUTHENTICATE_USER (QMgrName, SecurityParms, ApplicationContext,
                   IdentityContext, &CorrelationPtr, ComponentData,
                   &Continuation, &CompCode, &Reason);
```

宣告 傳遞至服務的參數，如下所示:

```
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-檢查權限

此函數由 MQZAS_VERSION_1 授權服務元件提供，並由佇列管理程式啟動，以檢查實體是否有權對指定的物件執行特定動作。

此函數 (適用於 MQZEP) 的函數 ID 為 MQZID_CHECK_AUTHORITY。

語法

MQZ_CHECK_AUTHORITY(*QMgrName*, *EntityName*, *EntityType*, *ObjectName*, *ObjectType*, *Authority*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

EntityName
類型: MQCHAR12 -輸入

實體名稱。 要檢查其物件授權的實體名稱。 字串的長度上限為 12 個字元; 如果它比用空白填補它的長度還短的話。 名稱不是以空值字元終止。

基礎安全服務不需要知道此實體。 如果不知道，則會使用特殊 **nobody** 群組 (假設所有實體都屬於該群組) 的授權進行檢查。 全空白名稱是有效的，可透過此方式使用。

EntityType
類型 :MQLONG-輸入

實體類型。 EntityName 指定的實體類型。 它必須是下列其中一個值:

MQZAET_PRINCIPAL
校長

MQZAET_GROUP
群組。

ObjectName
類型: MQCHAR48 -輸入

物件名稱。 需要存取權的物件名稱。 字串的長度上限為 48 個字元; 如果它比以空白填補它的長度還短的話。 名稱不是以空值字元終止。

如果 *ObjectType* 是 MQOT_Q_MGR ，則此名稱與 *QMgrName* 相同。

ObjectType
類型 :MQLONG-輸入

物件類型。 *ObjectName* 指定的實體類型。 它必須是下列其中一個值:

MQOT_AUTH_INFO
鑑別資訊。

MQ 通道
頻道

MQOT_CLNTCONN_CHANNEL
用戶端連線通道。

MQ 接聽器
接聽器。

MQOT_NAMELIST
名單。

MQ 處理程序
程序定義。

MQOT_Q
佇列。

MQOT_Q_MGR
佇列管理程式。

MQ 服務
服務。

authority
類型 :MQLONG-輸入

要檢查的權限。 如果正在檢查一個授權，則此欄位等於適當的授權作業 (MQZAO_ * 常數)。 如果正在檢查多個授權，則它是對應 MQZAO_ * 常數的位元 OR 運算。

下列授權適用於 MQI 呼叫的使用:

MQZAO_CONNECT
能夠使用 MQCONN 呼叫。

MQ 導覽 _ 瀏覽
能夠搭配使用 MQGET 呼叫與瀏覽選項。

這容許在 MQGET 呼叫上指定 MQGMO_BROWSE_FIRST、MQGMO_BROWSE_MSG_UNDER_CURSOR 或 MQGMO_BROWSE_NEXT 選項。

MQZAO_ 輸入
校長 能夠搭配使用 MQGET 呼叫與輸入選項。

這容許在 MQOPEN 呼叫上指定 MQOO_INPUT_SHARED、MQOO_INPUT_EXCLUSIVE 或 MQOO_INPUT_AS_Q_DEF 選項。

MQZAO_OUTPUT
能夠使用 MQPUT 呼叫。

這容許在 MQOPEN 呼叫上指定 MQOO_OUTPUT 選項。

MQZAO_INQUIRE
使用 MQINQ 呼叫的能力。

這容許在 MQOPEN 呼叫上指定 MQOO_INQUIRE 選項。

MQZAO_SET
使用 MQSET 呼叫的能力。

這容許在 MQOPEN 呼叫上指定 MQOO_SET 選項。

MQZAO_PASS_IDENTITY_CONTEXT
傳遞身分環境定義的能力。

這容許在 MQOPEN 呼叫上指定 MQOO_PASS_IDENTITY_CONTEXT 選項，並在 MQPUT 和 MQPUT1 呼叫上指定 MQPMO_PASS_IDENTITY_CONTEXT 選項。

MQZAO_PASS_ALL_CONTEXT
能夠傳遞所有環境定義。

這容許在 MQOPEN 呼叫上指定 MQOO_PASS_ALL_CONTEXT 選項，並在 MQPUT 和 MQPUT1 呼叫上指定 MQPMO_PASS_ALL_CONTEXT 選項。

MQZAO_SET_IDENTITY_CONTEXT
設定身分環境定義的能力。

這容許在 MQOPEN 呼叫上指定 MQOO_SET_IDENTITY_CONTEXT 選項，並在 MQPUT 和 MQPUT1 呼叫上指定 MQPMO_SET_IDENTITY_CONTEXT 選項。

MQZAO_SET_ALL_CONTEXT
設定所有環境定義的能力。

這容許在 MQOPEN 呼叫上指定 MQOO_SET_ALL_CONTEXT 選項，並在 MQPUT 和 MQPUT1 呼叫上指定 MQPMO_SET_ALL_CONTEXT 選項。

MQZAO_ALTERNATE_USER_AUTHORITY
使用替代使用者權限的能力。

這容許在 MQOPEN 呼叫上指定 MQOO_ALTERNATE_USER_AUTHORITY 選項，並在 MQPUT1 呼叫上指定 MQPMO_ALTERNATE_USER_AUTHORITY 選項。

MQZAO_ALL_MQI
所有 MQI 授權。

這會啟用所有授權。

下列授權適用於佇列管理程式的管理:

MQZAO_CREATE
能夠建立指定類型的物件。

MQZAO_DELETE
刪除指定物件的能力。

MQZAO_DISPLAY
能夠顯示指定物件的屬性。

MQZAO_CHANGE
能夠變更指定物件的屬性。

MQZAO_clear
從指定佇列刪除所有訊息的能力。

MQZAO_XX_ENCODE_CASE_ONE authorize
授權其他使用者使用指定物件的能力。

MQZAO_CONTROL
能夠啟動或停止接聽器、服務或非用戶端通道物件，以及能夠對非用戶端通道物件進行連線測試。

已延伸 MQZAO_CONTROL_EXTENDED
能夠重設序號，或解決非用戶端通道物件上的不確定訊息。

MQZAO_ALL_ADMIN
設定身分環境定義的能力。

MQZAO_CREATE 以外的所有管理授權。

下列授權適用於 MQI 的使用，以及佇列管理程式的管理:

MQZAO_ALL
MQZAO_CREATE 以外的所有授權。

MQZAO_NONE
無授權。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此資料由佇列管理程式代表此特定元件保留; 此元件提供的任何功能對其進行的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

佇列管理程式會在 MQZ_INIT_AUTHORITY 呼叫的 *ComponentDataLength* 參數中傳遞此資料區的長度。

接續
類型 :MQLONG-輸出

依元件設定的接續指示器。 可以指定下列值:

MQZCI_DEFAULT
繼續相依於佇列管理程式。

對於 MQZ_CHECK_AUTHORITY ，這具有與 MQZCI_STOP 相同的效果。

MQ 配置項目 _ 繼續
繼續處理下一個元件。

停止 MQZCI_STOP
請勿繼續處理下一個元件。

如果對元件的呼叫失敗 (即 *CompCode* 會傳回 MQCC_FAILED) ，且 *Continue* 參數是 MQZCI_DEFAULT 或 MQZCI_continue ，則佇列管理程式會繼續呼叫其他元件 (如果有的話)。

如果呼叫成功 (亦即， *CompCode* 會傳回 MQCC_OK) ，則不論 接續 的設定為何，都不會呼叫其他元件。

如果呼叫失敗，且 接續 參數是 MQZCI_STOP ，則不會呼叫其他元件，且會傳回錯誤給佇列管理程式。元件不知道先前的呼叫，因此在呼叫之前， 接續 參數一律設為 MQZCI_DEFAULT。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035 , X'7F3') 未獲授權存取。

MQRC_SERVICE_ERROR
(2289 , X'8F1') 存取服務時發生非預期的錯誤。

MQRC_SERVICE_NOT_AVAILABLE
(2285 , X'8ED') 無法使用基礎服務。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_CHECK_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,
                     ObjectType, Authority, ComponentData,
                     &Continuation, &CompCode, &Reason);
```

傳遞至服務的參數宣告如下:

```
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 -檢查權限 (延伸)

此函數由 MQZAS_VERSION_2 授權服務元件提供，並由佇列管理程式啟動，以檢查實體是否有權對指定物件執行特定動作。

此函數 (適用於 MQZEP) 的函數 ID 為 MQZID_CHECK_AUTHORITY。

MQZ_CHECK_AUTHORITY_2 類似於 MQZ_CHECK_AUTHORITY , 但具有由 *EntityData* 參數取代的 *EntityName* 參數。

語法
MQZ_CHECK_AUTHORITY_2(*QMgrName*, *EntityData*, *EntityType*, *ObjectName*, *ObjectType*, *Authority*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

EntityData
類型 :MQZED-輸入

實體資料。 與具有要檢查之物件授權的實體相關的資料。 請參閱第 1083 頁的『MQZED-實體描述子』，以取得詳細資料。

基礎安全服務不需要知道此實體。 如果不知道，則會使用特殊 **nobody** 群組 (假設所有實體都屬於該群組) 的授權進行檢查。 全空白名稱是有效的，可透過此方式使用。

EntityType
類型 :MQLONG-輸入

實體類型。 *EntityData* 指定的實體類型。 它必須是下列其中一個值:

MQZAET_PRINCIPAL
校長

MQZAET_GROUP
群組。

ObjectName
類型: MQCHAR48 -輸入

物件名稱。 需要存取權的物件名稱。 字串的長度上限為 48 個字元; 如果它比以空白填補它的長度還短的話。 名稱不是以空值字元終止。

如果 *ObjectType* 是 MQOT_Q_MGR ，則此名稱與 *QMgrName* 相同。

ObjectType
類型 :MQLONG-輸入

物件類型。 *ObjectName* 指定的實體類型。 它必須是下列其中一個值:

MQOT_AUTH_INFO
鑑別資訊。

MQ 通道
頻道

MQOT_CLNTCONN_CHANNEL
用戶端連線通道。

MQ 接聽器
接聽器。

MQOT_NAMELIST
名單。

MQ 處理程序
程序定義。

MQOT_Q
佇列。

MQOT_Q_MGR
佇列管理程式。

MQ 服務
服務。

MQOT_TOPIC
主題。

authority
類型 :MQLONG-輸入

要檢查的權限。 如果正在檢查一個授權，則此欄位等於適當的授權作業 (MQZAO_ * 常數)。 如果正在檢查多個授權，則它是對應 MQZAO_ * 常數的位元 OR 運算。

下列授權適用於 MQI 呼叫的使用:

MQZAO_CONNECT
能夠使用 MQCONN 呼叫。

MQ 導覽 _ 瀏覽
能夠搭配使用 MQGET 呼叫與瀏覽選項。

這容許在 MQGET 呼叫上指定 MQGMO_BROWSE_FIRST、MQGMO_BROWSE_MSG_UNDER_CURSOR 或 MQGMO_BROWSE_NEXT 選項。

MQZAO_ 輸入
校長 能夠搭配使用 MQGET 呼叫與輸入選項。

這容許在 MQOPEN 呼叫上指定 MQOO_INPUT_SHARED、MQOO_INPUT_EXCLUSIVE 或 MQOO_INPUT_AS_Q_DEF 選項。

MQZAO_OUTPUT
能夠使用 MQPUT 呼叫。

這容許在 MQOPEN 呼叫上指定 MQOO_OUTPUT 選項。

MQZAO_INQUIRE
使用 MQINQ 呼叫的能力。

這容許在 MQOPEN 呼叫上指定 MQOO_INQUIRE 選項。

MQZAO_SET
使用 MQSET 呼叫的能力。

這容許在 MQOPEN 呼叫上指定 MQOO_SET 選項。

MQZAO_PASS_IDENTITY_CONTEXT
傳遞身分環境定義的能力。

這容許在 MQOPEN 呼叫上指定 MQOO_PASS_IDENTITY_CONTEXT 選項，並在 MQPUT 和 MQPUT1 呼叫上指定 MQPMO_PASS_IDENTITY_CONTEXT 選項。

MQZAO_PASS_ALL_CONTEXT
能夠傳遞所有環境定義。

這容許在 MQOPEN 呼叫上指定 MQOO_PASS_ALL_CONTEXT 選項，並在 MQPUT 和 MQPUT1 呼叫上指定 MQPMO_PASS_ALL_CONTEXT 選項。

MQZAO_SET_IDENTITY_CONTEXT
設定身分環境定義的能力。

這容許在 MQOPEN 呼叫上指定 MQOO_SET_IDENTITY_CONTEXT 選項，並在 MQPUT 和 MQPUT1 呼叫上指定 MQPMO_SET_IDENTITY_CONTEXT 選項。

MQZAO_SET_ALL_CONTEXT
設定所有環境定義的能力。

這容許在 MQOPEN 呼叫上指定 MQOO_SET_ALL_CONTEXT 選項，並在 MQPUT 和 MQPUT1 呼叫上指定 MQPMO_SET_ALL_CONTEXT 選項。

MQZAO_ALTERNATE_USER_AUTHORITY
使用替代使用者權限的能力。

這容許在 MQOPEN 呼叫上指定 MQOO_ALTERNATE_USER_AUTHORITY 選項，並在 MQPUT1 呼叫上指定 MQPMO_ALTERNATE_USER_AUTHORITY 選項。

MQZAO_ALL_MQI
所有 MQI 授權。

這會啟用所有授權。

下列授權適用於佇列管理程式的管理:

MQZAO_CREATE
能夠建立指定類型的物件。

MQZAO_DELETE
刪除指定物件的能力。

MQZAO_DISPLAY
能夠顯示指定物件的屬性。

MQZAO_CHANGE
能夠變更指定物件的屬性。

MQZAO_clear
從指定佇列刪除所有訊息的能力。

MQZAO_XX_ENCODE_CASE_ONE authorize
授權其他使用者使用指定物件的能力。

MQZAO_CONTROL
能夠啟動或停止接聽器、服務或非用戶端通道物件，以及能夠對非用戶端通道物件進行連線測試。

已延伸 MQZAO_CONTROL_EXTENDED
能夠重設序號，或解決非用戶端通道物件上的不確定訊息。

MQZAO_ALL_ADMIN
設定身分環境定義的能力。

MQZAO_CREATE 以外的所有管理授權。

下列授權適用於 MQI 的使用，以及佇列管理程式的管理:

MQZAO_ALL
MQZAO_CREATE 以外的所有授權。

MQZAO_NONE
無授權。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此資料由佇列管理程式代表此特定元件保留; 此元件提供的任何功能對其進行的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

佇列管理程式會在 MQZ_INIT_AUTHORITY 呼叫的 *ComponentDataLength* 參數中傳遞此資料區的長度。

接續
類型 :MQLONG-輸出

依元件設定的接續指示器。 可以指定下列值:

MQZCI_DEFAULT
繼續相依於佇列管理程式。

對於 MQZ_CHECK_AUTHORITY ，這具有與 MQZCI_STOP 相同的效果。

MQ 配置項目 _ 繼續
繼續處理下一個元件。

停止 MQZCI_STOP
請勿繼續處理下一個元件。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035 , X'7F3') 未獲授權存取。

MQRC_SERVICE_ERROR
(2289 , X'8F1') 存取服務時發生非預期的錯誤。

MQRC_SERVICE_NOT_AVAILABLE
(2285 , X'8ED') 無法使用基礎服務。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_CHECK_AUTHORITY_2 (QMgrName, &EntityData, EntityType,
                       ObjectName, ObjectType, Authority, ComponentData,
                       &Continuation, &CompCode, &Reason);
```

傳遞至服務的參數宣告如下:

```
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-檢查使用者是否為特許使用者

此函數由 MQZAS_VERSION_6 授權服務元件提供，並由佇列管理程式呼叫以判定指定使用者是否為特許使用者。

此函數 (適用於 MQZEP) 的函數 ID 是 MQZID_CHECK_PRIVILEGED。

語法
MQZ_CHECK_PRIVILEGED(*QMgrName*, *EntityData*, *EntityType*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

EntityData
類型 :MQZED-輸入

實體資料。 與要檢查之實體相關的資料。 如需相關資訊，請參閱第 1083 頁的『MQZED-實體描述子』。

EntityType
類型 :MQLONG-輸入

實體類型。 EntityData 指定的實體類型。 它必須是下列其中一個值:

MQZAET_PRINCIPAL
校長

MQZAET_GROUP
群組。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此資料由佇列管理程式代表此特定元件保留; 此元件提供的任何功能對其進行的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

佇列管理程式會在 MQZ_INIT_AUTHORITY 呼叫的 *ComponentDataLength* 參數中傳遞此資料區的長度。

接續
類型 :MQLONG-輸出

依元件設定的接續指示器。 可以指定下列值:

MQZCI_DEFAULT
繼續相依於佇列管理程式。

對於 MQZ_CHECK_AUTHORITY ，這具有與 MQZCI_STOP 相同的效果。

MQ 配置項目 _ 繼續
繼續處理下一個元件。

停止 MQZCI_STOP
請勿繼續處理下一個元件。

如果對元件的呼叫失敗 (即 *CompCode* 會傳回 MQCC_FAILED) ， 且 *Continue* 參數是 MQZCI_DEFAULT 或 MQZCI_continue ，則佇列管理程式會繼續呼叫其他元件 (如果有的話)。

如果呼叫成功 (亦即， *CompCode* 會傳回 MQCC_OK) ，則不論 接續 的設定為何，都不會呼叫其他元件。

如果呼叫失敗，且 接續 參數是 MQZCI_STOP ，則不會呼叫其他元件，且會傳回錯誤給佇列管理程式。 元件不知道先前的呼叫，因此在呼叫之前， 接續 參數一律設為 MQZCI_DEFAULT。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_NOT_PRIVILEGED
(2584 , X'A18') 此使用者不是特許使用者 ID。

MQRC_UNKNOWN_ENTITY
(2292 , X'8F4') 無法提供服務的實體。

MQRC_SERVICE_ERROR
(2289 , X'8F1') 存取服務時發生非預期的錯誤。

MQRC_SERVICE_NOT_AVAILABLE
(2285 , X'8ED') 無法使用基礎服務。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_CHECK_PRIVILEGED (QMgrName, &EntityData, EntityType,
                      ComponentData, &Continuation,
                      &CompCode, &Reason);
```

傳遞至服務的參數宣告如下:

```
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-複製所有權限

此功能由授權服務元件提供。 佇列管理程式會啟動它，將參照物件目前有效的所有授權複製到另一個物件。

此函數的函數 ID (適用於 MQZEP) 是 MQZID_COPY_ALL_AUTHORITY。

語法

MQZ_COPY_ALL_AUTHORITY(*QMgrName*, *RefObjectName*, *ObjectName*, *ObjectType*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

***RefObject* 名稱**
類型: MQCHAR48 -輸入

參照物件名稱。 要複製其授權的參照物件名稱。 字串的長度上限為 48 個字元; 如果它比以空白填補它的長度還短的話。 名稱不是以空值字元終止。

ObjectName
類型: MQCHAR48 -輸入

物件名稱。 要設定其存取權的物件名稱。 字串的長度上限為 48 個字元; 如果它比以空白填補它的長度還短的話。 名稱不是以空值字元終止。

ObjectType
類型 :MQLONG-輸入

物件類型。 *RefObjectName* 和 *ObjectName* 指定的實體類型。 它必須是下列其中一個值:

MQOT_AUTH_INFO
鑑別資訊。

MQ 通道
頻道

MQOT_CLNTCONN_CHANNEL
用戶端連線通道。

MQ 接聽器
接聽器。

MQOT_NAMELIST
名單。

MQ 處理程序
程序定義。

MQOT_Q
佇列。

MQOT_Q_MGR
佇列管理程式。

MQ 服務
服務。

MQOT_TOPIC
主題。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此資料由佇列管理程式代表此特定元件保留; 此元件提供的任何功能對其進行的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

此資料區的長度由佇列管理程式在 MQZ_INIT_AUTHORITY 呼叫的 ComponentData 長度參數中傳遞。

接續
類型 :MQLONG-輸出

依元件設定的接續指示器。 可以指定下列值:

MQZCI_DEFAULT
繼續相依於佇列管理程式。

對於 MQZ_CHECK_AUTHORITY ， 這具有與 MQZCI_STOP 相同的效果。

MQ 配置項目 _ 繼續
繼續處理下一個元件。

停止 MQZCI_STOP
請勿繼續處理下一個元件。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_SERVICE_ERROR
(2289 , X'8F1') 存取服務時發生非預期的錯誤。

MQRC_SERVICE_NOT_AVAILABLE
(2285 , X'8ED') 無法使用基礎服務。

MQRC_UNKNOWN_REF_OBJECT
(2294 , X'8F6') 參照物件不明。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_COPY_ALL_AUTHORITY (QMgrName, RefObjectName, ObjectName, ObjectType,
                        ComponentData, &Continuation, &CompCode,
                        &Reason);
```

傳遞至服務的參數宣告如下:

```
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-刪除權限

此功能由授權服務元件提供，並由佇列管理程式啟動，以刪除與指定物件相關聯的所有授權。

此函數 (適用於 MQZEP) 的函數 ID 為 MQZID_DELETE_AUTHORITY。

語法

MQZ_DELETE_AUTHORITY(*QMgrName*, *ObjectName*, *ObjectType*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

ObjectName
類型: MQCHAR48 -輸入

物件名稱。 要刪除其存取權的物件名稱。 字串的長度上限為 48 個字元; 如果它比以空白填補它的長度還短的話。 名稱不是以空值字元終止。

如果 *ObjectType* 是 MQOT_Q_MGR , 則此名稱與 *QMgrName* 相同。

ObjectType
類型 :MQLONG-輸入

物件類型。 *ObjectName* 指定的實體類型。 它必須是下列其中一個值:

MQOT_AUTH_INFO
鑑別資訊。

MQ 通道
頻道

MQOT_CLNTCONN_CHANNEL
用戶端連線通道。

MQ 接聽器
接聽器。

MQOT_NAMELIST
名單。

MQ 處理程序
程序定義。

MQOT_Q
佇列。

MQOT_Q_MGR
佇列管理程式。

MQ 服務
服務。

MQOT_TOPIC
主題。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此資料由佇列管理程式代表此特定元件保留; 此元件提供的任何功能對其進行的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

此資料區的長度由佇列管理程式在 MQZ_INIT_AUTHORITY 呼叫的 ComponentData 長度參數中傳遞。

接續
類型 :MQLONG-輸出

依元件設定的接續指示器。 可以指定下列值:

MQZCI_DEFAULT
繼續相依於佇列管理程式。

對於 MQZ_CHECK_AUTHORITY ， 這具有與 MQZCI_STOP 相同的效果。

MQ 配置項目 _ 繼續
繼續處理下一個元件。

停止 MQZCI_STOP
請勿繼續處理下一個元件。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_SERVICE_ERROR
(2289 , X'8F1') 存取服務時發生非預期的錯誤。

MQRC_SERVICE_NOT_AVAILABLE
(2285 , X'8ED') 無法使用基礎服務。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_DELETE_AUTHORITY (QMgrName, ObjectName, ObjectType, ComponentData,
                      &Continuation, &CompCode, &Reason);
```

傳遞至服務的參數宣告如下:

```
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-列舉權限資料

此函數由 MQZAS_VERSION_4 授權服務元件提供，並由佇列管理程式反覆地啟動，以擷取符合第一次呼叫時指定的選取準則的所有權限資料。

此函數 (適用於 MQZEP) 的函數 ID 是 MQZID_ENUMERATE_AUTHORITY_DATA。

語法

MQZ_ENUMERATE_AUTHORITY_DATA(*QMgrName*, *StartEnumeration*, *Filter*, *AuthorityBufferLength*, *AuthorityBuffer*, *AuthorityDataLength*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

StartEnumeration
類型 :MQLONG-輸入

指出呼叫是否可以開始列舉的旗標。 這指出呼叫是否可以開始列舉權限資料，或繼續列舉由前一個 MQZ_ENUMERATE_AUTHORITY_DATA 呼叫所啟動的權限資料。 該值是下列其中一個值:

MQZSE_START
開始列舉。 會以這個值來啟動呼叫，以開始列舉權限資料。 *Filter* 參數指定選取準則，用來選取此呼叫及連續呼叫所傳回的權限資料。

MQ ZSE_continue
繼續列舉。 以這個值來啟動呼叫，以繼續列舉權限資料。 在此情況下，會忽略 *Filter* 參數，且可以指定為空值指標 (選取準則由 *StartEnumeration* 設為 MQZSE_START 的呼叫所指定的 *Filter* 參數決定)。

過濾器
類型 :MQZAD-輸入

過濾器。 如果 *StartEnumeration* 是 MQZSE_START，則 *Filter* 會指定用來選取要傳回之權限資料的選取準則。 如果 *Filter* 是空值指標，則不會使用任何選取準則，即會傳回所有權限資料。 如需可以使用的選取準則的詳細資料，請參閱 第 1080 頁的『MQZAD-權限資料』。

如果 *StartEnumeration* 是 MQZSE_CONTINUE，則會忽略 *Filter*，且可以指定為空值指標。

***AuthorityBuffer* 長度**
類型 :MQLONG-輸入

AuthorityBuffer 的長度。 這是 *AuthorityBuffer* 參數的長度 (以位元組為單位)。 權限緩衝區必須夠大，才能容納要傳回的資料。

AuthorityBuffer
類型 :MQZAD-輸出

權限資料。 這是傳回權限資料的緩衝區。 緩衝區必須夠大，才能容納 MQZAD 結構、MQZED 結構，以及所定義的最長實體名稱和最長網域名稱。

註: 附註: 此參數定義為 MQZAD，因為 MQZAD 一律會出現在緩衝區開頭。 不過，如果將緩衝區宣告為 MQZAD，則緩衝區會太小-它必須大於 MQZAD，才能容納 MQZAD、MQZED 以及實體和網域名稱。

***AuthorityData* 長度**
類型 :MQLONG-輸出

在 *AuthorityBuffer* 中傳回的資料長度。 如果權限緩衝區太小， *AuthorityDataLength* 會設為所需緩衝區的長度，且呼叫會傳回完成碼 MQCC_FAILED 及原因碼 MQRC_BUFFER_LENGTH_ERROR。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此資料由佇列管理程式代表此特定元件保留; 此元件提供的任何功能對其進行的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

此資料區的長度由佇列管理程式在 MQZ_INIT_AUTHORITY 呼叫的 ComponentData 長度參數中傳遞。

接續
類型 :MQLONG-輸出

依元件設定的接續指示器。 可以指定下列值:

MQZCI_DEFAULT
繼續相依於佇列管理程式。

MQZ_ENUMERATE_AUTHORITY_DATA 的效果與 MQZCI_CONTINUE 相同。

MQ 配置項目 _ 繼續
繼續處理下一個元件。

停止 MQZCI_STOP
請勿繼續處理下一個元件。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因

類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_BUFFER_LENGTH_ERROR
(2005 , X'7D5') 緩衝區長度參數無效。

MQRC_NO_DATA_AVAILABLE
(2379 , X'94B') 沒有可用的資料。

MQRC_SERVICE_ERROR
(2289 , X'8F1') 存取服務時發生非預期的錯誤。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_ENUMERATE_AUTHORITY_DATA (QMgrName, StartEnumeration, &Filter,
                              AuthorityBufferLength,
                              &AuthorityBuffer,
                              &AuthorityDataLength, ComponentData,
                              &Continuation, &CompCode,
                              &Reason);
```

傳遞至服務的參數宣告如下:

```
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-可用使用者

此函數由 MQZAS_VERSION_5 授權服務元件提供，並由佇列管理程式啟動以釋放相關聯的已配置資源。

當應用程式已完成在所有使用者環境定義下執行時 (例如在 MQDISC MQI 呼叫期間) ，即會啟動。

此函數的函數 ID (適用於 MQZEP) 是 MQZID_FREE_USER。

語法

MQZ_FREE_USER(*QMgrName*, *FreeParms*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

FreeParms
類型 :MQZFP-輸入

可用參數。 包含與要釋放之資源相關的資料的結構。 請參閱第 1085 頁的『MQZFP-免費參數』，以取得詳細資料。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此資料由佇列管理程式代表此特定元件保留; 此元件提供的任何功能對其進行的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

此資料區的長度由佇列管理程式在 MQZ_INIT_AUTHORITY 呼叫的 ComponentData 長度參數中傳遞。

接續
類型 :MQLONG-輸出

接續旗標。 可以指定下列值:

MQZCI_DEFAULT
繼續相依於其他元件。

停止 MQZCI_STOP
請勿繼續處理下一個元件。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_SERVICE_ERROR
(2289 ， X'8F1') 存取服務時發生非預期的錯誤。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_AUTHENTICATE_USER (QMgrName, SecurityParms, ApplicationContext,
                    IdentityContext, CorrelationPtr, ComponentData,
                    &Continuation, &CompCode, &Reason);
```

傳遞至服務的參數宣告如下:

```
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-取得權限

此函數由 MQZAS_VERSION_1 授權服務元件提供，並由佇列管理程式啟動，以擷取實體必須存取指定物件的權限，包括 (如果實體是主體) 主體所屬群組所擁有的權限。 同屬設定檔中的權限包括在傳回的權限集中。

此函數 (適用於 MQZEP) 的函數 ID 是 MQZID_GET_AUTHORITY。

語法

MQZ_GET_AUTHORITY(*QMgrName*, *EntityName*, *EntityType*, *ObjectName*, *ObjectType*, *Authority*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

EntityName
類型: MQCHAR12 -輸入

實體名稱。 要擷取其物件存取權的實體名稱。 字串的長度上限為 12 個字元; 如果它比用空白填補它的長度還短的話。 名稱不是以空值字元終止。

EntityType
類型 :MQLONG-輸入

實體類型。 *EntityName* 指定的實體類型。 它必須是下列其中一個值:

MQZAET_PRINCIPAL
校長

MQZAET_GROUP
群組。

ObjectName
類型: MQCHAR48 -輸入

物件名稱。 要擷取存取權的物件名稱。 字串的長度上限為 48 個字元; 如果它比以空白填補它的長度還短的話。 名稱不是以空值字元終止。

如果 *ObjectType* 是 MQOT_Q_MGR ，則此名稱與 *QMgrName* 相同。

ObjectType
類型 :MQLONG-輸入

物件類型。 *ObjectName* 指定的實體類型。 它必須是下列其中一個值:

MQOT_AUTH_INFO
鑑別資訊。

MQ 通道
頻道

MQOT_CLNTCONN_CHANNEL
用戶端連線通道。

MQ 接聽器
接聽器。

MQOT_NAMELIST
名單。

MQ 處理程序
程序定義。

MQOT_Q
佇列。

MQOT_Q_MGR
佇列管理程式。

MQ 服務
服務。

MQOT_TOPIC
主題。

authority
類型 :MQLONG-輸入

實體的權限。 如果實體具有一個權限，則此欄位等於適當的授權作業 (MQZAO_ * 常數)。 如果它具有多個權限，則此欄位是對應 MQZAO_ * 常數的位元 OR 運算。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此資料由佇列管理程式代表此特定元件保留; 此元件提供的任何功能對其進行的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

佇列管理程式會在 MQZ_INIT_AUTHORITY 呼叫的 *ComponentDataLength* 參數中傳遞此資料區的長度。

接續
類型 :MQLONG-輸出

依元件設定的接續指示器。 可以指定下列值:

MQZCI_DEFAULT
繼續相依於佇列管理程式。

對於 MQZ_GET_AUTHORITY ， 其效果與 MQZCI_continue 相同。

MQ 配置項目 _ 繼續
繼續處理下一個元件。

停止 MQZCI_STOP
請勿繼續處理下一個元件。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035 ， X'7F3') 未獲授權存取。

MQRC_SERVICE_ERROR
(2289 , X'8F1') 存取服務時發生非預期的錯誤。

MQRC_SERVICE_NOT_AVAILABLE
(2285 , X'8ED') 無法使用基礎服務。

MQRC_UNKNOWN_ENTITY
(2292 , X'8F4') 無法提供服務的實體。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_GET_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,
                   ObjectType, &Authority, ComponentData,
                   &Continuation, &CompCode, &Reason);
```

傳遞至服務的參數宣告如下:

```
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_AUTHORITY_2 -取得權限 (延伸)

此函數由 MQZAS_VERSION_2 授權服務元件提供，並由佇列管理程式啟動，以擷取實體存取指定物件所需的權限。

此函數 (適用於 MQZEP) 的函數 ID 是 MQZID_GET_AUTHORITY。

MQZ_GET_AUTHORITY_2 類似於 MQZ_GET_AUTHORITY , 但使用 *EntityData* 參數取代 *EntityName* 參數。

語法

MQZ_GET_AUTHORITY_2(*QMgrName*, *EntityData*, *EntityType*, *ObjectName*, *ObjectType*, *Authority*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

EntityData
類型 :MQZED-輸入

實體資料。 與要擷取物件授權之實體相關的資料。 請參閱第 1083 頁的『MQZED-實體描述子』，以取得詳細資料。

EntityType
類型 :MQLONG-輸入

實體類型。 *EntityData* 指定的實體類型。 它必須是下列其中一個值:

MQZAET_PRINCIPAL
校長

MQZAET_GROUP
群組。

ObjectName
類型: MQCHAR48 -輸入

物件名稱。 要擷取其實體權限的物件名稱。 字串的長度上限為 48 個字元; 如果它比以空白填補它的長度還短的話。 名稱不是以空值字元終止。

如果 *ObjectType* 是 MQOT_Q_MGR ， 則此名稱與 *QMgrName* 相同。

ObjectType
類型 :MQLONG-輸入

物件類型。 *ObjectName* 指定的實體類型。 它必須是下列其中一個值:

MQOT_AUTH_INFO
鑑別資訊。

MQ 通道
頻道

MQOT_CLNTCONN_CHANNEL
用戶端連線通道。

MQ 接聽器
接聽器。

MQOT_NAMELIST
名單。

MQ 處理程序
程序定義。

MQOT_Q
佇列。

MQOT_Q_MGR
佇列管理程式。

MQ 服務
服務。

MQOT_TOPIC
主題。

authority
類型 :MQLONG-輸入

實體的權限。 如果實體具有一個權限，則此欄位等於適當的授權作業 (MQZAO_ * 常數)。 如果它具有多個權限，則此欄位是對應 MQZAO_ * 常數的位元 OR 運算。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此資料由佇列管理程式代表此特定元件保留; 此元件提供的任何功能對其進行的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

佇列管理程式會在 MQZ_INIT_AUTHORITY 呼叫的 *ComponentDataLength* 參數中傳遞此資料區的長度。

接續
類型 :MQLONG-輸出

依元件設定的接續指示器。 可以指定下列值:

MQZCI_DEFAULT
繼續相依於佇列管理程式。

對於 MQZ_CHECK_AUTHORITY ， 這具有與 MQZCI_STOP 相同的效果。

MQ 配置項目 _ 繼續
繼續處理下一個元件。

停止 MQZCI_STOP
請勿繼續處理下一個元件。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035 ， X'7F3') 未獲授權存取。

MQRC_SERVICE_ERROR
(2289 ， X'8F1') 存取服務時發生非預期的錯誤。

MQRC_SERVICE_NOT_AVAILABLE
(2285 ， X'8ED') 無法使用基礎服務。

MQRC_UNKNOWN_ENTITY
(2292 ， X'8F4') 無法提供服務的實體。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

語法

MQZ_GET_AUTHORITY_2*(QMgrName, EntityData, EntityType, ObjectName, ObjectType, Authority, ComponentData, Continuation, CompCode, Reason)*

C 呼叫

```
MQZ_GET_AUTHORITY_2 (QMgrName, &EntityData, EntityType, ObjectName,
                     ObjectType, &Authority, ComponentData,
                     &Continuation, &CompCode, &Reason);
```

傳遞至服務的參數宣告如下:

```
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-取得明確權限

此函數由 MQZAS_VERSION_1 授權服務元件提供，並由佇列管理程式啟動，以擷取具名群組必須存取指定物件的權限 (但沒有 **nobody** 群組的其他權限)，或具名主體的主要群組必須存取指定物件的權限。

在 UNIX 平台上，對於內建 WebSphere MQ 物件權限管理程式 (OAM)，傳回的權限僅由主體的主要群組擁有。

此函數的函數 ID (適用於 MQZEP) 是 MQZID_GET_EXPLICIT_AUTHORITY。

語法

MQZ_GET_EXPLICIT_AUTHORITY(*QMgrName*, *EntityName*, *EntityType*, *ObjectName*, *ObjectType*, *Authority*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

EntityName
類型: MQCHAR12 -輸入

實體名稱。 要擷取其物件存取權的實體名稱。 字串的長度上限為 12 個字元; 如果它比用空白填補它的長度還短的話。 名稱不是以空值字元終止。

EntityType
類型 :MQLONG-輸入

實體類型。 *EntityName* 指定的實體類型。 它必須是下列其中一個值:

MQZAET_PRINCIPAL
校長

MQZAET_GROUP
群組。

ObjectName
類型: MQCHAR48 -輸入

物件名稱。 要擷取其實體權限的物件名稱。 字串的長度上限為 48 個字元; 如果它比以空白填補它的長度還短的話。 名稱不是以空值字元終止。

如果 *ObjectType* 是 MQOT_Q_MGR，則此名稱與 *QMgrName* 相同。

ObjectType
類型 :MQLONG-輸入

物件類型。 *ObjectName* 指定的實體類型。 它必須是下列其中一個值:

MQOT_AUTH_INFO
鑑別資訊。

MQ 通道
頻道

MQOT_CLNTCONN_CHANNEL
用戶端連線通道。

MQ 接聽器
接聽器。

MQOT_NAMELIST
名單。

MQ 處理程序
程序定義。

MQOT_Q
佇列。

MQOT_Q_MGR
佇列管理程式。

MQ 服務
服務。

MQOT_TOPIC
主題。

authority
類型 :MQLONG-輸入

實體的權限。 如果實體具有一個權限，則此欄位等於適當的授權作業 (MQZAO_ * 常數)。 如果它具有多個權限，則此欄位是對應 MQZAO_ * 常數的位元 OR 運算。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此資料由佇列管理程式代表此特定元件保留; 此元件提供的任何功能對其進行的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

佇列管理程式會在 MQZ_INIT_AUTHORITY 呼叫的 *ComponentDataLength* 參數中傳遞此資料區的長度。

接續
類型 :MQLONG-輸出

依元件設定的接續指示器。 可以指定下列值:

MQZCI_DEFAULT
繼續相依於佇列管理程式。

對於 MQZ_GET_AUTHORITY ， 其效果與 MQZCI_continue 相同。

MQ 配置項目 _ 繼續
繼續處理下一個元件。

停止 MQZCI_STOP
請勿繼續處理下一個元件。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035 ， X'7F3') 未獲授權存取。

MQRC_SERVICE_ERROR
(2289 , X'8F1') 存取服務時發生非預期的錯誤。

MQRC_SERVICE_NOT_AVAILABLE
(2285 , X'8ED') 無法使用基礎服務。

MQRC_UNKNOWN_ENTITY
(2292 , X'8F4') 無法提供服務的實體。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_GET_EXPLICIT_AUTHORITY (QMgrName, EntityName, EntityType,
                            ObjectName, ObjectType, &Authority,
                            ComponentData, &Continuation,
                            &CompCode, &Reason);
```

傳遞至服務的參數宣告如下:

```
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 -取得明確權限 (延伸)

此函數由 MQZAS_VERSION_2 授權服務元件提供，並由佇列管理程式啟動，以擷取具名群組必須存取指定物件的權限 (但沒有 **nobody** 群組的其他權限)，或具名主體的主要群組必須存取指定物件的權限。

此函數的函數 ID (適用於 MQZEP) 是 MQZID_GET_EXPLICIT_AUTHORITY。

MQZ_GET_EXPLICIT_AUTHORITY_2 類似於 MQZ_GET_EXPLICIT_AUTHORITY，但 *EntityName* 參數取代為 *EntityData* 參數。

語法

MQZ_GET_EXPLICIT_AUTHORITY_2(*QMgrName*, *EntityData*, *EntityType*, *ObjectName*, *ObjectType*, *Authority*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

EntityData
類型 :MQZED-輸入

實體資料。 與要擷取其物件授權之實體相關的資料。 請參閱第 1083 頁的『MQZED-實體描述子』，以取得詳細資料。

EntityType
類型 :MQLONG-輸入

實體類型。 *EntityData* 指定的實體類型。 它必須是下列其中一個值:

MQZAET_PRINCIPAL
校長

MQZAET_GROUP
群組。

ObjectName
類型: MQCHAR48 -輸入

物件名稱。 要擷取其實體權限的物件名稱。 字串的長度上限為 48 個字元; 如果它比以空白填補它的長度還短的話。 名稱不是以空值字元終止。

如果 *ObjectType* 是 MQOT_Q_MGR ， 則此名稱與 *QMgrName* 相同。

ObjectType
類型 :MQLONG-輸入

物件類型。 *ObjectName* 指定的實體類型。 它必須是下列其中一個值:

MQOT_AUTH_INFO
鑑別資訊。

MQ 通道
頻道

MQOT_CLNTCONN_CHANNEL
用戶端連線通道。

MQ 接聽器
接聽器。

MQOT_NAMELIST
名單。

MQ 處理程序
程序定義。

MQOT_Q
佇列。

MQOT_Q_MGR
佇列管理程式。

MQ 服務
服務。

MQOT_TOPIC
主題。

authority
類型 :MQLONG-輸入

實體的權限。 如果實體具有一個權限，則此欄位等於適當的授權作業 (MQZAO_ * 常數)。 如果它具有多個權限，則此欄位是對應 MQZAO_ * 常數的位元 OR 運算。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此資料由佇列管理程式代表此特定元件保留; 此元件提供的任何功能對其進行的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

佇列管理程式會在 MQZ_INIT_AUTHORITY 呼叫的 *ComponentDataLength* 參數中傳遞此資料區的長度。

接續
類型 :MQLONG-輸出

依元件設定的接續指示器。 可以指定下列值:

MQZCI_DEFAULT
繼續相依於佇列管理程式。

對於 MQZ_CHECK_AUTHORITY ， 這具有與 MQZCI_STOP 相同的效果。

MQ 配置項目 _ 繼續
繼續處理下一個元件。

停止 MQZCI_STOP
請勿繼續處理下一個元件。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035 ， X'7F3') 未獲授權存取。

MQRC_SERVICE_ERROR
(2289 ， X'8F1') 存取服務時發生非預期的錯誤。

MQRC_SERVICE_NOT_AVAILABLE
(2285 ， X'8ED') 無法使用基礎服務。

MQRC_UNKNOWN_ENTITY
(2292 ， X'8F4') 無法提供服務的實體。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_GET_EXPLICIT_AUTHORITY_2 (QMgrName, &EntityData, EntityType,
                              ObjectName, ObjectType, &Authority,
                              ComponentData, &Continuation,
                              &CompCode, &Reason);
```

傳遞至服務的參數宣告如下:

```
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-起始設定授權服務

此功能由授權服務元件提供，並由佇列管理程式在元件配置期間啟動。 預期會呼叫 MQZEP ，以提供資訊給佇列管理程式。

此函數 (適用於 MQZEP) 的函數 ID 為 MQZID_INIT_AUTHORITY。

語法

MQZ_INIT_AUTHORITY(*Hconfig*, *Options*, *QMgrName*, *ComponentDataLength*, *ComponentData*, *Version*, *CompCode*, *Reason*)

參數

Hconfig
類型 :MQHCONFIG-輸入

配置控點。 此控點代表正在起始設定的特定元件。 當使用 MQZEP 函數呼叫佇列管理程式時，元件會使用它。

選項
類型 :MQLONG-輸入

起始設定選項。 它必須是下列其中一個值:

MQZIO_PRIMARY
主要起始設定。

次要 MQZIO_SECONDARY
次要起始設定。

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

***ComponentData* 長度**
類型 :MQLONG-輸入

元件資料的長度。 *ComponentData* 區域的長度 (以位元組為單位)。 此長度定義在元件配置資料中。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 在呼叫元件主要起始設定函數之前，會將此起始設定全部起始設定為零。 此資料由佇列管理程式代表此特定元件保留; 此元件所提供的任何功能 (包括起始設定功能) 對它所做的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

佇列管理程式會在 MQZ_INIT_AUTHORITY 呼叫的 *ComponentDataLength* 參數中傳遞此資料區的長度。

版本
類型 :MQLONG-輸入/輸出

版本號碼。 在輸入起始設定功能時，這會識別佇列管理程式支援的最高版本號碼。 必要的話，起始設定功能必須將此變更為它支援的介面版本。 如果傳回時佇列管理程式不支援元件所傳回的版本，它會呼叫元件 MQZ_TERM_AUTHORITY 函數，且不會進一步使用這個元件。

支援下列值:

MQZAS_VERSION_1
第 1 版。

MQZAS_VERSION_2
第 2 版。

MQZAS_VERSION_3
第 3 版。

MQZAS_VERSION_4
第 4 版。

MQZAS_VERSION_5
第 5 版。

MQZAS_VERSION_6
第 6 版。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_INITIALIZATION_FAILED
(2286 , X'8EE') 起始設定失敗，原因未定義。

MQRC_SERVICE_NOT_AVAILABLE
(2285 , X'8ED') 無法使用基礎服務。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_INIT_AUTHORITY (Hconfig, Options, QMgrName, ComponentDataLength,
                    ComponentData, &Version, &CompCode,
                    &Reason);
```

傳遞至服務的參數宣告如下:

```
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-查詢授權服務

此函數由 MQZAS_VERSION_5 授權服務元件提供，並由佇列管理程式啟動以查詢支援的功能。

在使用多個服務元件的情況下，會以與服務元件安裝順序相反的順序來呼叫服務元件。

此函數 (適用於 MQZEP) 的函數 ID 為 MQZID_INQUIRE。

語法

MQZ_INQUIRE(*QMgrName*, *SelectorCount* , *Selectors* , *IntAttrCount* , *IntAttrs* , *CharAttrLength* , *CharAttrs* , *SelectorReturned* , *ComponentData* , *Continuation* , *CompCode* , *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

SelectorCount
類型 :MQLONG-輸入

選取元數目。 在 *Selectors* 參數中提供的選取元數目。

值必須在 0 到 256 的範圍內。

選取器
類型: MQLONGxSelector 計數-輸入

選取元的陣列。 每一個選取元都會識別必要的屬性，且必須是下列其中一項:

- MQIACF_INTERFACE_VERSION (整數)
- MQIACF_USER_ID_SUPPORT (整數)
- MQCACF_SERVICE_COMPONENT (字元)

可以按任何順序指定選取元。 陣列中的選取元數目由 *SelectorCount* 參數指出。

選取元所識別的整數屬性會以出現在 *Selectors* 中的相同順序，在 *IntAttrs* 參數中傳回。

選取元所識別的字元屬性會以出現在 *Selectors* 中的相同順序，在 *CharAttrs* 參數中傳回。

IntAttrCount
類型 :MQLONG-輸入

在 IntAttrs 參數中提供的整數屬性數目。

值必須在 0 到 256 的範圍內。

IntAttrs
類型 :MQLONG ×IntAttrCount-output

整數屬性。 整數屬性的陣列。 傳回整數屬性的順序與 *Selectors* 陣列中對應的整數選取器的順序相同。

***CharAttr* 計數**
類型 :MQLONG-輸入

字元屬性緩衝區的長度。 *CharAttrs* 參數的長度 (以位元組為單位)。

該值必須至少為所要求字元屬性的長度總和。 如果未要求任何字元屬性，則零是有效值。

CharAttrs
類型 :MQLONG ×CharAttrCount-output

字元屬性緩衝區。 包含字元屬性的緩衝區連結在一起。 傳回字元屬性的順序與 *Selectors* 陣列中對應的字元選取元的順序相同。

緩衝區的長度由 CharAttrCount 參數提供。

SelectorReturned
類型 :MQLONG ×SelectorCount -輸入

已傳回選取元。 值的陣列，識別選取元參數中選取元所要求的集已傳回哪些屬性。 此陣列中的值數目由 *SelectorCount* 參數指示。 陣列中的每一個值都與選取器陣列中對應位置的選取器相關。 每一個值都是下列其中一項:

MQZSL_returned
已傳回 *Selectors* 參數中對應選取元所要求的屬性。

MQZSL_NOT_RETURNED
尚未傳回 *Selectors* 參數中對應選取元所要求的屬性。

陣列已起始設定所有值為 *MQZSL_NOT_RETURNED*。 當授權服務元件傳回屬性時，它會將陣列中的適當值設為 *MQZSL_NOT_RETURNED* 。 這可讓進行查詢呼叫的任何其他授權服務元件識別已傳回的屬性。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此資料由佇列管理程式代表此特定元件保留; 此元件提供的任何功能對其進行的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

佇列管理程式會在 MQZ_INIT_AUTHORITY 呼叫的 *ComponentDataLength* 參數中傳遞此資料區的長度。

接續
類型 :MQLONG-輸出

依元件設定的接續指示器。 可以指定下列值:

MQZCI_DEFAULT
繼續相依於佇列管理程式。

對於 MQZ_CHECK_AUTHORITY ，這具有與 MQZCI_STOP 相同的效果。

停止 MQZCI_STOP
請勿繼續處理下一個元件。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_WARNING
局部完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_WARNING:

MQRC_CHAR_ATTRS_TOO_SHORT
字元屬性的空間不足。

MQRC_INT_COUNT_TOO_SMALL
整數屬性的空間不足。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_SELECTOR_COUNT_ERROR
選取元數目無效。

MQRC_SELECTOR_ERROR
屬性選取元無效。

已超出 MQRC_SELECTOR_LIMIT_EXCEEDED
指定了太多選取元。

MQRC_INT_ATTR_COUNT_ERROR
整數屬性數目無效。

MQRC_INT_ATTRS_ARRAY_ERROR
整數屬性陣列無效。

MQRC_CHAR_ATTR_LENGTH_ERROR
字元屬性數目無效。

MQRC_CHAR_ATTRS_ERROR
字元屬性字串無效。

MQRC_SERVICE_ERROR
(2289 , X'8F1') 存取服務時發生非預期的錯誤。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_INQUIRE (QMgrName, SelectorCount, Selectors, IntAttrCount,
                  &IntAttrs, CharAttrLength, &CharAttrs,
                  SelectorReturned, ComponentData, &Continuation,
                  &CompCode, &Reason);
```

傳遞至服務的參數宣告如下:

```
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];          /* Chatacter 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 */
```

MQZ_REFRESH_CACHE-重新整理所有授權

此函數由 MQZAS_VERSION_3 授權服務元件提供，並由佇列管理程式呼叫以重新整理元件內部保留的授權清單。

此函數 (適用於 MQZEP) 的函數 ID 為 MQZID_REFRESH_CACHE (8L)。

語法

MQZ_REFRESH_CACHE(*QMgrName*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何定義的方式來使用它。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 佇列管理程式會代表此特定元件保留此資料; 會保留此元件所提供的任何功能對其進行的任何變更，並在下次呼叫此元件的其中一個功能時呈現。

佇列管理程式會在 MQZ_INIT_AUTHORITY 呼叫的 *ComponentDataLength* 參數中傳遞此資料區的長度。

接續
類型 :MQLONG-輸出

依元件設定的接續指示器。 可以指定下列值:

MQZCI_DEFAULT
繼續相依於佇列管理程式。

對於 MQZ_CHECK_AUTHORITY ， 此具有與 MQZCI_STOP 相同的效果。

MQ 配置項目 _ 繼續
繼續處理下一個元件。

停止 MQZCI_STOP
請勿繼續處理下一個元件。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_WARNING:

MQRC_SERVICE_ERROR
(2289 ， X'8F1') 存取服務時發生非預期的錯誤。

C 呼叫

```
MQZ_REFRESH_CACHE (QMgrName, ComponentData,
                   &Continuation, &CompCode, &Reason);
```

宣告參數如下:

```
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-設定權限

此函數由 MQZAS_VERSION_1 授權服務元件提供，並由佇列管理程式啟動以設定實體存取指定物件的權限。

此函數的函數 ID (適用於 MQZEP) 是 MQZID_SET_AUTHORITY。

註: 此功能會置換任何現有的權限。 若要保留任何現存的權限，您必須使用此功能重新設定它們。

語法

MQZ_SET_AUTHORITY(*QMgrName*, *EntityName*, *EntityType*, *ObjectName*, *ObjectType*, *Authority*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

EntityName
類型: MQCHAR12 -輸入

實體名稱。 要擷取其物件存取權的實體名稱。 字串的長度上限為 12 個字元; 如果它比用空白填補它的長度還短的話。 名稱不是以空值字元終止。

EntityType
類型 :MQLONG-輸入

實體類型。 *EntityName* 指定的實體類型。 它必須是下列其中一個值:

MQZAET_PRINCIPAL
校長

MQZAET_GROUP
群組。

ObjectName
類型: MQCHAR48 -輸入

物件名稱。 需要存取權的物件名稱。 字串的長度上限為 48 個字元; 如果它比以空白填補它的長度還短的話。 名稱不是以空值字元終止。

如果 *ObjectType* 是 MQOT_Q_MGR ，則此名稱與 *QMgrName* 相同。

ObjectType
類型 :MQLONG-輸入

物件類型。 *ObjectName* 指定的實體類型。 它必須是下列其中一個值:

MQOT_AUTH_INFO
鑑別資訊。

MQ 通道
頻道

MQOT_CLNTCONN_CHANNEL
用戶端連線通道。

MQ 接聽器
接聽器。

MQOT_NAMELIST
名單。

MQ 處理程序
程序定義。

MQOT_Q
佇列。

MQOT_Q_MGR
佇列管理程式。

MQ 服務
服務。

MQOT_TOPIC
主題。

authority
類型 :MQLONG-輸入

實體的權限。 如果正在設定一個權限，則此欄位等於適當的授權作業 (MQZAO_ * 常數)。 如果設定多個權限，則此欄位是對應 MQZAO_ * 常數的位元 OR 運算。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此資料由佇列管理程式代表此特定元件保留; 此元件提供的任何功能對其進行的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

佇列管理程式會在 MQZ_INIT_AUTHORITY 呼叫的 *ComponentDataLength* 參數中傳遞此資料區的長度。

接續
類型 :MQLONG-輸出

依元件設定的接續指示器。 可以指定下列值:

MQZCI_DEFAULT
繼續相依於佇列管理程式。

對於 MQZ_GET_AUTHORITY ， 其效果與 MQZCI_continue 相同。

MQ 配置項目 _ 繼續
繼續處理下一個元件。

停止 MQZCI_STOP
請勿繼續處理下一個元件。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035 ， X'7F3') 未獲授權存取。

MQRC_SERVICE_ERROR
(2289 , X'8F1') 存取服務時發生非預期的錯誤。

MQRC_SERVICE_NOT_AVAILABLE
(2285 , X'8ED') 無法使用基礎服務。

MQRC_UNKNOWN_ENTITY
(2292 , X'8F4') 無法提供服務的實體。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_SET_AUTHORITY (QMgrName, EntityName, EntityType, ObjectName,
                   ObjectType, Authority, ComponentData,
                   &Continuation, &CompCode, &Reason);
```

傳遞至服務的參數宣告如下:

```
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 -設定權限 (延伸)

此函數由 MQZAS_VERSION_2 授權服務元件提供，並由佇列管理程式啟動，以設定實體存取指定物件所需的權限。

此函數的函數 ID (適用於 MQZEP) 是 MQZID_SET_AUTHORITY。

註: 此功能會置換任何現有的權限。 若要保留任何現存的權限，您必須使用此功能重新設定它們。

MQZ_SET_AUTHORITY_2 類似於 MQZ_SET_AUTHORITY , 但具有 *EntityData* 參數所取代的 *EntityName* 參數。

語法

MQZ_SET_AUTHORITY_2(*QMgrName*, *EntityData*, *EntityType*, *ObjectName*, *ObjectType*, *Authority*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

EntityData
類型 :MQZED-輸入

實體資料。 與要設定物件授權之實體相關的資料。 請參閱第 1083 頁的『MQZED-實體描述子』，以取得詳細資料。

EntityType
類型 :MQLONG-輸入

實體類型。 *EntityData* 指定的實體類型。 它必須是下列其中一個值:

MQZAET_PRINCIPAL
校長

MQZAET_GROUP
群組。

ObjectName
類型: MQCHAR48 -輸入

物件名稱。 要設定實體權限的物件名稱。 字串的長度上限為 48 個字元; 如果它比以空白填補它的長度還短的話。 名稱不是以空值字元終止。

如果 *ObjectType* 是 MQOT_Q_MGR ， 則此名稱與 *QMgrName* 相同。

ObjectType
類型 :MQLONG-輸入

物件類型。 *ObjectName* 指定的實體類型。 它必須是下列其中一個值:

MQOT_AUTH_INFO
鑑別資訊。

MQ 通道
頻道

MQOT_CLNTCONN_CHANNEL
用戶端連線通道。

MQ 接聽器
接聽器。

MQOT_NAMELIST
名單。

MQ 處理程序
程序定義。

MQOT_Q
佇列。

MQOT_Q_MGR
佇列管理程式。

MQ 服務
服務。

MQOT_TOPIC
主題。

authority
類型 :MQLONG-輸入

實體的權限。 如果正在設定一個權限，則此欄位等於適當的授權作業 (MQZAO_ * 常數)。 如果設定多個權限，則此欄位是對應 MQZAO_ * 常數的位元 OR 運算。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此資料由佇列管理程式代表此特定元件保留; 此元件提供的任何功能對其進行的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

佇列管理程式會在 MQZ_INIT_AUTHORITY 呼叫的 *ComponentDataLength* 參數中傳遞此資料區的長度。

接續
類型 :MQLONG-輸出

依元件設定的接續指示器。 可以指定下列值:

MQZCI_DEFAULT
繼續相依於佇列管理程式。

對於 MQZ_CHECK_AUTHORITY ， 這具有與 MQZCI_STOP 相同的效果。

MQ 配置項目 _ 繼續
繼續處理下一個元件。

停止 MQZCI_STOP
請勿繼續處理下一個元件。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_NOT_AUTHORIZED
(2035 ， X'7F3') 未獲授權存取。

MQRC_SERVICE_ERROR
(2289 ， X'8F1') 存取服務時發生非預期的錯誤。

MQRC_SERVICE_NOT_AVAILABLE
(2285 ， X'8ED') 無法使用基礎服務。

MQRC_UNKNOWN_ENTITY
(2292 ， X'8F4') 無法提供服務的實體。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_SET_AUTHORITY_2 (QMgrName, &EntityData, EntityType, ObjectName,
                     ObjectType, Authority, ComponentData,
                     &Continuation, &CompCode, &Reason);
```

傳遞至服務的參數宣告如下:

```
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-終止授權服務

此功能由授權服務元件提供，並由佇列管理程式在不再需要此元件的服務時啟動。 此功能必須執行元件所需的任何清除。

此函數 (適用於 MQZEP) 的函數 ID 是 MQZID_TERM_AUTHORITY。

語法

MQZ_TERM_AUTHORITY(*Hconfig*, *Options*, *QMgrName*, *ComponentData*, *CompCode*, *Reason*)

參數

Hconfig
類型 :MQHCONFIG-輸入

配置控點。 此控點代表要終止的特定元件。 當使用 MQZEP 函數呼叫佇列管理程式時，元件會使用它。

選項
類型 :MQLONG-輸入

終止選項。 它必須是下列其中一個值:

MQZTO_PRIMARY
主要終止。

次要 MQZTO_SECONDARY
次要終止。

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此資料由佇列管理程式代表此特定元件保留; 此元件提供的任何功能對其進行的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

此資料區的長度由佇列管理程式在 MQZ_INIT_AUTHORITY 呼叫上的 ComponentData 長度參數中傳遞。

當 MQZ_TERM_AUTHORITY 呼叫已完成時，佇列管理程式會捨棄此資料。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_SERVICE_NOT_AVAILABLE
(2285 , X'8ED') 無法使用基礎服務。

MQRC_TERMINATION_FAILED
(2287 , X'8FF') 終止失敗，原因未定義。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_TERM_AUTHORITY (Hconfig, Options, QMgrName, ComponentData,
                    &CompCode, &Reason);
```

傳遞至服務的參數宣告如下:

```
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-刪除名稱

此功能由名稱服務元件提供，並由佇列管理程式啟動以刪除指定佇列的項目。

此函數 (適用於 MQZEP) 的函數 ID 是 MQZID_DELETE_NAME。

語法

MQZ_DELETE_NAME(*QMgrName*, *QName*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

完整名稱
類型: MQCHAR48 -輸入

佇列名稱。 要刪除其項目的佇列名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此資料由佇列管理程式代表此特定元件保留; 此元件提供的任何功能對其進行的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

此資料區的長度由佇列管理程式在 MQZ_INIT_NAME 呼叫的 ComponentData 長度參數中傳遞。

接續
類型 :MQLONG-輸出

依元件設定的接續指示器。 它必須是下列其中一個值:

MQZCI_DEFAULT
繼續相依於佇列管理程式。

停止 MQZCI_STOP
請勿繼續處理下一個元件。

對於 **MQZ_DELETE_NAME** 指令，不論 **Continuation** 參數中傳回什麼，佇列管理程式都不會嘗試啟動另一個元件。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_WARNING
警告 (局部完成)。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_WARNING:

MQRC_UNKNOWN_NAME
(2288 ， X'8F0') 找不到佇列名稱。

註: 如果基礎服務以此案例成功回應，則可能無法傳回此代碼。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_SERVICE_ERROR
(2289 ， X'8F1') 存取服務時發生非預期的錯誤。

MQRC_SERVICE_NOT_AVAILABLE
(2285 ， X'8ED') 無法使用基礎服務。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_DELETE_NAME (QMgrName, QName, ComponentData, &Continuation,
                 &CompCode, &Reason);
```

傳遞至服務的參數宣告如下:

```
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-起始設定名稱服務

此功能由名稱服務元件提供，並由佇列管理程式在元件配置期間啟動。 預期會呼叫 MQZEP ， 以提供資訊給佇列管理程式。

此函數 (適用於 MQZEP) 的函數 ID 為 MQZID_INIT_NAME。

語法

MQZ_INIT_NAME(*Hconfig*, *Options*, *QMgrName*, *ComponentDataLength*, *ComponentData*, *Version*, *CompCode*, *Reason*)

參數

Hconfig
類型 :MQHCONFIG-輸入

配置控點。 此控點代表正在起始設定的特定元件。 當使用 MQZEP 函數呼叫佇列管理程式時，元件會使用它。

選項
類型 :MQLONG-輸入

起始設定選項。 它必須是下列其中一個值:

MQZIO_PRIMARY
主要起始設定。

次要 MQZIO_SECONDARY
次要起始設定。

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

***ComponentData* 長度**
類型 :MQLONG-輸入

元件資料的長度。 *ComponentData* 區域的長度 (以位元組為單位)。 此長度定義在元件配置資料中。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 在呼叫元件主要起始設定函數之前，會將此起始設定全部起始設定為零。 此資料由佇列管理程式代表此特定元件保留; 此元件所提供的任何功能 (包括起始設定功能) 對它所做的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

佇列管理程式會在 MQZ_INIT_AUTHORITY 呼叫的 *ComponentDataLength* 參數中傳遞此資料區的長度。

版本
類型 :MQLONG-輸入/輸出

版本號碼。 在輸入起始設定功能時，這會識別佇列管理程式支援的最高版本號碼。 必要的話，起始設定功能必須將此變更為它支援的介面版本。 如果傳回時佇列管理程式不支援元件所傳回的版本，則它會呼叫元件 MQZ_TERM_NAME 函數，且不會進一步使用此元件。

支援下列值:

MQZAS_VERSION_1
第 1 版。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_INITIALIZATION_FAILED
(2286 , X'8EE') 起始設定失敗，原因未定義。

MQRC_SERVICE_NOT_AVAILABLE
(2285 , X'8ED') 無法使用基礎服務。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_INIT_NAME (Hconfig, Options, QMgrName, ComponentDataLength,
               ComponentData, &Version, &CompCode, &Reason);
```

傳遞至服務的參數宣告如下:

```
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-插入名稱

此功能由名稱服務元件提供，並由佇列管理程式啟動以插入指定佇列的項目，其中包含擁有佇列的佇列管理程式名稱。 如果已在服務中定義佇列，則呼叫會失敗。

此函數 (適用於 MQZEP) 的函數 ID 是 MQZID_INSERT_NAME。

語法

MQZ_INSERT_NAME(*QMgrName*, *QName*, *ResolvedQMgrName*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

完整名稱
類型: MQCHAR48 -輸入

佇列名稱。 要插入項目的佇列名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

***ResolvedQMgr* 名稱**
類型: MQCHAR48 -輸入

已解析佇列管理程式名稱。 佇列解析成的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此資料由佇列管理程式代表此特定元件保留; 此元件所提供的任何功能 (包括起始設定功能) 對它所做的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

佇列管理程式會在 MQZ_INIT_NAME 呼叫的 *ComponentDataLength* 參數中傳遞此資料區的長度。

接續
類型 :MQLONG-輸入/輸出

依元件設定的接續指示器。 對於 MQZ_INSERT_NAME ， 佇列管理程式不會嘗試啟動另一個元件，不論 *Continuation* 參數中傳回了什麼。

支援下列值:

MQZCI_DEFAULT
繼續相依於佇列管理程式。

停止 MQZCI_STOP
請勿繼續處理下一個元件。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_Q_ALREADY_EXISTS
(2290 ， X'8F2') 佇列物件已存在。

MQRC_SERVICE_ERROR
(2289 ， X'8F1') 存取服務時發生非預期的錯誤。

MQRC_SERVICE_NOT_AVAILABLE
(2285 ， X'8ED') 無法使用基礎服務。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_INSERT_NAME (QMgrName, QName, ResolvedQMgrName, ComponentData,
                 &Continuation, &CompCode, &Reason);
```

傳遞至服務的參數宣告如下:

```
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-查閱名稱

此功能由名稱服務元件提供，並由佇列管理程式啟動，以擷取所指定佇列的擁有端佇列管理程式名稱。

此函數的函數 ID (適用於 MQZEP) 是 MQZID_LOOKUP_NAME。

語法

MQZ_LOOKUP_NAME(*QMgrName*, *QName*, *ResolvedQMgrName*, *ComponentData*, *Continuation*, *CompCode*, *Reason*)

參數

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

完整名稱
類型: MQCHAR48 -輸入

佇列名稱。 要解析其項目的佇列名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

***ResolvedQMgr* 名稱**
類型: MQCHAR48 -輸出

已解析佇列管理程式名稱。 如果功能順利完成，則這是擁有佇列的佇列管理程式名稱。

服務元件傳回的名稱必須以空白填補在參數完整長度的右邊; 名稱不得以空值字元終止，或包含前導或內含空白。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此資料由佇列管理程式代表此特定元件保留; 此元件所提供的任何功能 (包括起始設定功能) 對它所做的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

佇列管理程式會在 MQZ_INIT_NAME 呼叫的 *ComponentDataLength* 參數中傳遞此資料區的長度。

接續
類型 :MQLONG-輸出

依元件設定的接續指示器。 對於 MQZ_LOOKUP_NAME ， 佇列管理程式會指定是否要啟動另一個名稱服務元件，如下所示:

- 如果 *CompCode* 是 MQCC_OK ， 則不會啟動進一步的元件，無論 接續中傳回任何值。
- 如果 *CompCode* 不是 MQCC_OK ， 則會啟動另一個元件，除非 *Continuation* 是 MQZCI_STOP。

支援下列值:

MQZCI_DEFAULT
繼續相依於佇列管理程式。

MQ 配置項目 _ 繼續
繼續處理下一個元件。

停止 MQZCI_STOP
請勿繼續處理下一個元件。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 , X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_SERVICE_ERROR
(2289 , X'8F1') 存取服務時發生非預期的錯誤。

MQRC_SERVICE_NOT_AVAILABLE
(2285 , X'8ED') 無法使用基礎服務。

MQRC_UNKNOWN_Q_NAME
(2288 , X'8F0') 找不到佇列名稱。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_LOOKUP_NAME (QMgrName, QName, ResolvedQMgrName, ComponentData,
                 &Continuation, &CompCode, &Reason);
```

傳遞至服務的參數宣告如下:

```
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-終止名稱服務

此功能由名稱服務元件提供，並由佇列管理程式在不再需要此元件的服務時啟動。 此功能必須執行元件所需的任何清除。

此函數 (適用於 MQZEP) 的函數 ID 是 MQZID_TERM_NAME。

語法
MQZ_TERM_NAME(*Hconfig*, *Options*, *QMgrName*, *ComponentData*, *CompCode*, *Reason*)

參數

Hconfig
類型 :MQHCONFIG-輸入

配置控點。 此控點代表要終止的特定元件。 當使用 MQZEP 函數呼叫佇列管理程式時，元件會使用它。

選項
類型 :MQLONG-輸入

終止選項。 它必須是下列其中一個值:

MQZTO_PRIMARY
主要終止。

次要 MQZTO_SECONDARY
次要終止。

QMgrName
類型: MQCHAR48 -輸入

佇列管理程式名稱。 呼叫元件的佇列管理程式名稱。 此名稱以空白填補參數的完整長度; 名稱不是以空值字元結尾。

佇列管理程式名稱會傳遞至元件以取得資訊; 授權服務介面不需要元件以任何已定義的方式來使用它。

ComponentData
類型 :MQBYTE ×ComponentData 長度-輸入/輸出

元件資料。 此資料由佇列管理程式代表此特定元件保留; 此元件所提供的任何功能 (包括起始設定功能) 對它所做的任何變更都會保留，並在下次呼叫其中一個元件功能時呈現。

元件資料位於可供所有處理程序存取的共用記憶體中。

佇列管理程式會在 MQZ_INIT_NAME 呼叫的 *ComponentDataLength* 參數中傳遞此資料區的長度。

當 MQZ_TERM_NAME 呼叫完成時，佇列管理程式會捨棄此資料。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_TERMINATION_FAILED
(2287 ， X'8FF') 終止失敗，原因未定義。

MQRC_SERVICE_NOT_AVAILABLE
(2285 ， X'8ED') 無法使用基礎服務。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZ_TERM_NAME (Hconfig, Options, QMgrName, ComponentData, &CompCode,
               &Reason);
```

傳遞至服務的參數宣告如下:

```
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-應用程式環境定義

MQZAC 結構用於 *ApplicationContext* 參數的 MQZ_AUTHENTICATE_USER 呼叫。 此參數指定與呼叫端應用程式相關的資料。

表 *1* 彙總結構中的欄位。

表 *596: MQZAC* 中的欄位

欄位	說明
StrucId	結構 ID
版本	結構版本號碼
ProcessId	處理程序 ID
ThreadId	執行緒 ID
ApplName	應用程式名稱
UserID	使用者 ID
EffectiveUserID	有效使用者 ID
環境	環境
CallerType	呼叫程式類型
AuthenticationType	鑑別類型
BindType	連結類型

欄位

StrucId
類型: MQCHAR4 -輸入

結構 ID。 值如下所示:

MQZAC_STRUC_ID
應用程式環境定義結構的 ID。

對於 C 程式設計語言，也會定義常數 MQZAC_STR_ID_ARRAY; 此值與 MQZAC_STRUC_ID 相同，但卻是字元陣列而非字串。

版本
類型 :MQLONG-輸入

結構版本號碼。 值如下所示:

MQZAC_VERSION_1
Version-1 應用程式環境定義結構。 常數 MQZAC_CURRENT_VERSION 指定現行版本的版本號碼。

ProcessId
類型 :MQPID-輸入

應用程式的處理程序 ID。

ThreadId
類型 :MQTID-輸入

應用程式的執行緒 ID。

ApplName
類型: MQCHAR28 -輸入

應用程式名稱。

UserID
類型: MQCHAR12 -輸入

使用者 ID。 在 UNIX 系統上，此欄位指定應用程式的實際使用者 ID。 在 Windows 上，此欄位指定應用程式的使用者 ID。

EffectiveUserID
類型: MQCHAR12 -輸入

有效使用者 ID。 在 UNIX 系統上，此欄位指定應用程式的有效使用者 ID。 在 Windows 上，此欄位為空白。

environment
類型 :MQLONG-輸入

環境。 此欄位指定從中進行呼叫的環境。 欄位是下列其中一個值:

MQXE_COMMAND_SERVER
指令伺服器

MQXE_MQSC
runmqsc 指令直譯器

MQXE_MCA
訊息通道代理程式 MQXE_OTHER

MQXE_OTHER
未定義的環境

CallerType
類型 :MQLONG-輸入

呼叫程式類型。 此欄位指定進行呼叫的程式類型。 欄位是下列其中一個值:

MQXACT_EXTERNAL
此呼叫是佇列管理程式的外部呼叫。

MQXACT_INTERNAL
此呼叫是佇列管理程式的內部呼叫。

AuthenticationType
類型 :MQLONG-輸入

鑑別類型。 此欄位指定正在執行的鑑別類型。 欄位是下列其中一個值:

MQZAT_INITIAL_CONTEXT
鑑別呼叫是因為正在起始設定使用者環境定義。 此值在 MQCONN 或 MQCONNX 呼叫期間使用。

MQZAT_CHANGE_CONTEXT
鑑別呼叫是因為正在變更使用者環境定義。 當 MCA 變更使用者環境定義時，會使用此值。 上層主題 :MQZAC-

BindType
類型 :MQLONG-輸入

連結類型。 此欄位指定使用中連結的類型。 欄位是下列其中一個值:

MQCNO_FASTPATH_BINDING
捷徑連結。

MQCNO_SHARED_BINDING
共用連結。

MQCNO_ISOLATED_BINDING
隔離的連結。

C 宣告

宣告結構的欄位，如下所示:

```
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-權限資料

MQZAD 結構在 MQZ_ENUMERATE_AUTHORITY_DATA 呼叫中用於兩個參數: 一個輸入及一個輸出。

- MQZAD 用於輸入至呼叫的 *Filter* 參數。 此參數指定用來選取呼叫所傳回之權限資料的選取準則。
- MQZAD 也用於從呼叫輸出的 *AuthorityBuffer* 參數。 此參數指定設定檔名稱、物件類型及實體之組合的授權。

表 *1.* 彙總結構中的欄位。

表 *597: MQZAD* 中的欄位	
欄位	**說明**
StrucId	結構 ID
版本	結構版本號碼
ProfileName	處理程序 ID
ObjectType	執行緒 ID
權限管理中心	應用程式名稱
EntityDataPtr	使用者 ID
EntityType	環境
選項	呼叫程式類型

欄位

StrucId

類型: MQCHAR4 -輸入

結構 ID。 值如下所示:

MQZAC_STRUC_ID
應用程式環境定義結構的 ID。

對於 C 程式設計語言，也會定義常數 MQZAC_STR_ID_ARRAY; 此值與 MQZAC_STRUC_ID 相同，但卻是字元陣列而非字串。

版本

類型 :MQLONG-輸入

結構版本號碼。 值如下所示:

MQZAC_VERSION_1
Version-1 應用程式環境定義結構。 常數 MQZAC_CURRENT_VERSION 指定現行版本的版本號碼。

下列常數指定現行版本的版本號碼:

MQZAD_CURRENT_VERSION
權限資料結構的現行版本。

ProfileName
類型: MQCHAR48 -輸入

設定檔名稱。

對於 過濾器 參數，此欄位是需要權限資料的設定檔名稱。 如果名稱到欄位結尾或第一個空值字元為止都是空白，則會傳回所有設定檔名稱的權限資料。

對於 *AuthorityBuffer* 參數，此欄位是符合指定選取準則的設定檔名稱。

ObjectType
類型 :MQLONG-輸入

物件類型。

對於 過濾器 參數，此欄位是需要其權限資料的物件類型。 如果值為 MQOT_ALL ，則會傳回所有物件類型的權限資料。

對於 *AuthorityBuffer* 參數，此欄位是套用 *ProfileName* 參數所識別設定檔的物件類型。

該值是下列其中一項; 對於 *Filter* 參數， MQOT_ALL 值也有效:

MQOT_AUTH_INFO
鑑別資訊

MQ 通道
通道

MQOT_CLNTCONN_CHANNEL
用戶端連線通道

MQ 接聽器
接聽器

MQOT_NAMELIST
名稱清單

MQ 處理程序
程序定義

MQOT_Q
佇列

MQOT_Q_MGR
佇列管理程式

MQ 服務
服務

權限管理中心
類型 :MQLONG-輸入

權限。

對於 過濾器 參數，會忽略此欄位。

對於 *AuthorityBuffer* 參數，此欄位代表實體對 *ProfileName* 及 *ObjectType* 所識別物件的授權。 如果實體只有一個權限，則欄位等於適當的授權值 (MQZAO_ * 常數)。 如果實體具有多個權限，則欄位是對應 MQZAO_ * 常數的位元 OR 運算。

EntityDataPtr
類型 :PMQZED-輸入

識別實體的 MQZED 結構位址。

對於 過濾器 參數，此欄位指向 MQZED 結構，用於識別需要權限資料的實體。 如果 *EntityDataPtr* 是空值指標，則會傳回所有實體的權限資料。

對於 *AuthorityBuffer* 參數，此欄位會指向 MQZED 結構，用來識別已傳回其權限資料的實體。

EntityType
類型 :MQLONG-輸入

實體類型。

對於 過濾器 參數，此欄位指定需要權限資料的實體類型。 如果值為 MQZAET_NONE ，則會傳回所有實體類型的權限資料。

對於 *AuthorityBuffer* 參數，此欄位指定 *EntityDataPtr* 參數所指向 MQZED 結構所識別的實體類型。

該值是下列其中一項; 對於 過濾器 參數， MQZAET_NONE 值也有效:

MQZAET_PRINCIPAL
主體

MQZAET_GROUP
群組

選項
類型 :MQAUTHOPT-輸入

選項。 此欄位指定的選項可讓您控制所顯示的設定檔。 必須指定下列其中一個值:

MQAUTHOPT_NAME_ALL_MATCHING
顯示所有設定檔

MQAUTHOPT_NAME_EXPLICIT
顯示與 *ProfileName* 欄位中指定的名稱完全相同的設定檔。

此外，還必須指定下列其中一項:

MQAUTHOPT_ENTITY_SET
顯示用來計算實體對 *ProfileName* 參數所指定物件的累積權限的所有設定檔。 *ProfileName* 參數不得包含任何萬用字元。

如果指定的實體是主體，則針對集 {entity, groups} 的每一個成員，會顯示適用於物件的最適用設定檔。

如果指定的實體是群組，則會顯示套用至物件之群組中最適用的設定檔。

如果指定此值，則 *ProfileName*、 *ObjectType*、 *EntityType* 及 *EntityDataPtr* MQZED 結構中指定的實體名稱的值都必須是非空白。

如果您已指定 MQAUTHOPT_NAME_ALL_MATCHING ，則也可以指定下列值:

MQAUTHOPT_ENTITY_EXPLICIT
顯示與 *EntityDataPtr* MQZED 結構中所指定實體名稱完全相同的設定檔。

C 宣告

```
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 */
};
```

欄位

MQZED-實體描述子

在許多授權服務呼叫中使用 MQZED 結構，以指定要檢查其授權的實體。

表 *1.* 彙總結構中的欄位。

表 *598: MQZED* 中的欄位	
欄位	**說明**
StrucId	結構 ID
版本	版本
EntityName Ptr	實體名稱
EntityDomainPtr	實體網域指標
SecurityId	安全 ID
CorrelationPtr	相關性指標

欄位

StrucId
類型: MQCHAR4 -輸入

結構 ID。 值如下所示:

MQZED_STRUC_ID
實體描述子結構的 ID。

對於 C 程式設計語言，也會定義常數 MQZED_STRUC_ARRAY; 此值與 MQZED_STRUC_ID 相同，但它是字元陣列而非字串。

版本
類型 :MQLONG-輸入

結構版本號碼。 值如下所示:

MQZED_VERSION_1
Version-1 實體描述子結構。

下列常數指定現行版本的版本號碼:

MQZED_CURRENT_VERSION
實體描述子結構的現行版本。

EntityNamePtr
類型 :PMQCHAR-輸入

設定檔名稱。

實體名稱的位址。 這是指向要檢查其授權之實體名稱的指標。

EntityDomainPtr
類型 :PMQCHAR-輸入

實體網域名稱的位址。 這是指向網域名稱的指標，該網域包含要檢查其授權的實體定義。

SecurityId
類型: MQBYTE40 -輸入

權限。

安全 ID。 這是要檢查其授權的安全 ID。

CorrelationPtr
類型 :MQPTR-輸入

相關性指標。 這有助於在鑑別使用者函數與其他適當的 OAM 函數之間傳遞相關性資料。

C 宣告

```
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 */
```

欄位

MQZEP-新增元件進入點

在起始設定期間，服務元件會啟動此功能，以將進入點新增至該服務元件的進入點向量。

語法

MQZEP (*Hconfig*、 函數、 *EntryPoint*、 *CompCode*、 原因)

參數

Hconfig
類型 :MQHCONFIG-輸入

配置控點。 此控點代表針對此特定可安裝服務所配置的元件。 它必須與佇列管理程式在元件起始設定呼叫上傳遞給元件配置功能的元件相同。

函數
類型 :MQLONG-輸入

函數 ID。 這會針對每一個可安裝服務定義有效值。

如果針對相同函數多次呼叫 MQZEP ，則最後一次呼叫會提供所使用的進入點。

EntryPoint
類型 :PMQFUNC-輸入

函數進入點。 這是元件提供用來執行功能的進入點位址。

值 NULL 是有效的，指出此元件未提供此函數。 對於未使用 MQZEP 定義的進入點，假設為 NULL。

CompCode
類型 :MQLONG-輸出

完成碼。 它必須是下列其中一個值:

MQCC_OK
順利完成。

MQCC_FAILED
呼叫失敗。

原因
類型 :MQLONG-輸出

定義 *CompCode* 的原因碼。

如果 *CompCode* 是 MQCC_OK:

MQRC_NONE
(0 ， X'000 ') 沒有理由報告。

如果 *CompCode* 是 MQCC_FAILED:

MQRC_FUNCTION_ERROR
(2281 , X'8E9') 函數 ID 無效。

MQRC_HCONFIG_ERROR
(2280 , X'8E8') 配置控點無效。

如需這些原因碼的相關資訊，請參閱 API 原因碼。

C 呼叫

```
MQZEP (Hconfig, Function, EntryPoint, &CompCode, &Reason);
```

宣告參數如下:

```
MQHCONFIG  Hconfig;     /* Configuration handle */
MQLONG     Function;    /* Function identifier */
PMQFUNC    EntryPoint;  /* Function entry point */
MQLONG     CompCode;    /* Completion code */
MQLONG     Reason;      /* Reason code qualifying CompCode */
```

MQZFP-免費參數

MQZFP 結構用於 *FreeParms* 參數的 MQZ_FREE_USER 呼叫。 此參數指定與要釋放的資源相關的資料。

表 *1.* 彙總結構中的欄位。

表 *599: MQZFP* 中的欄位	
欄位	**說明**
StrucId	結構 ID
版本	版本
已保留	保留欄位
CorrelationPtr	相關性指標

欄位

StrucId
類型: MQCHAR4 -輸入

結構 ID。 值如下所示:

MQZIC_STRUC_ID
身分環境定義結構的 ID。 對於 C 程式設計語言，也會定義常數 MQZIC_STRUC_ID_ARRAY; 此值與 MQZIC_STRUC_ID 相同，但它是字元陣列而非字串。

版本
類型 :MQLONG-輸入

結構版本號碼。 值如下所示:

MQZFP_VERSION_1
Version-1 可用參數結構。

下列常數指定現行版本的版本號碼:

MQZFP_CURRENT_VERSION
免費參數結構的現行版本。

已保留
類型: MQBYTE8 -輸入

保留欄位。 起始值是空值。

CorrelationPtr
類型 :MQPTR-輸入

相關性指標。 與要釋放的資源相關的相關性資料位址。

C 宣告

```
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 */
};
```

欄位

MQZIC-身分環境定義

MQZIC 結構用於 *IdentityContext* 參數的 MQZ_AUTHENTICATE_USER 呼叫。

MQZIC 結構包含身分環境定義資訊，可識別第一次將訊息放置在佇列上的應用程式使用者:

- 佇列管理程式會在 *UserIdentifier* 欄位中填入識別使用者的名稱，其執行方式取決於應用程式執行所在的環境。
- 佇列管理程式會在 *AccountingToken* 欄位中填入它從放置訊息的應用程式所決定的記號或號碼。
- 應用程式可以使用 *ApplIdentity* 資料 欄位，以取得他們想要包含的任何關於使用者的額外資訊 (例如，已加密密碼)。

適當授權的應用程式可以使用 MQZ_AUTHENTICATE_USER 函數來設定身分環境定義。

當在 WebSphere MQ for Windows 下建立訊息時， Windows 系統安全 ID (SID) 會儲存在 *AccountingToken* 欄位中。 SID 可用來補充 *UserIdentifier* 欄位，以及建立使用者的認證。

表 *1.* 彙總結構中的欄位。

表 *600: MQZIC* 中的欄位	
欄位	**說明**
StrucId	結構 ID
版本	版本
UserIdentifier	使用者 ID
AccountingToken	帳戶記號
ApplIdentityData	應用程式身分資料

欄位

StrucId
類型: MQCHAR4 -輸入

結構 ID。 值如下所示:

MQZIC_STRUC_ID
身分環境定義結構的 ID。 對於 C 程式設計語言，也會定義常數 MQZIC_STRUC_ID_ARRAY; 此值與 MQZIC_STRUC_ID 相同，但它是字元陣列而非字串。

版本
類型 :MQLONG-輸入

結構版本號碼。 值如下所示:

MQZIC_VERSION_1
Version-1 身分環境定義結構。

下列常數指定現行版本的版本號碼:

MQZIC_CURRENT_VERSION
身分環境定義結構的現行版本。

UserIdentifier
類型: MQCHAR12 -輸入

使用者 ID。 這是訊息身分環境定義的一部分。 *UserIdentifier* 指定產生訊息之應用程式的使用者 ID。 佇列管理程式會將此資訊視為字元資料，但不會定義其格式。 如需 *UserIdentifier* 欄位的相關資訊，請參閱 第 400 頁的『UserIdentifier (MQCHAR12)』。

AccountingToken
類型: MQBYTE32 -輸入

結算記號。 這是訊息身分環境定義的一部分。 *AccountingToken* 可讓應用程式對因訊息而完成的工作收取適當的費用。 佇列管理程式會將此資訊視為位元字串，且不會檢查其內容。 如需 *AccountingToken* 欄位的相關資訊，請參閱 第 366 頁的『AccountingToken (MQBYTE32)』。

ApplIdentityData
類型: MQCHAR32 -輸入

與身分相關的應用程式資料。 這是訊息身分環境定義的一部分。 ApplIdentity 資料是由應用程式套組所定義的資訊，可用來提供訊息來源的其他相關資訊。 例如，它可以由以適當使用者權限執行的應用程式設定，以指出身分資料是否受信任。 如需 ApplIdentity 資料欄位的相關資訊，請參閱 第 368 頁的『ApplIdentity 資料 (MQCHAR32)』。

C 宣告

```
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 */
};
```

欄位

HTTP IBM WebSphere MQ 橋接器的參考資料

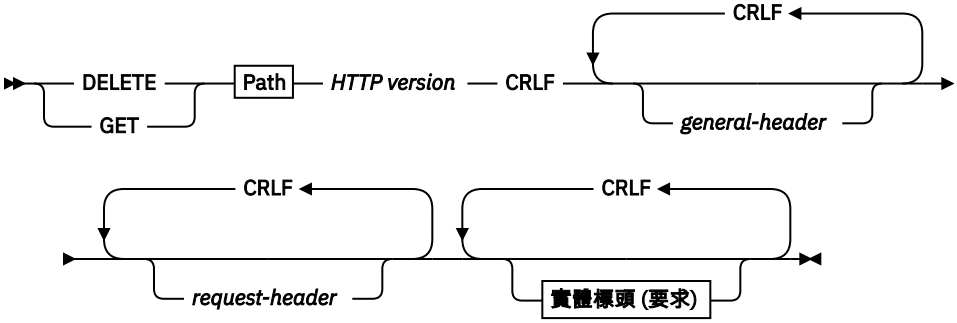
HTTP IBM WebSphere MQ 橋接器的參考主題，按字母順序排列

HTTP DELETE: WebSphere MQ for HTTP 指令橋接器

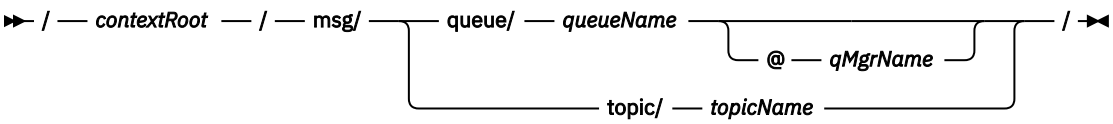
HTTP **DELETE** 作業會從 WebSphere MQ 佇列取得訊息，或從主題擷取發佈資訊。 從佇列中移除訊息。 如果保留發佈，則不會移除它。 回應訊息會傳回給用戶端，包括訊息的相關資訊。

語法

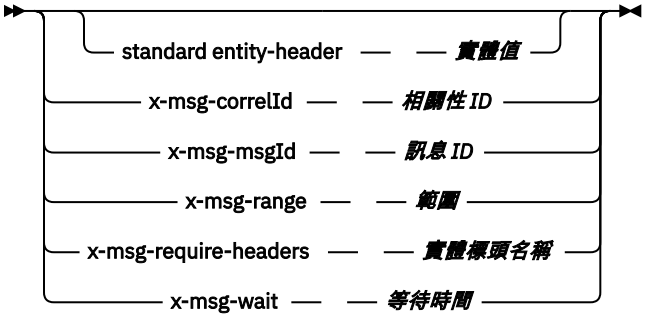
要求



Path



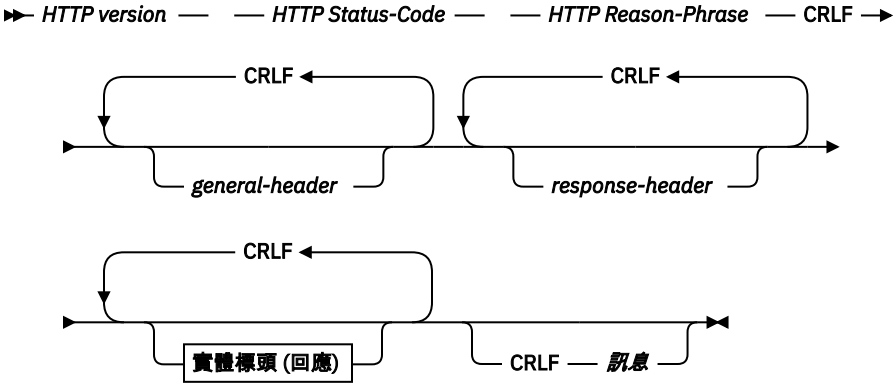
entity-header (要求)



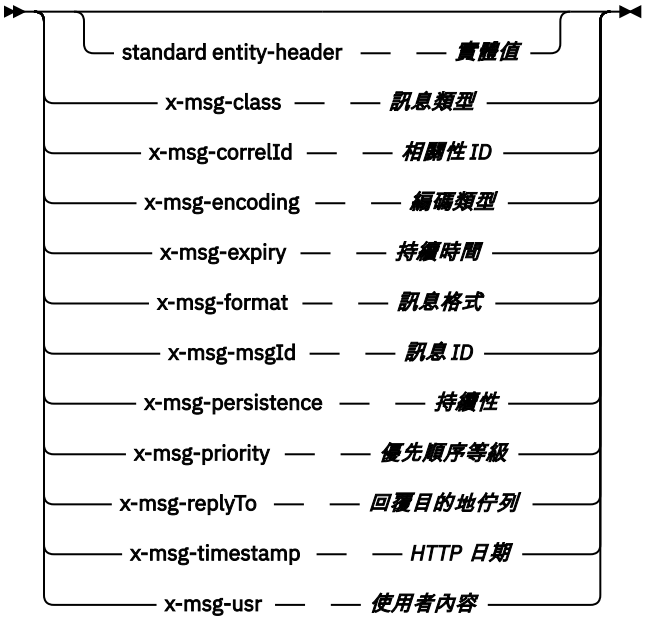
註:

1. If a question mark (?) is used it must be substituted with `%3f`. For example, `orange?topic` should be specified as `orange%3ftopic`.
2. @*qMgrName* is only valid on an HTTP **POST**

回應



entity-header (回應)



要求參數

路徑
請參閱 第 1120 頁的『URI 格式』。

***HTTP* 版本**
HTTP 版本; 例如 `HTTP/1.1`

一般標頭
請參閱 HTTP/1.1 - 4.5 一般標頭欄位。

要求標頭
請參閱 HTTP/1.1 - 5.3 要求標頭欄位。 在 HTTP/1.1 要求上， 主機 欄位是必要的。 通常由您用來建立用戶端要求的工具自動插入。

實體標頭 *(要求)*
請參閱 HTTP/1.1 - 7.1 實體標頭欄位。 要求語法圖中列出的其中一個實體標頭。

回應參數

路徑
請參閱 第 1120 頁的『URI 格式』。

***HTTP* 版本**
HTTP 版本; 例如 `HTTP/1.1`

一般標頭
請參閱 HTTP/1.1 - 4.5 一般標頭欄位。

回應標頭
請參閱 HTTP/1.1 - 6.2 回應標頭欄位。

實體標頭 *(回應)*
請參閱 HTTP/1.1 - 7.1 實體標頭欄位。 「回應」語法圖中列出的其中一個實體或回應標頭。 `Content-Length` 一律呈現在回應中。 如果沒有訊息內文，則設為零。

message
訊息內文。

說明

如果 HTTP **DELETE** 要求成功，回應訊息會包含從 WebSphere MQ 佇列擷取的資料。 HTTP `Content-Length` 標頭中會傳回訊息內文中的位元組數。 HTTP 回應的狀態碼設為 `200 OK`。 如果 `x-msg-range` 指定為 `0` 或 `0-0`，則 HTTP 回應的狀態碼是 `204 No Content`。

如果 HTTP **DELETE** 要求不成功，則回應包括 HTTP 錯誤訊息的 WebSphere MQ 橋接器及 HTTP 狀態碼。

HTTP DELETE 範例

HTTP **DELETE** 從佇列中取得並刪除訊息，或者擷取並刪除發佈資訊。 **HTTPDELETE** Java 範例是從佇列讀取訊息的 HTTP **DELETE** 要求範例。 您可以改用瀏覽器表單或 AJAX 工具箱來建立 HTTP **DELETE** 要求，而不使用 Java。

第 1090 頁的圖 37 是一個 HTTP 要求，用來刪除佇列上稱為 `myQueue` 的下一則訊息。 訊息內文將傳回給用戶端作為回應。 在 WebSphere MQ 術語中， HTTP **DELETE** 是破壞性取得。

此要求包含 HTTP 要求標頭 `x-msg-wait`，它指示用於 HTTP 的 WebSphere MQ 橋接器要等待多久，訊息才會抵達佇列。 此要求還包含 `x-msg-require-headers` 要求標頭，其指定用戶端將在回應中收到訊息相關性 ID。

```
DELETE /msg/queue/myQueue/ HTTP/1.1
Host: www.example.org
x-msg-wait: 10
x-msg-require-headers: correlID
```

圖 *37: HTTP **DELETE*** 要求範例

第 1090 頁的圖 38，是傳回給用戶端的回應。 相關性 ID 會傳回給用戶端，如該要求的 `x-msg-require-headers` 中所要求。

```
HTTP/1.1 200 OK
Date: Wed, 2 Jan 2007 22:38:34 GMT
Server: Apache-Coyote/1.1 WMQ-HTTP/1.1 JEE-Bridge/1.1
Content-Length: 50
Content-Type: text/plain; charset=utf-8
x-msg-correlId: 1234567890

Here is my message body that is retrieved from the queue.
```

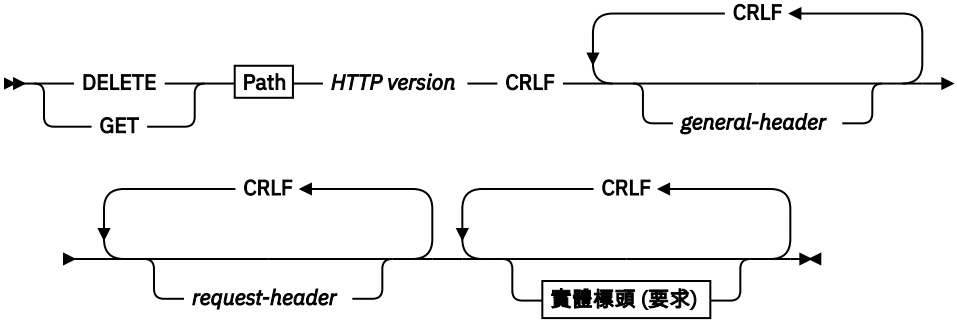
圖 *38: HTTP **DELETE*** 回應範例

HTTP GET: WebSphere MQ for HTTP 指令橋接器

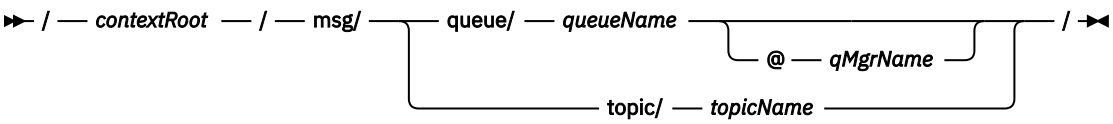
HTTP **GET** 作業會從 WebSphere MQ 佇列取得訊息。 訊息留在佇列上。 HTTP **GET** 作業相當於瀏覽 WebSphere MQ 佇列。

語法

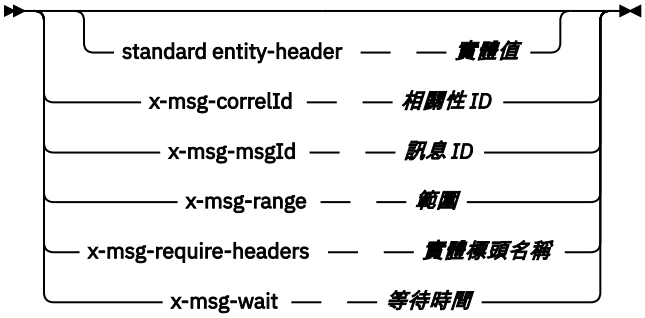
要求



Path



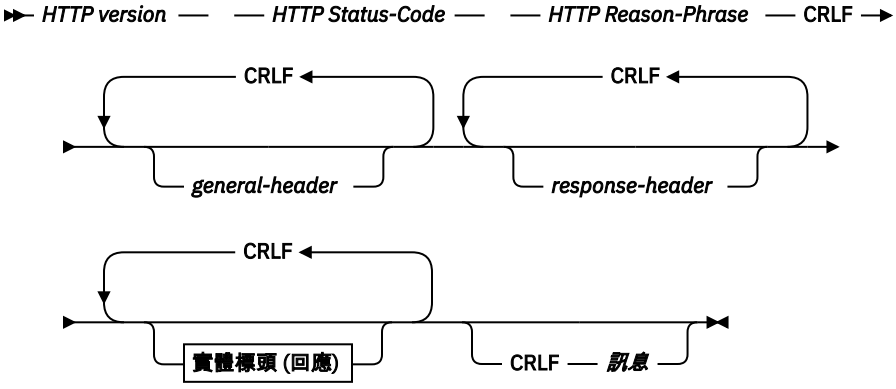
entity-header (要求)



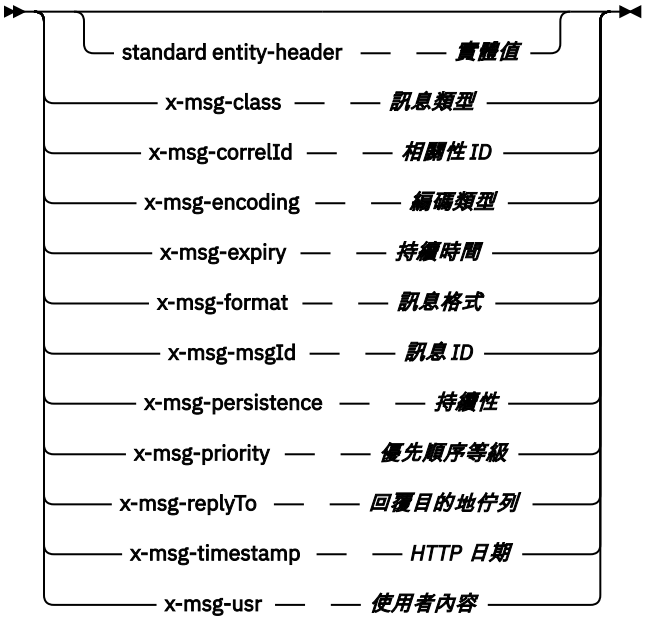
註:

1. If a question mark (?) is used it must be substituted with `%3f`. For example, `orange?topic` should be specified as `orange%3ftopic`.
2. @*qMgrName* is only valid on an HTTP **POST**

回應



entity-header (回應)



要求參數

路徑
請參閱 第 1120 頁的『URI 格式』。

***HTTP* 版本**
HTTP 版本; 例如 `HTTP/1.1`

一般標頭
請參閱 HTTP/1.1 - 4.5 一般標頭欄位。

要求標頭
請參閱 HTTP/1.1 - 5.3 要求標頭欄位。 在 HTTP/1.1 要求上， 主機 欄位是必要的。 通常由您用來建立用戶端要求的工具自動插入。

實體標頭 *(要求)*
請參閱 HTTP/1.1 - 7.1 實體標頭欄位。 要求語法圖中列出的其中一個實體標頭。

回應參數

路徑
請參閱 第 1120 頁的『URI 格式』。

***HTTP* 版本**
HTTP 版本; 例如 `HTTP/1.1`

一般標頭
請參閱 HTTP/1.1 - 4.5 一般標頭欄位。

回應標頭
請參閱 HTTP/1.1 - 6.2 回應標頭欄位。

實體標頭 *(回應)*
請參閱 HTTP/1.1 - 7.1 實體標頭欄位。 「回應」語法圖中列出的其中一個實體或回應標頭。 `Content-Length` 一律呈現在回應中。 如果沒有訊息內文，則設為零。

message
訊息內文。

說明

如果 HTTP **GET** 要求成功，回應訊息會包含從 WebSphere MQ 佇列擷取的資料。 HTTP `Content-Length` 標頭中會傳回訊息內文中的位元組數。 HTTP 回應的狀態碼設為 `200 OK`。 如果 `x-msg-range` 指定為 `0` 或 `0-0`，則 HTTP 回應的狀態碼是 `204 No Content`。

如果 HTTP **GET** 要求不成功，則回應包括 HTTP 錯誤訊息的 WebSphere MQ 橋接器及 HTTP 狀態碼。

HTTP GET 範例

HTTP **GET** 從佇列中取得訊息。 訊息會保留在佇列上。 在 WebSphere MQ 術語中， HTTP **GET** 是瀏覽要求。 您可以使用 Java 用戶端、瀏覽器表單或 AJAX 工具箱來建立 HTTP **GET** 要求。

第 1093 頁的圖 39 是一個 HTTP 要求，用來瀏覽佇列上稱為 `myQueue` 的下一個訊息。

此要求包含 HTTP 要求標頭 `x-msg-wait`，它指示用於 HTTP 的 WebSphere MQ 橋接器要等待多久，訊息才會抵達佇列。 此要求還包含 `x-msg-require-headers` 要求標頭，其指定用戶端將在回應中收到訊息相關性 ID。

```
GET /msg/queue/myQueue/ HTTP/1.1
Host: www.example.org
x-msg-wait: 10
x-msg-require-headers: correlID
```

圖 *39: HTTP **GET*** 要求範例

第 1093 頁的圖 40 是傳回給用戶端的回應。 相關性 ID 會傳回給用戶端，如該要求的 `x-msg-require-headers` 中所要求。

```
HTTP/1.1 200 OK
Date: Wed, 2 Jan 2007 22:38:34 GMT
Server: Apache-Coyote/1.1 WMQ-HTTP/1.1 JEE-Bridge/1.1
Content-Length: 50
Content-Type: text/plain; charset=utf-8
x-msg-correlId: 1234567890

Here is my message body that appears on the queue.
```

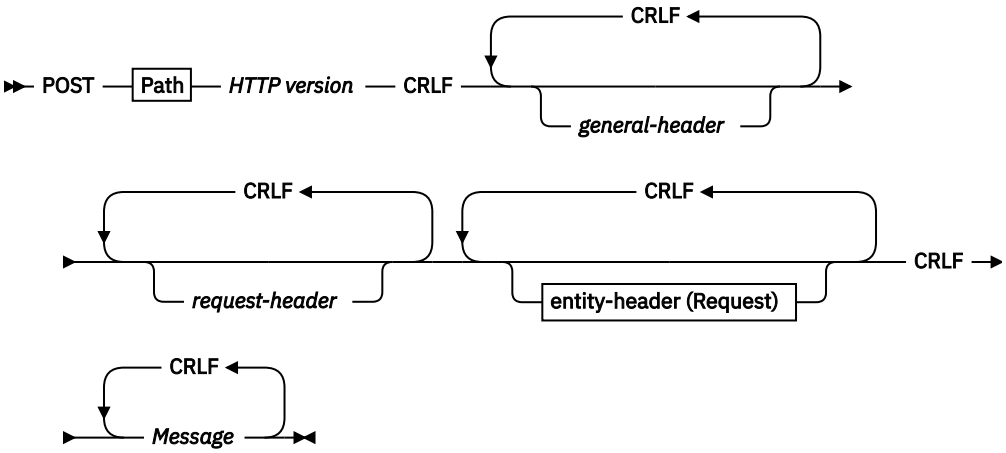
圖 *40: HTTP **GET*** 回應範例

HTTP POST: WebSphere MQ for HTTP 指令橋接器

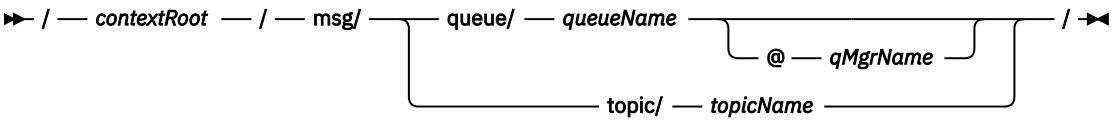
HTTP **POST** 作業會將訊息放置在 WebSphere MQ 佇列上，或將訊息發佈至主題。

Syntax

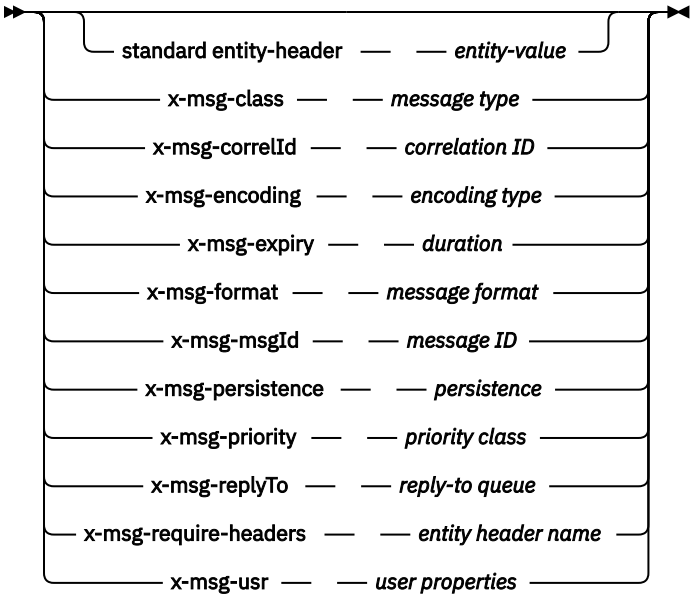
Request



Path



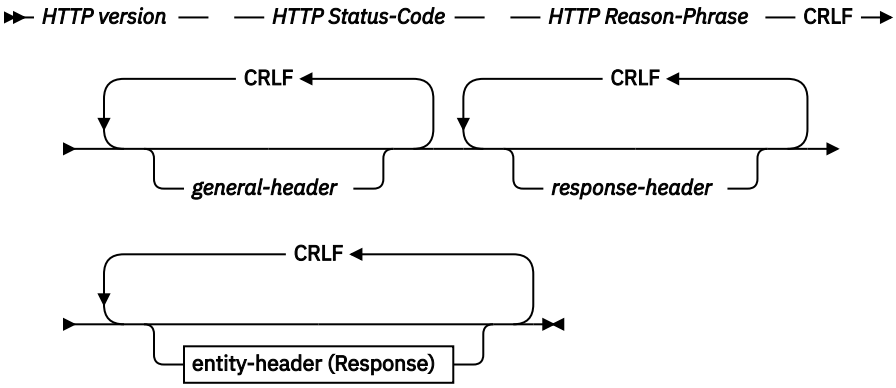
entity-header (Request)



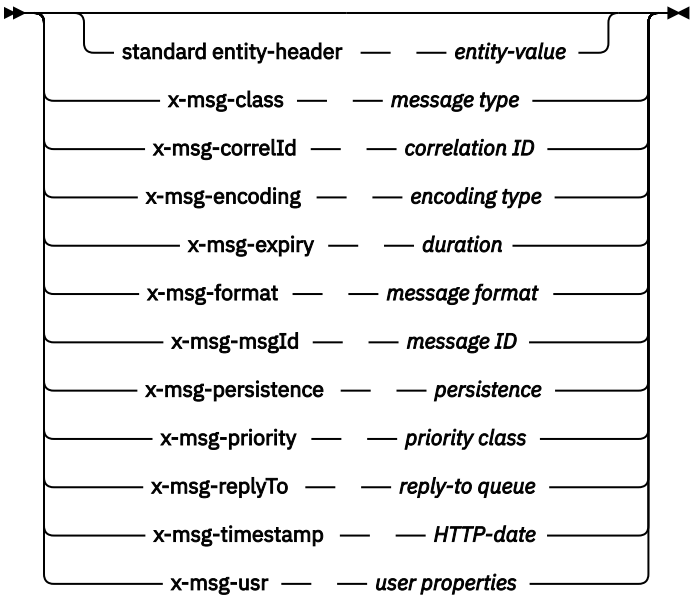
註:

1. If a question mark (?) is used it must be substituted with `%3f`. For example, `orange?topic` should be specified as `orange%3ftopic`.
2. @*qMgrName* is only valid on an HTTP **POST**

Response



entity-header (Response)



要求參數

路徑
請參閱 第 1120 頁的『URI 格式』。

***HTTP* 版本**
HTTP 版本; 例如 `HTTP/1.1`

一般標頭
請參閱 HTTP/1.1 - 4.5 一般標頭欄位。

要求標頭
請參閱 HTTP/1.1 - 5.3 要求標頭欄位。 在 HTTP/1.1 要求上， 主機 欄位是必要的。 通常由您用來建立用戶端要求的工具自動插入。

實體標頭 *(要求)*
請參閱 HTTP/1.1 - 7.1 實體標頭欄位。 要求語法圖中列出的其中一個實體標頭。 `Content-Length` 和 `Content-Type` 應該插入要求中，且通常由您用來建立用戶端要求的工具自動插入。 `Content-Type` 必須符合 `x-msg-class` 自訂 entity-header (如果有指定的話) 中定義的類型。

message
要放入佇列的訊息，或要公佈至主題的發佈。

回應參數

路徑
請參閱 第 1120 頁的『URI 格式』。

***HTTP* 版本**
HTTP 版本; 例如 `HTTP/1.1`

一般標頭
請參閱 HTTP/1.1 - 4.5 一般標頭欄位。

回應標頭
請參閱 HTTP/1.1 - 6.2 回應標頭欄位。

實體標頭 (回應)
請參閱 HTTP/1.1 - 7.1 實體標頭欄位。 「回應」語法圖中列出的其中一個實體或回應標頭。 `Content-Length` 一律呈現在回應中。 如果沒有訊息內文，則設為零。

說明

如果未包含 `x-msg-usr` 標頭，且訊息類別是 BYTES 或 TEXT，則放置在佇列上的訊息沒有 MQRFH2。

在 HTTP **POST** 要求中使用 HTTP 實體及要求標頭，來設定放置在佇列上的訊息內容。 您也可以使用 `x-msg-require-headers` 來要求在回應訊息中傳回哪些標頭。

如果 HTTP **POST** 要求成功，則回應訊息的實體是空的，且其 Content-Length 為零。 HTTP 狀態碼為 200 OK。

如果 HTTP **POST** 要求不成功，則回應包括 HTTP 錯誤訊息的 WebSphere MQ 橋接器及 HTTP 狀態碼。 WebSphere MQ 訊息不會放置在佇列或主題上。

HTTP POST 範例

HTTP **POST** 會將訊息放置到佇列，或者將發佈放置到主題。 **HTTPPOST** Java 範例是訊息對佇列的 HTTP **POST** 要求範例。 您可以改用瀏覽器表單或 AJAX 工具箱來建立 HTTP **POST** 要求，而不使用 Java。

第 1096 頁的圖 41 顯示將訊息放置在稱為 myQueue 的佇列上的 HTTP 要求。 此要求包含 HTTP 標頭 `x-msg-correlId` ，用來設定 WebSphere MQ 訊息的相關性 ID。

```
POST /msg/queue/myQueue/ HTTP/1.1
Host: www.example.org
Content-Type: text/plain
x-msg-correlID: 1234567890
Content-Length: 50

Here is my message body that is posted on the queue.
```

圖 *41:* 佇列的 *HTTP **POST*** 要求範例

第 1096 頁的圖 42 顯示傳回給用戶端的回應。 沒有任何回應內容。

```
HTTP/1.1 200 OK
Date: Wed, 2 Jan 2007 22:38:34 GMT
Server: Apache-Coyote/1.1 WMQ-HTTP/1.1 JEE-Bridge/1.1
Content-Length: 0
```

圖 *42: HTTP POST* 回應範例

HTTP 標頭

適用於 HTTP 的 WebSphere MQ 橋接器支援自訂要求 HTTP 標頭、自訂實體 HTTP 標頭，以及標準 HTTP 標頭的子集。

HTTP 實務是以 x- 作為所有自訂標頭的字首， HTTP 標頭的 WebSphere MQ Bridge 以 x-msg- 作為字首。 例如，若要設定優先順序標頭，請使用 x-msg-priority。

註:

- 大部分標頭值都區分大小寫。 例如，當使用 msgId 標頭時， NONE 是關鍵字，而 none 是 msgID。
- 會忽略拼錯的標頭。

自訂實體 HTTP 標頭

自訂實體 HTTP 標頭包含 WebSphere MQ 訊息的相關資訊。 使用實體標頭，您可以在訊息描述子 (MQMD) 中設定值，或在 MQMD 中設定查詢值。 其他實體標頭 x-msg-usr 會設定並傳回您要與要求相關聯的任何使用者內容資訊。

您可以在不同的 HTTP 要求環境定義中使用實體標頭:

DELETE
: 您只能搭配 **DELETE** HTTP 要求使用 x-msg-correlId 及/或 x-msg-msgId 實體標頭。 標頭的效果是依 MQGET 中的 MsgId 和 CorrelId 來選取特定訊息，並從其佇列中刪除該訊息。

GET
: 您只能搭配 **GET** HTTP 要求使用 x-msg-correlId 及/或 x-msg-msgId 實體標頭。 標頭的效果是依 MQGET 中的 MsgId 和 CorrelId 來選取特定訊息以進行瀏覽。

POST
: 您可以在 **POST** HTTP 要求中使用任何實體標頭，但 x-msg-timestamp 除外。

x-msg-require-headers
: 在任何 **GET**、 **POST** 或 **DELETE** HTTP 要求上，您可以在 x-msg-require-headers 要求標頭內新增多個實體標頭，以逗點區隔。 效果是在 HTTP 回應訊息中傳回指定的實體標頭，其中包含相關聯訊息內容的值。

每一個標頭的說明列出 WebSphere MQ HTTP 橋接器處理標頭的環境定義。 例如，在標頭 **POST**、 x-msg-require-headers 中，標頭是由 HTTP **POST** 要求中的 HTTP WebSphere MQ 橋接器所處理，或在 HTTP **POST**、 **GET** 或 **DELETE** 要求中的 x-msg-require-headers 要求標頭。 如果標頭包含在環境定義中，則在中不接受該標頭，則會忽略該標頭。 未報告任何錯誤。

您可以將任何標準 HTTP 標頭放入要由 Web 伺服器或其他要求處理程式處理的要求中。 同樣地，回應可能包含 Web 伺服器或其他回應處理程式所插入的其他標準 HTTP 標頭。

自訂要求 HTTP 標頭

三個自訂要求 HTTP 標頭 (x-msg-range、 x-msg-require-headers 及 x-msg-wait) 會將 HTTP 要求的相關資訊傳遞給伺服器。 它們充當要求修飾元。 使用 x-msg-range，您可以限制在回應中傳回的訊息資料量。 使用 x-msg-require-headers，您可以要求回應包含要求結果的相關資訊。 使用 x-msg-wait，您可以修改用戶端等待 HTTP 回應的時間。

標準 HTTP 標頭

主機 標準 HTTP 要求-標頭必須在 HTTP/1.1 要求中指定。

Content-Length 和 Content-Type 標準 HTTP 實體標頭可以在要求中指定。

可以傳回 Content-Length、 Content-Location、 Content-Range、 Content-Type 及 Server 標準 HTTP 實體標頭，以回應要求。 在要求訊息的 x-msg-request-header 標頭中指定一或多個標準 HTTP 標頭。

HTTP 標頭的英文字母清單

class:HTTP x-msg-class 實體標頭

設定或傳回訊息類型。

類型	敘述
HTTP 標頭名稱	x-msg-class
HTTP 標頭類型	實體標頭
在 HTTP 要求訊息中有效	POST， x-msg-require-headers
容許值	**BYTES** **MAP** **STREAM** **TEXT**
預設值	BYTES

說明

- 在 HTTP **POST** 要求中，設定所建立訊息的類型。
- 在 **GET** 或 **DELETE** 上指定類別標頭會傳回 400 Bad Request，實體主體為 MQHTTP40007。
- 在 x-msg-require-headers 中指定，將 HTTP 回應訊息中的 x-msg-class 設為訊息類型。
- 如果指定給此標頭的值無效，則會傳回 MQHTTP40005 訊息。
- 如果未指定 x-msg-class 標頭，且訊息的內容類型是 application/x-www-form-urlencoded，則會假設資料是 JMS 對映物件。

Content-Length:HTTP 實體標頭

設定或傳回訊息內文的長度 (以位元組為單位)。

類型	敘述
HTTP 標頭名稱	**內容長度**
HTTP 標頭類型	實體標頭
在 HTTP 要求訊息中有效	x-msg-require-headers
容許及回覆值	***Integer value*** 訊息內文的長度 (以位元組為單位)。

說明

- 在 HTTP 要求中， Content-Length 是選用項目。 對於 **GET** 或 **DELETE**，長度必須為零。 對於 **POST**，如果指定 Content-Length，且它不符合訊息行的長度，則會截斷訊息，或以空值填補指定的長度。
- 即使沒有內容，也一律在 HTTP 回應中傳回 Content-Length，在此情況下，此值為零。

Content-Location:HTTP 實體標頭

在 HTTP 回應訊息的標準 Content-Location 標頭中，傳回要求中所參照的佇列或主題。

類型	敘述
HTTP 標頭名稱	**內容-位置**
HTTP 標頭類型	實體標頭
在 HTTP 要求訊息中有效	x-msg-require-headers
回覆值	格式為的 URI，

類型	敘述
	`/msg/queue/queuename` or *`/msg/topic/topicname`*

說明

- 在 x-msg-require-headers 中要求時， Content-Location 實體標頭會傳回 HTTP 要求中所參照的佇列或主題。

Content-Range:HTTP 實體標頭

在 HTTP 回應的 Content-Range 標頭中，傳回從 WebSphere MQ 訊息選取的位元組範圍。

類型	敘述
HTTP 標頭名稱	**內容-範圍**
HTTP 標頭類型	實體標頭
在 HTTP 要求訊息中有效	x-msg-require-headers
回覆值	***String*** 傳回所傳回子字串的下限 *m* 及上限 *n* ， 以及整個訊息的 *length* 。 例如: *`m-n/length`*

說明

-
- 只有在包含 x-msg-range 要求標頭的 **GET** 或 **DELETE** 要求中指定 Content-Range 時，才會在 HTTP 回應中傳回 Content-Range 。
- 如果在 **GET** 或 **DELETE** 要求上指定 x-msg-range ，則會在回應中傳回 Content-Range 標頭中指定的位元組範圍。 例如，如果在包含 100 位元組的訊息要求中使用 x-msg-range: 0-60 ，則 content-range 標頭會保留字串 0-60/100
- x-msg-range 要求也會在 HTTP 回應的 x-msg-range 標頭中傳回內容範圍。

Content-Type:HTTP 實體標頭

根據 HTTP 內容類型，在 WebSphere MQ 訊息中設定或傳回 JMS 訊息的類別。

類型	敘述
HTTP 標頭名稱	**內容類型**
HTTP 標頭類型	實體標頭
在 HTTP 要求訊息中有效	**POST**， x-msg-require-headers
容許或傳回的值	***media-type*** 如需支援的媒體類型，請參閱 第 1100 頁的表 601。

表 *601: x-msg-class* 與 *HTTP 內容類型* 之間的對映	
x-msg-class	**HTTP 內容類型**
BYTES	application/octet-stream application/xml
TEXT	text/*
MAP	application/x-www-form-urlencoded application/xml （選用）
STREAM	application/xml（選用）

說明

- 在 HTTP **POST** 要求上，指定 Content-Type 或 x-msg-class。 如果您同時指定兩者，則它們必須一致或傳回 HTTP Bad Request 異常狀況 Status code 400 。 如果您同時省略 Content-Type 和 x-msg-class，則會假設 Content-Type 是 text/* 。
- 在回應中一律會設定 **內容類型** ， 以回應具有訊息內文的 HTTP **GET** 或 **DELETE** 。 **內容類型** 是根據 第 1100 頁的表 602 中的規則來設定。

表 *602:* 將訊息類型對映至 *x-msg-class* 和 *內容類型*			
訊息格式	**JMS 訊息類型**	**x-msg-class**	**內容類型**
MQFMT_STRING 以外的任何項目	無	BYTES	application/octet-stream
MQFMT_STRING	無	TEXT	text/plain
MQFMT_NONE	jms_bytes	BYTES	application/octet-stream
MQFMT_NONE	jms_text	TEXT	text/plain
MQFMT_NONE	jms_map	MAP	application/xml
MQFMT_NONE	jms_stream	STREAM	application/xml

correlId:HTTP x-msg-correlId entity-header

設定或傳回相關性 ID。

類型	**敘述**
HTTP 標頭名稱	x-msg-correlId
HTTP 標頭類型	實體標頭
在 HTTP 要求訊息中有效	**DELETE**、 **GET**、 **POST**、 x-msg-require-headers
容許值	***String value*** 例如: `x-msg-correlId: mycorrelationid` 允許以引號括住的字串; 例如:

類型	敘述

	`x-msg-correlId: "my id"`
	Hex value
	字首為 `0x:`的十六進位值; 例如:
	`x-msg-correlId: 0x:43c1d23a`
	下列 0x: 的十六進位值限制為 48 個字元，代表 24 個位元組。 系統不處理其他資料。

預設值	不適用

說明

- 在 HTTP **POST** 要求上，設定所建立訊息的相關性 ID。
- 在 HTTP **GET** 或 **DELETE** 要求上，從佇列或主題中選取訊息。 如果不存在具有指定相關性 ID 的訊息，則會傳回 `HTTP 504 Gateway Timeout` 回應。 `x-msg-correlId` 可以與 `x-msg-msgID` 一起使用，以從符合兩個選取器的佇列或主題中選取訊息。
- 在 `x-msg-require-headers` 中指定，將 HTTP 回應訊息中的 `x-msg-corelId` 設為訊息的相關性 ID。
- 在 `0x:` 字首之後容許水平空格。

註:

- 在 HTTP **GET** 或 **DELETE** 要求上指定不含值的 `x-msg-correlId` ; 例如， `"x-msg-correlId:"`會傳回佇列或主題上的下一則訊息，不論其相關性 ID 為何。
- 如果您指定 24 個字元或更少的選取元，或 0x: 後接 48 個字元或更少的選取元， WebSphere MQ for HTTP 橋接器會使用最佳化選取元，以增進效能。
- 從佇列中選取訊息時，會使用包含 JMSCorrelationID 的 JMS 訊息選取器。 此選取器的行為如 選取行為中所述。

encoding:HTTP `x-msg-encoding` entity-header

設定或傳回訊息編碼。

類型	敘述
HTTP 標頭名稱	`x-msg` **編碼**
HTTP 標頭類型	實體標頭
在 HTTP 要求訊息中有效	**POST**， `x-msg-require-headers`
容許值	下列值的逗點區隔清單: **DECIMAL_NORMAL** **DECIMAL_REVERSED** **FLOAT_IEEE_NORMAL** **FLOAT_IEEE_REVERSED** **FLOAT_S390** **INTEGER_NORMAL** **INTEGER_REVERSED** 例如: `x-msg-encoding: INTEGER_NORMAL,DECIMAL_NORMAL,FLOAT_IEEE_NORMAL`

類型	敘述
	註: 值區分大小寫
預設值	DECIMAL_NORMAL, FLOAT_IEEE_NORMAL, INTEGER_NORMAL

說明

- 在 HTTP **POST** 要求上，指定所建立訊息的編碼。
- 在 HTTP **GET** 或 **DELETE** 要求上，會忽略 x-msg-encoding 標頭。
- 在 x-msg-require-headers 中指定，將 HTTP 回應訊息中的 x-msg-encoding 設為訊息的 encoding 內容。

expiry:HTTP x-msg-expiry 實體標頭

設定或傳回訊息到期持續時間。

類型	敘述
HTTP 標頭名稱	x-msg-expiry
HTTP 標頭類型	實體標頭
在 HTTP 要求訊息中有效	**POST**, x-msg-require-headers

容許值	**UNLIMITED**
	例如:
	`x-msg-expiry: UNLIMITED`
	Integer value
	到期之前的毫秒數。
	例如:
	`x-msg-expiry: 10000`

預設值	UNLIMITED

說明

- 在 HTTP **POST** 要求上設定時，要求訊息會在指定的時間後到期。
- 在 HTTP **GET** 或 **DELETE** 要求上，會忽略 x-msg-expiry 標頭。
- 在 x-msg-require-headers 中指定，將 HTTP 回應訊息中的 x-msg-expiry 設為訊息的到期時間。
- UNLIMITED 指定訊息永不到期。
- 訊息從訊息到達佇列的時間開始到期，因此會忽略網路延遲。
- 上限值受 WebSphere MQ 限制為 214748364700 毫秒。 如果指定的值大於此值，則會假設可能的到期時間上限。

format:HTTP x-msg-format entity-header

設定或傳回 WebSphere MQ 訊息格式。

類型	敘述
HTTP 標頭名稱	x-msg-format
HTTP 標頭類型	實體標頭

類型	敘述
在 HTTP 要求訊息中有效	**POST**， x-msg-require-headers

容許值	**NONE**
	例如：
	`x-msg-format: NONE`
	String value
	任何使用者定義的值，最多 8 個字元。 例如：
	`x-msg-format: myformat`

預設值	None

說明

- 在 HTTP **POST** 要求上設定時，請設定要求訊息格式。
- 在 HTTP **GET** 或 **DELETE** 要求上，會忽略 x-msg-format 標頭。
- 在 x-msg-require-headers 中指定，將 HTTP 回應訊息中的 x-msg-format 設為訊息的格式。
- NONE 區分大小寫，並指出訊息格式為空白。
- 即使 x-msg-format 的值與 HTTP 要求的媒體類型相矛盾，也會使用它。 請參閱 第 1103 頁的表 603。

表 *603:* 將內容類型和 *x-msg-class* 對映至訊息格式

x-msg-class	**內容類型**	**佇列/主題上的訊息格式**
BYTES	• application/octet-stream • application/xml	WebSphere MQ 訊息: MQFMT 設為 MQC.MQFMT_NONE
TEXT	• text/*	WebSphere MQ 訊息: MQFMT 設為 MQC.MQFMT_STRING
MAP	• application/x-www-form-urlencoded • application/xml（選用）	JMSMap
STREAM	• application/xml （選用）	JMSStream

msgId:HTTP x-msg-msgId entity-header

設定或傳回訊息 ID。

類型	敘述
HTTP 標頭名稱	x-msg-msgId
HTTP 標頭類型	實體標頭
在 HTTP 要求訊息中有效	**DELETE**、 **GET**、 **POST**、 x-msg-require-headers
容許值	***String value*** 例如： `x-msg-msgId: mymsgid`

類型	敘述
	以引號括住的字串，例如 x-msg-msgId: "my id" ***Hex value*** 字首為 0x:的十六進位值; 例如, `x-msg-msgId: 0x:43c1d23a`
預設值	不適用

說明

- 在 HTTP **POST** 要求上，設定所建立訊息的訊息 ID。
- 在 HTTP **GET** 或 **DELETE** 要求上，從佇列或主題中選取訊息。 如果不存在具有指定訊息 ID 的訊息，則會傳回 HTTP 504 Gateway Timeout 回應。 x-msg-msgId 可以與 x-msg-correlID 搭配使用，以從符合兩個選取器的佇列或主題中選取訊息。
- 在 x-msg-require-headers 中指定，會在對訊息訊息 ID 的 HTTP 回應中傳回 x-msg-msgId 。
- 在 0x:字首之後容許水平空格。

註: 在 HTTP **GET** 或 **DELETE** 要求上指定不含值的 x-msg-msgId ; 例如， "x-msg-msgId:"會在佇列或主題上傳回下一則訊息，而不論其訊息 ID 為何。

從佇列中選取訊息時，會使用包含 JMSMessageID 的 JMS 訊息選取器。 此選取器的行為如 選取行為 中所述。

持續性:HTTP x-msg-persistence 實體標頭

設定或傳回訊息持續性。

類型	敘述
HTTP 標頭名稱	x-msg-persistence
HTTP 標頭類型	實體標頭
在 HTTP 要求訊息中有效	**POST**， x-msg-require-headers

容許值	**NON_PERSISTENT**
	此訊息無法在系統失敗或佇列管理程式重新啟動時存活。
	例如:
	`x-msg-persistence: NON_PERSISTENT`
	PERSISTENT
	訊息在系統失敗及佇列管理程式重新啟動之後仍然存在。
	例如:
	`x-msg-persistence: PERSISTENT`
	AS_DESTINATION
	僅適用於 **POST** 。
	使用訊息提供者所決定之目的地的預設持續性。
	註: 區分大小寫

預設值	NON_PERSISTENT

說明

- 在 HTTP **POST** 要求上設定時，請設定要求訊息持續性。
- 在 HTTP **GET** 或 **DELETE** 要求上，會忽略 x-msg-persistence 標頭。
- 在 x-msg-require-headers 中指定，將 HTTP 回應訊息中的 x-msg-persistence 設為訊息的持續性。

priority:HTTP x-msg-priority 實體標頭

設定或傳回訊息優先順序。

類型	敘述
HTTP 標頭名稱	x-msg-priority
HTTP 標頭類型	實體標頭
在 HTTP 要求訊息中有效	**POST**， x-msg-require-headers

容許值	**LOW**
	例如：
	`x-msg-priority: LOW`
	MEDIUM
	此優先順序等於 WebSphere MQ 優先順序層次 4。 例如：
	`x-msg-priority: MEDIUM`
	HIGH
	例如：
	`x-msg-priority: HIGH`
	Integer value
	0 到 9 範圍內整數的字串表示法; 例如,
	`x-msg-priority: 3`
	AS_DESTINATION
	僅適用於 **POST** 。
	使用訊息提供者所決定之目的地的預設優先順序。
	註: 區分大小寫

預設值	MEDIUM

說明

- 在 HTTP **POST** 要求上設定時，請設定要求訊息優先順序。
- 在 HTTP **GET** 或 **DELETE** 要求上，會忽略 x-msg-priority 標頭。
- 在 x-msg-require-headers 中指定，將 HTTP 回應訊息中的 x-msg-priority 設為訊息的優先順序。

priority:HTTP x-msg-priority 實體標頭

設定或傳回訊息優先順序。

類型	敘述
HTTP 標頭名稱	x-msg-priority

類型	敘述
HTTP 標頭類型	實體標頭
在 HTTP 要求訊息中有效	**POST**， `x-msg-require-headers`

容許值	**LOW**
	例如：
	`x-msg-priority: LOW`
	MEDIUM
	此優先順序等於 WebSphere MQ 優先順序層次 4。 例如：
	`x-msg-priority: MEDIUM`
	HIGH
	例如：
	`x-msg-priority: HIGH`
	Integer value
	0 到 9 範圍內整數的字串表示法; 例如，
	`x-msg-priority: 3`
	AS_DESTINATION
	僅適用於 **POST** 。
	使用訊息提供者所決定之目的地的預設優先順序。
	註: 區分大小寫

預設值	MEDIUM

說明

- 在 HTTP **POST** 要求上設定時，請設定要求訊息優先順序。
- 在 HTTP **GET** 或 **DELETE** 要求上，會忽略 `x-msg-priority` 標頭。
- 在 `x-msg-require-headers` 中指定，將 HTTP 回應訊息中的 `x-msg-priority` 設為訊息的優先順序。

replyTo:HTTP x-msg-replyTo entity-header

設定或傳回訊息回覆目的地佇列及佇列管理程式名稱。

類型	敘述
HTTP 標頭名稱	`x-msg-replyTo`
HTTP 標頭類型	實體標頭
在 HTTP 要求訊息中有效	**POST**， `x-msg-require-headers`

容許值	***URI***
	點對點 URI; 例如，
	`x-msg-replyTo: /msg/queue/myReplyQueue`
	`x-msg-replyTo: /msg/queue/myReplyQueue@myReplyQueueManager`
	註: 區分大小寫

預設值	MEDIUM

說明

- 在 HTTP **POST** 要求上設定時，請設定要求訊息 replyTo 目的地。
- 在 HTTP **GET** 或 **DELETE** 要求上，會忽略 `x-msg-replyTo` 標頭。
- 在 `x-msg-require-headers` 中指定，將 HTTP 回應訊息中的 `x-msg-replyTo` 設為 訊息的回覆目的地佇列及佇列管理程式名稱。

註: HTTP 回應中的 URI 可以包括 WebSphere 適用於 HTTP 的 MQ 橋接器所連接的佇列管理程式名稱。

伺服器:HTTP 回應標頭

傳回用戶端所連接伺服器及通訊協定的相關資訊。

類型	敘述
HTTP 標頭名稱	SERVER
HTTP 標頭類型	回應標頭
在 HTTP 要求訊息中有效	`x-msg-require-headers`

回覆值	`WMQ-HTTP/1.1 JEE-Bridge/1.1`
	or
	`Server: Product-token WMQ-HTTP/1.1 JEE-Bridge/1.1`

說明

- 如果 WebSphere MQ Bridge for HTTP 部署至應用程式伺服器，則 HTTP 詳細資料的 WebSphere MQ 橋接器會附加至伺服器回應標頭。 例如，適用於 HTTP 部署至 WebSphere Application Server Community Edition(稱為 Apache-Coyote) 的 WebSphere MQ 橋接器會提供回應:

```
Server: Apache-Coyote/1.1 WMQ-HTTP/1.1 JEE-Bridge/1.1
```

require-headers:HTTP x-msg-require-headers request-header

設定 HTTP 回應訊息中要傳回的標頭。

類型	敘述
HTTP 標頭名稱	`x-msg-require-headers`
HTTP 標頭類型	要求標頭
在 HTTP 要求訊息中有效	**POST**, **GET**, **DELETE**
容許值	以逗點區隔的實體標頭名稱清單: **ALL** **ALL-USR** **class** **content-location** **correlId** **encoding** **expiry** **format** **msgId**

類型	敘述

	NO_require-headers
	persistence
	priority
	replyTo
	server
	timestamp
	usr-*property name*
	例如:
	`x-msg-require-headers: msgId`
	或
	`x-msg-require-headers: expiry,correlId,timestamp`
	如果要要求特定的內容，請執行下列動作:
	`x-msg-require-headers: usr-myCustomProperty`
	如果要要求所有內容，請執行下列動作:
	`x-msg-require-headers: ALL-USR, ALL`

預設值	NO_require-headers

說明

- x-msg-require-headers 的值不區分大小寫，除非是 ALL、 NO_require-headers 和 ALL-USR 常數以及 *property-name* 變數。

timestamp:HTTP x-msg-timestamp 實體標頭

傳回訊息時間戳記。

類型	敘述
HTTP 標頭名稱	x-msg-timestamp
HTTP 標頭類型	實體標頭
在 HTTP 要求訊息中有效	x-msg-require-headers
回覆值	***HTTP-date*** 採用下列格式的日期: 日、日期月、年、時間-區域; 例如: `Sun, 06 Nov 1994 08:49:37 GMT` 由 RFC 822 定義，並在 RFC 1123 中更新。
預設值	不適用

說明

- 在 HTTP **POST**、 **GET** 或 **DELETE** 要求上，會忽略 x-msg-timestamp 標頭。
- 在 x-msg-require-headers 中指定，將 HTTP 回應訊息中的 x-msg-timestamp 設為訊息的時間戳記。

usr:HTTP x-msg-usr 實體標頭

設定或傳回使用者內容。

類型	敘述
HTTP 標頭名稱	x-msg-usr
HTTP 標頭類型	實體標頭
在 HTTP 要求訊息中有效	**POST**， x-msg-require-headers

容許值	請參閱 第 1109 頁的『語法』; 例如:
	`x-msg-usr: myProp1;5;i1,`
	`x-msg-usr: myProp2;"My String";string`

預設值	不適用

說明

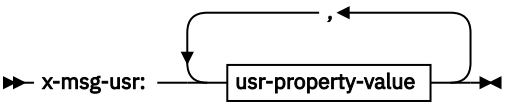
- 在 HTTP **POST** 要求上設定時，請設定要求訊息使用者內容。
- 在 HTTP **GET** 或 **DELETE** 要求上，會忽略 x-msg-usr 標頭。
- 在 x-msg-require-headers 中指定，將 HTTP 回應訊息中的 x-msg-usr 設為訊息的使用者內容。
- 在訊息上可以設定多個內容。 在單一 x-msg-usr 標頭中指定多個以逗點區隔的內容，或使用 x-msg-usr 標頭的兩個以上個別實例。
- 您可以要求在 **GET** 或 **DELETE** 要求的回應中傳回特定內容。 在要求的 x-msg-require-headers 標頭中指定內容的名稱，並使用字首 usr-。 例如:

```
x-msg-require-headers: usr-myProp1
```

- 若要要求在回應中傳回所有使用者內容，請使用 ALL-USR 常數。 例如:

```
x-msg-require-headers: ALL-USR
```

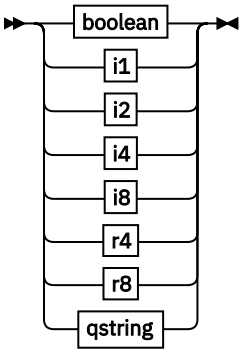
語法



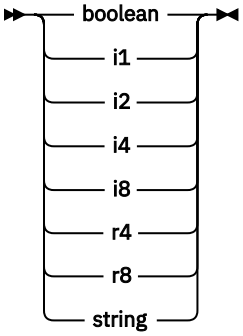
usr-property-value

property-name

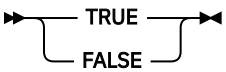
usr-value



usr-type



boolean

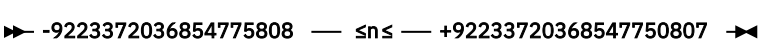


i1

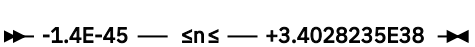
i2

i4

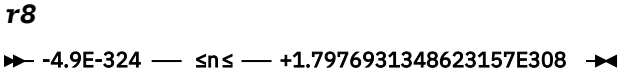
i8



r4



r8



qstring

wait:HTTP x-msg-wait request-header

設定在傳回 HTTP 504 Gateway Timeout 回應訊息之前等待訊息到達的時段。

類型	敘述
HTTP 標頭名稱	x-msg-wait

類型	敘述
HTTP 標頭類型	要求標頭
在 HTTP 要求訊息中有效	**GET**, **DELETE**

容許值	**NO_WAIT**
	例如:
	`x-msg-wait: NO_WAIT`
	Integer value
	HTTP 的 WebSphere MQ 橋接器等待訊息到達的毫秒數; 例如,
	`x-msg-wait: 8`

預設值	NO_WAIT

說明

- 在 HTTP **POST** 要求上，會忽略 `x-msg-wait` 標頭。
- 在 HTTP **GET** 或 **DELETE** 要求上， `x-msg-wait` 指定在傳回 `HTTP 504 Gateway Timeout` 回應之前等待訊息到達的時間。
- `NO_WAIT` 區分大小寫。
- 預設等待時間上限為 35000。 您可以設定 Servlet 的 `maximum_wait_time` 參數來變更預設值。 如需相關資訊，請參閱 安裝、配置及驗證 WebSphere MQ for HTTP 橋接器 一節。
- 如果您設定大於 `maximum_wait_time` 的值，則會改用 `maximum_wait_time` 。

HTTP 回覆碼

來自 HTTP 的 WebSphere MQ 橋接器的回覆碼清單

適用於 HTTP 的 WebSphere MQ 橋接器會傳回四種類型的錯誤:

Servlet 錯誤
MQHTTP0001 及 MQHTTP0002 是 Servlet 錯誤。 會記載它們，但不會傳回至 HTTP 用戶端。

成功作業
範圍 200-299 中的 HTTP 狀態碼指出作業成功。

用戶端錯誤數
400-499 範圍內的 HTTP 狀態碼指出用戶端錯誤。 WebSphere MQ Bridge for HTTP 回覆碼，範圍 MQHTTP40001 - MQHTTP49999 對應於用戶端錯誤。

伺服器錯誤數

500-599 範圍內的 HTTP 狀態碼指出用戶端錯誤。 WebSphere MQ Bridge for HTTP 回覆碼，範圍 MQHTTP50001 - MQHTTP59999 對應於伺服器錯誤。

如果發生伺服器錯誤，則會將完整堆疊追蹤輸出至應用程式伺服器錯誤日誌。 堆疊追蹤也會在 HTTP 回應中傳回給 HTTP 用戶端。 請處理用戶端應用程式中的堆疊追蹤，或將它交給應用程式伺服器管理者來解決問題。

如果堆疊追蹤包含資源配接器錯誤，請參閱資源配接器的說明文件。

回覆碼的英文字母清單

HTTP 200: 正常

此狀態碼類別指出已順利接收、瞭解及接受要求。

HTTP 狀態碼
200 OK

HTTP 204: 無內容

在要求中順利傳送 HTTP **GET** 或 **DELETE** 及 x-msg-range: 0 之後傳送。

HTTP 狀態碼
204 No Content

MQHTTP0001: Servlet 環境定義中未指定 Connection Factory

Servlet 錯誤

說明
Servlet 錯誤

HTTP 狀態碼
無

程式設計師回應
記載這些錯誤的位置是您應用程式伺服器特有的。 請參閱應用程式伺服器的說明文件。

MQHTTP0002: 無法使用 JNDI 名稱 *jndiNameTried* 取得 *queueOrTopic* 的連線管理程式

Servlet 錯誤

說明
Servlet 錯誤

HTTP 狀態碼
無

程式設計師回應
記載這些錯誤的位置是您應用程式伺服器特有的。 請參閱應用程式伺服器的說明文件。

MQHTTP40001: 保留

已保留

HTTP 狀態碼
400 Bad Request

MQHTTP40002: URI 對 HTTP 的 WebSphere MQ 傳輸無效

HTTP 要求中指定的 URI 無效。

說明
HTTP 要求中指定的 URI 無效。

HTTP 狀態碼
400 Bad Request

程式設計師回應
請確認指定 URI 的格式和語法正確。

MQHTTP40003: URI 無效。 @qmgr 僅在 POST 上有效

已在非 **POST** 要求的 HTTP 要求的 URI 中指定 `@qmgr` URI 選項。

說明
已在非 **POST** 要求的 HTTP 要求的 URI 中指定 `@qmgr` URI 選項。

HTTP 狀態碼
`400 Bad Request`

程式設計師回應
如果您嘗試使用 **POST** 動詞來放置訊息，請將 HTTP 要求變更為 **POST** 要求。 如果您嘗試使用 **DELETE** 或 **GET** 動詞來取得訊息，請從 URI 中移除 `@qmgr` 。

MQHTTP40004: 指定的 內容類型 無效

在 **POST** 要求上指定的 `Content-Type` 標頭欄位與 `x-msg-class` 標頭值不相容。

說明
在 **POST** 要求上指定的 `Content-Type` 標頭欄位與 `x-msg-class` 標頭值不相容。

HTTP 狀態碼
`400 Bad Request`

程式設計師回應
將 `Content-Type` 標頭欄位變更為受支援的標頭欄位。 `Content-Type` 標頭必須與指定的 `x-msg-class` 標頭欄位相容。

MQHTTP40005: 訊息標頭值不正確

指定的受支援標頭欄位具有對指定要求無效的值。

說明
指定的受支援標頭欄位具有對指定要求無效的值。

HTTP 狀態碼
`400 Bad Request`

程式設計師回應
請將指定給給定標頭欄位的值變更為有效的值。 請檢查指定值的大小寫，因為部分標頭欄位具有區分大小寫的值。

MQHTTP40006: *`Header_name`* 不是有效的要求標頭

HTTP 要求訊息中指定了只在 HTTP 回應訊息中有效的標頭。

說明
已在 HTTP 要求訊息中指定只在 HTTP 回應訊息中有效的標頭。

HTTP 狀態碼
400 Bad Request

程式設計師回應
從 HTTP 要求中移除僅在 HTTP 回應中有效的任何標頭; 例如， x-msg-timestamp。

MQHTTP40007: *Header_name* 僅適用於 ...

已在 HTTP 要求中指定標頭，但標頭欄位對給定的要求動詞無效。

說明
已在 HTTP 要求中指定標頭，但標頭欄位對給定的要求動詞無效。

HTTP 狀態碼
400 Bad Request

程式設計師回應
從 HTTP 要求中移除對給定要求動詞無效的任何標頭。 例如， x-msg-encoding 適用於 HTTP **POST** 要求，但不適用於 HTTP **GET** 或 HTTP **DELETE** 要求。

MQHTTP40008: *Header_name* 長度上限為 ...

已超出給定標頭欄位的長度上限。

說明
已超出給定標頭欄位的長度上限。

HTTP 狀態碼
400 Bad Request

程式設計師回應
將標頭欄位的值變更為標頭欄位所允許範圍內的值。

MQHTTP40009: 標頭欄位 *header_field* 不適用於 ...

HTTP 的 WebSphere MQ 橋接器所連接的傳訊提供者不支援 HTTP 要求中指定的標頭欄位。

說明
HTTP 的 WebSphere MQ 橋接器所連接的傳訊提供者不支援 HTTP 要求中指定的標頭欄位。 如果使用的傳訊提供者無法支援適用於 HTTP 的 WebSphere MQ 橋接器的所有特性，則會發生錯誤。

HTTP 狀態碼
400 Bad Request

程式設計師回應
請從 HTTP 要求中移除不受支援的標頭。

MQHTTP40010: 無法剖析具有 內容類型 *content_type* 的訊息

HTTP 要求的內容與要求的 Content-Type 不相容。

說明
HTTP 要求的內容與要求的 Content-Type 不相容。 常見原因是 application/x-www-form-urlencoded 或 application/xml data 格式不正確。

HTTP 狀態碼
400 Bad Request

程式設計師回應
請更正 HTTP 要求的內容，使其為要求 Content-Type 的正確格式。

MQHTTP40301: 禁止您存取 ...

HTTP 的 WebSphere MQ 橋接器無法鑑別指定目的地的本身。

說明
HTTP 的 WebSphere MQ 橋接器無法鑑別指定目的地的本身。

HTTP 狀態碼
403 Forbidden

程式設計師回應
變更目的地的鑑別內容，以便 WebSphere MQ Bridge for HTTP 獲得連接授權。 或者，指定 WebSphere MQ HTTP 橋接器獲授權連接的目的地。

MQHTTP40302: 禁止您 ...

HTTP 的 WebSphere MQ 橋接器無法連接至佇列管理程式。

說明
HTTP 的 WebSphere MQ 橋接器無法連接至佇列管理程式。 HTTP 安全配置的 WebSphere MQ 橋接器不正確。

HTTP 狀態碼
403 Forbidden

程式設計師回應
變更佇列管理程式的鑑別配置，讓 WebSphere MQ Bridge for HTTP 獲得連接授權。 或者，配置 HTTP 的 WebSphere MQ 橋接器，以連接至其獲授權連接的佇列管理程式。

MQHTTP40401: 找不到目的地 *destination_name*

HTTP 的 WebSphere MQ 橋接器無法找到 HTTP 要求 URI 中指定的目的地。

說明
HTTP 的 WebSphere MQ 橋接器無法找到 HTTP 要求 URI 中指定的目的地。

HTTP 狀態碼
404 Not found

程式設計師回應
請檢查 HTTP 要求 URI 中指定的目的地是否存在，或指定替代目的地。

MQHTTP40501: 不接受方法 *method_name*

HTTP 的 WebSphere MQ 橋接器不支援 HTTP 要求中指定的方法。

說明

HTTP 的 WebSphere MQ 橋接器不支援 HTTP 要求中指定的方法。

HTTP 狀態碼

405 Method not allowed

程式設計師回應

將 HTTP 要求中指定的方法變更為 HTTP WebSphere MQ 橋接器支援的方法。

MQHTTP41301: 所公佈的訊息對目的地而言太大

HTTP POST 要求 URI 中指定的目的地無法接受與 HTTP 要求中指定的訊息一樣長的訊息。

說明

HTTP POST 要求 URI 中指定的目的地無法接受與 HTTP 要求中指定的訊息一樣長的訊息。

HTTP 狀態碼

413 Request entity too large

程式設計師回應

請減少 HTTP 要求中指定的訊息大小。 或者，指定可支援所需長度之訊息的目的地。

MQHTTP41501: 不支援媒體類型字集

適用於 HTTP 的 WebSphere MQ 橋接器不支援 Content-Type 標頭欄位中指定的字集。

說明

適用於 HTTP 的 WebSphere MQ 橋接器不支援 Content-Type 標頭欄位中指定的字集。

HTTP 狀態碼

415 Unsupported media type

程式設計師回應

將 **內容類型** 標頭欄位的字集變更為適用於 HTTP 的 WebSphere MQ 橋接器所支援的字集。

MQHTTP41502: 媒體類型 *media-type* 不受支援 ...

HTTP 要求中指定的媒體類型不受指定 HTTP 動詞之 HTTP 的 WebSphere MQ 橋接器支援。

說明

HTTP 要求中指定的媒體類型不受指定 HTTP 動詞之 HTTP 的 WebSphere MQ 橋接器支援。

HTTP 狀態碼

415 Unsupported media type

程式設計師回應

將 HTTP 要求中指定的媒體類型變更為指定 HTTP 動詞的 WebSphere MQ Bridge for HTTP 所支援的媒體類型。

MQHTTP41503: 媒體類型 *media-type* 不受支援 ...

對於指定的 x-msg-class 標頭欄位，適用於 HTTP 的 WebSphere MQ 橋接器不支援 HTTP 要求中指定的媒體類型。

說明
對於指定的 x-msg-class 標頭欄位，適用於 HTTP 的 WebSphere MQ 橋接器不支援 HTTP 要求中指定的媒體類型。

HTTP 狀態碼
415 Unsupported media type

程式設計師回應
針對指定的 x-msg-class 標頭欄位，將 HTTP 要求中指定的媒體類型變更為 WebSphere MQ Bridge for HTTP 所支援的媒體類型。

MQHTTP41701: 不支援 HTTP 標頭 Expect

HTTP 的 WebSphere MQ 橋接器不支援 Expect 標頭欄位。

說明
已在 HTTP 要求中指定 Expect 標頭。 HTTP 的 WebSphere MQ 橋接器不支援 Expect 標頭欄位。

HTTP 狀態碼
417 Expectation failed

程式設計師回應
從 HTTP 要求中移除 Expect 標頭。

MQHTTP50001: 發生非預期的問題 ...

適用於 HTTP 的 WebSphere MQ 橋接器中發生錯誤。

說明
適用於 HTTP 的 WebSphere MQ 橋接器中發生錯誤。

HTTP 狀態碼
500 Internal server error

程式設計師回應
請聯絡 WebSphere MQ Bridge for HTTP 的系統管理者。

MQHTTP50201: HTTP 的 WebSphere MQ 橋接器與佇列管理程式之間發生錯誤

HTTP 的 WebSphere MQ 橋接器與佇列管理程式之間發生錯誤

說明
HTTP 的 WebSphere MQ 橋接器與佇列管理程式之間發生錯誤

HTTP 狀態碼
502 Bad Gateway

程式設計師回應
請聯絡 WebSphere MQ Bridge for HTTP 的系統管理者。

MQHTTP50401: 訊息擷取逾時

在逾時期間未傳回符合 HTTP **GET** 或 HTTP **DELETE** 中所指定要求參數的訊息。

說明
在逾時期間未傳回符合 HTTP **GET** 或 HTTP **DELETE** 中所指定要求參數的訊息。 回覆碼指出在 HTTP 要求的生命期間，任何時間都沒有可用的適當訊息。

HTTP 狀態碼
`504 Gateway timeout`

程式設計師回應
如果預期訊息，請檢查 HTTP 要求的標頭欄位，例如 `x-msg-correlId` 和 `x-msg-msgid`。 請檢查 HTTP 要求 URI 中指定的目的地是否正確。 請嘗試使用 `x-msg-wait` 標頭欄位來延伸 HTTP 要求的等待時間。

MQHTTP50501: HTTP 1.1 以上 ...

適用於 HTTP 的 WebSphere MQ 橋接器不支援 HTTP 要求中使用的 HTTP 通訊協定。

說明
適用於 HTTP 的 WebSphere MQ 橋接器不支援 HTTP 要求中使用的 HTTP 通訊協定。

HTTP 狀態碼
`505 HTTP version not supported`

程式設計師回應
變更 HTTP 要求以使用 HTTP 通訊協定 V1.1 或更高版本。

WebSphere Bridge for HTTP 的訊息類型及訊息對映

WebSphere 適用於 HTTP 的 MQ 橋接器支援四個訊息類別: TEXT、 BYTES、 STREAM 及 MAP。 訊息類別會對映至 JMS 訊息類型和 HTTP `Content-Type`。

HTTP POST

到達目的地的訊息類型取決於 HTTP 要求的 `x-msg-class` 標頭或 **內容類型** 的值。 第 1118 頁的表 604 顯示對應於每一個 `x-msg-class` 的 HTTP **內容類型** 類型。 任一欄位可用來設定訊息類型及訊息格式。 如果兩個欄位都已設定，且設定不一致，則會傳回 `Bad Request exception` (`HTTP 400, MQHTTP20004`)。

表 *604: `x-msg-class`* 與 *HTTP 內容類型* 之間的對映

x-msg-class	**HTTP 內容類型**
BYTES	`application/octet-stream` `application/xml`
TEXT	`text/*`
MAP	`application/x-www-form-urlencoded` `application/xml` （選用）

表 *604: x-msg-class* 與 *HTTP 內容類型* 之間的對映 (繼續)	
x-msg-class	**HTTP 內容類型**
STREAM	application/xml（選用）

如果在 MQRFH2 標頭中設定 JMS 訊息類型，則會根據 第 1119 頁的表 605 進行對映。

表 *605: x-msg-class* 與 *JMS* 訊息類型之間的對映。	
x-msg-class	**JMS 訊息類型**
BYTES	jms_bytes
TEXT	jms_text
MAP	jms_map
STREAM	jms_stream

JMS 訊息類型一律設定為 MAP 或 STREAM 訊息類別。 並非一律為 BYTES 或 TEXT 訊息類別設定。 如果要為要求建立 MQRFH2 ，則一律會設定 JMS 訊息類型。 否則，如果未建立 MQRFH2 ，則不會設定 JMS 訊息類型。 如果在要求中使用 x-msg-usr 標頭設定使用者內容，則會建立 MQRFH2 。

如果已設定 JMS 訊息類型，則訊息格式會設為 MQFMT_NONE，請參閱 第 1119 頁的表 607:

表 *606: x-msg-class* 與 *WebSphere MQ* 訊息格式之間的對映		
x-msg-class	**訊息中呈現 MQRFH2 的訊息格式**	**訊息中出現 *no* MQRFH2 的訊息格式**
BYTES	MQFMT_NONE	MQFMT_NONE
TEXT	MQFMT_NONE	MQFMT_STRING
MAP	MQFMT_NONE	不可能
STREAM	MQFMT_NONE	不可能

HTTP GET 或 DELETE

擷取的訊息類型或格式會決定 HTTP 回應的 x-msg-class 標頭和 Content-Type 的值。 只有在 x-msg-headers 要求中要求時，才會傳回 x-msg-class 標頭。

第 1119 頁的表 607 說明 x-msg-class 和 **內容類型**之間的對映，以及從佇列或主題擷取的訊息類型。

表 *607:* 將訊息類型對映至 *x-msg-class* 和 *內容類型*			
訊息格式	**JMS 訊息類型**	**x-msg-class**	**內容類型**
MQFMT_STRING 以外的任何項目	無	BYTES	application/octet-stream
MQFMT_STRING	無	TEXT	text/plain
MQFMT_NONE	jms_bytes	BYTES	application/octet-stream
MQFMT_NONE	jms_text	TEXT	text/plain
MQFMT_NONE	jms_map	MAP	application/xml
MQFMT_NONE	jms_stream	STREAM	application/xml

MAP 和 STREAM 訊息類別序列化

MAP 和 STREAM 訊息類別會在 HTTP 回應中序列化回用戶端，其方式與將訊息序列化至佇列的方式相同。

對於 MAP， XML 名稱、類型及值三項值編碼為:

```
<map>
    <elt name="elementname1" dt="datatype1">value1</elt>
    <elt name="elementname2" dt="datatype2">value2</elt>
    ...
</map>
```

STREAM 類似於 MAP，但沒有元素名稱:

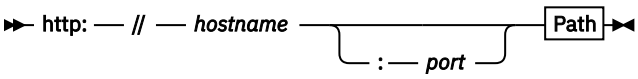
```
<stream>
    <elt dt="datatype1">value1</elt>
    <elt dt="datatype2">value2</elt>
    ...
</stream>
```

註: datatype 是定義用於定義使用者定義內容的其中一個資料類型，並列在 第 1109 頁的『usr:HTTP x-msg-usr 實體標頭』中。 字串元素會省略 dt="string" 屬性，因為預設資料類型是 string。

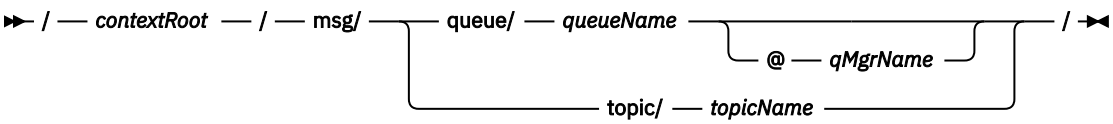
URI 格式

HTTP 的 WebSphere MQ 橋接器所截取的 URI。

Syntax



Path



註:

1. If a question mark (?) is used it must be substituted with %3f. For example, orange?topic should be specified as orange%3ftopic.
2. @*qMgrName* is only valid on an HTTP **POST**

說明

使用 *contextRoot* 環境定義根目錄，將 WebSphere MQ for HTTP Servlet 橋接器部署至 JEE 應用程式伺服器。 要求至

```
http://hostname:port/context_root/msg/queue/queueName@qMgrName
```

和

```
http://hostname:port/context_root/msg/topic/topicString
```

由適用於 HTTP 的 WebSphere MQ 橋接器截取。

IBM WebSphere MQ .NET 類別和介面

IBM WebSphere MQ .NET 類別和介面會按字母順序列出。 說明內容、方法和建構子。

MQAsyncStatus .NET 類別

使用 MQAsyncStatus 來查詢前一個 MQI 活動的狀態; 例如，查詢前一個非同步放置作業是否成功。 MQAsyncStatus 會封裝 MQSTS 資料結構的特性。

類別

```
System.Object
    └ IBM.WMQ.MQBase
        └ IBM.WMQ.MQBaseObject
            └ IBM.WMQ.MQAsyncStatus
```

```
public class IBM.WMQ.MQAsyncStatus extends IBM.WMQ.MQBaseObject;
```


- 第 1121 頁的『內容』
- 第 1122 頁的『建構子』

內容

測試在取得內容時是否擲出 MQException 。

public static int CompCode {get;}

第一個錯誤或警告的完成碼。

public static int Reason {get;}

第一個錯誤或警告的原因碼。

public static int PutSuccessCount {get;}

成功的非同步 MQI 放置呼叫數。

public static int PutWarningCount {get;}

成功但有警告的非同步 MQI 放置呼叫數。

public static int PutFailureCount {get;}

失敗的非同步 MQI 放置呼叫數。

public static int ObjectType {get;}

第一個錯誤的物件類型。 下列為可能的值:

- MQC.MQOT_ALIAS_Q
- MQC.MQOT_LOCAL_Q
- MQC.MQOT_MODEL_Q
- MQC.MQOT_Q
- MQC.MQOT_REMOTE_Q
- MQC.MQOT_TOPIC
- 0，表示未傳回任何物件

public static string ObjectName {get;}

物件名稱。

public static string ObjectQMgrName {get;}

物件佇列管理程式名稱。

public static string ResolvedObjectName {get;}

已解析的物件名稱。

public static string ResolvedObjectQMgrName {get;}

已解析的物件佇列管理程式名稱。

建構子

public MQAsyncStatus() throws MQException;

建構子方法，在適當的情況下，建構欄位起始設定為零或空白的物件。

MQAuthenticationInformationRecord .NET 類別

使用 MQAuthenticationInformationRecord 來指定要在 WebSphere MQ SSL 用戶端連線中使用之鑑別器的相關資訊。 MQAuthenticationInformationRecord 會封裝鑑別資訊記錄 MQAIR。

類別

```
System.Object
    └ IBM.WMQ.MQAuthenticationInformationRecord
```

public class IBM.WMQ.MQAuthenticationInformationRecord extends System.Object;

- 第 1122 頁的『內容』
- 第 1123 頁的『建構子』

內容

測試在取得內容時是否擲出 MQException 。

public long Version {get; set;}

結構版本號碼。

public long AuthInfoType {get; set;}

鑑別資訊的類型。 此屬性必須設為下列其中一個值:

- OCSP -使用 OCSP 完成憑證撤銷狀態檢查。
- CRLLDAP -使用 LDAP 伺服器上的「憑證撤銷清冊」完成憑證撤銷狀態檢查。

public string AuthInfoConnName {get; set;}

LDAP 伺服器執行所在主機的 DNS 名稱或 IP 位址，具有選用的埠號。 此關鍵字是必要的。

public string LDAPPassword {get; set;}

與存取 LDAP 伺服器之使用者的識別名稱相關聯的密碼。 僅當 **AuthInfoType** 設定為 CRLLDAP 時，此內容才適用。

public string LDAPUserName {get; set;}

存取 LDAP 伺服器之使用者的識別名稱。 當您設定此內容時，會自動正確設定 LDAPUserName **長度** 和 LDAPUserNamePtr 。 僅當 AuthInfo **類型** 設為 CRLLDAP 時，此內容才適用。

public string OCSPResponderURL {get; set;}

可用來聯絡 OCSP 回應端的 URL。 僅當 AuthInfo **類型** 設為 OCSP 時，此內容才適用

此欄位區分大小寫。 它必須以小寫字串 http:// 開頭。 視 OCSP 伺服器實作而定， URL 的其餘部分可能區分大小寫。

建構子

MQAuthenticationInformationRecord();

MQDestination .NET 類別

使用 MQDestination 來存取 MQQueue 和 MQTopic 共用的方法。 MQDestination 是抽象基礎類別，無法實例化。

類別

```
System.Object
    └ IBM.WMQ.MQBase
        └ IBM.WMQ.MQBaseObject
            └ IBM.WMQ.MQManagedObject
                └ IBM.WMQ.MQDestination
                    ├ IBM.WMQ.MQQueue
                    └ IBM.WMQ.MQTopic
```

public class IBM.WMQ.MQDestination extends IBM.WMQ.MQManagedObject;

- 第 1123 頁的『內容』
- 第 1123 頁的『方法』
- 第 1125 頁的『建構子』

內容

測試在取得內容時是否擲出 MQException 。

public DateTime CreationDateTime {get;}

建立佇列或主題的日期和時間。 最初包含在 MQQueue 中，此內容已移至基礎 MQDestination 類別。

沒有預設值。

public int DestinationType {get;}

整數值，說明所使用的目的地類型。 從子類別建構子 MQQueue 或 MQTopic 起始設定，此值可以採用下列其中一個值:

- MQOT_Q
- MQOT_TOPIC

沒有預設值。

方法

public void Get(MQMessage *message*);
public void Get(MQMessage *message*, MQGetMessageOptions *getMessageOptions*);
public void Get(MQMessage *message*, MQGetMessageOptions *getMessageOptions*, int *MaxMsgSize*);

擲出 MQException。

如果目的地是 MQQueue 物件，則從佇列取得訊息; 如果目的地是 MQTopic 物件，則使用預設 MQGetMessageOptions 實例來取得訊息。

如果取得失敗，則 MQMessage 物件保持不變。 如果成功， MQMessage 的訊息描述子和訊息資料部分會取代為送入訊息中的訊息描述子和訊息資料。

從特定 MQQueueManager 對 WebSphere MQ 的所有呼叫都是同步的。 因此，如果您執行 get with wait，則在完成 Get 呼叫之前，會封鎖所有其他使用相同 MQQueueManager 的執行緒進一步進行 WebSphere MQ 呼叫。 如果您需要多個執行緒來同步存取 WebSphere MQ，則每一個執行緒都必須建立自己的 MQQueueManager 物件。

message
包含訊息描述子及傳回的訊息資料。 訊息描述子中的部分欄位是輸入參數。 請務必確定 MessageId 和 CorrelationId 輸入參數已根據需要設定。

可重新連接的用戶端會針對在 MQGM_SYNCPOINT 下收到的訊息，在成功重新連線之後傳回原因碼 MQRC_BACKED_OUT。

getMessageOptions
控制 get 動作的選項。

從單位元組字元碼轉換為雙位元組碼時，使用選項 MQC.MQGMO_CONVERT 可能會導致異常狀況，原因碼為 MQC.MQRC_CONVERTED_STRING_TOO_BIG。 在此情況下，會將訊息複製到緩衝區而不進行轉換。

如果未指定 *getMessageOptions*，則使用的訊息選項為 MQGMO_NOWAIT。

如果您在可重新連接的用戶端中使用 MQGMO_LOGICAL_ORDER 選項，則會傳回 MQRC_RECONNECT_INCOMPATIBLE 原因碼。

MaxMsgSize
此訊息物件要接收的最大訊息。 如果佇列上的訊息大於此大小，則會發生下列兩種情況之一:

- 如果在 MQGetMessageOptions 物件中設定 MQGMO_ACCEPT_TRUNCATED_MSG 旗標，則訊息會盡可能填入訊息資料。 擲出異常狀況，含有 MQCC_WARNING 完成碼和 MQRC_TRUNCATED_MSG_ACCEPTED 原因碼。
- 如果未設定 MQGMO_ACCEPT_TRUNCATED_MSG 旗標，則訊息會留在佇列上。 擲出異常狀況，含有 MQCC_WARNING 完成碼和 MQRC_TRUNCATED_MSG_FAILED 原因碼。

如果未指定 *MaxMsgSize*，則會擷取整個訊息。

```
public void Put(MQMessage message);
public void Put(MQMessage message, MQPutMessageOptions putMessageOptions);
```

擲出 MQException。

如果目的地是 MQQueue 物件，則將訊息放入佇列; 如果目的地是 MQTopic 物件，則將訊息發佈至主題。

完成「放置」呼叫之後對 MQMessage 物件的修改不會影響 WebSphere MQ 佇列或發佈主題上的實際訊息。

Put 會更新 MQMessage 物件的 MessageId 和 CorrelationId 內容，且不會清除訊息資料。 進一步的 Put 或 Get 呼叫會參照 MQMessage 物件中的更新資訊。 例如，在下列程式碼 Snippet 中，第一個訊息包含 a 和第二個 ab。

```
msg.WriteString("a");
q.Put(msg,pmo);
msg.WriteString("b");
q.Put(msg,pmo);
```

message
包含訊息描述子資料及要傳送之訊息的 MQMessage 物件。 此方法可以變更訊息描述子。 在此方法完成之後立即在訊息描述子中的值是已放入佇列或發佈至主題的值。

下列原因碼會傳回至可重新連接的用戶端:

- 如果在持續訊息上執行「放置」呼叫時連線中斷，且重新連線成功，則為 MQRC_CALL_INTERRUPTED。
- MQRC_NONE 如果在對非持續訊息執行 Put 呼叫時連線成功 (請參閱 應用程式回復)。

putMessageOptions
控制 put 動作的選項。

如果未指定 *putMessageOptions* ，則會使用 MQPutMessageOptons 的預設實例。

如果您在可重新連接的用戶端中使用 MQPMO_LOGICAL_ORDER 選項，則會傳回 MQRC_RECONNECT_INCOMPATIBLE 原因碼。

註: 為了簡單和效能，如果您想要將單一訊息放入佇列，請使用 MQQueueManager.Put 物件。 您應該為此具有 MQQueue 物件。

建構子

MQDestination 是抽象基礎類別，無法實例化。 使用 MQQueue 和 MQTopic 建構子或使用 MQQueueManager.AccessQueue 和 MQQueueManager.AccessTopic methods 來存取目的地。

MQEnvironment .NET 類別

使用 MQEnvironment 來控制如何呼叫 MQQueueManager 建構子，以及選取 WebSphere MQ MQI 用戶端連線。 MQEnvironment 類別包含控制 WebSphere MQ 行為的內容。

類別

```
System.Object
    │
    └─ IBM.WMQ.MQEnvironment
```

```
public class IBM.WMQ.MQEnvironment extends System.Object;
```


- 第 1125 頁的『內容-僅限用戶端』
- 第 1126 頁的『內容』
- 第 1127 頁的『建構子』

內容-僅限用戶端

測試在取得內容時是否擲出 MQException 。

public static int CertificateValPolicy {get; set;}

設定使用哪個 SSL/TLS 憑證驗證原則來驗證從遠端夥伴系統收到的數位憑證。 有效值為:

- MQC.CERTIFICATE_VALIDATION_POLICY_ANY
- MQC.CERTIFICATE_VALIDATION_POLICY_RFC5280

public static ArrayList EncryptionPolicySuiteB {get; set;}

設定 Suite B 相容加密法的層次。 有效值為:

- MQC.MQ_SUITE_B_NONE -這是預設值。
- MQC.MQ_SUITE_B_128_BIT
- MQC.MQ_SUITE_B_192_BIT

public static string Channel {get; set;}

要連接至目標佇列管理程式的通道名稱。 在用戶端模式下實例化 MQQueueManager 實例之前，您 必須設定通道內容。

public static int FipsRequired {get; set;}

如果在 WebSphere MQ 中執行加密法，請指定 MQC.MQSSL_FIPS_YES 只使用 FIPS 認證的演算法。 預設值為 MQC.MQSSL_FIPS_NO。

如果已配置加密硬體，則使用的加密模組是硬體產品所提供的那些加密模組。 視使用中的硬體而定，這些可能未經過 FIPS 認證達到特定層次。

public static string Hostname {get; set;}

WebSphere MQ 伺服器所在電腦的 TCP/IP 主機名稱。 如果未設定主機名稱，且未設定置換內容，則會使用伺服器連結模式來連接本端佇列管理程式。

public static int Port {get; set;}
要連接的埠。 這是 WebSphere MQ 伺服器接聽送入連線要求的埠。 預設值是 1414。

public static string SSLCipherSpec {get; set;}
將 SSLCipherSpec 設為 SVRCONN 通道上設定的 CipherSpec 值，以啟用連線的 SSL。 預設值為「空值」，且未針對連線啟用 SSL。

public static string sslPeerName {get; set;}

識別名稱型樣。 如果設定 sslCipher **規格**，此變數可用來確保使用正確的佇列管理程式。 如果設為空值 (預設值)，則不會執行佇列管理程式的 DN。 如果 sslCipherSpec 是空值，則會忽略 sslPeer **名稱**。

內容

測試在取得內容時是否擲出 MQException。

public static ArrayList HdrCompList {get; set;}
標頭資料壓縮清單

public static int KeyResetCount {get; set;}
指出在重新協議秘密金鑰之前，在 SSL 交談內傳送及接收的未加密位元組數。

public static ArrayList MQAIRArray {get; set;}
MQAuthenticationInformationRecord 物件的陣列。

public static ArrayList MsgCompList {get; set;}
訊息資料壓縮清單

public static string Password {get; set;}
要鑑別的密碼。 透過設定此「密碼」內容，移入從 MQCSP 結構參照的密碼。

public static string ReceiveExit {get; set;}
接收結束程式可讓您檢查及變更從佇列管理程式收到的資料。 它通常與佇列管理程式中對應的傳送結束程式搭配使用。 如果 ReceiveExit 設為空值，則不會呼叫任何接收結束程式。

public static string ReceiveUserData {get; set;}
與接收結束程式相關聯的使用者資料。 限制為 32 個字元。

public static string SecurityExit {get; set;}
安全結束程式可讓您自訂嘗試連接至佇列管理程式時所發生的安全流程。 如果 SecurityExit 設為空值，則不會呼叫安全結束程式。

public static string SecurityUserData {get; set;}
與安全結束程式相關聯的使用者資料。 限制為 32 個字元。

public static string SendExit {get; set;}
傳送結束程式可讓您檢查或變更傳送至佇列管理程式的資料。 它通常與佇列管理程式中對應的接收結束程式搭配使用。 如果 SendExit 設為空值，則不會呼叫傳送結束程式。

public static string SendUserData {get; set;}
與傳送結束程式相關聯的使用者資料。 限制為 32 個字元。

public static string SharingConversations {get; set;}

當 .NET 應用程式沒有使用用戶端通道定義表 (CCDT) 時，這些應用程式的連線會使用 SharingConversations 欄位。

SharingConversations 決定在與此連線相關聯的 Socket 上可以共用的交談數上限。

值 0 表示通道的運作方式與 WebSphere MQ 7.0 版之前一樣，與交談共用、先讀及活動訊號相關。

當實例化 WebSphere MQ 佇列管理程式時，該欄位會在內容的雜湊表中作為 SHARING_CONVERSATIONS_PROPERTY 傳遞。

如果未指定 SharingConversations，則會使用預設值 10。

public static string SSLCryptoHardware {get; set;}

設定必要的參數字串名稱，以配置系統上呈現的加密硬體。 如果 sslCipherSpec 是空值，則會忽略 SSLCryptoHardware。

public static string SSLKeyRepository {get; set;}

設定金鑰儲存庫的完整檔名。

如果 SSLKeyRepository 設為空值 (預設值)，則會使用憑證 MQSSLKEYR 環境變數來尋找金鑰儲存庫。 如果 sslCipherSpec 是空值，則會忽略 SSLCryptoHardware 。

註: .kdb 副檔名是檔名的必要部分，但不會併入作為參數值的一部分。 您指定的目錄必須存在。 WebSphere MQ 會在第一次存取新的金鑰儲存庫時建立檔案，除非該檔案已存在。

public static string UserId {get; set;}

要鑑別的使用者 ID。 透過設定 UserId，移入從 MQCSP 結構參照的使用者 ID。 使用 API 或安全結束程式來鑑別 UserId 。

建構子

public MQEnvironment()

MQException .NET 類別

使用 MQException 來找出失敗 WebSphere MQ 函數的完成碼和原因碼。 每當發生 WebSphere MQ 錯誤時，都會擲出 MQException 。

類別

```
System.Object
    └ System.Exception
        └ System.ApplicationException
            └ IBM.WMQ.MQException
```

```
public class IBM.WMQ.MQException extends System.ApplicationException;
```


- 第 1127 頁的『內容』
- 第 1127 頁的『建構子』

內容

public int CompletionCode {get; set;}
與錯誤相關聯的 WebSphere MQ 完成碼。 可能值包括:

- MQException.MQCC_OK
- MQException.MQCC_WARNING
- MQException.MQCC_FAILED

public int ReasonCode {get; set;}
說明錯誤的 WebSphere MQ 原因碼。

建構子

public MQException(int *completionCode*, int *reasonCode*)

completionCode
WebSphere MQ 完成碼。

reasonCode
WebSphere MQ 完成碼。

MQGetMessageOptions .NET 類別

使用 MQGetMessageOptions 來指定擷取訊息的方式。 它會修改 MQDestination.Get 的行為。

類別

```
System.Object
    └ IBM.WMQ.MQBase
        └ IBM.WMQ.MQBaseObject
            └ IBM.WMQ.MQGetMessageOptions
```

```
public class IBM.WMQ.MQGetMessageOptions extends IBM.WMQ.MQBaseObject;
```


- 第 1128 頁的『內容』
- 第 1130 頁的『建構子』

內容

註: 此類別中部分可用選項的行為取決於使用這些選項的環境。 這些元素會以星號 * 標示。

測試在取得內容時是否擲出 MQException 。

public int GroupStatus {get;}*
GroupStatus 指出擷取的訊息是否位於群組中，以及是否為群組中的最後一則訊息。 可能的值為:

MQC.MQGS_LAST_MSG_IN_GROUP
訊息是群組中最後一個或唯一的訊息。

MQC.MQGS_MSG_IN_GROUP
訊息位於群組中，但不是群組中的最後一個。

MQC.MQGS_NOT_IN_GROUP
訊息不在群組中。

public int MatchOptions {get; set;}*
MatchOptions 可決定如何選取訊息。 可以設定下列比對選項:

MQC.MQMO_MATCH_CORREL_ID
要比對的相關性 ID。

MQC.MQMO_MATCH_GROUP_ID
要比對的群組 ID。

MQC.MQMO_MATCH_MSG_ID
要比對的訊息 ID。

MQC.MQMO_MATCH_MSG_SEQ_NUMBER
符合訊息序號。

MQC.MQMO_NONE
不需要相符。

public int Options {get; set;}
選項 控制 MQQueue.get 的動作。 可以指定下列任何值。 如果需要多個選項，則可以新增值，或使用位元 OR 運算子結合。

MQC.MQGMO_ACCEPT_TRUNCATED_MSG
容許截斷訊息資料。

MQC.MQGMO_ALL_MSGS_AVAILABLE*
只有在群組中的所有訊息都可用時，才會從群組中擷取訊息。

MQC.MQGMO_ALL_SEGMENTS_AVAILABLE*
僅當群組中的所有區段都可用時，才擷取邏輯訊息的區段。

MQC.MQGMO_BROWSE_FIRST
從佇列開頭瀏覽。

MQC.MQGMO_BROWSE_MSG_UNDER_CURSOR*
瀏覽游標下的瀏覽訊息。

MQC.MQGMO_BROWSE_NEXT
從佇列中的現行位置瀏覽。

MQC.MQGMO_COMPLETE_MSG*
僅擷取完整邏輯訊息。

MQC.MQGMO_CONVERT
要求轉換應用程式資料，以符合 MQMessage 的 CharacterSet 及 Encoding 屬性，然後再將資料複製到訊息緩衝區。 因為從訊息緩衝區擷取資料時也會套用資料轉換，所以應用程式不會設定此選項。

使用此選項可能會在從單位元組字集轉換為雙位元組字集時造成問題。 相反地，在遞送訊息之後，請使用 readString、 readLine 和 writeString 方法來執行轉換。

MQC.MQGMO_FAIL_IF_QUIESCING
如果佇列管理程式在靜止中，則會失敗。

MQC.MQGMO_LOCK*
鎖定已瀏覽的訊息。

MQC.MQGMO_LOGICAL_ORDER*
以群組形式傳回訊息，並以邏輯順序傳回邏輯訊息區段。

如果您在可重新連接的用戶端中使用 MQGMO_LOGICAL_ORDER 選項，則會將 MQRC_RECONNECT_INCOMPATIBLE 原因碼傳回給應用程式。

MQC.MQGMO_MARK_SKIP_BACKOUT*
容許取消工作單元，而不還原佇列上的訊息。

MQC.MQGMO_MSG_UNDER_CURSOR
在瀏覽游標下取得訊息。

MQC.MQGMO_NONE
未指定其他選項; 所有選項都採用其預設值。

MQC.MQGMO_NO_PROPERTIES
不會擷取訊息的內容，但訊息描述子 (或延伸) 中包含的內容除外。

MQC.MQGMO_NO_SYNCPOINT
取得沒有同步點控制的訊息。

MQC.MQGMO_NO_WAIT
如果沒有適當的訊息，請立即傳回。

MQC.MQGMO_PROPERTIES_AS_Q_DEF
擷取 MQQueue 的 PropertyControl 屬性所定義的訊息內容。 對訊息描述子或延伸中訊息內容的存取權不受 PropertyControl 屬性的影響。

MQC.MQGMO_PROPERTIES_COMPATIBILITY
擷取 MQRFH2 標頭中字首為 mcd、 jms、 usr 或 mqext 的訊息內容。 訊息的其他內容 (訊息描述子或延伸中包含的內容除外) 會被捨棄。

MQC.MQGMO_PROPERTIES_FORCE_MQRFH2
擷取訊息內容，但 MQRFH2 標頭中的訊息描述子或延伸中所包含的內容除外。 在預期擷取內容但無法變更為使用訊息控點的應用程式中使用 MQC.MQGMO_PROPERTIES_FORCE_MQRFH2 。

MQC.MQGMO_PROPERTIES_IN_HANDLE
使用 MsgHandle 擷取訊息內容。

MQC.MQGMO_SYNCPOINT
在同步點控制下取得訊息。 訊息會標示為其他應用程式無法使用，但只有在確定工作單元時，才會從佇列中刪除它。 如果工作單元已取消，則訊息會重新變成可用。

MQC.MQGMO_SYNCPOINT_IF_PERSISTENT*
如果訊息持續存在，則取得具有同步點控制的訊息。

MQC.MQGMO_UNLOCK*
解除鎖定先前鎖定的訊息。

MQC.MQGMO_WAIT
等待訊息到達。

public string ResolvedQueueName {get;}
佇列管理程式會將 ResolvedQueueName 設為從中擷取訊息之佇列的本端名稱。 ResolvedQueueName 不同於開啟別名佇列或模型佇列時用來開啟佇列的名稱。

public char Segmentation {get;}*
分段 指出您是否可以對擷取的訊息進行分段。 可能的值為:

MQC.MQSEG_INHIBITED
不容許分段。

MQC.MQSEG_ALLOWED
容許分段

public byte SegmentStatus {get;}*
SegmentStatus 是一個輸出欄位，指出擷取的訊息是否為邏輯訊息的區段。 如果訊息是區段，則旗標會指出它是否為最後一個區段。 可能的值為:

MQC.MQSS_LAST_SEGMENT
訊息是邏輯訊息的最後一個或唯一區段。

MQC.MQSS_NOT_A_SEGMENT
訊息不是區段。

MQC.MQSS_SEGMENT
訊息是區段，但不是邏輯訊息的最後一個區段。

public int WaitInterval {get; set;}
WaitInterval 是 MQQueue.get 呼叫等待適當訊息到達的時間上限 (毫秒)。 將 WaitInterval 與 MQC.MQGMO_WAIT 搭配使用。 將 MQC.MQWI_UNLIMITED 值設為等待訊息的時間無限制。

建構子

public MQGetMessageOptions()
建構新的 MQGetMessageOptions 物件，其中 **選項** 設為 MQC.MQGMO_NO_WAIT， WaitInterval 設為零， ResolvedQueue **名稱** 設為空白。

MQManagedObject .NET 類別

使用 MQManagedObject 來查詢及設定 MQDestination、 MQProcess、 MQQueueManager 及 MQSubscription 的屬性。 MQManagedObject 是這些類別的超類別。

類別

```
System.Object
    └ IBM.WMQ.MQBase
        └ IBM.WMQ.MQBaseObject
            └ IBM.WMQ.MQManagedObject
                ├ IBM.WMQ.MQProcess
                ├ IBM.WMQ.MQQueueManager
                ├ IBM.WMQ.MQSubscription
                └ IBM.WMQ.MQDestination
```

```
public class IBM.WMQ.MQManagedObject extends IBM.WMQ.MQBaseObject;
```


- 第 1131 頁的『內容』
- 第 1131 頁的『方法』

- 第 1132 頁的『建構子』

內容

測試在取得內容時是否擲出 MQException 。

public string AlternateUserId {get; set;}
開啟資源時所設定的替代使用者 ID (如果有的話)。 當針對已開啟的物件發出 AlternateUserID.set 時，會忽略它。 AlternateUserId 對訂閱無效。

public int CloseOptions {get; set;}

設定此屬性以控制關閉資源的方式。 預設值為 MQC.MQCO_NONE。 除了永久動態佇列、暫時動態佇列、訂閱及建立它們的物件所存取的主題之外， MQC.MQCO_NONE 是所有資源唯一允許的值。

對於佇列及主題，允許下列其他值:

MQC.MQCO_DELETE
如果沒有訊息，請刪除佇列。

MQC.MQCO_DELETE_PURGE
刪除佇列，並清除其中的任何訊息。

MQC.MQCO_QUIESCE
要求關閉佇列，如果仍有任何訊息，則會收到警告 (容許在最終關閉之前擷取這些訊息)。

對於訂閱，允許下列其他值:

MQC.MQCO_KEEP_SUB
未刪除訂閱。 只有在原始訂閱可延續時，此選項才有效。 MQC.MQCO_KEEP_SUB 是可延續主題的預設值。

MQC.MQCO_REMOVE_SUB
已刪除訂閱。 MQC.MQCO_REMOVE_SUB 是不可延續、未受管理主題的預設值。

MQC.MQCO_PURGE_SUB
已刪除訂閱。 MQC.MQCO_PURGE_SUB 是不可延續的受管理主題的預設值。

public MQQueueManager ConnectionReference {get;}
此資源所屬的佇列管理程式。

public string MQDescription {get;}
佇列管理程式所保留資源的說明。 MQDescription 會傳回訂閱及主題的空字串。

public boolean IsOpen {get;}
指出資源目前是否開啟。

public string Name {get;}
資源的名稱。 名稱是在存取方法上提供，或由佇列管理程式配置給動態佇列。

public int OpenOptions {get; set;}
當開啟 WebSphere MQ 物件時，會設定 OpenOptions 。 OpenOptions.set 方法會被忽略，且不會導致錯誤。 訂閱沒有 OpenOptions。

方法

public virtual void Close();
擲出 MQException。

關閉物件。 在呼叫 Close 之後，不允許對此資源執行進一步作業。 若要變更 Close 方法的行為，請設定 closeOptions 屬性。

public string GetAttributeString(int *selector*, int *length*);
擲出 MQException。

取得屬性字串。

selector
整數，指出要查詢的屬性。

length
整數，指出所需字串的長度。

public void Inquire(int[] *selectors*, int[] *intAttrs*, byte[] *charAttrs*);
擲出 MQException。

傳回整數陣列及一組字串，其中包含佇列、處理程序或佇列管理程式的屬性。 要查詢的屬性指定在選取元陣列中。

註: 可以使用 MQManagedObject、 MQQueue 和 MQQueueManager 中定義的 Get 方法來查詢許多較常見的屬性。

selectors
整數陣列，識別具有要查詢的值的屬性。

intAttrs
傳回整數屬性值的陣列。 傳回整數屬性值的順序與選取元陣列中整數屬性選取元的順序相同。

charAttrs
在其中傳回字元屬性的緩衝區 (已連結)。 字元屬性會以與選取元陣列中字元屬性選取元相同的順序傳回。 每一個屬性字串的長度都是固定的。

public void Set(int[] *selectors*, int[] *intAttrs*, byte[] *charAttrs*);
擲出 MQException。

設定選取元向量中定義的屬性。 要設定的屬性指定在 selectors 陣列中。

selectors
整數陣列，識別具有要設定的值的屬性。

intAttrs
要設定的整數屬性值陣列。 這些值必須與選取元陣列中整數屬性選取元的順序相同。

charAttrs
連結要設定之字元屬性的緩衝區。 這些值必須與選取元陣列中字元屬性選取元的順序相同。 每一個字元屬性的長度都是固定的。

public void SetAttributeString(int *selector*, string *value*, int *length*);
擲出 MQException。

設定屬性字串。

selector
整數，指出要設定的屬性。

value
要設為屬性值的字串。

length
整數，指出所需字串的長度。

建構子

protected MQManagedObject()
建構子方法。 此物件是抽象基礎類別，無法自行實例化。

MQMessage .NET 類別

使用 MQMessage 來存取 WebSphere MQ 訊息的訊息描述子及資料。 MQMessage 會封裝 WebSphere MQ 訊息。

類別

```
System.Object
    └ IBM.WMQ.MQBase
        └ IBM.WMQ.MQBaseObject
            └ IBM.WMQ.MQMessage
```

```
public class IBM.WMQ.MQMessage extends IBM.WMQ.MQBaseObject;
```

建立 MQMessage 物件，然後使用 Read 及 Write 方法，在訊息與應用程式中其他物件之間傳送資料。 使用 MQDestination、 MQQueue 和 MQTopic 類別的 Put 和 Get 方法來傳送和接收 MQMessage 物件。

使用 MQMessage 的內容來取得並設定訊息描述子的內容。 使用 SetProperty 和 GetProperty 方法來設定及取得延伸訊息內容。

- 第 1133 頁的『內容』
- 第 1138 頁的『Read 和 Write 訊息方法』
- 第 1140 頁的『緩衝區方法』
- 第 1141 頁的『內容方法』
- 第 1143 頁的『建構子』

內容

測試在取得內容時是否擲出 MQException 。

public string AccountingToken {get; set;}
訊息的部分身分環境定義; 它可協助應用程式收取因訊息而完成的工作費用。 預設值為 MQC.MQACT_NONE。

public string ApplicationIdData {get; set;}
訊息身分環境定義的一部分。 ApplicationId **資料** 是應用程式套組所定義的資訊，可用來提供訊息或其發送端的其他相關資訊。 預設值為 ""。

public string ApplicationOriginData {get; set;}
應用程式所定義的資訊，可用來提供訊息來源的其他相關資訊。 預設值為 ""。

public int BackoutCount {get;}
MQQueue.Get 呼叫先前在工作單元中所傳回及取消訊息的次數。 預設值為零。

public int CharacterSet {get; set;}

訊息中字元資料的編碼字集 ID。

設定 CharacterSet ， 以識別訊息中字元資料的字集。 取得 CharacterSet ， 以找出訊息中用來編碼字元資料的字集。

.NET 應用程式一律以 Unicode 執行，而在其他環境中，應用程式則以佇列管理程式的相同字集執行。

ReadString 和 ReadLine 方法會為您將訊息中的字元資料轉換成 Unicode。

WriteString 方法會從 Unicode 轉換為 CharacterSet 中編碼的字集。 如果 CharacterSet 設為其預設值 MQC.MQCCSI_Q_MGR(即 0) ， 則不會進行轉換，且 CharacterSet 會設為 1200。 如果您將 CharacterSet 設為其他值， WriteString 會從 Unicode 轉換為替代值。

註: 其他讀取及寫入方法不使用 CharacterSet。

- ReadChar 及 WriteChar 會在訊息緩衝區中來回讀取及寫入 Unicode 字元，而不進行轉換。
- ReadUTF 和 WriteUTF 會在應用程式中的 Unicode 字串與訊息緩衝區中的 UTF-8 字串 (字首為 2 位元組長度欄位) 之間進行轉換。
- 位元組方法會在應用程式與訊息緩衝區之間傳送位元組，而不會變更。

public byte[] CorrelationId {get; set;}

- 對於 MQQueue.Get 呼叫，這是要擷取之訊息的相關性 ID。 佇列管理程式會傳回第一個訊息，其中包含符合訊息描述子欄位的訊息 ID 及相關性 ID。 預設值 MQC.MQCI_NONE 可協助任何相關性 ID 相符。
- 若為 MQQueue.Put 呼叫，則為要設定的相關性 ID。

public int DataLength {get;}
剩餘要讀取的訊息資料位元組數。

public int DataOffset {get; set;}
訊息資料內的現行游標位置。 讀取及寫入會在現行位置生效。

public int Encoding {get; set;}
用於應用程式訊息資料中數值的表示法。 **編碼** 適用於二進位、聚集十進位及浮點數資料。 這些數值格式的讀取和寫入方法的行為會相應地變更。 從這三個區段中各新增一個值，以建構編碼欄位的值。 或者，使用位元 OR 運算子來建構結合三個區段中每一個的值的值。

1. 二進位整數

   MQC.MQENC_INTEGER_NORMAL
   大序排列法整數。

   MQC.MQENC_INTEGER_REVERSED
   小序排列法整數，在 Intel 架構中使用。

2. 壓縮十進位

   MQC.MQENC_DECIMAL_NORMAL
   大序排列法聚集十進位，由 z/OS 使用。

   MQC.MQENC_DECIMAL_REVERSED
   小序排列法聚集十進位。

3. 浮點數

   MQC.MQENC_FLOAT_IEEE_NORMAL
   大序排列法 IEEE 浮點數。

   MQC.MQENC_FLOAT_IEEE_REVERSED
   小序排列法 IEEE 浮點數，用作 Intel 架構。

   MQC.MQENC_FLOAT_S390
   z/OS 格式化浮點數。

預設值為:

```
MQC.MQENC_INTEGER_REVERSED |
MQC.MQENC_DECIMAL_REVERSED |
MQC.MQENC_FLOAT_IEEE_REVERSED
```

預設值會使 WriteInt 寫入小序排列法整數，而 ReadInt 讀取小序排列法整數。 如果您改為設定旗標 MQC.MQENC_INTEGER_NORMAL 旗標，則 WriteInt 會寫入大序排列法整數，而 ReadInt 會讀取大序排列法整數。

註: 從 IEEE 格式浮點轉換為 zSeries 格式浮點時，可能會失去精準度。

public int Expiry {get; set;}
由放置訊息的應用程式所設定的到期時間 (以十分之一秒為單位)。 在經歷訊息的到期時間之後，即可由佇列管理程式捨棄。 如果訊息指定其中一個 MQC.MQRO_EXPIRATION 旗標，則會在捨棄訊息時產生報告。 預設值為 MQC.MQEI_UNLIMITED，表示訊息永不到期。

public int Feedback {get; set;}
將 **意見回饋** 與類型為 MQC.MQMT_REPORT 的訊息搭配使用，以指出報告的本質。 系統會定義下列回饋碼:

- 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

也可以使用 MQC.MQFB_APPL_FIRST 至 MQC.MQFB_APPL_LAST 範圍內的應用程式定義回饋值。 此欄位的預設值為 MQC.MQFB_NONE，表示未提供任何意見。

public string Format {get; set;}
訊息傳送者用來向接收端指出訊息中資料本質的格式名稱。 您可以使用自己的格式名稱，但以字母 MQ 開頭的名稱具有佇列管理程式所定義的意義。 佇列管理程式內建格式如下:

MQC.MQFMT_ADMIN
指令伺服器要求/回覆訊息。

MQC.MQFMT_COMMAND_1
鍵入 1 指令回覆訊息。

MQC.MQFMT_COMMAND_2
鍵入 2 指令回覆訊息。

MQC.MQFMT_DEAD_LETTER_HEADER
無法傳送的郵件標頭。

MQC.MQFMT_EVENT
事件訊息。

MQC.MQFMT_NONE
沒有格式名稱。

MQC.MQFMT_PCF
可程式化指令格式的使用者定義訊息。

MQC.MQFMT_STRING
完全由字元組成的訊息。

MQC.MQFMT_TRIGGER
觸發訊息

MQC.MQFMT_XMIT_Q_HEADER
傳輸佇列標頭。

預設值為 MQC.MQFMT_NONE。

public byte[] GroupId {get; set;}
識別實體訊息所屬之訊息群組的位元組字串。 預設值為 MQC.MQGI_NONE。

public int MessageFlags {get; set;}
控制訊息分段及狀態的旗標。

public byte[] MessageId {get; set;}

對於 MQQueue.Get 呼叫，此欄位指定要擷取之訊息的訊息 ID。 通常，佇列管理程式會傳回第一個訊息，其中包含符合訊息描述子欄位的訊息 ID 及相關性 ID。 使用特殊值 MQC.MQMI_NONE 容許任何訊息 ID 相符。

對於 MQQueue.Put 呼叫，此欄位指定要使用的訊息 ID。 如果指定 MQC.MQMI_NONE ，則佇列管理程式會在放置訊息時產生唯一訊息 ID。 放置之後會更新此成員變數的值，以指出所使用的訊息 ID。 預設值為 MQC.MQMI_NONE。

public int MessageLength {get;}
MQMessage 物件中訊息資料的位元組數。

public int MessageSequenceNumber {get; set;}
群組內邏輯訊息的序號。

public int MessageType {get; set;}
指出訊息的類型。 下列值目前由系統定義:

- MQC.MQMT_DATAGRAM
- MQC.MQMT_REPLY
- MQC.MQMT_REPORT
- MQC.MQMT_REQUEST

也可以使用應用程式定義值，範圍為 MQC.MQMT_APPL_FIRST 至 MQC.MQMT_APPL_LAST。 此欄位的預設值為 MQC.MQMT_DATAGRAM。

public int Offset {get; set;}
在分段的訊息中，實體訊息中資料從邏輯訊息開始的偏移。

public int OriginalLength {get; set;}
分段訊息的原始長度。

public int Persistence {get; set;}

訊息持續性。 已定義下列值:

- MQC.MQPER_NOT_PERSISTENT

  如果您在可重新連接的用戶端中設定此選項，當連線成功時，MQRC_NONE 原因碼會傳回給應用程式。
- MQC.MQPER_PERSISTENT

  如果您在可重新連接的用戶端中設定此選項，則在連線成功之後， MQRC_CALL_INTERRUPTED 原因碼會傳回給應用程式。
- MQC.MQPER_PERSISTENCE_AS_Q_DEF

預設值為 MQC.MQPER_PERSISTENCE_AS_Q_DEF，它會從目的地佇列的預設持續性屬性取得訊息的持續性。

public int Priority {get; set;}
訊息優先順序。 也可以在出埠訊息中設定特殊值 MQC.MQPRI_PRIORITY_AS_Q_DEF 。 然後會從目的地佇列的預設優先順序屬性取得訊息的優先順序。 預設值為 MQC.MQPRI_PRIORITY_AS_Q_DEF。

public int PropertyValidation {get; set;}
指定是否在設定訊息的內容時進行內容驗證。 可能的值為:

- MQCMHO_DEFAULT_VALIDATION
- MQCMHO_VALIDATE
- MQCMHO_NO_VALIDATION

預設值為 MQCMHO_DEFAULT_VALIDATION。

public string PutApplicationName {get; set;}
放置訊息的應用程式名稱。 預設值為 ""。

public int PutApplicationType {get; set;}
放置訊息的應用程式類型。 PutApplication **類型** 可以是系統定義或使用者定義的值。 系統會定義下列值:

- 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

預設值為 MQC.MQAT_NO_CONTEXT，表示訊息中沒有環境定義資訊。

public DateTime PutDateTime {get; set;}
放置訊息的時間和日期。

public string ReplyToQueueManagerName {get; set;}
要傳送回覆或報告訊息的佇列管理程式名稱。 預設值為 ""，且佇列管理程式會提供 ReplyToQueueManager **名稱**。

public string ReplyToQueueName {get; set;}
發出訊息取得要求的應用程式向其傳送 MQC.MQMT_REPLY 及 MQC.MQMT_REPORT 訊息的訊息佇列名稱。 預設 ReplyToQueueName 是 ""。

public int Report {get; set;}
使用 **報告** 來指定報告及回覆訊息的相關選項:

- 是否需要報告。
- 應用程式訊息資料是否要併入報告中。
- 如何在報告或回覆中設定訊息及相關性 ID。

可以要求四種報告類型的任何組合:

- 指定四種報告類型的任何組合。 根據是否要將應用程式訊息資料併入報告訊息中，為每一種報告類型選取三個選項中的任何一個。

  1. 收到時確認
     - MQC.MQRO_COA
     - MQC.MQRO_COA_WITH_DATA
     - MQC.MQRO_COA_WITH_FULL_DATA**
  2. 遞送時確認
     - MQC.MQRO_COD
     - MQC.MQRO_COD_WITH_DATA
     - MQC.MQRO_COD_WITH_FULL_DATA**
  3. 異常狀況
     - MQC.MQRO_EXCEPTION
     - MQC.MQRO_EXCEPTION_WITH_DATA
     - MQC.MQRO_EXCEPTION_WITH_FULL_DATA**
  4. 期限
     - MQC.MQRO_EXPIRATION
     - MQC.MQRO_EXPIRATION_WITH_DATA
     - MQC.MQRO_EXPIRATION_WITH_FULL_DATA**

  註: z/OS 佇列管理程式不支援清單中標示為 ** 的值。 如果您的應用程式可能存取 z/OS 佇列管理程式，不論應用程式執行所在的平台為何，請不要使用它們。
- 指定下列其中一項，以控制如何為報告或回覆訊息產生訊息 ID:
  - MQC.MQRO_NEW_MSG_ID
  - MQC.MQRO_PASS_MSG_ID
- 指定下列其中一項，以控制如何設定報告或回覆訊息的相關性 ID:
  - MQC.MQRO_COPY_MSG_ID_TO_CORREL_ID

- MQC.MQRO_PASS_CORREL_ID

- 當原始訊息無法遞送至目的地佇列時，請指定下列其中一項來控制原始訊息的處置:
  - MQC.MQRO_DEAD_LETTER_Q
  - MQC.MQRO_DISCARD_MSG**
- 如果未指定報告選項，則預設值為:

```
MQC.MQRO_NEW_MSG_ID |
MQC.MQRO_COPY_MSG_ID_TO_CORREL_ID |
MQC.MQRO_DEAD_LETTER_Q
```

- 您可以指定下列其中一項或兩項，以要求接收端應用程式傳送正面動作或負面動作報告訊息。
  - MQC.MQRO_PAN
  - MQC.MQRO_NAN

public int TotalMessageLength {get;}
訊息中儲存在從中接收此訊息之訊息佇列上的位元組總數。

public string UserId {get; set;}
UserId 是訊息身分環境定義的一部分。 佇列管理程式通常會提供此值。 如果您有權設定身分環境定義，則可以置換此值。

public int Version {get; set;}
使用中 MQMD 結構的版本。

Read 和 Write 訊息方法

Read 和 Write 方法執行的功能與 .NET System.IO 名稱空間中 BinaryReader 和 BinaryWriter 類別的成員相同。 如需完整語言語法和用法範例，請參閱 MSDN。 從訊息緩衝區中現行位置讀取或寫入的方法。 它們會將現行位置向前移動讀取或寫入的位元組數。

註: 如果訊息資料包含 MQRFH 或 MQRFH2 標頭，您必須使用 ReadBytes 方法來讀取資料。

- 所有方法都會擲出 IOException。
- ReadFully 方法會自動調整目標 byte 或 sbyte 陣列的大小，以完全符合訊息。 也會調整空值陣列的大小。
- Read 方法擲出 EndOfStreamException。
- WriteDecimal 方法擲出 MQException。
- ReadString、 ReadLine 及 WriteString 方法會在 Unicode 與訊息字集之間轉換; 請參閱 CharacterSet。
- 根據 **編碼**的值， Decimal 方法會讀取及寫入以大序排列法、 MQC.MQENC_DECIMAL_NORMAL 或小序排列法 MQC.MQENC_DECIMAL_REVERSE 格式編碼的聚集十進位數。 十進位範圍及對應的 .NET 類型如下:

  Decimal2/short
  -999 至 999

  Decimal4/int
  -9999999 至 9999999

  Decimal8/long
  -9999999999999 至 999999999999999
- Double 和 Float 方法會根據 **編碼**的值，讀取和寫入以 IEE 大序排列法和小序排列法格式 MQC.MQENC_FLOAT_IEEE_NORMAL 和 MQC.MQENC_FLOAT_IEEE_REVERSED 或 S/390 格式 MQC.MQENC_FLOAT_S390 編碼的浮動值。
- Int 方法會根據 **編碼**的值來讀取及寫入以大序排列法、 MQC.MQENC_INTEGER_NORMAL 或小序排列法 MQC.MQENC_INTEGER_REVERSED 格式編碼的整數值。 除了加上不帶正負號的 2 位元組整數類型之外，整數都是帶正負號的。 整數大小以及 .NET 和 WebSphere MQ 類型如下:

  2 位元組
  short, Int2, ushort, UInt2

4 位元組
int, Int4

8 位元組
long, Int8

- WriteObject 會將物件的類別、其非 transient 及非 static 欄位的值，以及其超類型的欄位傳送至訊息緩衝區。
- ReadObject 會從物件的類別、類別的簽章、其非暫時性及非靜態欄位的值，以及其超類型的欄位來建立物件。

表 *608:* 讀取及寫入訊息方法

目標類型	方法簽章

Boolean	`public bool ReadBoolean();`
	
	`public void WriteBoolean(bool value);`

Byte	`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)`

Bytes	`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)`

Decimal2	`public void WriteDecimal2(short value)`
Decimal4	`public void WriteDecimal4(short value)`
Decimal8	`public void WriteDecimal8(short value)`

Double	`public double ReadDouble()`
	
	`public void WriteDouble(double value)`

Float	`public float ReadFloat()`
	
	`public void WriteFloat(float value)`

Int2	`public void WriteInt2(int value)`

表 *608:* 讀取及寫入訊息方法 *(*繼續*)*

目標類型	方法簽章

Int4	`public int readDecimal4()`
	`public int ReadInt()`
	`public int ReadInt4()`
	
	`public void WriteInt(int value)`
	`public void WriteInt4(int value)`

Int8	`public void WriteInt8(long value)`

Long	`public long ReadDecimal8()`
	`public long ReadLong()`
	`public long ReadInt8()`
	
	`public void WriteLong(long value)`

Object	`public Object ReadObject()`
	
	`public void WriteObject(Object object)`

Short	`public short ReadShort()`
	`public short ReadDecimal2()`
	`public short ReadInt2()`
	
	`public void WriteShort(int value)`

string	`public string ReadString(int length)`
	
	`public void WriteString(string string)`

Unsigned Short	`public ushort ReadUnsignedShort()` `public ushort ReadUInt2()`

Unicode	`public string ReadLine()`
	`public char ReadChar()`
	
	`public void WriteChar(int value)`
	`public void WriteChars(string string)`

UTF	`public string ReadUTF()`
	
	`public void WriteUTF(string string)`

緩衝區方法

public void ClearMessage();

擲出 IOException。

捨棄訊息緩衝區中的任何資料，並將資料偏移設回零。

public void ResizeBuffer(int *size*)

擲出 IOException。

MQMessage 物件關於後續取得作業可能需要的緩衝區大小的提示。 如果訊息目前包含訊息資料，且新大小小於現行大小，則會截斷訊息資料。

public void Seek(int *pos*)

擲出 IOException、 ArgumentOutOfRangeException、 ArgumentException。

將游標移至 *pos* 所提供訊息緩衝區中的絕對位置。 後續的讀取和寫入會在緩衝區中的這個位置執行。

public int SkipBytes(int *i*)

擲出 IOException， EndOfStreamException。

在訊息緩衝區中向前移動 n 個位元組，並傳回 n(已跳過的位元組數)。

SkipBytes 方法會封鎖，直到發生下列其中一個事件為止:

- 跳過所有位元組
- 偵測到訊息緩衝區結尾
- 擲出異常狀況

內容方法

public void DeleteProperty(string *name*);

擲出 MQException。

從訊息中刪除具有指定名稱的內容。

name
要刪除的內容名稱。

public System.Collections.IEnumerator GetPropertyNames(string *name*)

擲出 MQException。

傳回符合指定名稱之所有內容名稱的 IEnumerator 。 名稱結尾可以使用百分比符號 '%' 作為萬用字元，以過濾訊息內容，以零或多個字元 (包括句點) 進行比對。

name
要比對的內容名稱。

- 所有 SetProperty 和 GetProperty 方法都會擲出 MQException

表 *609: SetProperty* 和 *GetProperty* 方法

類型	方法簽章

Boolean	`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);`

Byte	`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);`

表 609: *SetProperty* 和 *GetProperty* 方法 (繼續)

類型	方法簽章

Bytes	`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);`

Double	`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);`

Float	`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);`

Int2	`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);`

Int4	`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);`

Int8	`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);`

Long	`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);`

Object	`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);`

表 609: *SetProperty* 和 *GetProperty* 方法 (繼續)	
類型	**方法簽章**

Short	`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);`

string	`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);`

建構子

public MQMessage();
建立具有預設訊息描述子資訊及空訊息緩衝區的 MQMessage 物件。

MQProcess .NET 類別

使用 MQProcess 來查詢 WebSphere MQ 處理程序的屬性。 使用建構子或 MQQueueManager AccessProcess 方法來建立 MQProcess 物件。

類別

```
System.Object
    └ IBM.WMQ.MQBase
        └ IBM.WMQ.MQBaseObject
            └ IBM.WMQ.MQManagedObject
                └ IBM.WMQ.MQProcess
```

public class IBM.WMQ.MQProcess extends IBM.WMQ.MQManagedObject;

- 第 1143 頁的『內容』
- 第 1144 頁的『建構子』

內容

測試在取得內容時是否擲出 MQException 。

public string ApplicationId {get;}

取得識別要啟動之應用程式的字串。 ApplicationId 由觸發監視器應用程式使用。 ApplicationId 會作為觸發訊息的一部分傳送至起始佇列。

預設值為空值。

public int ApplicationType {get;}

識別要由觸發監視器應用程式啟動的處理程序類型。 已定義標準類型，但可以使用其他類型:

- MQAT_AIX
- MQAT_CICS
- MQAT_IMS

- MQAT_MVS
- MQAT_NATIVE
- MQAT_OS400
- MQAT_UNIX
- MQAT_WINDOWS
- MQAT_JAVA
- MQAT_USER_FIRST
- MQAT_USER_LAST

預設值為 MQAT_NATIVE。

public string EnvironmentData {get;}

取得要啟動之應用程式環境的相關資訊。

預設值為空值。

public string UserData {get;}

取得使用者所提供要啟動之應用程式的相關資訊。

預設值為空值。

建構子

public MQProcess(MQQueueManager *queueManager*, string *processName*, int *openOptions*);
public MQProcess(MQQueueManager *qMgr*, string *processName*, int *openOptions*, string *queueManagerName*, string *alternateUserId*);

擲出 MQException。

存取佇列管理程式 *qMgr* 上的 WebSphere MQ 處理程序，以查詢處理程序屬性。

qMgr
要存取的佇列管理程式。

processName
要開啟的處理程序名稱。

openOptions
控制程序開啟的選項。 可以新增或使用位元 OR 運算結合的有效選項如下:

- MQC.MQOO_FAIL_IF_QUIESCING
- MQC.MQOO_INQUIRE
- MQC.MQOO_SET
- MQC.MQOO_ALTERNATE_USER_AUTHORITY

queueManagerName
在其中定義處理程序的佇列管理程式名稱。 如果佇列管理程式與處理程序正在存取的佇列管理程式相同，您可以保留空白或空值佇列管理程式名稱。

alternateUserId

如果在 *openOptions* 參數中指定 MQC.MQOO_ALTERNATE_USER_AUTHORITY，*alternateUserId* 會指定用來檢查動作授權的替代使用者 ID。 如果未指定 MQOO_ALTERNATE_USER_AUTHORITY，則 *alternateUserId* 可以是空白或空值。

如果未指定 MQC.MQOO_ALTERNATE_USER_AUTHORITY，則會使用預設使用者權限來連線至佇列管理程式。

public MQProcess MQQueueManager.AccessProcess(string *processName*, int *openOptions*);
public MQProcess MQQueueManager.AccessProcess(string *processName*, int *openOptions*, string *queueManagerName*, string *alternateUserId*);

擲出 MQException。

存取此佇列管理程式上的 WebSphere MQ 處理程序，以查詢處理程序屬性。

processName
要開啟的處理程序名稱。

openOptions
控制程序開啟的選項。 可以新增或使用位元 OR 運算結合的有效選項如下:

- MQC.MQOO_FAIL_IF_QUIESCING
- MQC.MQOO_INQUIRE
- MQC.MQOO_SET
- MQC.MQOO_ALTERNATE_USER_AUTHORITY

queueManagerName
在其中定義處理程序的佇列管理程式名稱。 如果佇列管理程式與處理程序正在存取的佇列管理程式相同，您可以保留空白或空值佇列管理程式名稱。

alternateUserId

如果在 *openOptions* 參數中指定 MQC.MQOO_ALTERNATE_USER_AUTHORITY，*alternateUserId* 會指定用來檢查動作授權的替代使用者 ID。 如果未指定 MQOO_ALTERNATE_USER_AUTHORITY，則 *alternateUserId* 可以是空白或空值。

如果未指定 MQC.MQOO_ALTERNATE_USER_AUTHORITY，則會使用預設使用者權限來連線至佇列管理程式。

MQPropertyDescriptor .NET 類別

使用 MQPropertyDescriptor 作為 MQMessage GetProperty 和 SetProperty 方法的參數。 MQPropertyDescriptor 說明 MQMessage 內容。

類別

```
System.Object
        └ IBM.WMQ.MQPropertyDescriptor
```

public class IBM.WMQ.MQPropertyDescriptor extends System.Object;

- 第 1145 頁的『內容』
- 第 1146 頁的『建構子』

內容

測試在取得內容時是否擲出 MQException。

public int Context {get; set;}

內容所屬的訊息環境定義。 可能的值為:

MQC.MQPD_NO_CONTEXT
內容未與訊息環境定義相關聯。

MQC.MQPD_USER_CONTEXT
內容與使用者環境定義相關聯。

如果使用者已獲授權，則會在擷取訊息時儲存與使用者環境定義相關聯的內容。 後續參照已儲存環境定義的 Put 方法可以將內容傳遞至新訊息。

public int CopyOptions {get; set;}

CopyOptions 說明可複製內容的訊息類型。

當佇列管理程式接收包含 WebSphere MQ 定義內容的訊息時，如果佇列管理程式將該內容視為不正確，則佇列管理程式會更正 CopyOptions 欄位的值。

可以指定下列選項的任何組合。 透過新增值或使用位元 OR 來結合選項。

MQC.MQCOPY_ALL
此內容會複製到所有類型的後續訊息。

MQC.MQCOPY_FORWARD
內容會複製到正在轉遞的訊息中。

MQC.MQCOPY_PUBLISH
當發佈訊息時，此內容會複製到訂閱者所接收的訊息中。

MQC.MQCOPY_REPLY
內容會複製到回覆訊息中。

MQC.MQCOPY_REPORT
內容會複製到報告訊息中。

MQC.MQCOPY_DEFAULT
該值指出未指定其他複製選項。 內容與後續訊息之間不存在任何關係。 一律會針對訊息描述子內容傳回 MQC.MQCOPY_DEFAULT。

MQC.MQCOPY_NONE
與 MQC.MQCOPY_DEFAULT 相同

public int Options { set; }
選項 預設為 CMQC.MQPD_NONE。 您無法設定任何其他值。

public int Support { get; set; }

設定 **支援**，以指定 WebSphere MQ 定義訊息內容所需的支援層次。 所有其他內容的支援是選用的。 可以指定下列任何值，也可以不指定下列任何值

MQC.MQPD_SUPPORT_OPTIONAL
即使不支援此內容，佇列管理程式也會接受此內容。 可以捨棄此內容，以便訊息流向不支援訊息內容的佇列管理程式。 此值也會指派給未定義 WebSphere MQ 的內容。

MQC.MQPD_SUPPORT_REQUIRED
需要支援內容。 如果您將訊息放到不支援 WebSphere MQ 定義內容的佇列管理程式中，方法會失敗。 它會傳回完成碼 MQC.MQCC_FAILED 和原因碼 MQC.MQRC_UNSUPPORTED_PROPERTY。

MQC.MQPD_SUPPORT_REQUIRED_IF_LOCAL

如果訊息以本端佇列為目的地，則需要支援內容。 如果您將訊息放到不支援 WebSphere MQ 定義內容的佇列管理程式上的本端佇列，方法會失敗。 它會傳回完成碼 MQC.MQCC_FAILED 和原因碼 MQC.MQRC_UNSUPPORTED_PROPERTY。

如果將訊息放置到遠端佇列管理程式，則不會進行檢查。

建構子

PropertyDescriptor();
建立內容描述子。

MQPutMessageOptions .NET 類別

使用 MQPutMessageOptions 來指定如何傳送訊息。 它會修改 MQDestination.Put 的行為。

類別

```
System.Object
    └ IBM.WMQ.MQBase
        └ IBM.WMQ.MQBaseObject
            └ IBM.WMQ.MQPutMessageOptions
```

```
public class IBM.WMQ.MQPutMessageOptions extends IBM.WMQ.MQBaseObject;
```

• 第 1147 頁的『內容』第 1149 頁的『建構子』

內容

測試在取得內容時是否擲出 MQException 。

註: 此類別中部分可用選項的行為取決於使用這些選項的環境。 這些元素會以星號 * 標示。

public MQQueue ContextReference {get; set;}

如果 options 欄位包括 MQC.MQPMO_PASS_IDENTITY_CONTEXT 或 MQC.MQPMO_PASS_ALL_CONTEXT，請設定此欄位以參照從中取得環境定義資訊的 MQQueue 。

此欄位的起始值是空值。

public int InvalidDestCount {get;}*

通常，用於配送清單， InvalidDest **計數** 指出無法傳送至配送清單中佇列的訊息數目。 計數包括無法開啟的佇列，以及已順利開啟但放置作業失敗的佇列。

.NET 不支援配送清單，但在開啟單一佇列時，會設定 InvalidDest **計數** 。

public int KnownDestCount {get;} *

通常用於配送清單， KnownDest **計數** 指出現行呼叫已順利傳送至解析為本端佇列之佇列的訊息數。

.NET 不支援配送清單，但在開啟單一佇列時，會設定 InvalidDest **計數** 。

public int Options {get; set;}

控制 MQDestination.put 和 MQQueueManager.put 動作的選項。 可以指定下列任何值，也可以不指定下列任何值。 如果需要多個選項，可以使用位元 OR 運算子來新增或結合值。

MQC.MQPMO_ASYNC_RESPONSE
此選項會導致使用部分回應資料非同步進行 MQDestination.put 呼叫。

MQC.MQPMO_DEFAULT_CONTEXT
建立預設環境定義與訊息的關聯。

MQC.MQPMO_FAIL_IF_QUIESCING
如果佇列管理程式在靜止中，則會失敗。

MQC.MQPMO_LOGICAL_ORDER*
將訊息群組中的邏輯訊息和區段放入其邏輯順序。

如果您在可重新連接的用戶端中使用 MQPMO_LOGICAL_ORDER 選項，則會將 MQRC_RECONNECT_INCOMPATIBLE 原因碼傳回給應用程式。

MQC.MQPMO_NEW_CORREL_ID*
為每一個傳送的訊息產生新的相關性 ID。

MQC.MQPMO_NEW_MSG_ID*
為每一個已傳送訊息產生新的訊息 ID。

MQC.MQPMO_NONE
未指定選項。 請勿與其他選項搭配使用。

MQC.MQPMO_NO_CONTEXT
沒有任何環境定義與訊息相關聯。

MQC.MQPMO_NO_SYNCPOINT
放置沒有同步點控制的訊息。 如果未指定同步點控制選項，則會假設預設值為無同步點。

MQC.MQPMO_PASS_ALL_CONTEXT
從輸入佇列控點傳遞所有環境定義。

MQC.MQPMO_PASS_IDENTITY_CONTEXT
從輸入佇列控點傳遞身分環境定義。

MQC.MQPMO_RESPONSE_AS_Q_DEF

對於 MQDestination.put 呼叫，此選項會從佇列的 DEFPRESP 屬性取得放置回應類型。

對於 MQQueueManager.put 呼叫，此選項會導致同步進行呼叫。

MQC.MQPMO_RESPONSE_AS_TOPIC_DEF
MQC.MQPMO_RESPONSE_AS_TOPIC_DEF 是 MQC.MQPMO_RESPONSE_AS_Q_DEF 與主題物件搭配使用的同義字。

MQC.MQPMO_RETAIN

佇列管理程式會保留所傳送的發佈資訊。 如果使用此選項且無法保留發佈，則不會發佈訊息，且呼叫會因 MQC.MQRC_PUT_NOT_RETAINED 而失敗。

在發佈此出版品的時間之後，透過呼叫 MQSubscription.RequestPublicationUpdate 方法來要求此出版品的副本。 儲存的發佈會傳送至建立訂閱的應用程式，而不設定 MQC.MQSO_NEW_PUBLICATIONS_ONLY 選項。 當收到發佈資訊時，請檢查該發佈資訊的 MQIsRetained 訊息內容，以找出它是否為保留的發佈資訊。

當訂閱者要求保留的發佈資訊時，所使用的訂閱可能在主題字串中包含萬用字元。 如果主題樹狀結構中有多個符合訂閱的保留發佈，則會全部傳送。

MQC.MQPMO_SET_ALL_CONTEXT
從應用程式設定所有環境定義。

MQC.MQPMO_SET_IDENTITY_CONTEXT
從應用程式設定身分環境定義。

MQC.MQPMO_SYNC_RESPONSE
此選項會使 MQDestination.put 或 MQQueueManager.put 呼叫與完整回應資料同步進行。

MQC.MQPMO_SUPPRESS_REPLYTO
任何填入發佈的 ReplyToQueueName 及 ReplyToQueueManagerName 欄位的資訊都不會傳遞給訂閱者。 如果此選項與需要 ReplyToQueueName 的報告選項一起使用，則呼叫會失敗，並傳回 MQC.MQRC_MISSING_REPLY_TO_Q。

MQC.MQPMO_SYNCPOINT
放置具有同步點控制的訊息。 在確定工作單元之前，在工作單元之外看不到訊息。 如果工作單元已取消，則會刪除訊息。

public int RecordFields {get; set;} *

配送清單的相關資訊。 .NET 中不支援配送清單。

public string ResolvedQueueManagerName {get;}

佇列管理程式設為佇列管理程式名稱的輸出欄位，該佇列管理程式擁有遠端佇列名稱所指定的佇列。 ResolvedQueueManagerName 可能不同於從中存取佇列的佇列管理程式名稱 (如果佇列是遠端佇列的話)。

只有在物件是單一佇列時，才會傳回非空白值。 如果物件是配送清單或主題，則傳回的值未定義。

public string ResolvedQueueName {get;}

由佇列管理程式設定為放置訊息之佇列名稱的輸出欄位。 如果開啟的佇列是別名或模型佇列，則 ResolvedQueue **名稱** 可能不同於用來開啟佇列的名稱。

只有在物件是單一佇列時，才會傳回非空白值。 如果物件是配送清單或主題，則傳回的值未定義。

public int UnknownDestCount {get;} *

通常用於配送清單， UnknownDest **計數** 是佇列管理程式設定的輸出欄位。 它會報告現行呼叫已順利傳送至解析為遠端佇列之佇列的訊息數。

.NET 不支援配送清單，但在開啟單一佇列時，會設定 InvalidDest **計數** 。

建構子

public MQPutMessageOptions();

建構未設定選項的新 MQPutMessageOptions 物件，以及空白 ResolvedQueueName 和 ResolvedQueueManagerName。

MQQueue .NET 類別

使用 MQQueue 來傳送及接收訊息，以及查詢 WebSphere MQ 佇列的屬性。 使用建構子或 MQQueueManager.AccessProcess 方法來建立 MQQueue 物件。

類別

```
System.Object
    └ IBM.WMQ.MQBase
        └ IBM.WMQ.MQBaseObject
            └ IBM.WMQ.MQManagedObject
                └ IBM.WMQ.MQDestination
                    └ IBM.WMQ.MQQueue
```

public class IBM.WMQ.MQQueue extends IBM.WMQ.MQDestination;

- 第 1149 頁的『內容』
- 第 1151 頁的『方法』
- 第 1153 頁的『建構子』

內容

測試在取得內容時是否擲出 MQException 。

public int ClusterWorkLoadPriority {get;}
指定佇列的優先順序。 此參數僅適用於本端、遠端及別名佇列。

public int ClusterWorkLoadRank {get;}
指定佇列的等級。 此參數僅適用於本端、遠端及別名佇列。

public int ClusterWorkLoadUseQ {get;}
指定當目標佇列具有本端實例及至少一個遠端叢集實例時， MQPUT 作業的行為。 如果 MQPUT 源自叢集通道，則此參數不適用。 此參數僅對本端佇列有效。

public DateTime CreationDateTime {get;}
建立此佇列的日期和時間。

public int CurrentDepth {get;}
取得目前在佇列上的訊息數。 在 put 呼叫期間及取消 get 呼叫期間，此值會增加。 在非瀏覽取得期間及取消 put 呼叫期間，它會減少。

public int DefinitionType {get;}
定義佇列的方式。 可能值包括:

- MQC.MQQDT_PREDEFINED
- MQC.MQQDT_PERMANENT_DYNAMIC
- MQC.MQQDT_TEMPORARY_DYNAMIC

public int InhibitGet {get; set;}
控制您是否可以取得此佇列或此主題的訊息。 可能值包括:

- MQC.MQQA_GET_INHIBITED
- MQC.MQQA_GET_ALLOWED

public int InhibitPut {get; set;}
控制您是否可以將訊息放置在此佇列或此主題。 可能值包括:

- MQQA_PUT_INHIBITED
- MQQA_PUT_ALLOWED

public int MaximumDepth {get;}
任何一次可以存在於佇列上的訊息數目上限。 嘗試將訊息放入已包含這許多訊息的佇列失敗，原因碼為 MQC.MQRC_Q_FULL。

public int MaximumMessageLength {get;}
可以存在於此佇列上每一則訊息中的應用程式資料長度上限。 嘗試放置大於此值的訊息失敗，原因碼為 MQC.MQRC_MSG_TOO_BIG_FOR_Q。

public int NonPersistentMessageClass {get;}
置入這個佇列之非持續訊息的可靠性層次。

public int OpenInputCount {get;}
目前適用於從佇列中移除訊息的控點數目。 OpenInput **計數** 是本端佇列管理程式已知的有效輸入控點總數，而不只是應用程式所建立的控點。

public int OpenOutputCount {get;}
目前適用於將訊息新增至佇列的控點數目。 OpenOutput **計數** 是本端佇列管理程式已知的有效輸出控點總數，不只是應用程式所建立的控點。

public int QueueAccounting {get;}
指定您是否可以啟用佇列的帳戶資訊收集。

public int QueueMonitoring {get;}
指定您是否可以啟用佇列的監視。

public int QueueStatistics {get;}
指定是否可以啟用佇列的統計資料收集。

public int QueueType {get;}
此佇列的類型具有下列其中一個值:

- MQC.MQQT_ALIAS
- MQC.MQQT_LOCAL
- MQC.MQQT_REMOTE
- MQC.MQQT_CLUSTER

public int Shareability {get;}
是否可以多次開啟佇列以供輸入。 可能值包括:

- MQC.MQQA_SHAREABLE
- MQC.MQQA_NOT_SHAREABLE

public string TPIPE {get;}
TPIPE 名稱，用於使用 WebSphere MQ IMS 橋接器與 OTMA 進行通訊。

public int TriggerControl {get; set;}
是否將觸發訊息寫入起始佇列，以啟動應用程式來處理佇列。 可能值包括:

- MQC.MQTC_OFF
- MQC.MQTC_ON

public string TriggerData {get; set;}
佇列管理程式插入觸發訊息中的可用格式資料。 當到達此佇列的訊息導致觸發訊息寫入起始佇列時，它會插入 TriggerData 。 字串允許的長度上限由 MQC.MQ_TRIGGER_DATA_LENGTH 提供。

public int TriggerDepth {get; set;}
當觸發類型設為 MQC.MQTT_DEPTH 時，在寫入觸發訊息之前必須在佇列上的訊息數。

public int TriggerMessagePriority {get; set;}
訊息不會參與產生觸發訊息的訊息優先順序。 也就是說，在決定是否產生觸發程式時，佇列管理程式會忽略這些訊息。 值零會導致所有訊息產生觸發訊息。

public int TriggerType {get; set;}
由於訊息到達此佇列而寫入觸發訊息的條件。 可能值包括:

- MQC.MQTT_NONE
- MQC.MQTT_FIRST
- MQC.MQTT_EVERY
- MQC.MQTT_DEPTH

方法

public void Get(MQMessage *message*);
public void Get(MQMessage *message*, MQGetMessageOptions *getMessageOptions*);
public void Get(MQMessage *message*, MQGetMessageOptions *getMessageOptions*, int *MaxMsgSize*);

擲出 MQException。

從佇列取得訊息。

如果取得失敗，則 MQMessage 物件保持不變。 如果成功， MQMessage 的訊息描述子和訊息資料部分會取代為送入訊息中的訊息描述子和訊息資料。

從特定 MQQueueManager 對 WebSphere MQ 的所有呼叫都是同步的。 因此，如果您執行 get with wait ，則在完成 Get 呼叫之前，會封鎖所有其他使用相同 MQQueueManager 的執行緒進一步進行 WebSphere MQ 呼叫。 如果您需要多個執行緒來同步存取 WebSphere MQ ，則每一個執行緒都必須建立自己的 MQQueueManager 物件。

message
包含訊息描述子及傳回的訊息資料。 訊息描述子中的部分欄位是輸入參數。 請務必確定 MessageId 和 CorrelationId 輸入參數已根據需要設定。

可重新連接的用戶端會針對在 MQGM_SYNCPOINT 下收到的訊息，在成功重新連線之後傳回原因碼 MQRC_BACKED_OUT 。

getMessageOptions
控制 get 動作的選項。

從單位元組字元碼轉換為雙位元組碼時，使用選項 MQC.MQGMO_CONVERT 可能會導致異常狀況，原因碼為 MQC.MQRC_CONVERTED_STRING_TOO_BIG 。 在此情況下，會將訊息複製到緩衝區而不進行轉換。

如果未指定 *getMessageOptions* ，則使用的訊息選項為 MQGMO_NOWAIT。

如果您在可重新連接的用戶端中使用 MQGMO_LOGICAL_ORDER 選項，則會傳回 MQRC_RECONNECT_INCOMPATIBLE 原因碼。

MaxMsgSize
此訊息物件要接收的最大訊息。 如果佇列上的訊息大於此大小，則會發生下列兩種情況之一:

- 如果在 MQGetMessageOptions 物件中設定 MQGMO_ACCEPT_TRUNCATED_MSG 旗標，則訊息會盡可能填入訊息資料。 擲出異常狀況，含有 MQCC_WARNING 完成碼和 MQRC_TRUNCATED_MSG_ACCEPTED 原因碼。
- 如果未設定 MQGMO_ACCEPT_TRUNCATED_MSG 旗標，則訊息會留在佇列上。 擲出異常狀況，含有 MQCC_WARNING 完成碼和 MQRC_TRUNCATED_MSG_FAILED 原因碼。

如果未指定 *MaxMsgSize* ，則會擷取整個訊息。

public void Put(MQMessage *message*);
public void Put(MQMessage *message*, MQPutMessageOptions *putMessageOptions*);

擲出 MQException。

將訊息放入佇列。

完成「放置」呼叫之後對 MQMessage 物件的修改不會影響 WebSphere MQ 佇列或發佈主題上的實際訊息。

Put 會更新 MQMessage 物件的 MessageId 和 CorrelationId 內容，且不會清除訊息資料。 進一步的 Put 或 Get 呼叫會參照 MQMessage 物件中的更新資訊。 例如，在下列程式碼 Snippet 中，第一個訊息包含 a 和第二個 ab。

```
msg.WriteString("a");
q.Put(msg,pmo);
msg.WriteString("b");
q.Put(msg,pmo);
```

message
包含訊息描述子資料及要傳送之訊息的 MQMessage 物件。 此方法可以變更訊息描述子。 在此方法完成之後立即在訊息描述子中的值是已放入佇列或發佈至主題的值。

下列原因碼會傳回至可重新連接的用戶端:

- 如果在持續訊息上執行「放置」呼叫時連線中斷，且重新連線成功，則為 MQRC_CALL_INTERRUPTED 。
- MQRC_NONE 如果在對非持續訊息執行 Put 呼叫時連線成功 (請參閱 應用程式回復)。

putMessageOptions
控制 put 動作的選項。

如果未指定 *putMessageOptions* ，則會使用 MQPutMessageOptons 的預設實例。

如果您在可重新連接的用戶端中使用 MQPMO_LOGICAL_ORDER 選項，則會傳回 MQRC_RECONNECT_INCOMPATIBLE 原因碼。

註: 為了簡單和效能，如果您想要將單一訊息放入佇列，請使用 MQQueueManager.Put 物件。 您應該為此具有 MQQueue 物件。

public void PutForwardMessage(MQMessage *message*);
public void PutForwardMessage(MQMessage *message*, MQPutMessageOptions *putMessageOptions*);

擲出 MQException

將正在轉遞的訊息放入佇列，其中 *message* 是原始訊息。

message
包含訊息描述子資料及要傳送之訊息的 MQMessage 物件。 此方法可以變更訊息描述子。 在此方法完成之後立即在訊息描述子中的值是已放入佇列或發佈至主題的值。

下列原因碼會傳回至可重新連接的用戶端:

- 如果在持續訊息上執行「放置」呼叫時連線中斷，且重新連線成功，則為 MQRC_CALL_INTERRUPTED 。
- MQRC_NONE 如果在對非持續訊息執行 Put 呼叫時連線成功 (請參閱 應用程式回復)。

putMessageOptions
控制 put 動作的選項。

如果未指定 *putMessageOptions* ，則會使用 MQPutMessageOptons 的預設實例。

如果您在可重新連接的用戶端中使用 MQPMO_LOGICAL_ORDER 選項，則會傳回 MQRC_RECONNECT_INCOMPATIBLE 原因碼。

public void PutReplyMessage(MQMessage *message*)
public void PutReplyMessage(MQMessage *message*, MQPutMessageOptions *putMessageOptions*)

擲出 MQException。

將回覆訊息放入佇列中，其中 *message* 是原始訊息。

message
包含訊息描述子及傳回的訊息資料。 訊息描述子中的部分欄位是輸入參數。 請務必確定 MessageId 和 CorrelationId 輸入參數已根據需要設定。

可重新連接的用戶端會針對在 MQGM_SYNCPOINT 下收到的訊息，在成功重新連線之後傳回原因碼 MQRC_BACKED_OUT 。

putMessageOptions
控制 put 動作的選項。

如果未指定 *putMessageOptions* ，則會使用 MQPutMessageOptons 的預設實例。

如果您在可重新連接的用戶端中使用 MQPMO_LOGICAL_ORDER 選項，則會傳回 MQRC_RECONNECT_INCOMPATIBLE 原因碼。

public void PutReportMessage(MQMessage *message*)
public void PutReportMessage(MQMessage *message*, MQPutMessageOptions *putMessageOptions*)

擲出 MQException。

將報告訊息放入佇列，其中 *message* 是原始訊息。

message
包含訊息描述子及傳回的訊息資料。 訊息描述子中的部分欄位是輸入參數。 請務必確定 MessageId 和 CorrelationId 輸入參數已根據需要設定。

可重新連接的用戶端會針對在 MQGM_SYNCPOINT 下收到的訊息，在成功重新連線之後傳回原因碼 MQRC_BACKED_OUT 。

putMessageOptions
控制 put 動作的選項。

如果未指定 *putMessageOptions* ，則會使用 MQPutMessageOptons 的預設實例。

如果您在可重新連接的用戶端中使用 MQPMO_LOGICAL_ORDER 選項，則會傳回 MQRC_RECONNECT_INCOMPATIBLE 原因碼。

建構子

public MQQueue MQQueueManager.AccessQueue(string *queueName*, int *openOptions*);
public MQQueue MQQueueManager.AccessQueue(string *queueName*, int *openOptions*, string *queueManagerName*, string *dynamicQueueName*, string *alternateUserId*);

擲出 MQException。

存取此佇列管理程式上的佇列。

您可以取得或瀏覽訊息、放置訊息、查詢佇列的屬性，或設定佇列的屬性。 如果名為的佇列是模型佇列，則會建立動態本端佇列。 查詢結果 MQQueue 物件的 name 屬性，以找出動態佇列的名稱。

queueName
要開啟的佇列名稱。

openOptions
控制佇列開啟的選項。

MQC.MQOO_ALTERNATE_USER_AUTHORITY
使用指定的使用者 ID 進行驗證。

MQC.MQOO_BIND_AS_QDEF
使用佇列的預設連結。

MQC.MQOO_BIND_NOT_FIXED
請勿連結至特定目的地。

MQC.MQOO_BIND_ON_OPEN
開啟佇列時，將控點連結至目的地。

MQC.MQOO_BROWSE
開啟以瀏覽訊息。

MQC.MQOO_FAIL_IF_QUIESCING
如果佇列管理程式在靜止中，則會失敗。

MQC.MQOO_INPUT_AS_Q_DEF
開啟以使用佇列定義的預設值來取得訊息。

MQC.MQOO_INPUT_SHARED
開啟以取得具有共用存取權的訊息。

MQC.MQOO_INPUT_EXCLUSIVE
開啟以取得具有獨佔性存取權的訊息。

MQC.MQOO_INQUIRE
開啟查詢-如果您想要查詢內容，則為必要。

MQC.MQOO_OUTPUT
開啟以放置訊息。

MQC.MQOO_PASS_ALL_CONTEXT
容許傳遞所有環境定義。

MQC.MQOO_PASS_IDENTITY_CONTEXT
容許傳遞身分環境定義。

MQC.MQOO_SAVE_ALL_CONTEXT
擷取訊息時儲存環境定義。

MQC.MQOO_SET
開啟以設定屬性-如果您想要設定內容，則為必要。

MQC.MQOO_SET_ALL_CONTEXT
容許設定所有環境定義。

MQC.MQOO_SET_IDENTITY_CONTEXT
容許設定身分環境定義。

queueManagerName
在其中定義佇列的佇列管理程式名稱。 完全空白或空值的名稱表示 MQQueueManager 物件所連接的佇列管理程式。

dynamicQueueName
除非 queueName 指定模型佇列的名稱，否則會忽略 *dynamicQueueName* 。 如果有，*dynamicQueueName* 會指定要建立的動態佇列名稱。 如果 queueName 指定模型佇列的名稱，則空白或空值名稱無效。 如果名稱中的最後一個非空白字元是星號 *，則佇列管理程式會將星號取代為字元字串。 這些字元保證為佇列產生的名稱在此佇列管理程式上是唯一的。

alternateUserId
如果在 openOptions 參數中指定 MQC.MQOO_ALTERNATE_USER_AUTHORITY，則 *alternateUserId* 會指定用來檢查開啟授權的替代使用者 ID。 如果未指定 MQC.MQOO_ALTERNATE_USER_AUTHORITY，則 *alternateUserId* 可以保留空白或空值。

public MQQueue(MQQueueManager *queueManager*, string *queueName*, int *openOptions*, string *queueManagerName*, string *dynamicQueueName*, string *alternateUserId*);

擲出 MQException。

存取 queueManager 上的佇列。

您可以取得或瀏覽訊息、放置訊息、查詢佇列的屬性，或設定佇列的屬性。 如果名為的佇列是模型佇列，則會建立動態本端佇列。 查詢結果 MQQueue 物件的 name 屬性，以找出動態佇列的名稱。

queueManager
用來存取佇列的佇列管理程式。

queueName
要開啟的佇列名稱。

openOptions
控制佇列開啟的選項。

MQC.MQOO_ALTERNATE_USER_AUTHORITY
使用指定的使用者 ID 進行驗證。

MQC.MQOO_BIND_AS_QDEF
使用佇列的預設連結。

MQC.MQOO_BIND_NOT_FIXED
請勿連結至特定目的地。

MQC.MQOO_BIND_ON_OPEN
開啟佇列時，將控點連結至目的地。

MQC.MQOO_BROWSE
開啟以瀏覽訊息。

MQC.MQOO_FAIL_IF_QUIESCING
如果佇列管理程式在靜止中，則會失敗。

MQC.MQOO_INPUT_AS_Q_DEF
開啟以使用佇列定義的預設值來取得訊息。

MQC.MQOO_INPUT_SHARED
開啟以取得具有共用存取權的訊息。

MQC.MQOO_INPUT_EXCLUSIVE
開啟以取得具有獨佔性存取權的訊息。

MQC.MQOO_INQUIRE
開啟查詢-如果您想要查詢內容，則為必要。

MQC.MQOO_OUTPUT
開啟以放置訊息。

MQC.MQOO_PASS_ALL_CONTEXT
容許傳遞所有環境定義。

MQC.MQOO_PASS_IDENTITY_CONTEXT
容許傳遞身分環境定義。

MQC.MQOO_SAVE_ALL_CONTEXT
擷取訊息時儲存環境定義。

MQC.MQOO_SET
開啟以設定屬性-如果您想要設定內容，則為必要。

MQC.MQOO_SET_ALL_CONTEXT
容許設定所有環境定義。

MQC.MQOO_SET_IDENTITY_CONTEXT
容許設定身分環境定義。

queueManagerName
在其中定義佇列的佇列管理程式名稱。 完全空白或空值的名稱表示 MQQueueManager 物件所連接的佇列管理程式。

dynamicQueueName
除非 queueName 指定模型佇列的名稱，否則會忽略 *dynamicQueueName* 。 如果有，*dynamicQueueName* 會指定要建立的動態佇列名稱。 如果 queueName 指定模型佇列的名稱，則空白或空值名稱無效。 如果名稱中的最後一個非空白字元是星號 *，則佇列管理程式會將星號取代為字元字串。 這些字元保證為佇列產生的名稱在此佇列管理程式上是唯一的。

alternateUserId
如果在 openOptions 參數中指定 MQC.MQOO_ALTERNATE_USER_AUTHORITY，則 *alternateUserId* 會指定用來檢查開啟授權的替代使用者 ID。 如果未指定 MQC.MQOO_ALTERNATE_USER_AUTHORITY，則 *alternateUserId* 可以保留空白或空值。

MQQueueManager .NET 類別

使用「MQQueueManager」來連接佇列管理程式及存取佇列管理程式物件。 它也會控制交易。 MQQueueManager 建構子會建立用戶端或伺服器連線。

類別

```
System.Object
    └ IBM.WMQ.MQBase
        └ IBM.WMQ.MQBaseObject
            └ IBM.WMQ.ManagedObject
                └ IBM.WMQ.MQQueueManager
```

```
public class IBM.WMQ.MQQueueManager extends IBM.WMQ.MQManagedObject;
```


- 第 1156 頁的『內容』
- 第 1159 頁的『方法』
- 第 1164 頁的『建構子』

內容

測試在取得內容時是否擲出 MQException 。

public int AccountingConnOverride {get;}
應用程式是否可以置換 MQI 帳戶及佇列帳戶值的設定。

public int AccountingInterval {get;}
寫入中間統計記錄之前的時間 (以秒為單位)。

public int ActivityRecording {get;}
控制活動報告的產生。

public int AdoptNewMCACheck {get;}
指定要檢查哪些元素，以決定在偵測到新的入埠通道時是否採用 MCA。 若要採用， MCA 名稱必須符合作用中 MCA 的名稱。

public int AdoptNewMCAInterval {get;}
新通道等待孤立通道結束的時間量 (秒)。

public int AdoptNewMCAType {get;}
當偵測到符合 AdoptNewMCACheck 值的新入埠通道要求時，是否要採用 (重新啟動) 孤立 MCA 實例。

public int BridgeEvent {get;}
是否產生 IMS Bridge 事件。

public int ChannelEvent {get;}
是否產生頻道事件。

public int ChannelInitiatorControl {get;}
當佇列管理程式啟動時，通道起始程式是否自動啟動。

public int ChannelInitiatorAdapters {get;}
要處理 WebSphere MQ 呼叫的配接器子作業數目。

public int ChannelInitiatorDispatchers {get;}
用於通道起始程式的分派器數目。

public int ChannelInitiatorTraceAutoStart {get;}
指定是否自動啟動通道起始程式追蹤。

public int ChannelInitiatorTraceTableSize {get;}
通道 起始程式的追蹤資料空間大小 (MB)。

public int ChannelMonitoring {get;}
是否使用通道監視。

public int ChannelStatistics {get;}
控制通道統計資料的收集。

public int CharacterSet {get;}
傳回佇列管理程式的編碼字集 ID (CCSID)。 CharacterSet 由佇列管理程式用於應用程式設計介面中的所有字串欄位。

public int ClusterSenderMonitoring {get;}
控制自動定義叢集傳送端通道的線上監視資料收集。

public int ClusterSenderStatistics {get;}
控制自動定義叢集傳送端通道的統計資料收集。

public int ClusterWorkLoadMRU {get;}
出埠叢集通道的數目上限。

public int ClusterWorkLoadUseQ {get;}
MQQueue 內容的預設值 ClusterWorkLoadUseQ(如果它指定值 QMGR)。

public int CommandEvent {get;}
指定是否產生指令事件。

public string CommandInputQueueName {get;}
傳回在佇列管理程式上定義的指令輸入佇列名稱。 應用程式可以將指令傳送至此佇列 (如果已獲授權這麼做的話)。

public int CommandLevel {get;}
指出佇列管理程式的功能層次。 對應於特定函數層次的函數集取決於平台。 在特定平台上，您可以依賴每個佇列管理程式來支援所有佇列管理程式共用的最低功能層次的功能。

public int CommandLevel {get;}
當佇列管理程式啟動時是否自動啟動指令伺服器。

public string DNSGroup {get;}
處理佇列共用群組之入埠傳輸的 TCP 接聽器必須結合的群組名稱。 當使用 Workload Manager for Dynamic Domain Name Services 支援 (DDNS) 時，它會加入此群組。

public int DNSWLM {get;}
處理佇列共用群組之入埠傳輸的 TCP 接聽器是否必須向「工作量管理程式」登錄 DDNS。

public int IPAddressVersion {get;}
要用於通道連線的 IP 通訊協定 (IPv4 或 IPv6)。

public boolean IsConnected {get;}
傳回 isConnected 的值。

如果為 true，則已建立與佇列管理程式的連線，且不知道是否已中斷。 對 IsConnected 的任何呼叫都不會主動嘗試呼叫佇列管理程式，因此實體連線功能可能中斷，但 IsConnected 仍可能傳回 true。 只有在佇列管理程式上執行活動 (例如，放置訊息、取得訊息) 時，才會更新 IsConnected 狀態。

如果為 false，則表示尚未建立佇列管理程式的連線，或已中斷連線，或已中斷連線。

public int KeepAlive {get;}
指定是否要使用 TCP KEEPALIVE 機能來檢查連線的另一端是否仍然可用。 如果無法使用，則會關閉通道。

public int ListenerTimer {get;}
在 APPC 或 TCP/IP 失敗之後， WebSphere MQ 嘗試重新啟動接聽器的時間間隔 (以秒為單位)。

public int LoggerEvent {get;}
是否產生日誌程式事件。

public string LU62ARMSuffix {get;}
SYS1.PARMLIB。 這個字尾代表這個通道起始程式的 LUADD。 當自動重新啟動管理程式 (ARM) 重新啟動通道起始程式時，會發出 z/OS 指令 SET APPC=xx。

public string LUGroupName {get; z/os}
處理佇列共用群組之入埠傳輸的 LU 6.2 接聽器要使用的一般 LU 名稱。

public string LUName {get;}
用於出埠 LU 6.2 傳輸的 LU 名稱。

public int MaximumActiveChannels {get;}
任何時間都會處於作用中狀態的通道數目上限。

public int MaximumCurrentChannels {get;}
可隨時保持現行狀態的通道數上限 (包括具有已連接用戶端的伺服器連線通道)。

public int MaximumLU62Channels {get;}
使用 LU 6.2 傳輸通訊協定的現行或可連接的通道數上限。

public int MaximumMessageLength {get;}
傳回佇列管理程式可處理的訊息長度上限 (以位元組為單位)。 任何佇列都無法定義大於 MaximumMessage **長度**的訊息長度上限。

public int MaximumPriority {get;}
傳回佇列管理程式支援的訊息優先順序上限。 優先順序範圍從零 (最低) 到這個值。 如果您在中斷佇列管理程式的連線之後呼叫此方法，則會擲出 MQException 。

public int MaximumTCPChannels {get;}
使用 TCP/IP 傳輸通訊協定的現行或可連接的用戶端通道數上限。

public int MQIAccounting {get;}
控制 MQI 資料的帳戶資訊收集。

public int MQIStatistics {get;}
控制收集佇列管理程式的統計資料監視資訊。

public int OutboundPortMax {get;}
連結送出通道時要使用的埠號範圍內的最大值。

public int OutboundPortMin {get;}
連結送出通道時要使用的埠號範圍最小值。

public int QueueAccounting {get;}
類別 3 帳戶 (執行緒層次和佇列層次帳戶) 資料是否要用於所有佇列。

public int QueueMonitoring {get;}
控制收集佇列的線上監視資料。

public int QueueStatistics {get;}
控制收集佇列的統計資料。

public int ReceiveTimeout {get;}
在回到非作用中狀態之前， TCP/IP 通道等待從其友機接收資料 (包括活動訊號) 的時間長度。

public int ReceiveTimeoutMin {get;}
在回到非作用中狀態之前， TCP/IP 通道等待從其友機接收資料 (包括活動訊號) 的時間長度下限。

public int ReceiveTimeoutType {get;}
套用至 ReceiveTimeout 中的值的限定元。

public int SharedQueueQueueManagerName {get;}
指定如何將訊息遞送至共用佇列。 如果 put 指定來自相同佇列共用群組的不同佇列管理程式作為目標佇列管理程式，則會以兩種方式遞送訊息:

MQC.MQSQQM_USE
訊息在放入共用佇列之前，會先遞送至物件佇列管理程式。

MQCMQSQQM_IGNORE
訊息會直接放置在共用佇列上。

public int SSLEvent {get;}
是否產生 SSL 事件。

public int SSLFips {get;}
如果在 WebSphere MQ 中執行加密法，而不是在加密硬體中執行，是否只使用 FIPS 認證的演算法。

public int SSLKeyResetCount {get;}
指出在重新協議秘密金鑰之前，在 SSL 交談內傳送及接收的未加密位元組數。

public int ClusterSenderStatistics {get;}
指定連續聚集統計資料之間的間隔 (分鐘)。

public int SyncpointAvailability {get;}
指出佇列管理程式是否支援使用 MQQueue.get 及 MQQueue.put 方法的工作單元及同步點。

public string TCPName {get;}
要使用的唯一或預設 TCP/IP 系統名稱，視 TCPStackType 的值而定。

public int TCPStackType {get;}
指定通道起始程式是否只使用 TCPName 中指定的 TCP/IP 位址空間。 或者，通道起始程式可以連結至任何 TCP/IP 位址。

public int TraceRouteRecording {get;}
控制路徑追蹤資訊的記錄。

方法

public MQProcess AccessProcess(string *processName*, int *openOptions*);
public MQProcess AccessProcess(string *processName*, int *openOptions*, string *queueManagerName*, string *alternateUserId*);

擲出 MQException。

存取此佇列管理程式上的 WebSphere MQ 處理程序，以查詢處理程序屬性。

processName
要開啟的處理程序名稱。

openOptions
控制程序開啟的選項。 可以新增或使用位元 OR 運算結合的有效選項如下:

- MQC.MQOO_FAIL_IF_QUIESCING
- MQC.MQOO_INQUIRE
- MQC.MQOO_SET
- MQC.MQOO_ALTERNATE_USER_AUTHORITY

queueManagerName
在其中定義處理程序的佇列管理程式名稱。 如果佇列管理程式與處理程序正在存取的佇列管理程式相同，您可以保留空白或空值佇列管理程式名稱。

alternateUserId

如果在 *openOptions* 參數中指定 MQC.MQOO_ALTERNATE_USER_AUTHORITY，*alternateUserId* 會指定用來檢查動作授權的替代使用者 ID。 如果未指定 MQOO_ALTERNATE_USER_AUTHORITY，則 *alternateUserId* 可以是空白或空值。

如果未指定 MQC.MQOO_ALTERNATE_USER_AUTHORITY，則會使用預設使用者權限來連線至佇列管理程式。

public MQQueue AccessQueue(string *queueName*, int *openOptions*);
public MQQueue AccessQueue(string *queueName*, int *openOptions*, string *queueManagerName*, string *dynamicQueueName*, string *alternateUserId*);

擲出 MQException。

存取此佇列管理程式上的佇列。

您可以取得或瀏覽訊息、放置訊息、查詢佇列的屬性，或設定佇列的屬性。 如果名為的佇列是模型佇列，則會建立動態本端佇列。 查詢結果 MQQueue 物件的 name 屬性，以找出動態佇列的名稱。

queueName
要開啟的佇列名稱。

openOptions
控制佇列開啟的選項。

MQC.MQOO_ALTERNATE_USER_AUTHORITY
使用指定的使用者 ID 進行驗證。

MQC.MQOO_BIND_AS_QDEF
使用佇列的預設連結。

MQC.MQOO_BIND_NOT_FIXED
請勿連結至特定目的地。

MQC.MQOO_BIND_ON_OPEN
開啟佇列時，將控點連結至目的地。

MQC.MQOO_BROWSE
開啟以瀏覽訊息。

MQC.MQOO_FAIL_IF_QUIESCING
如果佇列管理程式在靜止中，則會失敗。

MQC.MQOO_INPUT_AS_Q_DEF
開啟以使用佇列定義的預設值來取得訊息。

MQC.MQOO_INPUT_SHARED
開啟以取得具有共用存取權的訊息。

MQC.MQOO_INPUT_EXCLUSIVE
開啟以取得具有獨佔性存取權的訊息。

MQC.MQOO_INQUIRE
開啟查詢-如果您想要查詢內容，則為必要。

MQC.MQOO_OUTPUT
開啟以放置訊息。

MQC.MQOO_PASS_ALL_CONTEXT
容許傳遞所有環境定義。

MQC.MQOO_PASS_IDENTITY_CONTEXT
容許傳遞身分環境定義。

MQC.MQOO_SAVE_ALL_CONTEXT
擷取訊息時儲存環境定義。

MQC.MQOO_SET
開啟以設定屬性-如果您想要設定內容，則為必要。

MQC.MQOO_SET_ALL_CONTEXT
容許設定所有環境定義。

MQC.MQOO_SET_IDENTITY_CONTEXT
容許設定身分環境定義。

queueManagerName
在其中定義佇列的佇列管理程式名稱。 完全空白或空值的名稱表示 MQQueueManager 物件所連接的佇列管理程式。

dynamicQueueName
除非 queueName 指定模型佇列的名稱，否則會忽略 *dynamicQueueName* 。 如果有，*dynamicQueueName* 會指定要建立的動態佇列名稱。 如果 queueName 指定模型佇列的名稱，則空白或空值名稱無效。 如果名稱中的最後一個非空白字元是星號 *，則佇列管理程式會將星號取代為字元字串。 這些字元保證為佇列產生的名稱在此佇列管理程式上是唯一的。

alternateUserId
如果在 openOptions 參數中指定 MQC.MQOO_ALTERNATE_USER_AUTHORITY，則 *alternateUserId* 會指定用來檢查開啟授權的替代使用者 ID。 如果未指定 MQC.MQOO_ALTERNATE_USER_AUTHORITY，則 *alternateUserId* 可以保留空白或空值。

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*);

存取此佇列管理程式上的主題。

MQTopic 物件與管理主題物件密切相關，管理主題物件有時稱為主題物件。 在輸入上， topicObject 指向管理主題物件。 MQTopic 建構子會從主題物件取得主題字串，並將它與 topicName 結合，以建立主題名稱。 topicObject 或 topicName 可以是空值。 主題名稱與主題樹狀結構相符，並在 topicObject 中傳回最相符管理主題物件的名稱。

與 MQTopic 物件相關聯的主題是結合兩個主題字串的結果。 第一個主題字串是由 *topicObject* 所識別的管理主題物件所定義。 第二個主題字串是 *topicString*。 與 MQTopic 物件相關聯的結果主題字串可以透過包括萬用字元來識別多個主題。

視是否開啟主題來發佈或訂閱而定，您可以使用 MQTopic.Put 方法來發佈主題，或使用 MQTopic.Get 方法來接收主題的發佈。 如果您想要發佈及訂閱相同的主題，則必須存取該主題兩次，一次用於發佈，一次用於訂閱。

如果您為訂閱建立 MQTopic 物件，但未提供 MQDestination 物件，則會採用受管理訂閱。 如果您傳遞佇列作為 MQDestination 物件，則會採用未受管理的訂閱。 您必須確定您設定的訂閱選項與受管理或未受管理的訂閱一致。

destination
destination 是 MQQueue 實例。 藉由提供 *destination*， MQTopic 會以未受管理的訂閱方式開啟。 主題上的發佈會遞送至以 *destination* 身分存取的佇列。

topicName
主題名稱第二部分的主題字串。 *topicName* 與 *topicObject* 管理主題物件中定義的主題字串連結。 您可以將 *topicName* 設為空值，在此情況下，主題名稱由 *topicObject* 中的主題字串所定義。

topicObject
在輸入上， *topicObject* 是主題物件的名稱，包含形成主題名稱第一部分的主題字串。 *topicObject* 中的主題字串會與 *topicName* 連結。 主題名稱的建構規則定義在 結合主題字串 中。

在輸出上， *topicObject* 包含管理主題物件的名稱，該管理主題物件在主題樹狀結構中與主題名稱所識別的主題最相符。

openAs
存取主題以發佈或訂閱。 參數只能包含下列其中一個選項:

- MQC.MQTOPIC_OPEN_AS_SUBSCRIPTION
- MQC.MQTOPIC_OPEN_AS_PUBLICATION

options

結合控制開啟發佈或訂閱主題的選項。 使用 MQC.MQSO_* 常數來存取訂閱的主題，並使用 MQC.MQOO_* 常數來存取發佈的主題。

如果需要多個選項，請將值新增在一起，或使用位元 OR 運算子來結合選項值。

alternateUserId

指定替代使用者 ID，用來檢查完成作業所需的授權。 如果在 options 參數中設定 MQC.MQOO_ALTERNATE_USER_AUTHORITY 或 MQC.MQSO_ALTERNATE_USER_AUTHORITY，則必須指定 *alternateUserId*。

subscriptionName

如果提供選項 MQC.MQSO_DURABLE 或 MQC.MQSO_ALTER，則需要 *subscriptionName*。 在這兩種情況下，都會隱含地開啟 MQTopic 進行訂閱。 如果已設定 MQC.MQSO_DURABLE，且訂閱存在，或如果已設定 MQC.MQSO_ALTER，且訂閱不存在，則會擲出異常狀況。

properties

設定使用雜湊表列出的任何特殊訂閱內容。 雜湊表中的指定項目會以輸出值更新。 項目不會新增至雜湊表以報告輸出值。

- 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();

擲出 MQException

傳回 MQAsyncStatus 物件，代表佇列管理程式連線的非同步活動。

public void Backout();

擲出 MQException。

取消自前次同步點以來在同步點內讀取或寫入的任何訊息。

使用 MQC.MQPMO_SYNCPOINT 旗標集寫入的訊息會從佇列中移除。 使用 MQC.MQGMO_SYNCPOINT 旗標讀取的訊息會在它們來自的佇列上恢復。 如果訊息持續存在，則會記載變更。

對於可重新連接的用戶端，在重新連線成功之後，會將 MQRC_NONE 原因碼傳回給用戶端。

public void Begin();

擲出 MQException。

Begin 僅在伺服器連結模式中受支援。 它會啟動廣域工作單元。

public void Commit();

擲出 MQException。

確定自前次同步點以來在同步點內讀取或寫入的任何訊息。

以 MQC.MQPMO_SYNCPOINT 旗標集撰寫的訊息可供其他應用程式使用。 使用 MQC.MQGMO_SYNCPOINT 旗標集擷取的訊息會被刪除。 如果訊息持續存在，則會記載變更。

下列原因碼會傳回至可重新連接的用戶端:

- 如果在執行確定呼叫時失去連線，則為 MQRC_CALL_INTERRUPTED。
- 如果在重新連線之後發出確定呼叫，則為 MQRC_BACKED_OUT。

Disconnect();

擲出 MQException。

關閉佇列管理程式的連線。 此應用程式無法再存取在此佇列管理程式上存取的所有物件。 若要重新存取物件，請建立 MQQueueManager 物件。

通常，會確定作為工作單元一部分執行的任何工作。 不過，如果工作單元是由 .NET 管理，則可能會回復工作單元。

```
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);
```

擲出 MQException。

將單一訊息放在佇列或主題上，而不先建立 MQQueue 或 MQTopic 物件。

queueName
要放置訊息的佇列名稱。

destinationName
目的地物件的名稱。 視 *type* 的值而定，它是佇列或主題。

type
目的地物件的類型。 您不得結合選項。

MQC.MQOT_Q
佇列

MQC.MQOT_TOPIC
主題

queueManagerName

在其中定義佇列的佇列管理程式或佇列管理程式別名的名稱。 如果指定類型 MQC.MQOT_TOPIC，則會忽略此參數。

如果佇列是模型佇列，且解析的佇列管理程式名稱不是此佇列管理程式，則會擲出 MQException。

topicString

topicString 與 *destinationName* 主題物件中的主題名稱結合。

如果 *destinationName* 是佇列，則會忽略 *topicString*。

message
要傳送的訊息。 訊息是輸入/輸出物件。

下列原因碼會傳回至可重新連接的用戶端:

- MQRC_CALL_INTERRUPTED 如果在對持續訊息執行「放置」呼叫時連線中斷。
- MQRC_NONE，如果在對非持續訊息執行 Put 呼叫時連線成功 (請參閱 應用程式回復)。

putMessageOptions

控制放置動作的選項。

如果您省略 *putMessageOptions*，則會建立 *putMessageOptions* 的預設實例。 *putMessageOptions* 是輸入/輸出物件。

如果您在可重新連接的用戶端中使用 MQPMO_LOGICAL_ORDER 選項，則會傳回 MQRC_RECONNECT_INCOMPATIBLE 原因碼。

alternateUserId

指定在佇列上放置訊息時，用來檢查授權的替代使用者 ID。

如果您未在 *putMessageOptions* 中設定 MQC.MQOO_ALTERNATE_USER_AUTHORITY，則可以省略 *alternateUserId*。 如果您設定 MQC.MQOO_ALTERNATE_USER_AUTHORITY，也必須設定 *alternateUserId*。 除非您同時設定 MQC.MQOO_ALTERNATE_USER_AUTHORITY，否則 *alternateUserId* 沒有作用。

建構子

```
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);
```

擲出 MQException。

建立與佇列管理程式的連線。 在建立用戶端連線或伺服器連線之間選取。

嘗試連接至佇列管理程式時，您必須對佇列管理程式具有查詢 (inq) 權限。 如果沒有查詢權限，連線嘗試會失敗。

如果符合下列其中一個條件，則會建立用戶端連線:

1. *channel* 或 *connName* 指定在建構子中。
2. *HostName*、 *Port* 或 *Channel* 指定在 *properties* 中。
3. 已指定 *MQEnvironment.HostName*、 *MQEnvironment.Port* 或 *MQEnvironment.Channel*。

連線內容的值會依顯示的順序預設。 建構子中的 *channel* 和 *connName* 優先於建構子中的內容值。 建構子內容值優先採用 MQEnvironment 內容。

主機名稱、通道名稱及埠定義在 MQEnvironment 類別中。

queueManagerName
要連接的佇列管理程式或佇列管理程式群組的名稱。

省略參數，或將其保留為空值，或保留空白以選取預設佇列管理程式。 伺服器上的預設佇列管理程式連線是指向伺服器上的預設佇列管理程式。 用戶端連線上的預設佇列管理程式連線是指向接聽器所連接的佇列管理程式。

options
指定 MQCNO 連線選項。 這些值必須適用於所建立的連線類型。 比方說，如果您指定用戶端連線的下列伺服器連線內容，則會擲出 MQException。

- MQC.MQCNO_FASTPATH_BINDING
- MQC.MQCNO_STANDARD_BINDING

properties

properties 參數採用一系列鍵/值配對，這些鍵/值配對會置換 MQEnvironment 所設定的內容; 請參閱範例 第 1166 頁的『置換 MQEnvironment 內容』。 可以置換下列內容:

- 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_CRYPTO_HARDWARE_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

channel
伺服器連線通道的名稱

connName
格式為 *HostName* (埠)的連線名稱。

您可以使用 CONNECTION_NAME_PROPERTY，提供 主機名稱 及 埠 的清單作為建構子 MQQueueManager(String queueManagerName, Hashtable properties) 的引數。

例如:

```
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);
```

嘗試建立連線時，會依序處理連線名稱清單。 如果第一個主機名稱和埠的連線嘗試失敗，則會嘗試第二個屬性配對的連線。 用戶端會重複此處理程序，直到成功建立連線或清單耗盡為止。 如果清單已耗盡，則會傳回適當的原因碼及完成碼給用戶端應用程式。

當未提供埠號給連線名稱時，會使用預設埠 (在 mqclient.ini 中配置)。

設定連線清單

當設定自動用戶端重新連線選項時，您可以使用下列方法來設定連線清單:

透過 MQSERVER 設定連線清單
您可以透過命令提示字元來設定連線清單。

在命令提示字元中，設定

```
MQSERVER=SYSTEM.DEF.SVRCONN/TCP/Hostname1(Port1),Hostname2(Por2),Hostname3(Port3)
For Example:
MQSERVER=SYSTEM.DEF.SVRCONN/TCP/fred.mq.com(5266),nick.mq.com(6566),jack.mq.com(8413)
```

如果您在 MQSERVER 中設定連線，請不要在應用程式中設定它。

如果您在應用程式中設定連線清單，應用程式會改寫 MQSERVER 環境變數中所設定的任何內容。

透過應用程式設定連線清單

您可以指定主機名稱和埠內容，在應用程式中設定連線清單。

```
String connName = "fred.mq.com(2344), nick.mq.com(3746), chris.mq.com(4288)";
MQQueueManager qm = new MQQueueManager("QM1", "TestChannel", connName);
```

透過 app.config 設定連線清單

App.config 是您在其中指定鍵值組的 XML 檔案。

在連線清單中指定

```
<app.Settings>
<add key="Connection1" value="Hostname1(Port1)"/>
<add key="Connection2" value="Hostname2(Port2)"/>
<app.Settings>
```

例如：

```
<app.Settings>
<add key>="Connection1" value="fred.mq.com(2966)"/>
<add key>="Connection2" value="alex.mq.com(6533)"/>
<app.Settings>
```

您可以直接變更 app.config 檔案中的連線清單。

透過 MQEnvironment 設定連線清單

如果要透過 MQEnvironment 來設定連線清單，請使用 *ConnectionName* 內容。

```
MQEnvironment.ConnectionName = "fred.mq.com(4288),"alex.mq.com(5211);
```

ConnectionName 內容會改寫 MQEnvironment 中設定的主機名稱和埠內容。

建立用戶端連線

下列範例顯示如何建立與佇列管理程式的用戶端連線。 在建立新的 MQQueueManager 物件之前，您可以先設定 MQEnvironment 變數來建立用戶端連線。

```
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 WebSphere MQ port)
MQEnvironment.Channel  = "channel.name"; // the case sensitive
                                         //  name of the
                                         // SVR CONN channel on
                                         // the queue manager
MQQueueManager qMgr    = new MQQueueManager("MYQM");
```

圖 *43:* 用戶端連線

置換 MQEnvironment 內容

下列範例顯示如何建立佇列管理程式，並在雜湊表中定義其使用者 ID 和密碼。

```
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);
}
```

圖 *44:* 置換 *MQEnvironment* 內容

建立可重新連接的連線

下列範例顯示如何自動將用戶端重新連接至佇列管理程式。

```
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
                                                    // 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 reconnect
happens

MQ QueueManager qmgr = new MQQueueManager("qmgrname", properties);
```

圖 *45:* 自動將用戶端重新連接至佇列管理程式

MQSubscription .NET 類別

使用 MQSubscription 來要求將保留的發佈資訊傳送給訂閱者。 MQSubscription 是針對訂閱開啟之 MQTopic 物件的內容。

類別

```
System.Object
    └ IBM.WMQ.MQBase
        └ IBM.WMQ.MQBaseObject
            └ IBM.WMQ.MQManagedObject
                └ IBM.WMQ.MQSubscription
```

```
public class IBM.WMQ.MQSubscription extends IBM.WMQ.MQManagedObject;
```

- 第 1167 頁的『內容』
- 第 1167 頁的『方法』
- 第 1168 頁的『建構子』

內容

使用 MQManagedObject 類別來存取訂閱內容; 請參閱 第 1131 頁的『內容』。

方法

使用 MQManagedObject 類別存取訂閱 Inquire、 Set 及 Get 方法; 請參閱 第 1131 頁的『方法』。

public int RequestPublicationUpdate(int *options*);

擲出 MQException。

要求現行主題的更新發佈資訊。 如果佇列管理程式具有主題的保留發佈資訊，則會將它們傳送至訂閱者。

在呼叫 RequestPublicationUpdate 之前，請開啟訂閱的主題，以取得 MQSubscription 物件。

通常，使用 MQC.MQSO_PUBLICATIONS_ON_REQUEST 選項開啟訂閱。 如果主題字串中沒有萬用字元，則此呼叫只會傳送一個發佈。 如果主題字串包含萬用字元，則可能會傳送許多出版品。 此方法會傳回傳送至訂閱佇列的保留發佈數。 不保證會收到這麼多出版品，特別是當它們是非持續訊息時。

options

MQC.MQSRO_FAIL_IF_QUIESCING
如果佇列管理程式處於靜止狀態，則此方法會失敗。 在 z/OS 上，對於 CICS 或 IMS 應用程式，MQC.MQSRO_FAIL_IF_QUIESCING 也會在連線處於靜止狀態時強制方法失敗。

MQC.MQSRO_NONE
未指定任何選項。

建構子

無 Public 建構子。

在為訂閱開啟的 MQTopic 物件的 SubscriptionReference 內容中傳回 MQSubscription 物件。

呼叫 RequestPublicationUpdate 方法。 MQSubscription 是 MQManagedObject 的子類別。 使用參照來存取 MQManagedObject 的內容和方法。

MQTopic .NET 類別

使用 MQTopic 來發佈或訂閱主題的訊息，或查詢或設定主題的屬性。 使用建構子或 MQQueueManager.AccessTopic 方法來建立 MQTopic 物件，以進行發佈或訂閱。

類別

```
System.Object
    └ IBM.WMQ.MQBase
        └ IBM.WMQ.MQBaseObject
            └ IBM.WMQ.MQManagedObject
                └ IBM.WMQ.MQDestination
                    └ IBM.WMQ.MQTopic
```

```
public class IBM.WMQ.MQTopic extends IBM.WMQ.MQDestination;
```


- 第 1168 頁的『內容』
- 第 1169 頁的『方法』
- 第 1170 頁的『建構子』

內容

測試在取得內容時是否擲出 MQException 。

public Boolean IsDurable {get;}
唯讀內容，如果訂閱是可延續的，則會傳回 True ， 否則會傳回 False 。 如果已開啟主題進行發佈，則會忽略該內容，且一律會傳回 False。

public Boolean IsManaged {get;};
如果訂閱由佇列管理程式管理，則傳回 True 的唯讀內容，否則會傳回 False 。 如果已開啟主題進行發佈，則會忽略該內容，且一律會傳回 False。

public Boolean IsSubscribed {get;};
唯讀內容，如果已開啟主題進行訂閱，則會傳回 True ; 如果已開啟主題進行發佈，則會傳回 False 。

public MQSubscription SubscriptionReference {get;};
唯讀內容，傳回與針對訂閱開啟的主題物件相關聯的 MQSubscription 物件。 如果您想要修改關閉選項或啟動任何物件方法，則可以使用此參照。

public MQDestination UnmanagedDestinationReference {get;};
唯讀內容，傳回與未受管理的訂閱相關聯的 MQQueue 。 它是建立主題物件時指定的目的地。 對於開啟以進行發佈或具有受管理訂閱的任何主題物件，此內容會傳回空值。

方法

public void Put(MQMessage *message*);
public void Put(MQMessage *message*, MQPutMessageOptions *putMessageOptions*);
擲出 MQException。

將訊息發佈至主題。

完成「放置」呼叫之後對 MQMessage 物件的修改不會影響 WebSphere MQ 佇列或發佈主題上的實際訊息。

Put 會更新 MQMessage 物件的 MessageId 和 CorrelationId 內容，且不會清除訊息資料。 進一步的 Put 或 Get 呼叫會參照 MQMessage 物件中的更新資訊。 例如，在下列程式碼 Snippet 中，第一個訊息包含 a 和第二個 ab。

```
msg.WriteString("a");
q.Put(msg,pmo);
msg.WriteString("b");
q.Put(msg,pmo);
```

message
包含訊息描述子資料及要傳送之訊息的 MQMessage 物件。 此方法可以變更訊息描述子。 在此方法完成之後立即在訊息描述子中的值是已放入佇列或發佈至主題的值。

下列原因碼會傳回至可重新連接的用戶端:

- 如果在持續訊息上執行「放置」呼叫時連線中斷，且重新連線成功，則為 MQRC_CALL_INTERRUPTED 。
- MQRC_NONE 如果在對非持續訊息執行 Put 呼叫時連線成功 (請參閱 應用程式回復)。

putMessageOptions
控制 put 動作的選項。

如果未指定 *putMessageOptions* ，則會使用 MQPutMessageOptons 的預設實例。

如果您在可重新連接的用戶端中使用 MQPMO_LOGICAL_ORDER 選項，則會傳回 MQRC_RECONNECT_INCOMPATIBLE 原因碼。

註: 為了簡單和效能，如果您想要將單一訊息放入佇列，請使用 MQQueueManager.Put 物件。 您應該為此具有 MQQueue 物件。

public void Get(MQMessage *message*);
public void Get(MQMessage *message*, MQGetMessageOptions *getMessageOptions*);
public void Get(MQMessage *message*, MQGetMessageOptions *getMessageOptions*, int *MaxMsgSize*);
擲出 MQException。

從主題擷取訊息。

此方法使用預設 MQGetMessageOptions 實例來執行取得。 使用的訊息選項為 MQGMO_NOWAIT。

如果取得失敗，則 MQMessage 物件保持不變。 如果成功， MQMessage 的訊息描述子和訊息資料部分會取代為送入訊息中的訊息描述子和訊息資料。

從特定 MQQueueManager 對 WebSphere MQ 的所有呼叫都是同步的。 因此，如果您執行 get with wait ，則在完成 Get 呼叫之前，會封鎖所有其他使用相同 MQQueueManager 的執行緒進一步進行 WebSphere MQ 呼叫。 如果您需要多個執行緒來同步存取 WebSphere MQ ，則每一個執行緒都必須建立自己的 MQQueueManager 物件。

message
包含訊息描述子及傳回的訊息資料。 訊息描述子中的部分欄位是輸入參數。 請務必確定 MessageId 和 CorrelationId 輸入參數已根據需要設定。

可重新連接的用戶端會針對在 MQGM_SYNCPOINT 下收到的訊息，在成功重新連線之後傳回原因碼 MQRC_BACKED_OUT 。

getMessageOptions
控制 get 動作的選項。

從單位元組字元碼轉換為雙位元組碼時，使用選項 MQC.MQGMO_CONVERT 可能會導致異常狀況，原因碼為 MQC.MQRC_CONVERTED_STRING_TOO_BIG 。 在此情況下，會將訊息複製到緩衝區而不進行轉換。

如果未指定 *getMessageOptions* ，則使用的訊息選項為 MQGMO_NOWAIT。

如果您在可重新連接的用戶端中使用 MQGMO_LOGICAL_ORDER 選項，則會傳回 MQRC_RECONNECT_INCOMPATIBLE 原因碼。

MaxMsgSize
此訊息物件要接收的最大訊息。 如果佇列上的訊息大於此大小，則會發生下列兩種情況之一:

- 如果在 MQGetMessageOptions 物件中設定 MQGMO_ACCEPT_TRUNCATED_MSG 旗標，則訊息會盡可能填入訊息資料。 擲出異常狀況，含有 MQCC_WARNING 完成碼和 MQRC_TRUNCATED_MSG_ACCEPTED 原因碼。
- 如果未設定 MQGMO_ACCEPT_TRUNCATED_MSG 旗標，則訊息會留在佇列上。 擲出異常狀況，含有 MQCC_WARNING 完成碼和 MQRC_TRUNCATED_MSG_FAILED 原因碼。

如果未指定 *MaxMsgSize* ，則會擷取整個訊息。

建構子

```
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);
```

存取 *queueManager* 上的主題。

MQTopic 物件與管理主題物件密切相關，管理主題物件有時稱為主題物件。 在輸入上， topicObject 指向管理主題物件。 MQTopic 建構子會從主題物件取得主題字串，並將它與 topicName 結合，以建立

主題名稱。 topicObject 或 topicName 可以是空值。 主題名稱與主題樹狀結構相符，並在 topicObject 中傳回最相符管理主題物件的名稱。

與 MQTopic 物件相關聯的主題是結合兩個主題字串的結果。 第一個主題字串是由 *topicObject* 所識別的管理主題物件所定義。 第二個主題字串是 *topicString*。 與 MQTopic 物件相關聯的結果主題字串可以透過包括萬用字元來識別多個主題。

視是否開啟主題來發佈或訂閱而定，您可以使用 MQTopic.Put 方法來發佈主題，或使用 MQTopic.Get 方法來接收主題的發佈。 如果您想要發佈及訂閱相同的主題，則必須存取該主題兩次，一次用於發佈，一次用於訂閱。

如果您為訂閱建立 MQTopic 物件，但未提供 MQDestination 物件，則會採用受管理訂閱。 如果您傳遞佇列作為 MQDestination 物件，則會採用未受管理的訂閱。 您必須確定您設定的訂閱選項與受管理或未受管理的訂閱一致。

queueManager
在上存取主題的佇列管理程式。

destination
destination 是 MQQueue 實例。 藉由提供 *destination*， MQTopic 會以未受管理的訂閱方式開啟。 主題上的發佈會遞送至以 *destination* 身分存取的佇列。

topicName
主題名稱第二部分的主題字串。 *topicName* 與 *topicObject* 管理主題物件中定義的主題字串連結。 您可以將 *topicName* 設為空值，在此情況下，主題名稱由 *topicObject* 中的主題字串所定義。

topicObject
在輸入上， *topicObject* 是主題物件的名稱，包含形成主題名稱第一部分的主題字串。 *topicObject* 中的主題字串會與 *topicName* 連結。 主題名稱的建構規則定義在 結合主題字串 中。

在輸出上， *topicObject* 包含管理主題物件的名稱，該管理主題物件在主題樹狀結構中與主題名稱所識別的主題最相符。

openAs
存取主題以發佈或訂閱。 參數只能包含下列其中一個選項:

- MQC.MQTOPIC_OPEN_AS_SUBSCRIPTION
- MQC.MQTOPIC_OPEN_AS_PUBLICATION

options

結合控制開啟發佈或訂閱主題的選項。 使用 MQC.MQSO_* 常數來存取訂閱的主題，並使用 MQC.MQOO_* 常數來存取發佈的主題。

如果需要多個選項，請將值新增在一起，或使用位元 OR 運算子來結合選項值。

alternateUserId
指定替代使用者 ID，用來檢查完成作業所需的授權。 如果在 options 參數中設定 MQC.MQOO_ALTERNATE_USER_AUTHORITY 或 MQC.MQSO_ALTERNATE_USER_AUTHORITY，則必須指定 *alternateUserId*。

subscriptionName
如果提供選項 MQC.MQSO_DURABLE 或 MQC.MQSO_ALTER，則需要 *subscriptionName*。 在這兩種情況下，都會隱含地開啟 MQTopic 進行訂閱。 如果已設定 MQC.MQSO_DURABLE，且訂閱存在，或如果已設定 MQC.MQSO_ALTER，且訂閱不存在，則會擲出異常狀況。

properties
設定使用雜湊表列出的任何特殊訂閱內容。 雜湊表中的指定項目會以輸出值更新。 項目不會新增至雜湊表以報告輸出值。

- 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*);

存取此佇列管理程式上的主題。

MQTopic 物件與管理主題物件密切相關，管理主題物件有時稱為主題物件。 在輸入上， topicObject 指向管理主題物件。 MQTopic 建構子會從主題物件取得主題字串，並將它與 topicName 結合，以建立主題名稱。 topicObject 或 topicName 可以是空值。 主題名稱與主題樹狀結構相符，並在 topicObject 中傳回最相符管理主題物件的名稱。

與 MQTopic 物件相關聯的主題是結合兩個主題字串的結果。 第一個主題字串是由 *topicObject* 所識別的管理主題物件所定義。 第二個主題字串是 *topicString*。 與 MQTopic 物件相關聯的結果主題字串可以透過包括萬用字元來識別多個主題。

視是否開啟主題來發佈或訂閱而定，您可以使用 MQTopic.Put 方法來發佈主題，或使用 MQTopic.Get 方法來接收主題的發佈。 如果您想要發佈及訂閱相同的主題，則必須存取該主題兩次，一次用於發佈，一次用於訂閱。

如果您為訂閱建立 MQTopic 物件，但未提供 MQDestination 物件，則會採用受管理訂閱。 如果您傳遞佇列作為 MQDestination 物件，則會採用未受管理的訂閱。 您必須確定您設定的訂閱選項與受管理或未受管理的訂閱一致。

destination
destination 是 MQQueue 實例。 藉由提供 *destination*， MQTopic 會以未受管理的訂閱方式開啟。 主題上的發佈會遞送至以 *destination* 身分存取的佇列。

topicName
主題名稱第二部分的主題字串。 *topicName* 與 *topicObject* 管理主題物件中定義的主題字串連結。 您可以將 *topicName* 設為空值，在此情況下，主題名稱由 *topicObject* 中的主題字串所定義。

topicObject
在輸入上， *topicObject* 是主題物件的名稱，包含形成主題名稱第一部分的主題字串。 *topicObject* 中的主題字串會與 *topicName* 連結。 主題名稱的建構規則定義在 結合主題字串 中。

在輸出上， *topicObject* 包含管理主題物件的名稱，該管理主題物件在主題樹狀結構中與主題名稱所識別的主題最相符。

openAs
存取主題以發佈或訂閱。 參數只能包含下列其中一個選項:

- MQC.MQTOPIC_OPEN_AS_SUBSCRIPTION
- MQC.MQTOPIC_OPEN_AS_PUBLICATION

options

結合控制開啟發佈或訂閱主題的選項。 使用 MQC.MQSO_* 常數來存取訂閱的主題，並使用 MQC.MQOO_* 常數來存取發佈的主題。

如果需要多個選項，請將值新增在一起，或使用位元 OR 運算子來結合選項值。

alternateUserId

指定替代使用者 ID，用來檢查完成作業所需的授權。 如果在 options 參數中設定 MQC.MQOO_ALTERNATE_USER_AUTHORITY 或 MQC.MQSO_ALTERNATE_USER_AUTHORITY，則必須指定 *alternateUserId*。

subscriptionName

如果提供選項 MQC.MQSO_DURABLE 或 MQC.MQSO_ALTER，則需要 *subscriptionName*。 在這兩種情況下，都會隱含地開啟 MQTopic 進行訂閱。 如果已設定 MQC.MQSO_DURABLE，且訂閱存在，或如果已設定 MQC.MQSO_ALTER，且訂閱不存在，則會擲出異常狀況。

properties

設定使用雜湊表列出的任何特殊訂閱內容。 雜湊表中的指定項目會以輸出值更新。 項目不會新增至雜湊表以報告輸出值。

- 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 介面

實作 IMQObjectTrigger，以處理 **runmqdnm** .NET 監視器所傳遞的訊息。

介面

```
public interface IBM.WMQMonitor.IMQObjectTrigger();
```

視同步點控制是否在 **runmqdnm** 指令中指定而定，在 Execute 方法傳回之前或之後，會從佇列中移除訊息。

方法

void Execute (MQQueueManager *queueManager*, MQQueue *queue*, MQMessage *message*, string *param*);

queueManager

管理所監視佇列的佇列管理程式。

queue

正在監視佇列。

message

從佇列讀取的訊息。

param

從 UserParameter 傳遞的資料。

MQC .NET 介面

在常數名稱前面加上 MQC.，以參照 MQI 常數。 MQC 定義 MQI 使用的所有常數。

介面

```
System.Object
    └─ IBM.WMQ.MQC
```

```
public interface IBM.WMQ.MQC extends System.Object;
```

範例

```
MQQueue queue;
queue.closeOptions = MQC.MQCO_DELETE;
```

.NET 應用程式的字集 ID

您可以選取用來編碼 .NET IBM WebSphere MQ 訊息的字集說明

字集	說明
37	ibm037
437	ibm437 /PC 原始
500	ibm500
819	iso-8859-1 / latin1 / ibm819
1200	Unicode
1208	UTF-8
273	ibm273
277	ibm277
278	ibm278
280	ibm280
284	ibm284
285	ibm285
297	ibm297
420	ibm420
424	ibm424
737	ibm737 /PC 希臘文
775	ibm775 /PC 波羅的海
813	iso-8859-7 /greek/ ibm813
838	ibm838
850	ibm850 /PC 拉丁文 1
852	ibm852 /PC 拉丁文 2
855	ibm855 /PC 斯拉夫文

字集	說明
856	ibm856
857	ibm857 /PC 土耳其文
860	ibm860 /PC 葡萄牙文
861	ibm861 /PC 冰島文
862	ibm862 /PC 希伯來文
863	ibm863 /PC 加拿大法文
864	ibm864 /PC 阿拉伯文
865	ibm865 /PC Nordic
866	ibm866 /PC 俄文
868	ibm868
869	ibm869 /PC 現代希臘文
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 /斯拉夫語/ ibm915
916	iso-8859-8 /hebrew/ ibm916
918	ibm918
920	iso-8859-9 / latin5 / ibm920
921	ibm921
922	ibm922
930	ibm930
932	PC 日文
933	ibm933
935	ibm935
937	ibm937
939	ibm939
942	ibm942
943	ibm943
948	ibm948
949	ibm949
950	ibm950 /Big 5 繁體中文

字集	說明
954	EUCJIS
964	ibm964 /CNS 11643 繁體中文
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 拉丁文 2
1251	Windows 斯拉夫文
1252	Windows 拉丁文 1
1253	Windows 希臘文
1254	Windows 土耳其文
1255	Windows 希伯來文
1256	Windows 阿拉伯文
1257	Windows 波羅的海文
1258	Windows 越南文
1381	ibm1381
1383	ibm1383
2022	JIS
5601	ksc-5601 韓文
33722	ibm33722

IBM WebSphere MQ C++ 類別

IBM WebSphere MQ C++ 類別會封裝「IBM WebSphere MQ 訊息佇列介面 (MQI)」。 有一個單一 C++ 標頭檔 **imqi.hpp**，它涵蓋所有這些類別。

對於每一個類別，會顯示下列資訊:

類別階層圖
類別圖，顯示類別與其直接母類別的繼承關係 (如果有的話)。

其他相關類別
文件鏈結至其他相關類別，例如母類別，以及方法簽章中使用的物件類別。

物件屬性
類別的屬性。 這些是針對任何母類別所定義的屬性之外的其他屬性。 許多屬性反映 WebSphere MQ 資料結構成員 (請參閱 第 1178 頁的『C++ 及 MQI 交互參照』)。 如需詳細說明，請參閱 第 700 頁的『物件的屬性』。

建構子
用來建立類別物件的特殊方法簽章。

物件方法 (public)
方法的簽章，這些方法需要類別的實例來進行其作業，且沒有使用限制。

在適用的情況下，也會顯示下列資訊:

類別方法 (public)
不需要類別實例來進行其作業，且沒有使用限制的方法簽章。

超載 (母類別) 方法
在母類別中定義的那些虛擬方法的簽章，但針對此類別呈現不同的多型行為。

物件方法 (protected)
方法的簽章，這些方法需要類別的實例來進行其作業，並保留供衍生類別的實作使用。 此區段僅對類別寫出器感興趣，相對於類別使用者。

物件資料 (受保護)
衍生類別實作可用的物件實例資料的實作詳細資料。 此區段僅對類別寫出器感興趣，相對於類別使用者。

原因碼
MQRC_ * 值 (請參閱 API 原因碼) 可預期來自那些失敗的方法。 如需類別物件可能發生之原因碼的詳盡清單，請參閱母類別文件。 所記載的類別原因碼清單不包含母類別的原因碼。

註:

1. 這些類別的物件不是安全執行緒。 這可確保最佳效能，但請小心不要從多個執行緒存取任何物件。
2. 對於多執行緒程式，建議將個別「ImqQueue 管理程式」物件用於每一個執行緒。 每一個管理程式物件都必須有自己的其他物件獨立集合，以確保不同執行緒中的物件彼此隔離。

類別如下:

- 第 1191 頁的『ImqAuthentication 記錄 C++ 類別』
- 第 1193 頁的『ImqBinary C++ 類別』
- 第 1195 頁的『ImqCache C++ 類別』
- 第 1198 頁的『ImqChannel C++ 類別』
- 第 1203 頁的『ImqCICSBridge 標頭 C++ 類別』
- 第 1208 頁的『ImqDeadLetterHeader C++ 類別』
- 第 1211 頁的『ImqDistribution 列出 C++ 類別』
- 第 1212 頁的『ImqError C++ 類別』
- 第 1213 頁的『ImqGetMessageOptions C++ 類別』
- 第 1216 頁的『ImqHeader C++ 類別』
- 第 1218 頁的『ImqIMSBridge 標頭 C++ 類別』
- 第 1220 頁的『ImqItem C++ 類別』
- 第 1222 頁的『ImqMessage C++ 類別』
- 第 1228 頁的『ImqMessageTracker C++ 類別』
- 第 1231 頁的『ImqNamelist C++ 類別』
- 第 1233 頁的『ImqObject C++ 類別』
- 第 1238 頁的『ImqProcess C++ 類別』
- 第 1239 頁的『ImqPutMessageOptions C++ 類別』
- 第 1241 頁的『ImqQueue C++ 類別』

- 第 1251 頁的『ImqQueue 管理程式 C++ 類別』
- 第 1266 頁的『ImqReference 標頭 C++ 類別』
- 第 1269 頁的『ImqString C++ 類別』
- 第 1273 頁的『ImqTrigger C++ 類別』
- 第 1276 頁的『ImqWork 標頭 C++ 類別』

C++ 及 MQI 交互參照

此主題集合包含與 MQI 相關的 C++ 資訊。

請與 第 214 頁的『MQI 中使用的資料類型』一起閱讀此資訊。

此表格會將 MQI 資料結構與 C++ 類別及併入檔相關聯。 下列主題顯示每一個 C++ 類別的交互參照資訊。 這些交互參照與使用基礎 WebSphere MQ 程序化介面相關。 類別 ImqBinary、 ImqDistributionList 及 ImqString 沒有屬於此種類且已排除的屬性。

表 *610:* 資料結構、類別及併入檔交互參照

資料結構	**類別**	**併入檔**
MQAIR	ImqAuthentication 記錄	imqair.hpp
	ImqBinary	imqbin.hpp
	ImqCache	imqcac.hpp
MQCD	ImqChannel	imqchl.hpp
MQCIH	ImqCICSBridge 標頭	imqcih.hpp
MQDLH	ImqDeadLetterHeader	imqdlh.hpp
MQOR	ImqDistribution 清單	imqdst.hpp
	ImqError	imqerr.hpp
MQGMO	ImqGetMessageOptions	imqgmo.hpp
	ImqHeader	imqhdr.hpp
MQIIH	ImqIMSBridge 標頭	imqiih.hpp
	ImqItem	imqitm.hpp
MQMD	ImqMessage	imqmsg.hpp
	ImqMessage 追蹤器	imqmtr.hpp
	ImqNamelist	imqnml.hpp
MQOD、MQRR	ImqObject	imqobj.hpp
MQPMO、MQPMR、MQRR	ImqPutMessageOptions	imqpmo.hpp
	ImqProcess	imqpro.hpp
	ImqQueue	imqque.hpp
MQBO、MQCNO、MQCSP	ImqQueue 管理程式	imqmgr.hpp
MQRMH	ImqReference 標頭	imqrfh.hpp
	ImqString	imqstr.hpp
MQTM	ImqTrigger	imqtrg.hpp
MQTMC		

表 *610:* 資料結構、類別及併入檔交互參照 *(繼續)*		
資料結構	**類別**	**併入檔**
MQTMC2	ImqTrigger	imqtrg.hpp
MQXQH		
MQWIH	ImqWork 標頭	imqwih.hpp

ImqAuthentication 記錄交互參照

ImqAuthentication 記錄 C++ 類別之屬性、資料結構、欄位及呼叫的交互參照。

屬性	**資料結構**	**欄位**	**呼叫**
連線名稱	MQAIR	AuthInfoConnName	MQCONNX
密碼	MQAIR	LDAPPassword	MQCONNX
類型	MQAIR	AuthInfoType	MQCONNX
使用者名稱	MQAIR	LDAPUserNamePtr	MQCONNX
	MQAIR	LDAPUserName 偏移	MQCONNX
	MQAIR	LDAPUserName 長度	MQCONNX

ImqCache 交互參照

ImqCache C++ 類別之屬性和呼叫的交互參照。

屬性	**呼叫**
自動緩衝區	MQGET
緩衝區長度	MQGET
緩衝區指標	MQGET、MQPUT
資料長度	MQGET
資料偏移	MQGET
資料指標	MQGET
訊息長度	MQGET、MQPUT

ImqChannel 交互參照

ImqChannel C++ 類別之屬性、資料結構、欄位及呼叫的交互參照。

屬性	**資料結構**	**欄位**	**呼叫**
批次活動訊號	MQCD	BatchHeartbeat	MQCONNX
通道名稱	MQCD	ChannelName	MQCONNX
連線名稱	MQCD	ConnectionName	MQCONNX
	MQCD	ShortConnection 名稱	MQCONNX
標頭壓縮	MQCD	HdrComp 清單	MQCONNX
活動訊號間隔	MQCD	HeartbeatInterval	MQCONNX
持續作用間隔	MQCD	KeepAliveInterval	MQCONNX

屬性	資料結構	欄位	呼叫
本端位址	MQCD	LocalAddress	MQCONNX
訊息長度上限	MQCD	MaxMsgLength	MQCONNX
訊息壓縮	MQCD	MsgComp 清單	MQCONNX
模式名稱	MQCD	ModeName	MQCONNX
密碼	MQCD	密碼	MQCONNX
接收結束程式計數	MQCD		MQCONNX
接收結束程式名稱	MQCD	ReceiveExit	MQCONNX
	MQCD	ReceiveExits 已定義	MQCONNX
	MQCD	ReceiveExitPtr	MQCONNX
接收使用者資料	MQCD	ReceiveUserData	MQCONNX
	MQCD	ReceiveUserDataPtr	MQCONNX
安全結束程式名稱	MQCD	SecurityExit	MQCONNX
安全使用者資料	MQCD	SecurityUser 資料	MQCONNX
傳送結束程式計數	MQCD		MQCONNX
傳送結束程式名稱	MQCD	SendExit	MQCONNX
	MQCD	SendExits	MQCONNX
	MQCD	SendExitPtr	MQCONNX
傳送使用者資料	MQCD	SendUserData	MQCONNX
	MQCD	SendUserDataPtr	MQCONNX
SSL CipherSpec	MQCD	sslCipher 規格	MQCONNX
SSL 用戶端鑑別類型	MQCD	sslClient 鑑別	MQCONNX
SSL 對等名稱	MQCD	SSLPeerName	MQCONNX
交易程式名稱	MQCD	TpName	MQCONNX
傳輸類型	MQCD	TransportType	MQCONNX
使用者 ID	MQCD	UserIdentifier	MQCONNX

ImqCICSBridge 標頭交互參照

ImqCICSBridgeHeader C++ 類別之屬性、資料結構及欄位的交互參照。

屬性	資料結構	欄位
橋接器異常終止碼	MQCIH	AbendCode
ADS 描述子	MQCIH	AdsDescriptor
警示 ID	MQCIH	AttentionId
鑑別者 (authenticator)	MQCIH	鑑別程式
橋接器完成碼	MQCIH	BridgeCompletion 代碼
橋接器錯誤偏移	MQCIH	ErrorOffset
橋接器原因碼	MQCIH	BridgeReason

屬性	資料結構	欄位
橋接器取消程式碼	MQCIH	CancelCode
交談式作業	MQCIH	ConversationalTask
游標位置	MQCIH	CursorPosition
機能記號	MQCIH	機能
機能保留時間	MQCIH	FacilityKeep 時間
類似設施	MQCIH	FacilityLike
函數 (function)	MQCIH	函數
取得等待間隔	MQCIH	GetWait 間隔
鏈結類型	MQCIH	LinkType
下一個交易 ID	MQCIH	NextTransactionID
輸出資料長度	MQCIH	OutputData 長度
回覆格式	MQCIH	ReplyTo 格式
橋接器回覆碼	MQCIH	ReturnCode
起始碼	MQCIH	StartCode
作業結束狀態	MQCIH	TaskEnd 狀態
交易 ID	MQCIH	TransactionId
uow 控制項	MQCIH	UowControl
version	MQCIH	版本

ImqDeadLetterHeader 交互參照

ImqDeadLetterHeader C++ 類別之屬性、資料結構及欄位的交互參照。

屬性	資料結構	欄位
無法傳送郵件的原因碼	MQDLH	原因
目的地佇列管理程式名稱	MQDLH	DestQMgrName
目的地佇列名稱	MQDLH	DestQName
放置應用程式名稱	MQDLH	PutApplName
放置應用程式類型	MQDLH	PutApplType
放置日期	MQDLH	PutDate
放置時間	MQDLH	PutTime

ImqError 交互參照

ImqError C++ 類別之屬性及呼叫的交互參照。

屬性	呼叫
完成碼 (completion code)	MQBACK、MQBEGIN、MQCLOSE、MQCMIT、MQCONN、MQCONNX、MQDISC、MQGET、MQINQ、MQOPEN、MQPUT、MQSET

屬性	呼叫
原因碼 (reason code)	MQBACK、MQBEGIN、MQCLOSE、MQCMIT、MQCONN、MQCONNX、MQDISC、MQGET、MQINQ、MQOPEN、MQPUT、MQSET

ImqGetMessageOptions 交互參照

ImqGetMessageOptions C++ 類別之屬性、資料結構及欄位的交互參照。

屬性	資料結構	欄位
群組狀態	MQGMO	GroupStatus
符合選項	MQGMO	MatchOptions
訊息記號 (message token)	MQGMO	MessageToken
選項	MQGMO	選項
解析的佇列名稱	MQGMO	ResolvedQName
傳回長度	MQGMO	ReturnedLength
斷詞法 (segmentation)	MQGMO	分區段
區段狀態	MQGMO	SegmentStatus
	MQGMO	Signal1
	MQGMO	Signal2
同步點參與	MQGMO	選項
等待間隔	MQGMO	WaitInterval

ImqHeader 交互參照

ImqHeader C++ 類別之屬性、資料結構及欄位的交互參照。

屬性	資料結構	欄位
字元集	MQDLH、MQIIH	CodedCharSetId
encoding	MQDLH、MQIIH	編碼
格式 (format)	MQDLH、MQIIH	格式
標頭旗標	MQIIH、MQRMH	旗標

ImqIMSBridge 標頭交互參照

ImqAuthentication 記錄 C++ 類別之屬性、資料結構和欄位的交互參照。

屬性	資料結構	欄位
鑑別者 (authenticator)	MQIIH	鑑別程式
確定模式	MQIIH	CommitMode
邏輯終端機置換	MQIIH	LTermOverride
訊息格式服務對映圖名稱	MQIIH	MFSMapName
回覆格式	MQIIH	ReplyTo 格式
安全範圍	MQIIH	SecurityScope

屬性	資料結構	欄位
交易實例 ID	MQIIH	TranInstanceID
交易狀態	MQIIH	TranState

ImqItem 交互參照

ImqItem C++ 類別之屬性和呼叫的交互參照。

屬性	呼叫
結構 ID	MQGET

ImqMessage 交互參照

ImqMessage C++ 類別之屬性、資料結構、欄位及呼叫的交互參照。

屬性	資料結構	欄位	呼叫
應用程式 ID 資料	MQMD	ApplIdentityData	
應用程式原始資料	MQMD	ApplOriginData	
取消計數	MQMD	BackoutCount	
字元集	MQMD	CodedCharSetId	
encoding	MQMD	編碼	
到期	MQMD	期限	
格式 (format)	MQMD	格式	
訊息旗標	MQMD	MsgFlags	
訊息類型	MQMD	MsgType	
偏移	MQMD	偏移	
原始長度	MQMD	OriginalLength	
持續性	MQMD	持續性	
priority	MQMD	優先順序	
放置應用程式名稱	MQMD	PutApplName	
放置應用程式類型	MQMD	PutApplType	
放置日期	MQMD	PutDate	
放置時間	MQMD	PutTime	
回覆目的地佇列管理程式名稱	MQMD	回覆目的地佇列管理程式	
回覆目的地佇列名稱	MQMD	ReplyToQ	
報告	MQMD	報告	
序號 (sequence number)	MQMD	MsgSeqNumber	
訊息長度總計		DataLength	MQGET
使用者 ID	MQMD	UserIdentifier	

ImqMessage 追蹤器交互參照

ImqMessageTracker C++ 類別之屬性、資料結構及欄位的交互參照。

屬性	資料結構	欄位
帳戶記號	MQMD	AccountingToken
相關性 ID	MQMD	CorrelId
回饋	MQMD	意見
群組 ID	MQMD	GroupId
訊息 ID	MQMD	MsgId

ImqNamelist 交互參照

ImqNamelist C++ 類別之屬性、查詢及呼叫的交互參照。

屬性	查詢	呼叫
名稱計數	MQIA_NAME_COUNT	MQINQ
名稱清單名稱	MQCA_NAMELIST_NAME	MQINQ

ImqObject 交互參照

ImqObject C++ 類別之屬性、資料結構、欄位、查詢及呼叫的交互參照。

屬性	資料結構	欄位	查詢	呼叫
變更日期			MQCA_ALTERATION_DATE	MQINQ
變更時間			MQCA_ALTERATION_TIME	MQINQ
替代使用者 ID	MQOD	AlternateUserid		
替代安全 ID				
關閉選項				MQCLOSE
說明			MQCA_Q_DESC、 MQCA_Q_MGR_DESC、 MQCA_PROCESS_DESC	MQINQ
名	MQOD	ObjectName	MQCA_Q_MGR_NAME、 MQCQ_Q_NAME、 MQCA_PROCESS_NAME	MQINQ
開啟選項				MQOPEN
開啟狀態				MQOPEN、 MQCLOSE
佇列管理程式 ID	佇列管理程式 ID		MQCA_Q_MGR_IDENTIFIER	MQINQ

ImqProcess 交互參照

對 ImqAuthentication 記錄 C++ 類別的屬性、查詢及呼叫的交互參照。

屬性	查詢	呼叫
應用程式 ID	MQCA_APPL_ID	MQINQ
應用程式類型	MQIA_APPL_TYPE	MQINQ
環境資料	MQCA_ENV_DATA	MQINQ
使用者資料	MQCA_USER_DATA	MQINQ

ImqPutMessageOptions 交互參照

ImqAuthentication 記錄 C++ 類別之屬性、資料結構和欄位的交互參照。

表 *611: ImqPutMessageOptions* 交互參照		
屬性	**資料結構**	**欄位**
環境定義參照	MQPMO	環境定義
	MQPMO	InvalidDestCount
	MQPMO	KnownDestCount
選項	MQPMO	選項
記錄欄位	MQPMO	PutMsgRecFields
解析的佇列管理程式名稱	MQPMO	ResolvedQMgr 名稱
解析的佇列名稱	MQPMO	ResolvedQName
	MQPMO	逾時
	MQPMO	UnknownDestCount
同步點參與	MQPMO	選項

ImqQueue 交互參照

ImqQueue C++ 類別的屬性、資料結構、欄位、查詢及呼叫的交互參照。

表 *612: ImqQueue* 交互參照				
屬性	**資料結構**	**欄位**	**查詢**	**呼叫**
取消/重入佇列名稱			MQCA_BACKOUT_REQ_Q_NAME	MQINQ
取消臨界值			MQIA_BACKOUT_THRESHOLD	MQINQ
基本佇列名稱			MQCA_BASE_Q_NAME	MQINQ
叢集名稱			MQCA_CLUSTER_NAME	MQINQ
叢集名單名稱			MQCA_CLUSTER_NAMELIST	MQINQ
叢集工作量等級			MQIA_CLWL_Q_RANK	MQINQ
叢集工作量優先順序			MQIA_CLWL_Q_PRIORITY	MQINQ
叢集工作量使用佇列			MQIA_CLWL_USEQ	MQINQ
建立日期			MQCA_CREATION_DATE	MQINQ
建立時間			MQCA_CREATION_TIME	MQINQ
現行深度			MQIA_CURRENT_Q_DEPTH	MQINQ
預設連結			MQIA_DEF_BIND	MQINQ
預設的輸入開啟選項			MQIA_DEF_INPUT_OPEN_OPTION	MQINQ
預設持續性			MQIA_DEF_持續性	MQINQ
預設優先順序			MQIA_DEF_優先順序	MQINQ
定義類型			MQIA_DEFINITION_TYPE	MQINQ
深度高事件			MQIA_Q_DEPTH_HIGH_EVENT	MQINQ
深度上限			MQIA_Q_DEPTH_HIGH_LIMIT	MQINQ

表 612: ImqQueue 交互參照 (繼續)				
屬性	**資料結構**	**欄位**	**查詢**	**呼叫**
深度低事件			MQIA_Q_DEPTH_LOW_EVENT	MQINQ
深度下限			MQIA_Q_DEPTH_LOW_LIMIT	MQINQ
深度事件上限			MQIA_Q_DEPTH_MAX_LIMIT	MQINQ
配送清單			MQIA_DIST_清單	MQINQ、MQSET
動態佇列名稱	MQOD	DynamicQName		
強制取消			MQIA_HARDEN_GET_BACKOUT	MQINQ
索引類型			MQIA_INDEX_TYPE	MQINQ
禁止取得			MQIA_INHIT_get	MQINQ、MQSET
抑制放置			MQIA_INHIT_PLT	MQINQ、MQSET
起始佇列名稱			MQCA_INITIATION_Q_NAME	MQINQ
深度上限			MQIA_MAX_Q_DEPTH	MQINQ
訊息長度上限			MQIA_MAX_MSG_length	MQINQ
訊息遞送順序			MQIA_MSG_DELIVERY_SEQUENCE	MQINQ
下一個分散式佇列				
非持續訊息類別			MQIA_NPM_CLASS	MQINQ
開啟輸入計數			MQIA_OPEN_INPUT_COUNT	MQINQ
開啟輸出計數			MQIA_OPEN_OUTPUT_COUNT	MQINQ
前一個分散式佇列				
程序名稱			MQCA_PROCESS_NAME	MQINQ
佇列計數			MQIA_ACCOUNTING_Q	MQINQ
佇列管理程式名稱	MQOD	ObjectQMgrName		
監視佇列			MQIA_MONITORING_Q	MQINQ
佇列統計資料			MQI_STATISTICS_Q	MQINQ
佇列類型			MQIA_Q_TYPE	MQINQ
遠端佇列管理程式名稱			MQCA_REMOTE_Q_MGR_NAME	MQINQ
遠端佇列名稱			MQCA_REMOTE_Q_NAME	MQINQ
解析的佇列管理程式名稱	MQOD	ResolvedQMgr 名稱		
解析的佇列名稱	MQOD	ResolvedQName		
保留間隔			MQIA_RETENTION_INTERVAL	MQINQ
範圍			MQIA_SCOPE	MQINQ

表 612: ImqQueue 交互參照 (繼續)				
屬性	**資料結構**	**欄位**	**查詢**	**呼叫**
服務間隔 (service interval)			MQIA_Q_SERVICE_INTERVAL	MQINQ
服務間隔事件 (service interval event)			MQIA_Q_SERVICE_INTERVAL_EVENT	MQINQ
共用性			MQIA_SHAREABILITY	MQINQ
儲存類別 (storage class)			MQCA_STORAGE_CLASS	MQINQ
傳輸佇列名稱			MQCA_XMIT_Q_NAME	MQINQ
觸發控制			MQIA_TRIGGER_CONTROL	MQINQ、MQSET
觸發資料			MQ 卡 _ 觸發程式資料	MQINQ、MQSET
觸發深度			MQIA_TRIGGER_DEPTH	MQINQ、MQSET
觸發訊息優先順序			MQIA_TRIGGER_MSG_PRIORITY	MQINQ、MQSET
觸發函式類型			MQIA_TRIGGER_TYPE	MQINQ、MQSET
用法			MQIA_USAGE	MQINQ

ImqQueue 管理程式交互參照

ImqQueueManager C++ 類別之屬性、資料結構、欄位、查詢及呼叫的交互參照。

屬性	**資料結構**	**欄位**	**查詢**	**呼叫**
帳戶連線置換			MQIA_ACCOUNTING_CONN_OVERRIDE	MQINQ
計數間隔時間			MQIA_ACCOUNTING_INTERVAL	MQINQ
活動記錄中			MQIA_ACTIVITY_RECORDING	MQINQ
採用新的 MCA 檢查			MQIA_ADOPTNEWMCA_CHECK	MQINQ
採用新的 MCA 類型			MQIA_ADOPTNEWMCA_TYPE	MQINQ
鑑別類型	MQCSP	AuthenticationType		MQCONNX
權限事件			MQIA_AUTHORITY_EVENT	MQINQ
開始選項	MQBO	選項		MQBEGIN
橋接器事件			MQIA_BRIDGE_EVENT	MQINQ
通道自動定義			MQIA_CHANNEL_AUTO_DEF	MQINQ
通道自動定義事件			MQIA_CHANNEL_AUTO_EVENT	MQIA
通道自動定義結束程式			MQIA_CHANNEL_AUTO_EXIT	MQIA
通道事件 (channel event)			MQIA_CHANNEL_EVENT	MQINQ

屬性	資料結構	欄位	查詢	呼叫
通道起始程式配接器			MQIA_CHINIT_ADAPTERS	MQINQ
通道起始程式控制			MQIA_CHINIT_CONTROL	MQINQ
通道起始程式分派器			MQIA_CHINIT_DISPATCHER	MQINQ
通道起始程式追蹤自動啟動			MQIA_CHINIT_TRACE_AUTO_START	MQINQ
通道起始程式追蹤表格大小			MQIA_CHINIT_TRACE_TABLE_SIZE	MQINQ
監視通道			MQIA_MONITORING_CHANNEL	MQINQ
通道參照	MQCD	ChannelType		MQCONNX
通道統計資料			MQIA_STATISTICS_CHANNEL	MQINQ
字元集			MQIA_CODED_CHAR_SET_ID	MQINQ
叢集傳送端監視			MQIA_MONITORING_AUTO_CLUSSDR	MQINQ
叢集傳送端統計資料			MQIA_STATISTICS_AUTO_CLUSSDR	MQINQ
叢集工作量資料			MQCA_CLUSTER_WORKLOAD_DATA	MQINQ
叢集工作量結束程式			MQCA_CLUSTER_WORKLOAD_EXIT	MQINQ
叢集工作量長度			MQIA_CLUSTER_WORKLOAD_LENGTH	MQINQ
叢集工作量 mru			MQIA_CLWL_MRU_CHANNELS	MQINQ
叢集工作量使用佇列			MQIA_CLWL_USEQ	MQINQ
指令事件 (command event)			MQIA_COMMAND_Event	MQINQ
指令輸入佇列名稱			MQCA_COMMAND_INPUT_Q_NAME	MQINQ
指令層次			MQIA_COMMAND_LEVEL	MQINQ
指令伺服器控制			MQIA_CMD_SERVER_CONTROL	MQINQ
連線選項	MQCNO	選項		MQCONN、MQCONNX
連線 ID	MQCNO	ConnectionId		MQCONNX
連線狀態				MQCONN、MQCONNX、MQDISC
連線標記	MQCD	ConnTag		MQCONNX
加密硬體	MQSCO	CryptoHardware		MQCONNX
無法傳送郵件的佇列名稱			MQCA_DEAD_LETTER_Q_NAME	MQINQ
預設傳輸佇列名稱			MQCA_DEF_XMIT_Q_NAME	MQINQ

屬性	資料結構	欄位	查詢	呼叫
配送清單			MQIA_DIST_清單	MQINQ
dns 群組			MQCA_DNS_GROUP	MQINQ
dns wlm			MQIA_DNS_WLM	MQINQ
第一個鑑別記錄	MQSCO	AuthInfoRecOffset		MQCONNX
	MQSCO	AuthInfoRecPtr		MQCONNX
禁止事件			MQIA_INHIT_Event	MQINQ
IP 位址版本			MQIA_IP_ADDRESS_VERSION	MQINQ
金鑰儲存庫 (key repository)	MQSCO	KeyRepository		MQCONNX
金鑰重設計數	MQSCO	KeyReset 計數		MQCONNX
接聽器計時器			MQIA_LISTENER_TIMER	MQINQ
本端事件			MQIA_LOCAL_EVENT	MQINQ
日誌程式事件			MQIA_LOGGER_EVENT	MQINQ
LU 群組名稱			MQCA_LU_GROUP_NAME	MQINQ
LU 名稱			MQCA_LU_NAME	MQINQ
lu62 臂字尾			MQCA_LU62_ARM_SUFFIX	MQINQ
lu62 通道			MQIA_LU62_CHANNELS	MQINQ
作用中通道數上限			MQIA_ACTIVE_CHANNELS	MQINQ
通道數上限			MQIA_MAX_CHANNELS	MQINQ
控點數目上限			MQIA_MAX_HANDLES	MQINQ
訊息長度上限			MQIA_MAX_MSG_length	MQINQ
最大優先順序			MQIA_MAX_PRIORITY	MQINQ
未確定的訊息數上限			MQIA_MAX_UNCOMMITTED_MSGS	MQINQ
MQI 計數			MQIA_ACCOUNTING_MQI	MQINQ
MQI 統計資料			MQI_STATISTICS_MQI	MQINQ
出埠連接埠上限			MQIA_OUTBOUND_PORT_MAX	MQINQ
出埠連接埠下限			MQIA_OUTBOUND_PORT_MIN	MQINQ
密碼	MQCSP	CSPPasswordPtr		MQCONNX
	MQCSP	CSPPasswordOffset		MQCONNX
	MQCSP	CSPPasswordLength		MQCONNX
效能事件 (performance event)			MQIA_PERFORMANCE_EVENT	MQINQ
platform			MQIA_PLATFORM	MQINQ

屬性	資料結構	欄位	查詢	呼叫
佇列計數			MQIA_ACCOUNTING_Q	MQINQ
監視佇列			MQIA_MONITORING_Q	MQINQ
佇列統計資料			MQI_STATISTICS_Q	MQINQ
接收逾時			MQIA_RECEIVE_TIMEOUT	MQINQ
接收逾時下限			MQIA_RECEIVE_TIMEOUT_MIN	MQINQ
接收逾時類型			MQIA_RECEIVE_TIMEOUT_TYPE	MQINQ
遠端事件			MQIA_REMOTE_EVENT	MQINQ
REPOSITORY NAME			MQ 卡 _repository_name	MQINQ
儲存庫名單			MQCA_REPOSITORY_NAMELIST	MQINQ
共用佇列佇列管理程式名稱			MQIA_SHARED_Q_Q_MGR_NAME	MQINQ
ssl 事件			MQIA_SSL_EVENT	MQINQ
SSL fips			MQIA_SSL_FIPS_REQUIRED	MQINQ
SSL 金鑰重設計數			MQIA_SSL_RESET_COUNT	MQINQ
開始-停止事件			MQIA_START_STOP_EVENT	MQINQ
統計時間間隔			MQIA_STATISTICS_INTERVAL	MQINQ
同步點可用性			MQIA_SYNCPOINT	MQINQ
tcp channels			MQIA_TCP_CHANNELS	MQINQ
TCP 保持作用中			MQIA_TCP_KEEP_ALIVE	MQINQ
TCP 名稱			MQCA_TCP_NAME	MQINQ
TCP 堆疊類型			MQIA_TCP_STACK_TYPE	MQINQ
追蹤路徑記錄			MQIA_TRACE_ROUTE_RECORDING	MQINQ
觸發間隔			MQIA_TRIGGER_INTERVAL	MQINQ
使用者 ID	MQCSP	CSPUserIdPtr		MQCONNX
	MQCSP	CSPUserId 偏移		MQCONNX
	MQCSP	CSPUserId 長度		MQCONNX

ImqReference 標頭交互參照

ImqAuthentication 記錄 C++ 類別之屬性、資料結構和欄位的交互參照。

屬性	資料結構	欄位
目的地環境	MQRMH	DestEnv 長度， DestEnv 偏移
目的地名稱	MQRMH	DestName 長度， DestName 偏移
實例 ID	MQRMH	ObjectInstanceID
邏輯長度	MQRMH	DataLogical 長度

屬性	資料結構	欄位
邏輯偏移	MQRMH	DataLogical 偏移
邏輯偏移 2	MQRMH	DataLogicalOffset2
參照類型	MQRMH	ObjectType
來源環境	MQRMH	SrcEnv 長度， SrcEnv 偏移
來源名稱	MQRMH	SrcName 長度， SrcName 偏移

ImqTrigger 交互參照

ImqAuthentication 記錄 C++ 類別之屬性、資料結構和欄位的交互參照。

表 *613: ImqTrigger* 交互參照		
屬性	**資料結構**	**欄位**
應用程式 ID	MQTM	ApplId
應用程式類型	MQTM	ApplType
環境資料	MQTM	EnvData
程序名稱	MQTM	ProcessName
佇列名稱	MQTM	完整名稱
觸發資料	MQTM	TriggerData
使用者資料	MQTM	UserData

ImqWork 標頭交互參照

ImqAuthentication 記錄 C++ 類別之屬性、資料結構和欄位的交互參照。

屬性	資料結構	欄位
訊息記號 (message token)	MQWIH	MessageToken
服務名稱	MQWIH	ServiceName
服務步驟	MQWIH	ServiceStep

ImqAuthentication 記錄 C++ 類別

此類別會封裝鑑別資訊記錄 (MQAIR)， 以在執行自訂 SSL 用戶端連線的 ImqQueueManager: :connect 方法期間使用。

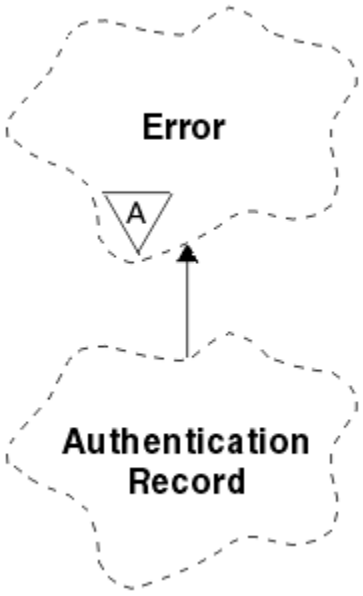


圖 *46: ImqAuthentication* 記錄類別

如需詳細資料，請參閱 ImqQueue 管理程式: :connect 方法的說明。 此類別在 z/OS 平台上無法使用。

- 第 1192 頁的『物件屬性』
- 第 1192 頁的『建構子』
- 第 1192 頁的『物件方法 (public)』
- 第 1193 頁的『物件方法 (protected)』

物件屬性

連線名稱
LDAP CRL 伺服器的連線名稱。 這是 IP 位址或 DNS 名稱，後面選擇性地接著埠號 (以括弧括住)。

連線參照
ImqQueueManager 物件的參照，提供與 (本端) 佇列管理程式的必要連線。 起始值為零。 請勿將此名稱與識別具名佇列之佇列管理程式 (可能是遠端) 的佇列管理程式名稱混淆。

下一個鑑別記錄
此類別的下一個物件 (無特定順序) 與此物件具有相同的 **連線參照** 。 起始值為零。

密碼
提供用於 LDAP CRL 伺服器連線鑑別的密碼。

前一個鑑別記錄
此類別的前一個物件 (無特定順序) 與此物件具有相同的 **連線參照** 。 起始值為零。

類型
記錄中包含的鑑別資訊類型。

使用者名稱
提供給 LDAP CRL 伺服器授權的使用者 ID。

建構子

ImqAuthentication 記錄 ();
預設建構子。

物件方法 (public)

void operator = (const ImqAuthenticationRecord & 空氣);
從 *air* 複製實例資料，並取代現有的實例資料。

const ImqString & connectionName ( ) const ;
傳回 **連線名稱**。

void setConnectionName (const ImqString & 名);
設定 **連線名稱**。

void setConnectionName (const char * *name* = 0);
設定 **連線名稱**。

ImqQueue 管理程式 * connectionReference () const;
傳回 **連線參照**。

void setConnectionReference (ImqQueueManager & 經理);
設定 **連線參照**。

void setConnectionReference (ImqQueueManager * *manager* = 0);
設定 **連線參照**。

void copyOut (MQAIR * *pAir*);
將實例資料複製到 *pAir*，以取代現有的實例資料。 這可能涉及配置相依儲存體。

void clear (MQAIR * *pAir*);
清除結構並釋放 *pAir* 所參照的相依儲存體。

ImqAuthentication 記錄 * nextAuthentication 記錄 () const;
傳回 **下一個鑑別記錄**。

const ImqString & password ( ) const ;
傳回 **password**。

void setPassword (const ImqString & 密碼);
設定 **密碼**。

void setPassword (const char * *password* = 0);
設定 **密碼**。

ImqAuthenticationRecord * previousAuthenticationRecord () const;
傳回 **前一個鑑別記錄**。

MQLONG 類型 () const;
傳回 **類型**。

void setType (const MQLONG *type*);
設定 **類型**。

const ImqString & userName ( ) const ;
傳回 **使用者名稱**。

void setUserName (const ImqString & 名);
設定 **使用者名稱**。

void setUserName (const char * *name* = 0);
設定 **使用者名稱**。

物件方法 (protected)

void setNextAuthenticationRecord (ImqAuthenticationRecord * *pAir* = 0);
設定 **下一個鑑別記錄**。

注意: 僅當您確定此功能不會岔斷鑑別記錄清單時，才使用此功能。

void setPreviousAuthenticationRecord (ImqAuthenticationRecord * *pAir* = 0);
設定 **前一個鑑別記錄**。

注意: 僅當您確定此功能不會岔斷鑑別記錄清單時，才使用此功能。

ImqBinary C++ 類別

此類別封裝可用於 ImqMessage **結算記號**、 **相關性 ID** 及 **訊息 ID** 值的二進位位元組陣列。 它容許輕鬆指派、複製及比較。

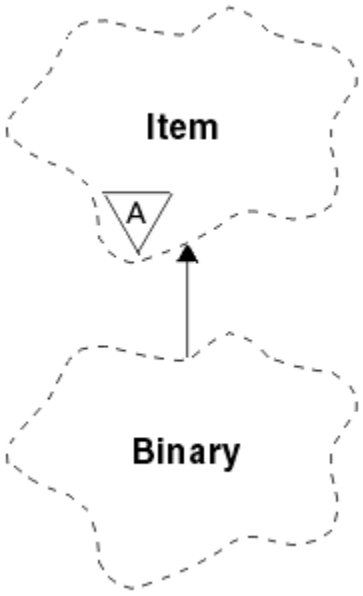


圖 *47: ImqBinary* 類別

- 第 1194 頁的『物件屬性』
- 第 1194 頁的『建構子』
- 第 1194 頁的『超載 ImqItem 方法』
- 第 1195 頁的『物件方法 (public)』
- 第 1195 頁的『物件方法 (protected)』
- 第 1195 頁的『原因碼』

物件屬性

Data
二進位資料的位元組陣列。 起始值是空值。

資料長度
位元組數。 起始值為零。

資料指標
資料第一個位元組的位址。 起始值為零。

建構子

ImqBinary( );
預設建構子。

ImqBinary(const ImqBinary & 二進位);
複製建構子。

ImqBinary(const void * *data*, const size_t *length*);
從 資料複製 *length* 個位元組。

超載 ImqItem 方法

虛擬 ImqBoolean copyOut(ImqMessage & 訊息);
將 **資料** 複製到訊息緩衝區，以取代任何現有的內容。 將 *msg* **格式** 設為 MQFMT_NONE。

如需進一步詳細資料，請參閱 ImqItem 類別方法說明。

虛擬 ImqBoolean pasteIn(ImqMessage & 訊息);
透過從訊息緩衝區傳送剩餘資料，並取代現有 **資料**，來設定 **資料** 。

若要成功， ImqMessage **格式** 必須是 MQFMT_NONE。

如需進一步詳細資料，請參閱 ImqItem 類別方法說明。

物件方法 (public)

void operator = (const ImqBinary & 二進位);
從 二進位複製位元組。

ImqBoolean 運算子 == (const ImqBinary & 二進位);
比較此物件與 *binary*。 如果不等於則傳回 FALSE ，否則傳回 TRUE。 如果物件具有相同的 **資料長度** 且位元組相符，則物件相等。

ImqBoolean copyOut(void * *buffer*， const size_t *length*， const char *pad* = 0);
將 **資料指標** 最多 長度 個位元組複製到 緩衝區。 如果 **資料長度** 不足，則 *buffer* 中的剩餘空間會填入 *pad* 位元組。 如果 *length* 也是零，則 *buffer* 可以是零。 *length* 不得為負數。 如果成功，它會傳回 TRUE。

size_t dataLength() const ;
傳回 **資料長度**。

ImqBoolean setDataLength(const size_t *length*);
設定 **資料長度**。 如果因為此方法而變更 **資料長度** ，則會取消起始設定物件中的資料。 如果成功，它會傳回 TRUE。

void * dataPointer() const ;
傳回 **資料指標**。

ImqBoolean isNull() const ;
如果 **資料長度** 為零，或如果所有 **資料** 位元組都為零，則傳回 TRUE。 否則會傳回 FALSE。

ImqBoolean set(const void * *buffer*， const size_t *length*);
從 緩衝區複製 長度 位元組。 如果成功，它會傳回 TRUE。

物件方法 (protected)

void clear();
將 **資料長度** 減少為零。

原因碼

- MQRC_NO_BUFFER
- MQRC_STORAGE_NOT_AVAILABLE
- MQRC_INCONSISTENT_FORMAT

ImqCache C++ 類別

使用此類別來保留或配置記憶體中的資料。

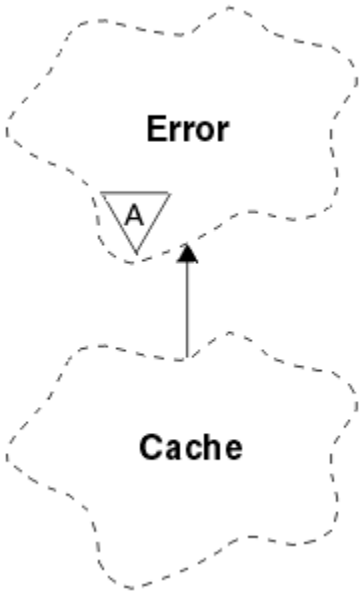


圖 *48: ImqCache* 類別

使用此類別來保留或配置記憶體中的資料。 您可以指定固定大小的記憶體緩衝區，或者系統可以自動提供彈性的記憶體數量。 此類別與 第 1179 頁的『ImqCache 交互參照』中列出的 MQI 呼叫相關。

- 第 1196 頁的『物件屬性』
- 第 1197 頁的『建構子』
- 第 1197 頁的『物件方法 (public)』
- 第 1198 頁的『原因碼』

物件屬性

自動緩衝區
指出緩衝區記憶體是由系統自動管理 (TRUE) 還是由使用者提供 (FALSE)。 一開始會將它設為 TRUE。

未直接設定此屬性。 它是使用 **useEmptyBuffer** 或 **useFullBuffer** 方法間接設定的。

如果提供使用者儲存體，則此屬性為 FALSE ，緩衝區記憶體無法成長，且可能會發生緩衝區溢位錯誤。 緩衝區的位址和長度維持不變。

如果未提供使用者儲存體，則此屬性為 TRUE ，且緩衝區記憶體可以漸進式成長，以容納任意數量的訊息資料。 不過，當緩衝區成長時，緩衝區的位址可能會變更，因此在使用 **緩衝區指標** 及 **資料指標**時請小心。

緩衝區長度
緩衝區中記憶體的位元組數。 起始值為零。

緩衝區指標
緩衝區記憶體的位址。 起始值是空值。

資料長度
資料指標之後的位元組數。 這必須等於或小於 **訊息長度**。 起始值為零。

資料偏移
資料指標之前的位元組數。 這必須等於或小於 **訊息長度**。 起始值為零。

資料指標
要寫入或讀取下一個緩衝區的部分位址。 起始值是空值。

訊息長度
緩衝區中有效資料的位元組數。 起始值為零。

建構子

ImqCache( );
預設建構子。

ImqCache(const ImqCache & 快取);
複製建構子。

物件方法 (public)

void operator = (const ImqCache & 快取);
將 *cache* 物件中的資料，最多 **訊息長度** 個位元組複製到物件。 如果 **自動緩衝區** 為 FALSE ，則 **緩衝區長度** 必須已足以容納所複製的資料。

ImqBoolean automaticBuffer() const ;
傳回 **自動緩衝區** 值。

size_t bufferLength() const ;
傳回 **緩衝區長度**。

char * bufferPointer() const ;
傳回 **緩衝區指標**。

void clearMessage();
將 **訊息長度** 及 **資料偏移** 設為零。

size_t dataLength() const ;
傳回 **資料長度**。

size_t dataOffset() const ;
傳回 **資料偏移**。

ImqBoolean setDataOffset(const size_t *offset*);
設定 **資料偏移**。 必要的話，會增加 **訊息長度** ，以確保它不小於 **資料偏移**。 如果成功，此方法會傳回 TRUE。

char * dataPointer() const ;
傳回 **資料指標**的副本。

size_t messageLength() const ;
傳回 **訊息長度**。

ImqBoolean setMessageLength(const size_t *length*);
設定 **訊息長度**。 必要的話，請增加 **緩衝區長度** ，以確保 **訊息長度** 不大於 **緩衝區長度**。 必要的話，減少 **資料偏移** ，以確保它不大於 **訊息長度**。 如果成功，它會傳回 TRUE。

ImqBoolean moreBytes(const size_t 位元組-必要);
確保在 **資料指標** 與緩衝區結尾之間，有 *bytes-required* 更多位元組可供寫入 (寫入)。 如果成功，它會傳回 TRUE。

如果 **自動緩衝區** 為 TRUE ，則會根據需要獲得更多記憶體; 否則， **緩衝區長度** 必須已足夠。

ImqBoolean 讀(常數大小 _t 長度, char * & 外部緩衝區);
從 **資料指標** 位置開始的緩衝區，將 *length* 個位元組複製到 外部緩衝區。 複製資料之後， **資料偏移** 會增加 長度。 如果成功，此方法會傳回 TRUE。

ImqBoolean resizeBuffer(const size_t *length*);
改變 **緩衝區長度**，前提是 **自動緩衝區** 為 TRUE。 這是透過重新配置緩衝區記憶體來達成。 來自現有緩衝區的資料最多 **訊息長度** 個位元組會複製到新的緩衝區。 複製的數目上限是 *length* 個位元組。 **緩衝區指標** 已變更。 **訊息長度** 及 **資料偏移** 會盡可能在新緩衝區的範圍內保留。 如果成功，則會傳回 TRUE; 如果 **自動緩衝區** 為 FALSE ，則會傳回 FALSE。

註: 如果系統資源有任何問題，此方法可能會因 MQRC_STORAGE_NOT_AVAILABLE 而失敗。

ImqBoolean useEmptyBuffer(const char * *external-buffer*， const size_t *length*);
識別空的使用者緩衝區，將 **緩衝區指標** 設為指向 外部緩衝區，將 **緩衝區長度** 設為 長度，並將 **訊息長度** 設為零。 執行 **clearMessage**。 如果緩衝區已完全準備好資料，請改用 **useFullBuffer** 方法。 如果緩衝區部分準備了資料，請使用 **setMessageLength** 方法來指出正確的數量。 如果成功，此方法會傳回 TRUE。

此方法可用來識別固定記憶體數量，如先前所述 (*external-buffer* 不是空值且 *length* 為非零) ，在此情況下， **自動緩衝區** 會設為 FALSE ，或者可用來回復至系統管理的彈性記憶體 (*external-buffer* 是空值且 *length* 為零) ，在此情況下， **自動緩衝區** 會設為 TRUE。

ImqBoolean useFullBuffer(const char * *externalBuffer*， const size_t *length*);
至於 **useEmpty 緩衝區**，除了 **訊息長度** 設為 *length* 之外。 如果成功，它會傳回 TRUE。

ImqBoolean write(const size_t *length*， const char * *external-buffer*);
將 *length* 個位元組從 外部緩衝區複製到從 **資料指標** 位置開始的緩衝區。 複製資料之後， **資料偏移** 會增加 長度，必要的話會增加 **訊息長度** ，以確保它不小於新的 **資料偏移** 值。 如果成功，此方法會傳回 TRUE。

如果 **自動緩衝區** 為 TRUE ，則保證記憶體數量足夠; 否則，最終 **資料偏移** 不得超出 **緩衝區長度**。

原因碼

- MQRC_BUFFER_NOT_AUTOMATIC
- MQRC_DATA_TRUNCATED
- MQRC_INSUFFICIENT_BUFFER
- MQRC_INSUFFICIENT_DATA
- MQ RC_NULL_POINTER
- MQRC_STORAGE_NOT_AVAILABLE
- MQRC_ZERO_LENGTH

ImqChannel C++ 類別

這個類別會封裝通道定義 (MQCD) ，以便在執行自訂用戶端連線的 Manager: :connect 方法期間使用。

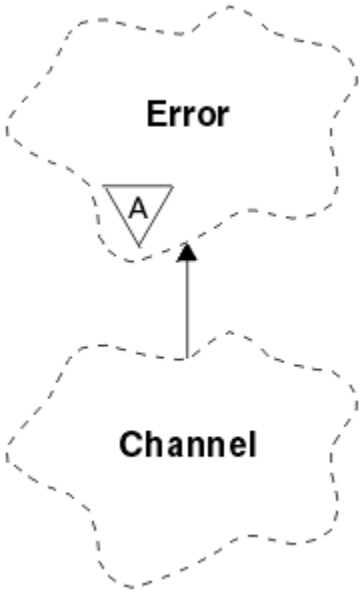


圖 *49: ImqChannel* 類別

如需詳細資料，請參閱 Manager: :connect 方法及 範例程式 HELLO WORLD (imqwrld.cpp) 的說明。 並非所有列出的方法都適用於所有平台; 如需詳細資料，請參閱 MQSC 參照 中 DEFINE CHANNEL 及 ALTER CHANNEL 指令的說明。 z/OS 不支援 ImqChannel 類別。

- 第 1199 頁的『物件屬性』
- 第 1200 頁的『建構子』
- 第 1200 頁的『物件方法 (public)』
- 第 1203 頁的『原因碼』

物件屬性

批次活動訊號
檢查遠端通道是否處於作用中的間隔毫秒數。 起始值為 0。

通道名稱
通道的名稱。 起始值是空值。

連線名稱
連線的名稱。 例如，主機的 IP 位址。 起始值是空值。

標頭壓縮
通道支援的標頭資料壓縮技術清單。 起始值全部設為 MQCOMPRESS_NOT_AVAILABLE。

活動訊號間隔
檢查連線是否仍在運作的間隔秒數。 起始值為 300。

持續作用間隔
指定通道保持作用中計時，傳給通訊堆疊的秒數。 起始值為 MQKA_AUTO。

本端位址
通道的本端通訊位址。

訊息長度上限
通道在單一通訊中支援的訊息長度上限。 起始值為 4 194 304。

訊息壓縮
通道支援的訊息資料壓縮技術清單。 起始值全部設為 MQCOMPRESS_NOT_AVAILABLE。

模式名稱
模式的名稱。 起始值是空值。

password
提供用於連線鑑別的密碼。 起始值是空值。

接收結束程式計數
接收結束程式數目。 起始值為零。 這個屬性是唯讀的。

接收結束程式名稱
接收結束程式的名稱。

接收使用者資料
與接收結束程式相關聯的資料。

安全結束程式名稱
要在連線伺服器端呼叫的安全結束程式名稱。 起始值是空值。

安全使用者資料
要傳遞至安全結束程式的資料。 起始值是空值。

傳送結束程式計數
傳送結束程式數目。 起始值為零。 這個屬性是唯讀的。

傳送結束程式名稱
傳送結束程式的名稱。

傳送使用者資料
與傳送結束程式相關聯的資料。

SSL CipherSpec
CipherSpec ， 與 SSL 搭配使用。

SSL 用戶端鑑別類型
與 SSL 搭配使用的用戶端鑑別類型。

SSL 對等名稱
與 SSL 搭配使用的同層級名稱。

交易程式名稱
交易程式的名稱。 起始值是空值。

傳輸類型
連線的傳輸類型。 起始值為 MQXPT_LU62。

使用者 ID
提供授權的使用者 ID。 起始值是空值。

建構子

ImqChannel( ) ;
預設建構子。

ImqChannel(const ImqChannel & *channel*);
複製建構子。

物件方法 (public)

void operator = (const ImqChannel & *channel*);
從 通道複製實例資料，並取代任何現有的實例資料。

MQLONG batchHeartBeat () const;
傳回 **批次活動訊號**。

ImqBoolean setBatchHeartBeat(const MQLONG *heartbeat* = 0L);
設定 **批次活動訊號** 。 如果成功，此方法會傳回 TRUE。

ImqString channelName() const;
傳回 **通道名稱**。

ImqBoolean setChannelName (const char * *name* = 0);
設定 **通道名稱**。 如果成功，此方法會傳回 TRUE。

ImqString connectionName() const;
傳回 **連線名稱**。

ImqBoolean setConnectionName (const char * *name* = 0);
設定 **連線名稱**。 如果成功，此方法會傳回 TRUE。

size_t headerCompressionCount () const;
傳回支援的標頭資料壓縮技術計數。

ImqBoolean headerCompression(const size_t count , MQLONG compress []) const;
傳回 **compress** 中支援的標頭資料壓縮技術的副本。 如果成功，此方法會傳回 TRUE。

ImqBoolean setHeaderCompression (const size_t count , const MQLONG compress [];

將支援的標頭資料壓縮技術設為 **compress**。

將支援的標頭資料壓縮技術計數設為 **count**。

如果成功，此方法會傳回 TRUE。

MQLONG heartBeat 間隔 () const;
傳回 **活動訊號間隔**。

ImqBoolean setHeartBeatInterval(const MQLONG *interval* = 300L);
設定 **活動訊號間隔**。 如果成功，此方法會傳回 TRUE。

MQLONG keepAlive 間隔 () const;
傳回 **保持作用中間隔**。

ImqBoolean setKeepAliveInterval(const MQLONG *interval* = MQKAI_AUTO);
設定 **保持作用中間隔**。 如果成功，此方法會傳回 TRUE。

ImqString localAddress() const;
傳回 **本端位址**。

ImqBoolean setLocalAddress (const char * *address* = 0);
設定 **本端位址**。 如果成功，此方法會傳回 TRUE。

MQLONG maximumMessage 長度 () const;
傳回 **訊息長度上限**。

ImqBoolean setMaximumMessageLength(const MQLONG *length* = 4194304L);
設定 **訊息長度上限**。 如果成功，此方法會傳回 TRUE。

size_t messageCompressionCount () const;
傳回支援的訊息資料壓縮技術計數。

ImqBoolean messageCompression(const size_t count , MQLONG compress []) const;
傳回 **compress** 中受支援訊息資料壓縮技術的副本。 如果成功，此方法會傳回 TRUE。

ImqBoolean setMessageCompression (const size_t count , const MQLONG compress []);

設定要壓縮的受支援訊息資料壓縮技術。

將支援的訊息資料壓縮技術計數設為計數。

如果成功，此方法會傳回 TRUE。

ImqString modeName() const;
傳回 **模式名稱**。

ImqBoolean setModeName (const char * *name* = 0);
設定 **模式名稱**。 如果成功，此方法會傳回 TRUE。

ImqString 密碼 () const;
傳回 **password**。

ImqBoolean setPassword(const char * *password* = 0);
設定 **密碼**。 如果成功，此方法會傳回 TRUE。

size_t receiveExitCount () const;
傳回 **接收結束計數**。

ImqString receiveExitName ();
傳回第一個 **接收結束程式名稱**(如果有的話)。 如果 **receive exit count** 為零，則會傳回空字串。

ImqBoolean receiveExitNames (const size_t *count*, ImqString * *names* []);
以 名稱傳回 **接收結束程式名稱** 的副本。 將任何超出 **receive exit count** 的 名稱 設為空字串。 如果成功，此方法會傳回 TRUE。

ImqBoolean setReceiveExitName(const char * *name* = 0);
將 **接收結束程式名稱** 設為單一 名稱。 *name* 可以是空白或空值。 將 **接收結束程式計數** 設為 1 或零。 清除 **接收使用者資料**。 如果成功，此方法會傳回 TRUE。

ImqBoolean setReceiveExitNames(const size_t *count*, const char * *names* []);
將 **接收結束程式名稱** 設為 名稱。 個別 名稱 值不得為空白或空值。 將 **接收結束計數** 設為 計數。 清除 **接收使用者資料**。 如果成功，此方法會傳回 TRUE。

ImqBoolean setReceiveExitNames(const size_t *count*, const ImqString * *names* []);
將 **接收結束程式名稱** 設為 名稱。 個別 名稱 值不得為空白或空值。 將 **接收結束計數** 設為 計數。 清除 **接收使用者資料**。 如果成功，此方法會傳回 TRUE。

ImqString receiveUserData ();
傳回第一個 **接收使用者資料** 項目 (如果有的話)。 如果 **接收結束計數** 為零，則會傳回空字串。

ImqBoolean receiveUserData (const size_t *count*, ImqString * *data* []);
傳回 *data* 中 **receive user data** 項目的副本。 將超出 **接收結束計數** 的任何 資料 設為空字串。 如果成功，此方法會傳回 TRUE。

ImqBoolean setReceiveUserData(const char * *data* = 0);
將 **接收使用者資料** 設為單一項目 資料。 如果 *data* 不是空值，則 **receive exit count** 必須至少為 1。 如果成功，此方法會傳回 TRUE。

ImqBoolean setReceiveUserData(const size_t *count*, const char * *data* []);
將 **接收使用者資料** 設為 資料。 *count* 不得大於 **receive exit count**。 如果成功，此方法會傳回 TRUE。

ImqBoolean setReceiveUserData(const size_t *count*, const ImqString * *data* []);
將 **接收使用者資料** 設為 資料。 *count* 不得大於 **receive exit count**。 如果成功，此方法會傳回 TRUE。

ImqString securityExit 名稱 () const;
傳回 **安全結束程式名稱**。

ImqBoolean setSecurityExitName(const char * *name* = 0);
設定 **安全結束程式名稱**。 如果成功，此方法會傳回 TRUE。

ImqString securityUserData () const;
傳回 **安全使用者資料**。

ImqBoolean setSecurityUserData(const char * *data* = 0);
設定 **安全使用者資料**。 如果成功，此方法會傳回 TRUE。

size_t sendExitCount () const;
傳回 **傳送結束程式計數**。

ImqString sendExitName ();
傳回第一個 **傳送結束程式名稱**(如果有的話)。 如果 **send exit count** 為零，則傳回空字串。

ImqBoolean sendExitNames (const size_t *count*, ImqString * *names* []);
以 名稱傳回 **傳送結束程式名稱** 的副本。 將任何超出 **send exit count** 的 名稱 設為空字串。 如果成功，此方法會傳回 TRUE。

ImqBoolean setSendExitName(const char * *name* = 0);
將 **傳送結束程式名稱** 設為單一 名稱。 *name* 可以是空白或空值。 將 **傳送結束計數** 設為 1 或零。 清除 **傳送使用者資料**。 如果成功，此方法會傳回 TRUE

ImqBoolean setSendExitNames(const size_t *count*, const char * *names* []);
將 **傳送結束程式名稱** 設為 名稱。 個別 名稱 值不得為空白或空值。 將 **傳送結束程式計數** 設為 計數。 清除 **傳送使用者資料**。 如果成功，此方法會傳回 TRUE。

ImqBoolean setSendExitNames(const size_t *count*, const ImqString * *names* []);
將 **傳送結束程式名稱** 設為 名稱。 個別 名稱 值不得為空白或空值。 將 **傳送結束程式計數** 設為 計數。 清除 **傳送使用者資料**。 如果成功，此方法會傳回 TRUE。

ImqString sendUserData ();
傳回第一個 **傳送使用者資料** 項目 (如果有的話)。 如果 **send exit count** 為零，則傳回空字串。

ImqBoolean sendUserData (const size_t *count*, ImqString * *data* []);
傳回 *data* 中 **send user data** 項目的副本。 將任何超出 **傳送結束計數** 的 資料 設為空字串。 如果成功，此方法會傳回 TRUE。

ImqBoolean setSendUserData(const char * *data* = 0);
將 **傳送使用者資料** 設為單一項目 資料。 如果 *data* 不是空值，則 **傳送結束程式計數** 必須至少為 1。 如果成功，此方法會傳回 TRUE。

ImqBoolean setSendUserData(const size_t *count*, const char * *data* []);
將 **傳送使用者資料** 設為 資料。 *count* 不得大於 **send exit count**。 如果成功，此方法會傳回 TRUE。

ImqBoolean setSendUserData(const size_t *count*, const ImqString * *data* []);
將 **傳送使用者資料** 設為 資料。 *count* 不得大於 **send exit count**。 如果成功，此方法會傳回 TRUE。

ImqString sslCipherSpecification () const;
傳回 SSL 密碼規格。

ImqBoolean setSslCipherSpecification(const char * *name* = 0);
設定 SSL 密碼規格。 如果成功，此方法會傳回 TRUE。

MQLONG sslClient 鑑別 () const;
傳回 SSL 用戶端鑑別類型。

ImqBoolean setSslClientAuthentication(const MQLONG *auth* = MQSCA_REQUIRED);
設定 SSL 用戶端鑑別類型。 如果成功，此方法會傳回 TRUE。

ImqString sslPeerName () const;
傳回 SSL 同層級名稱。

ImqBoolean setSslPeerName(const char * *name* = 0);
設定 SSL 同層級名稱。 如果成功，此方法會傳回 TRUE。

ImqString transactionProgramName () const;
傳回 **交易程式名稱**。

ImqBoolean setTransactionProgramName(const char * *name* = 0);
設定 **交易程式名稱**。 如果成功，此方法會傳回 TRUE。

MQLONG transportType() const;
傳回 **傳輸類型**。

ImqBoolean setTransportType (const MQLONG *type* = MQXPT_LU62);
設定 **傳輸類型**。 如果成功，此方法會傳回 TRUE。

ImqString userId() const;
傳回 **使用者 ID**。

ImqBoolean setUserId (const char * *id* = 0);
設定 **使用者 ID**。 如果成功，此方法會傳回 TRUE。

原因碼

- MQRC_DATA_LENGTH_ERROR
- MQRC_ITEM_COUNT_ERROR
- MQ RC_NULL_POINTER
- MQRC_SOURCE_BUFFER_ERROR

ImqCICSBridge 標頭 C++ 類別

此類別封裝 MQCIH 資料結構的特定特性。

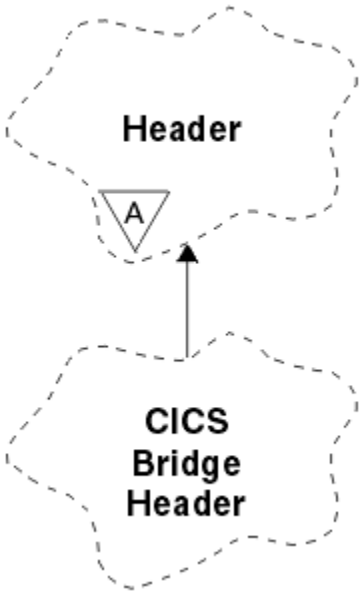


圖 *50: ImqCICSBridge* 標頭類別

此類別的物件由透過 WebSphere MQ for z/OS 傳送訊息至 CICS 橋接器的應用程式使用。

- 第 1203 頁的『物件屬性』
- 第 1205 頁的『建構子』
- 第 1206 頁的『超載 ImqItem 方法』
- 第 1206 頁的『物件方法 (public)』
- 第 1208 頁的『物件資料 (受保護)』
- 第 1208 頁的『原因碼』
- 第 1208 頁的『回覆碼』

物件屬性

ADS 描述子
傳送/接收 ADS 描述子。 這是使用 MQCADSD_NONE 來設定。 起始值為 MQCADSD_NONE。 下列是可能的其他值:

- MQCADSD_NONE
- MQCADSD_SEND
- MQCADSD_RECV
- MQCADSD_MSGFORMAT

警示 ID
AID 金鑰。 欄位長度必須為 MQ_ATTENTION_ID_LENGTH。

鑑別者 (authenticator)
RACF 密碼或通行證。 起始值包含長度 MQ_AUTHENTICATOR_LENGTH 的空白。

橋接器異常終止碼
橋接器異常終止碼，長度為 MQ_ABEND_CODE_LENGTH。 起始值為四個空白字元。 此欄位中傳回的值取決於回覆碼。 如需詳細資料，請參閱 第 1208 頁的表 614。

橋接器取消程式碼
橋接器異常終止交易碼。 欄位已保留，必須包含空白，且長度為 MQ_CANCEL_CODE_LENGTH。

橋接器完成碼
完成碼，可包含 WebSphere MQ 完成碼或 CICS EIBRESP 值。 此欄位具有起始值 MQCC_OK。 此欄位中傳回的值取決於回覆碼。 如需詳細資料，請參閱 第 1208 頁的表 614。

橋接器錯誤偏移
橋接器錯誤偏移。 起始值為零。 這個屬性是唯讀的。

橋接器原因碼
原因碼。 此欄位可以包含 WebSphere MQ 原因或 CICS EIBRESP2 值。 欄位的起始值為 MQRC_NONE。 此欄位中傳回的值取決於回覆碼。 如需詳細資料，請參閱 第 1208 頁的表 614。

橋接器回覆碼
來自 CICS 橋接器的回覆碼。 起始值為 MQCRC_OK。

交談式作業
作業是否可以交談。 起始值為 MQCCT_NO。 下列是可能的其他值:

- MQCCT_YES
- MQCCT_NO

游標位置
游標位置。 起始值為零。

機能保留時間
CICS 橋接器機能釋放時間。

facility like
終端機模擬屬性。 欄位長度必須為 MQ_FACILITY_LIKE_LENGTH。

機能記號
BVT 記號值。 欄位長度必須為 MQ_FACILITY_LENGTH。 起始值為 MQCFAC_NONE。

函數
函數，可包含 WebSphere MQ 呼叫名稱或 CICS EIBFN 函數。 欄位的起始值為 MQCFUNC_NONE ，長度為 MQ_FUNCTION_LENGTH。 此欄位中傳回的值取決於回覆碼。 如需其他詳細資料，請參閱第 1208 頁的表 614。

當 **函數** 包含 WebSphere MQ 呼叫名稱時，可以使用下列其他值:

- MQCFUNC_MQCONN
- MQCFUNC_MQGET
- MQCFUNC_MQINQ
- MQCFUNC_NONE
- MQCFUNC_MQOPEN
- MQCFUNC_PUT
- MQCFUNC_MQPUT1

取得等待間隔
CICS 橋接器作業發出的 MQGET 呼叫的等待間隔。 起始值是 MQCGWI_DEFAULT。 只有在 **uow 控制項** 具有 MQCUOWC_FIRST 值時，此欄位才適用。 下列是可能的其他值:

- MQCGWI_DEFAULT
- MQWI_UNLIMITED

鏈結類型
鏈結類型。 起始值為 MQCLT_PROGRAM。 下列是可能的其他值:

- MQCLT_PROGRAM
- MQCLT_TRANSACTION

下一個交易 ID
要連接的下一個交易 ID。 欄位長度必須為 MQ_TRANSACTION_ID_length。

輸出資料長度
COMMAREA 資料長度。 起始值為 MQCODL_AS_INPUT。

reply-to 格式
回覆訊息的格式名稱。 起始值是 MQFMT_NONE ， 長度為 MQ_FORMAT_LENGTH。

起始碼
交易起始碼。 欄位長度必須為 MQ_START_CODE_LENGTH。 起始值為 MQCSC_NONE。 下列是可能的其他值:

- MQCSC_START
- MQCSC_STARTDATA
- MQCSC_TERMINPUT
- MQCSC_NONE

作業結束狀態
作業結束狀態。 起始值為 MQCTES_NOSYNC。 下列是可能的其他值:

- MQCTES_COMMIT
- MQCTES_BACKOUT
- MQCTES_ENDTASK
- MQCTES_NOSYNC

交易 ID
要連接的交易 ID。 起始值必須包含空白，且長度必須為 MQ_TRANSACTION_ID_LENGTH。 僅當 **uow 控制項** 具有值 MQCUOWC_FIRST 或 MQCUOWC_ONLY 時，此欄位才適用。

UOW 控制項
UOW 控制。 起始值為 MQCUOWC_ONLY。 下列是可能的其他值:

- MQCUOWC_FIRST
- MQCUOWC_MIDDLE
- MQCUOWC_LAST
- 僅 MQCUOWC_ONLY
- MQCUOWC_COMMIT
- MQCUOWC_BACKOUT
- MQCUOWC_CONTINUE

version
MQCIH 版本號碼。 起始值為 MQCIH_VERSION_2。 唯一其他受支援的值是 MQCIH_VERSION_1。

建構子

ImqCICSBridge 標頭 ();
預設建構子。

ImqCICSBridgeHeader(const ImqCICSBridgeHeader & 標頭);
複製建構子。

超載 ImqItem 方法

virtual ImqBoolean copyOut(ImqMessage & 訊息);
在開始時將 MQCIH 資料結構插入訊息緩衝區，進一步移動現有的訊息資料，並將訊息格式設為 MQFMT_CICS。

如需詳細資料，請參閱母類別方法說明。

virtual ImqBoolean pasteIn(ImqMessage & 訊息);
從訊息緩衝區讀取 MQCIH 資料結構。 若要成功， *msg* 物件的編碼必須是 MQENC_NATIVE。 擷取具有 MQGMO_CONVERT 至 MQENC_NATIVE 的訊息。 若要成功， ImqMessage 格式必須是 MQFMT_CICS。

如需詳細資料，請參閱母類別方法說明。

物件方法 (public)

void operator = (const ImqCICSBridgeHeader & 標頭);
從 標頭複製實例資料，以取代現有的實例資料。

MQLONG ADSDDescriptor () const;
傳回 **ADS 描述子**的副本。

void setADSDescriptor(const MQLONG *descriptor* = MQCADSD_NONE);
設定 **ADS 描述子**。

ImqString attentionIdentifier() const;
傳回 **警示 ID** 的副本，並以尾端空格填補長度 MQ_ATTENTION_ID_LENGTH。

void setAttentionIdentifier (const char * *data* = 0);
設定 **警示 ID**，並以尾端空白填補長度 MQ_ATTENTION_ID_LENGTH。 如果未提供 資料 ， 請將 **注意 ID** 重設為起始值。

ImqString 鑑別器 () const;
傳回 **authenticator** 的副本，並以尾端空白填補長度 MQ_AUTHENTICATOR_LENGTH。

void setAuthenticator(const char * *data* = 0);
設定 **authenticator**，並以尾端空白填補長度 MQ_AUTHENTICATOR_LENGTH。 如果未提供 資料 ， 請將 **authenticator** 重設為起始值。

ImqString bridgeAbendCode () const;
傳回 **橋接器異常終止碼**的副本，並以尾端空白填補長度 MQ_ABEND_CODE_LENGTH。

ImqString bridgeCancelCode () const;
傳回 **橋接器取消代碼**的副本，並以尾端空格填補，長度為 MQ_CANCEL_CODE_LENGTH。

void setBridgeCancelCode(const char * *data* = 0);
設定 **橋接器取消碼**，並以尾端空白填補長度 MQ_CANCEL_CODE_LENGTH。 如果未提供 資料 ， 請將 **橋接器取消碼** 重設為起始值。

MQLONG bridgeCompletion 代碼 () const;
傳回 **橋接器完成碼**的副本。

MQLONG bridgeError 偏移 () const;
傳回 **橋接器錯誤偏移**的副本。

MQLONG bridgeReason 代碼 () const;
傳回 **橋接器原因碼**的副本。

MQLONG bridgeReturn 代碼 () const;
傳回 **橋接器回覆碼**。

MQLONG conversationalTask() const;
傳回 **交談式作業**的副本。

void setConversationalTask (const MQLONG *task* = MQCCT_NO);
設定 **交談式作業**。

MQLONG cursorPosition() const;
傳回 **游標位置**的副本。

void setCursorPosition (const MQLONG *position* = 0);
設定 **游標位置**。

MQLONG facilityKeep 時間 () const;
傳回 **facility keep time** 的副本。

void setFacilityKeepTime(const MQLONG *time* = 0);
設定 **機能保留時間**。

ImqString facilityLike() const;
傳回 **機能 (如)** 的副本，並以尾端空格填補長度 MQ_FACILITY_LIKE_LENGTH。

void setFacilityLike (const char * *name* = 0);
設定 **機能 (如)**，並以尾端空白填補長度 MQ_FACILITY_LIKE_LENGTH。 如果未提供 *name*，則會重設 **facility like** 起始值。

ImqBinary facilityToken() const;
傳回 **facility token** 的副本。

ImqBoolean setFacilityToken(const ImqBinary & 記號);
設定 **機能記號**。 *token* 的 **資料長度** 必須是零或 MQ_FACILITY_LENGTH。 如果成功，它會傳回 TRUE。

void setFacilityToken (const MQBYTE8 *token* = 0);
設定 **機能記號**。 *token* 可以是零，這與指定 MQCFAC_NONE 相同。 如果 *token* 非零，則必須處理二進位資料的 MQ_FACILITY_LENGTH 個位元組。 使用 MQCFAC_NONE 之類的預定值時，您可能需要進行強制轉型，以確保簽章相符。 例如， (MQBYTE *) MQCFAC_NONE。

ImqString 函數 () const;
傳回 **function** 的副本，並以尾端空格填補至長度 MQ_FUNCTION_LENGTH。

MQLONG getWait 間隔 () const;
傳回 **get wait interval** 的副本。

void setGetWaitInterval(const MQLONG *interval* = MQCGWI_DEFA
設定 **取得等待間隔**。

MQLONG linkType() const;
傳回 **鏈結類型**的副本。

void setLinkType (const MQLONG *type* = MQCLT_PROGRAM);
設定 **鏈結類型**。

ImqString nextTransactionID () const;
傳回 **下一個交易 ID** 資料的副本，並以尾端空白填補長度 MQ_TRANSACTION_ID_LENGTH。

MQLONG outputData 長度 () const;
傳回 **輸出資料長度**的副本。

void setOutputDataLength(const MQLONG *length* = MQCODL_AS_INPUT);
設定 **輸出資料長度**。

ImqString replyToFormat () const;
傳回 **reply-to 格式** 名稱的副本，並以尾端空格填補長度 MQ_FORMAT_LENGTH。

void setReplyToFormat(const char * *name* = 0);
設定 **reply-to 格式**，並以尾端空白填補長度 MQ_FORMAT_LENGTH。 如果未提供 *name*，請將 **reply-to 格式** 重設為起始值。

ImqString startCode() const;
傳回 **起始碼**的副本，並以長度 MQ_START_CODE_LENGTH 的尾端空白填補。

void setStartCode (const char * *data* = 0);
設定 **起始碼** 資料，並以尾端空白填補長度 MQ_START_CODE_LENGTH。 如果未提供 資料，請將 **start code** 重設為起始值。

MQLONG taskEnd 狀態 () const;
傳回 **作業結束狀態**的副本。

ImqString transactionIdentifier() const;
傳回 **交易 ID** 資料的副本，並以尾端空格填補長度 MQ_TRANSACTION_ID_LENGTH。

void setTransactionIdentifier (const char * *data* = 0);
設定 **交易 ID**，並以尾端空白填補長度 MQ_TRANSACTION_ID_LENGTH。 如果未提供 資料 ， 請將 **交易 ID** 重設為起始值。

MQLONG UOWControl () const;
傳回 **UOW 控制項**的副本。

void setUOWControl(const MQLONG *control* = MQCUOWC_ONLY);
設定 **UOW 控制項**。

MQLONG 版本 () const;
傳回 **version** 數字。

ImqBoolean setVersion(const MQLONG *version* = MQCIH_VERSION_2);
設定 **版本** 號碼。 如果成功，它會傳回 TRUE。

物件資料 (受保護)

MQLONG *olVersion*
在配置給 *opcih* 的儲存體中可容納的 MQCIH 版本號碼上限。

PMQCIH *opcih*
MQCIH 資料結構的位址。 配置的儲存體數量由 *olVersion* 指出。

原因碼

- MQRC_BINARY_DATA_LENGTH_ERROR
- MQRC_WRONG_VERSION

回覆碼

表 *614: ImqCICSBridge* 標頭類別回覆碼

回覆碼	函數	CompCode	原因	異常終止碼
MQCRC_OK				
MQCRC_BRIDGE_ERROR			MQFB_CICS	
MQCRC_MQ_API_ERROR	WebSphere MQ 呼叫名稱	WebSphere MQ CompCode	WebSphere MQ 原因	
MQ CRC_BRIDGE_TIMEOUT	WebSphere MQ 呼叫名稱	WebSphere MQ CompCode	WebSphere MQ 原因	
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++ 類別

此類別封裝 MQDLH 資料結構的特性。

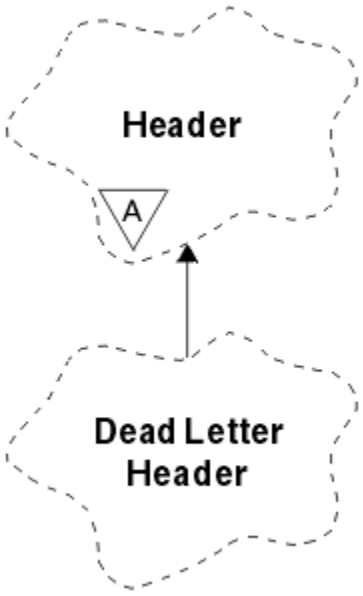


圖 *51: ImqDeadLetterHeader* 類別

此類別的物件通常由遇到無法處理之訊息的應用程式使用。 包含無法傳送郵件的標頭和訊息內容的新訊息被放置在無法傳送郵件的佇列上，並且訊息被丟棄。

- 第 1209 頁的『物件屬性』
- 第 1210 頁的『建構子』
- 第 1210 頁的『超載 ImqItem 方法』
- 第 1210 頁的『物件方法 (public)』
- 第 1211 頁的『物件資料 (受保護)』
- 第 1211 頁的『原因碼』

物件屬性

無法傳送郵件的原因碼
訊息抵達無法傳送郵件的佇列的原因。 起始值是 MQRC_NONE。

目的地佇列管理程式名稱
原始目的地佇列管理程式的名稱。 名稱是長度為 MQ_Q_MGR_NAME_LENGTH 的字串。 其起始值是空值。

目的地佇列名稱
原始目的地佇列的名稱。 名稱是長度為 MQ_Q_NAME_LENGTH 的字串。 其起始值是空值。

放置應用程式名稱
將訊息放到無法傳送郵件的佇列的應用程式名稱。 名稱是長度為 MQ_PUT_APPL_NAME_LENGTH 的字串。 其起始值是空值。

放置應用程式類型
將訊息放置在無法傳送郵件的佇列上的應用程式類型。 起始值為零。

放置日期
訊息放到無法傳送郵件的佇列的日期。 日期是長度為 MQ_PUT_DATE_LENGTH 的字串。 其起始值是空字串。

放置時間
訊息放到無法傳送郵件的佇列的時間。 時間是長度為 MQ_PUT_TIME_LENGTH 的字串。 其起始值是空字串。

建構子

ImqDeadLetterHeader( );
預設建構子。

ImqDeadLetterHeader(const ImqDeadLetterHeader & 標頭);
複製建構子。

超載 ImqItem 方法

虛擬 ImqBoolean copyOut(ImqMessage & 訊息);
在開始時將 MQDLH 資料結構插入訊息緩衝區，並進一步移動現有的訊息資料。 將 *msg* **格式** 設為 MQFMT_DEAD_LETTER_HEADER。

如需進一步詳細資料，請參閱 第 1216 頁的『ImqHeader C++ 類別』 頁面上的 ImqHeader 類別方法說明。

虛擬 ImqBoolean pasteIn(ImqMessage & 訊息);
從訊息緩衝區讀取 MQDLH 資料結構。

若要成功， ImqMessage **格式** 必須為 MQFMT_DEAD_LETTER_HEADER。

如需進一步詳細資料，請參閱 第 1216 頁的『ImqHeader C++ 類別』 頁面上的 ImqHeader 類別方法說明。

物件方法 (public)

void operator = (const ImqDeadLetterHeader & 標頭);
從 標頭複製實例資料，以取代現有的實例資料。

MQLONG deadLetterReasonCode() const ;
傳回 **無法傳送郵件的原因碼**。

void setDeadLetterReasonCode(const MQLONG *reason*);
設定 **無法傳送郵件的原因碼**。

ImqString destinationQueueManagerName() const ;
傳回 **目的地佇列管理程式名稱**，並除去任何尾端空白。

void setDestinationQueueManagerName(const char * *name*);
設定 **目的地佇列管理程式名稱**。 截斷超過 MQ_Q_MGR_NAME_LENGTH (48 個字元) 的資料。

ImqString destinationQueueName() const ;
傳回 **目的地佇列名稱**的副本，並除去任何尾端空白。

void setDestinationQueueName(const char * *name*);
設定 **目的地佇列名稱**。 截斷長於 MQ_Q_NAME_LENGTH (48 個字元) 的資料。

ImqString putApplicationName() const ;
傳回 **put 應用程式名稱**的副本，並除去任何尾端空白。

void setPutApplicationName(const char * *name* = 0);
設定 **put 應用程式名稱**。 截斷長於 MQ_PUT_APPL_NAME_LENGTH (28 個字元) 的資料。

MQLONG putApplication 類型() const ;
傳回 **put 應用程式類型**。

void setPutApplicationType(const MQLONG *type* = MQAT_NO_CONTEXT);
設定 **put 應用程式類型**。

ImqString putDate() const ;
傳回 **put date** 的副本，其中除去任何尾端空白。

void setPutDate(const char * *date* = 0);
設定 **放置日期**。 截斷超過 MQ_PUT_DATE_LENGTH (8 個字元) 的資料。

ImqString putTime() const ;
傳回 **放置時間**的副本，並除去任何尾端空白。

void setPutTime(const char * *time* = 0);
設定 **放置時間**。 截斷超過 MQ_PUT_TIME_LENGTH (8 個字元) 的資料。

物件資料 (受保護)

MQDLH *omqdlh*
MQDLH 資料結構。

原因碼

- MQRC_INCONSISTENT_FORMAT
- MQRC_STRUC_ID_ERROR
- MQRC_ENCODING_ERROR

ImqDistribution 列出 C++ 類別

此類別會封裝參照一個以上佇列的動態配送清單，以將一或多個訊息傳送至多個目的地。

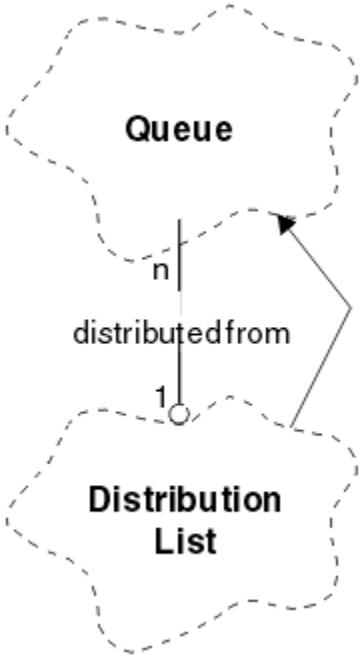


圖 *52: ImqDistribution* 清單類別

- 第 1211 頁的『物件屬性』
- 第 1212 頁的『建構子』
- 第 1212 頁的『物件方法 (public)』
- 第 1212 頁的『物件方法 (protected)』

物件屬性

第一個分散式佇列
類別中一個以上物件的第一個物件 (無特定順序)，其中 **配送清單參照** 會處理此物件。

最初沒有這類物件。 若要順利開啟 ImqDistribution 清單，必須至少有一個此類物件。

註: 當開啟 ImqDistributionList 物件時，會自動關閉任何參照它的已開啟物件。

建構子

ImqDistributionList ();
預設建構子。

ImqDistribution 清單 (const ImqDistributionList & 清單);
複製建構子。

物件方法 (public)

void operator = (const ImqDistributionList & 清單);
複製之前，會先取消參照參照 **此** 物件的所有物件。 在呼叫此方法之後，沒有任何物件會參照 **此** 物件。

*** firstDistributedQueue() const ;**
傳回 **第一個分散式佇列**。

物件方法 (protected)

void setFirstDistributedQueue(* *queue* = 0);
設定 **第一個分散式佇列**。

ImqError C++ 類別

此抽象類別提供與物件相關聯的錯誤相關資訊。

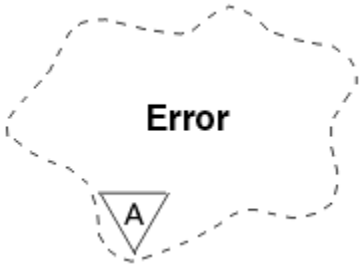


圖 *53: ImqError* 類別

- 第 1212 頁的『物件屬性』
- 第 1212 頁的『建構子』
- 第 1213 頁的『物件方法 (public)』
- 第 1213 頁的『物件方法 (protected)』
- 第 1213 頁的『原因碼』

物件屬性

完成碼 (completion code)
最新完成碼。 起始值為零。 下列是可能的其他值:

- MQCC_OK
- MQCC_WARNING
- MQCC_FAILED

原因碼 (reason code)
最新原因碼。 起始值為零。

建構子

ImqError( );
預設建構子。

ImqError(const ImqError & 錯誤);
複製建構子。

物件方法 (public)

void operator = (const ImqError & 錯誤);
從 *error* 複製實例資料，並取代現有的實例資料。

void clearErrorCodes();
將 **完成碼** 和 **原因碼** 都設為零。

MQLONG completionCode() const ;
傳回 **完成碼**。

MQLONG reasonCode() const ;
傳回 **原因碼**。

物件方法 (protected)

ImqBoolean checkReadPointer(const void * *pointer*, const size_t *length*);
驗證指標與長度的組合是否適用於唯讀存取權，並在成功時傳回 TRUE。

ImqBoolean checkWritePointer(const void * *pointer*, const size_t *length*);
驗證指標與長度的組合是否適用於讀寫存取，並在成功時傳回 TRUE。

void setCompletionCode(const MQLONG *code* = 0);
設定 **完成碼**。

void setReasonCode(const MQLONG *code* = 0);
設定 **原因碼**。

原因碼

- MQRC_BUFFER_ERROR

ImqGetMessageOptions C++ 類別

此類別封裝 MQGMO 資料結構

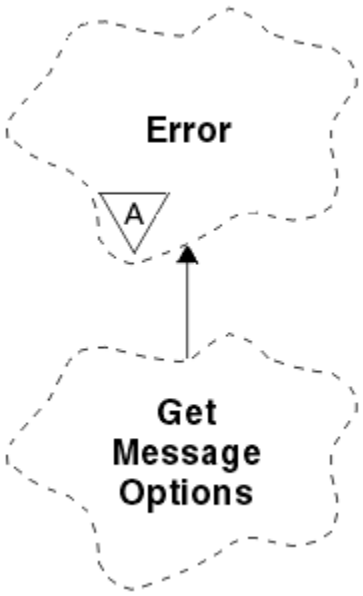


圖 *54: ImqGetMessageOptions* 類別

- 第 1214 頁的『物件屬性』
- 第 1215 頁的『建構子』

- 第 1215 頁的『物件方法 (public)』
- 第 1216 頁的『物件方法 (protected)』
- 第 1216 頁的『物件資料 (受保護)』
- 第 1216 頁的『原因碼』

物件屬性

群組狀態
訊息群組的訊息狀態。 起始值是 MQGS_NOT_IN_GROUP。 下列是可能的其他值:

- MQGS_MSG_IN_GROUP
- MQGS_LAST_MSG_IN_GROUP

符合選項
用於選取送入訊息的選項。 起始值為 MQMO_MATCH_MSG_ID | MQMO_MATCH_CORREL_ID。 下列是可能的其他值:

- MQMO_GROUP_ID
- MQMO_MATCH_MSG_SEQ_NUMBER
- MQMO_MATCH_OFFSET
- MQMO_MSG_TOKEN
- MQMO_NONE

訊息記號 (message token)
訊息記號。 長度為 MQ_MSG_TOKEN_LENGTH 的二進位值 (MQBYTE16)。 起始值為 MQMTOK_NONE。

選項
適用於訊息的選項。 起始值為 MQGMO_NO_WAIT。 下列是可能的其他值:

- MQGMO_WAIT
- MQ GMO_同步點
- 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
- MQ GMO_CONVERT
- MQ GMO_LOGICAL_ORDER
- MQ GMO_COMPLETE_MSG
- MQGMO_ALL_MSGS_AVAILABLE
- MQGMO_ALL_SEGMENTS_AVAILABLE
- MQGMO_NONE

解析的佇列名稱
已解析佇列名稱。 這個屬性是唯讀的。 名稱長度永不超過 48 個字元，且可以用空值填補該長度。 起始值是空字串。

傳回長度
傳回長度。 起始值為 MQRL_UNDEFINED。 這個屬性是唯讀的。

斷詞法 (segmentation)
將訊息分段的能力。 起始值為 MQSEG_INHIBITED。 可以使用其他值 MQSEG_ALLOWED。

區段狀態
訊息的分段狀態。 起始值為 MQSS_NOT_A_SEGMENT。 下列是可能的其他值:

- MQ SS_SEGMENT
- MQ SS_LAST_SEGMENT

同步點參與
在同步點控制下擷取訊息時為 TRUE。

等待間隔
類別 **get** 方法在等待適當訊息送達時暫停的時間長度 (如果尚無法使用的話)。 起始值為零，這會影響無限期等待。 其他值 MQWI_UNLIMITED 是可能的。 除非 **選項** 包括 MQGMO_WAIT ，否則會忽略此屬性。

建構子

ImqGetMessageOptions( );
預設建構子。

ImqGetMessageOptions(const ImqGetMessageOptions & 格莫);
複製建構子。

物件方法 (public)

void operator = (const ImqGetMessageOptions & 格莫);
從 *gmo* 複製實例資料，以取代現有實例資料。

MQCHAR groupStatus() const ;
傳回 **群組狀態**。

void setGroupStatus(const MQCHAR *status*);
設定 **群組狀態**。

MQLONG matchOptions() const ;
傳回 **比對選項**。

void setMatchOptions(const MQLONG *options*);
設定 **比對選項**。

ImqBinary messageToken() const;
傳回 **訊息記號**。

ImqBoolean setMessageToken(const ImqBinary & 記號);
設定 **訊息記號**。 *token* 的 **資料長度** 必須是零或 MQ_MSG_TOKEN_LENGTH。 如果成功，此方法會傳回 TRUE。

void setMessageToken (const MQBYTE16 *token* = 0);
設定 **訊息記號**。 *token* 可以是零，這與指定 MQMTOK_NONE 相同。 如果 *token* 為非零，則它必須處理二進位資料的 MQ_MSG_TOKEN_LENGTH 位元組。

使用預定值 (例如 MQMTOK_NONE) 時，您可能不需要進行強制轉型以確保簽章相符，例如 (MQBYTE *) MQMTOK_NONE。

MQLONG 選項() const ;
傳回 **選項**。

void setOptions(const MQLONG *options*);
設定 **選項**，包括 **同步點參與** 值。

ImqString resolvedQueueName() const ;
傳回 **已解析佇列名稱**的副本。

MQLONG returnedLength() const;
傳回 **returned length**。

MQCHAR segmentation() const ;
傳回 **segmentation**。

void setSegmentation(const MQCHAR *value*);
設定 **分段**。

MQCHAR segmentStatus() const ;
傳回 **區段狀態**。

void setSegmentStatus(const MQCHAR *status*);
設定 **區段狀態**。

ImqBoolean syncPointParticipation() const ;
傳回 **同步點參與** 值，如果 **選項** 包括 MQGMO_SYNCPOINT 或 MQGMO_SYNCPOINT_IF_PERSISTENT，則此值為 TRUE。

void setSyncPointParticipation(const ImqBoolean *sync*);
設定 **同步點參與** 值。 如果 *sync* 為 TRUE，請變更 **選項** 以包括 MQGMO_SYNCPOINT，以及同時排除 MQGMO_NO_SYNCPOINT 和 MQGMO_SYNCPOINT_IF_PERSISTENT。 如果 *sync* 為 FALSE，請變更 **選項** 以包括 MQGMO_NO_SYNCPOINT，以及同時排除 MQGMO_SYNCPOINT 和 MQGMO_SYNCPOINT_IF_PERSISTENT。

MQLONG waitInterval() const ;
傳回 **等待間隔**。

void setWaitInterval(const MQLONG *interval*);
設定 **等待間隔**。

物件方法 (protected)

static void setVersionSupported(const MQLONG);
設定 **MQGMO** 版本。 預設值為 **MQGMO_VERSION_3**。

物件資料 (受保護)

MQGMO *omqgmo*
MQGMO 第 2 版資料結構。 僅存取 MQGMO_VERSION_2 支援的 MQGMO 欄位。

PMQGMO *opgmo*
MQGMO 資料結構的位址。 此位址的版本號碼在 *olVersion* 中指出。 請先檢查版本號碼，然後再存取 MQGMO 欄位，以確保它們存在。

MQLONG *olVersion*
opgmo 所說明的 MQGMO 資料結構版本號碼。

原因碼

- MQRC_BINARY_DATA_LENGTH_ERROR

ImqHeader C++ 類別

此抽象類別封裝 MQDLH 資料結構的一般特性。

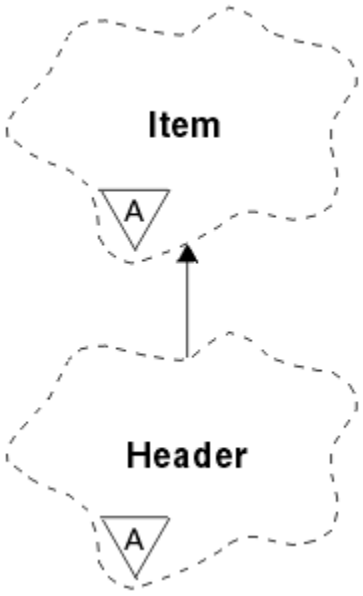


圖 *55: ImqHeader* 類別

- 第 1217 頁的『物件屬性』
- 第 1217 頁的『建構子』
- 第 1217 頁的『物件方法 (public)』

物件屬性

字元集
原始編碼字集 ID。 起始 MQCCSI_Q_MGR。

encoding
原始編碼。 起始 MQENC_NATIVE。

格式
原始格式。 起始 MQFMT_NONE。

標頭旗標
起始值如下:

- 零表示 ImqDeadLetterHeader 類別的物件
- ImqIMSBridge 標頭類別的物件 MQIIH_NONE
- ImqReference 標頭類別之物件的 MQRMHF_LAST
- ImqCICSBridge 標頭類別的物件 MQCIH_NONE
- MQWIH_NONE 適用於「ImqWork 標頭」類別的物件

建構子

ImqHeader( );
預設建構子。

ImqHeader(const ImqHeader & 標頭);
複製建構子。

物件方法 (public)

void operator = (const ImqHeader & 標頭);
從 標頭複製實例資料，以取代現有的實例資料。

虛擬 MQLONG characterSet() const ;
傳回 **字集**。

虛擬 void setCharacterSet(const MQLONG *ccsid* = MQCCSI_Q_MGR);
設定 **字集**。

虛擬 MQLONG encoding() const ;
傳回 **encoding**。

虛擬 void setEncoding(const MQLONG *encoding* = MQENC_NATIVE);
設定 **編碼**。

virtual ImqString format() const ;
傳回 **格式**的副本，包括尾端空白。

virtual void setFormat(const char * *name* = 0);
設定 **格式**，以尾端空白填補 8 個字元。

虛擬 MQLONG headerFlags() const ;
傳回 **標頭旗標**。

虛擬 void setHeaderFlags(const MQLONG *flags* = 0);
設定 **標頭旗標**。

ImqIMSBridge 標頭 C++ 類別

此類別封裝 MQIIH 資料結構的特性。

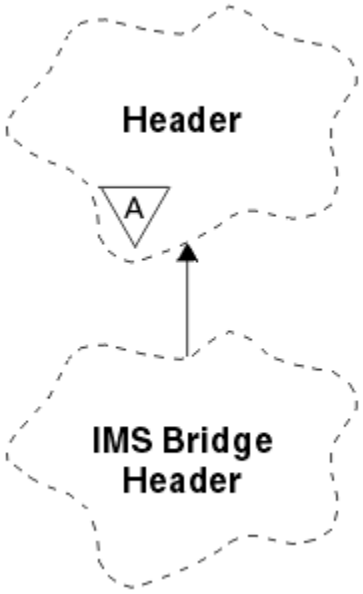


圖 *56: ImqIMSBridge* 標頭類別

此類別的物件由透過 WebSphere MQ for z/OS 傳送訊息至 IMS 橋接器的應用程式使用。

註: ImqHeader **字集** 及 **編碼** 必須具有預設值，且不得設為任何其他值。

- 第 1219 頁的『物件屬性』
- 第 1219 頁的『建構子』
- 第 1219 頁的『超載 ImqItem 方法』
- 第 1219 頁的『物件方法 (public)』
- 第 1220 頁的『物件資料 (受保護)』
- 第 1220 頁的『原因碼』

物件屬性

鑑別者 (authenticator)
RACF 密碼或通行證，長度為 MQ_AUTHENTICATOR_LENGTH。 起始值為 MQIAUT_NONE。

確定模式
確定模式。 如需 IMS 確定模式的相關資訊，請參閱 *OTMA* 使用手冊 。 起始值為 MQICM_COMMIT_THEN_SEND。 可以使用其他值 MQICM_SEND_THEN_COMMIT。

邏輯終端機置換
邏輯終端機置換，長度為 MQ_LTERM_OVERRIDE_LENGTH。 起始值是空字串。

訊息格式服務對映圖名稱
MFS 對映名稱，長度為 MQ_MFS_MAP_NAME_LENGTH。 起始值是空字串。

reply-to 格式
任何回覆的格式，長度為 MQ_FORMAT_LENGTH。 起始值為 MQFMT_NONE。

安全範圍
IMS 安全處理程序的範圍。 起始值是 MQISS_CHECK。 可以使用其他值 MQISS_FULL。

交易實例 ID
交易實例身分，長度為 MQ_TRAN_INSTANCE_ID_LENGTH 的二進位 (MQBYTE16) 值。 起始值為 MQITII_NONE。

交易狀態
IMS 交談的狀態。 起始值為 MQITS_NOT_IN_CONVERSATION。 可以使用額外值 MQITS_IN_CONVERSATION。

建構子

ImqIMSBridgeHeader ();
預設建構子。

ImqIMSBridge 標頭 (const ImqIMSBridgeHeader & 標頭);
複製建構子。

超載 ImqItem 方法

虛擬 ImqBoolean copyOut(ImqMessage & 訊息);
在開始時將 MQIIH 資料結構插入訊息緩衝區，並進一步移動現有的訊息資料。 將 *msg* **格式** 設為 MQFMT_IMS。

如需進一步詳細資料，請參閱母類別方法說明。

虛擬 ImqBoolean pasteIn(ImqMessage & 訊息);
從訊息緩衝區讀取 MQIIH 資料結構。

若要成功， *msg* 物件的 **編碼** 必須是 MQENC_NATIVE。 擷取具有 MQGMO_CONVERT 至 MQENC_NATIVE 的訊息。

若要成功， ImqMessage **格式** 必須是 MQFMT_IMS。

如需進一步詳細資料，請參閱母類別方法說明。

物件方法 (public)

void operator = (const ImqIMSBridgeHeader & 標頭);
從 標頭複製實例資料，以取代現有的實例資料。

ImqString 鑑別器() const ;
傳回 **authenticator** 的副本，並以尾端空白填補長度 MQ_AUTHENTICATOR_LENGTH。

void setAuthenticator(const char * *name*);
設定 **鑑別器**。

MQCHAR commitMode() const ;
傳回 **確定模式**。

void setCommitMode(const MQCHAR *mode*);
設定 **確定模式**。

ImqString logicalTerminalOverride() const ;
傳回 **邏輯終端機置換**的副本。

void setLogicalTerminalOverride(const char * *override*);
設定 **邏輯終端機置換**。

ImqString messageFormatServicesMap 名稱() const ;
傳回 **訊息格式服務對映名稱**的副本。

void setMessageFormatServicesMapName(const char * *name*);
設定 **訊息格式服務對映名稱**。

ImqString replyToFormat() const ;
傳回 **reply-to 格式**的副本，並以尾端空格填補長度 MQ_FORMAT_LENGTH。

void setReplyToFormat(const char * *format*);
設定 **reply-to 格式**，並以尾端空白填補長度 MQ_FORMAT_LENGTH。

MQCHAR securityScope() const ;
傳回 **安全範圍**。

void setSecurityScope(const MQCHAR *scope*);
設定 **安全範圍**。

ImqBinary transactionInstanceId() const ;
傳回 **交易實例 ID** 的副本。

ImqBoolean setTransactionInstanceId(const ImqBinary & *ID*);
設定 **交易實例 ID**。 *token* 的 **資料長度** 必須是零或 MQ_TRAN_INSTANCE_ID_LENGTH。 如果成功，此方法會傳回 TRUE。

void setTransactionInstanceId(const MQBYTE16 *id* = 0);
設定 **交易實例 ID**。 *id* 可以是零，這與指定 MQITII_NONE 相同。 如果 *id* 不是零，則必須解決二進位資料的 MQ_TRAN_INSTANCE_ID_LENGTH 個位元組。 使用預定值 (例如 MQITII_NONE) 時，您可能需要進行強制轉型以確保簽章相符，例如 (MQBYTE *) MQITII_NONE。

MQCHAR transactionState() const ;
傳回 **交易狀態**。

void setTransactionState(const MQCHAR *state*);
設定 **交易狀態**。

物件資料 (受保護)

MQIIH *omqiih*
MQIIH 資料結構。

原因碼

- MQRC_BINARY_DATA_LENGTH_ERROR
- MQRC_INCONSISTENT_FORMAT
- MQRC_ENCODING_ERROR
- MQRC_STRUC_ID_ERROR

ImqItem C++ 類別

這個抽象類別代表訊息內的項目 (可能是其中一個)。

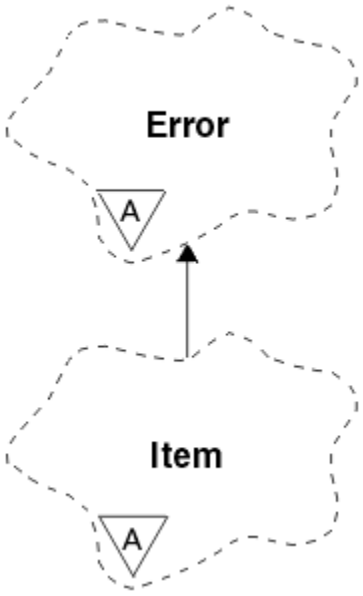


圖 *57: ImqItem* 類別

項目在訊息緩衝區中連結在一起。 每一個特殊化都與以結構 ID 開頭的特定資料結構相關聯。

此抽象類別中的多型方法容許在訊息中來回複製項目。 ImqMessage 類別 **readItem** 和 **writeItem** 方法提供另一種呼叫這些多型方法的樣式，這些多型方法對於應用程式更自然。

- 第 1221 頁的『物件屬性』
- 第 1221 頁的『建構子』
- 第 1221 頁的『類別方法 (public)』
- 第 1221 頁的『物件方法 (public)』
- 第 1222 頁的『原因碼』

物件屬性

結構 ID
資料結構開頭為四個字元的字串。 這個屬性是唯讀的。 請考量衍生類別的這個屬性。 它不會自動併入。

建構子

ImqItem( );
預設建構子。

ImqItem(const ImqItem & 項目);
複製建構子。

類別方法 (public)

static ImqBoolean structureId 是(const char * *structure-id-to-test*， const ImqMessage & 訊息);
如果送入 *msg* 中的下一個 ImqItem 的 **結構 ID** 與 *structure-id-to-test* 相同，則會傳回 TRUE。 下一個項目識別為目前由 ImqCache **資料指標**所定址的訊息緩衝區部分。 此方法根據 **結構 ID** ，因此不保證適用於所有 ImqItem 衍生類別。

物件方法 (public)

void operator = (const ImqItem & 項目);
從 項目複製實例資料，以取代現有的實例資料。

虛擬 ImqBoolean copyOut(ImqMessage & 訊息) = 0;
將此物件作為送出訊息緩衝區中的下一個項目寫入，並將它附加至任何現有項目。 如果寫入作業成功，請增加 ImqCache **資料長度**。 如果成功，此方法會傳回 TRUE。

置換此方法以使用特定子類別。

虛擬 ImqBoolean pasteIn(ImqMessage & 訊息) = 0;
從送入訊息緩衝區 破壞性 讀取此物件。 讀取具有破壞性，因為 ImqCache **資料指標** 已移動。 不過，緩衝區內容仍然相同，因此可以重設 ImqCache **資料指標**來重新讀取資料。

此物件的 (子) 類別必須與接下來在 *msg* 物件的訊息緩衝區中找到的 **結構 ID** 一致。

msg 物件的 **編碼** 應該是 MQENC_NATIVE。 建議使用 ImqMessage **編碼** 設為 MQENC_NATIVE ， 並使用 ImqGetMessageOptions **選項** (包括 MQGMO_CONVERT) 來擷取訊息。

如果讀取作業成功，則會減少 ImqCache **資料長度** 。 如果成功，此方法會傳回 TRUE。

置換此方法以使用特定子類別。

原因碼

- MQRC_ENCODING_ERROR
- MQRC_STRUC_ID_ERROR
- MQRC_INCONSISTENT_FORMAT
- MQRC_INSUFFICIENT_BUFFER
- MQRC_INSUFFICIENT_DATA

ImqMessage C++ 類別

此類別封裝 MQMD 資料結構，並同時處理訊息資料的建構及重新建構。

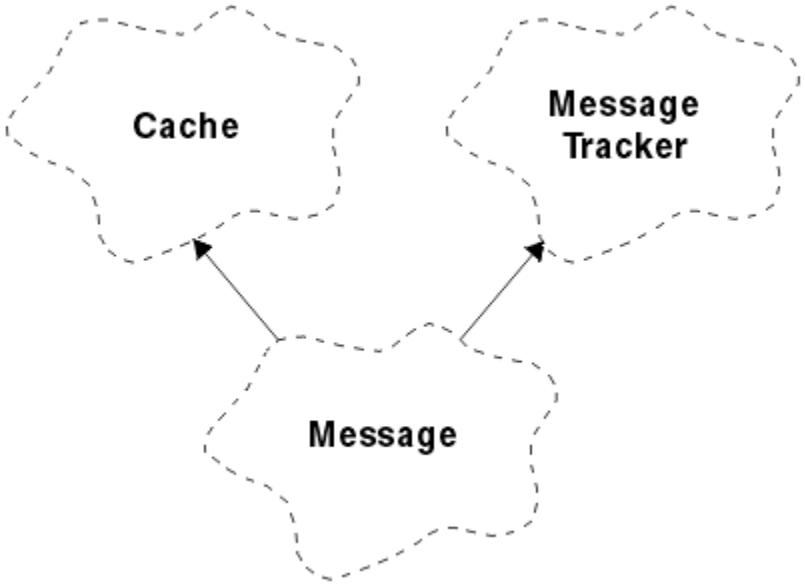


圖 *58: ImqMessage* 類別

- 第 1223 頁的『物件屬性』
- 第 1226 頁的『建構子』
- 第 1226 頁的『物件方法 (public)』
- 第 1228 頁的『物件方法 (protected)』
- 第 1228 頁的『物件資料 (受保護)』

物件屬性

應用程式 ID 資料
與訊息相關聯的識別資訊。 起始值是空字串。

應用程式原始資料
與訊息相關聯的原始資訊。 起始值是空字串。

取消計數
暫時擷取訊息並隨後取消的次數。 起始值為零。 這個屬性是唯讀的。

字元集
編碼字集 ID。 起始值為 MQCCSI_Q_MGR。 下列是可能的其他值:

- MQCCSI_INHERIT
- MQCCSI_Embedded

您也可以使用您選擇的「編碼字集 ID」。 如需此作業的相關資訊，請參閱 第 817 頁的『字碼頁轉換』。

encoding
訊息資料的機器編碼。 起始值為 MQENC_NATIVE。

到期
時間相依數量，用於控制 WebSphere MQ 在捨棄之前保留未擷取訊息的時間長度。 起始值為 MQEI_UNLIMITED。

格式
說明緩衝區中資料佈置的格式 (範本) 名稱。 超過 8 個字元的名稱會被截斷成 8 個字元。 名稱一律以空白填補至 8 個字元。 起始常數值是 MQFMT_NONE。 下列是可能的其他常數:

- 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

您也可以使用您選擇的應用程式特定字串。 如需此作業的相關資訊，請參閱訊息描述子 (MQMD) 的 第 376 頁的『格式 (MQCHAR8)』 欄位。

訊息旗標
分段控制資訊。 起始值為 MQMF_SEGMENTATION_INHIBITED。 下列是可能的其他值:

- 容許 MQMF_SEGMENTATION_ALLOWED
- MQMF_MSG_IN_GROUP
- MQMF_LAST_MSG_IN_GROUP

- MQMF_SEGMENT
- MQMF_LAST_SEGMENT
- 無 MQMF_NONE

訊息類型
訊息的廣泛分類。 起始值為 MQMT_DATAGRAM。 下列是可能的其他值:

- MQMT_SYSTEM_FIRST
- MQMT_SYSTEM_LAST
- MQMT_DATAGRAM
- MQMT_REQUEST
- MQMT_REPLY
- MQMT_REPORT
- MQMT_APPL_FIRST
- MQMT_APPL_LAST

您也可以使用您選擇的應用程式特定值。 如需此作業的相關資訊，請參閱訊息描述子 (MQMD) 的 第 384 頁的『MsgType (MQLONG)』 欄位。

偏移
偏移資訊。 起始值為零。

原始長度
分段訊息的原始長度。 起始值為 MQOL_UNDEFINED。

持續保存
指出訊息很重要，且必須隨時使用持續性儲存體來備份。 此選項意味著效能損失。 起始值為 MQPER_PERSISTENCE_AS_Q_DEF。 下列是可能的其他值:

- MQPER_PERSISTENT
- MQPER_NOT_PERSISTENT

priority
傳輸及遞送的相對優先順序。 具有相同優先順序的訊息通常會以提供的相同順序遞送 (雖然必須滿足數個準則才能保證如此)。 起始值為 MQPRI_PRIORITY_AS_Q_DEF。

內容驗證
指定是否應在設定訊息內容時驗證內容。 起始值為 **MQCMHO_DEFAULT_VALIDATION**。 下列是可能的其他值:

- MQ 指令 _ 驗證
- MQCMHO_NO_VALIDATION

下列方法會處理 **內容驗證**:

MQLONG propertyValidation() const;
傳回 **內容驗證** 選項。

void setPropertyValidation (const MQLONG *option*);
設定 **內容驗證** 選項。

放置應用程式名稱
放置訊息的應用程式名稱。 起始值是空字串。

放置應用程式類型
放置訊息的應用程式類型。 起始值為 MQAT_NO_CONTEXT。 下列是可能的其他值:

- 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
- MQ 視窗
- MQAT_WINDOWS_NT
- MQAT_XCF
- MQAT_DEFAULT
- MQAT_UNKNOWN
- MQAT_USER_FIRST
- MQAT_USER_LAST

您也可以使用您選擇的應用程式特定字串。 如需此作業的相關資訊，請參閱訊息描述子 (MQMD) 的 第 389 頁的『PutAppl 類型 (MQLONG)』 欄位。

放置日期
放置訊息的日期。 起始值是空字串。

放置時間
放置訊息的時間。 起始值是空字串。

回覆目的地佇列管理程式名稱
應將任何回覆傳送至其中的佇列管理程式名稱。 起始值是空字串。

回覆目的地佇列名稱
應將任何回覆傳送至其中的佇列名稱。 起始值是空字串。

report
與訊息相關聯的意見資訊。 起始值為 MQRO_NONE。 下列是可能的其他值:

- 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

其中 * 指出 WebSphere MQ for z/OS 上不支援的值。

序號 (sequence number)
識別群組內訊息的順序資訊。 起始值為 1。

訊息長度總計
在最近一次嘗試讀取訊息期間可用的位元組數。 如果最後一則訊息被截斷，或由於可能發生截斷而未讀取最後一則訊息，則此數字將大於 ImqCache **訊息長度** 。 這個屬性是唯讀的。 起始值為零。

在任何涉及截斷訊息的狀況下，此屬性都很有用。

使用者 ID
與訊息相關聯的使用者身分。 起始值是空字串。

建構子

ImqMessage( );
預設建構子。

ImqMessage(const ImqMessage & 訊息);
複製建構子。 如需詳細資料，請參閱 **operator =** 方法。

物件方法 (public)

void operator = (const ImqMessage & 訊息);
從 *msg* 複製 MQMD 和訊息資料。 如果使用者已提供此物件的緩衝區，則複製的資料量會限制為可用的緩衝區大小。 否則，系統會確保有足夠大小的緩衝區可供複製的資料使用。

ImqString applicationIdData() const ;
傳回 **應用程式 ID 資料**的副本。

void setApplicationIdData(const char * *data* = 0);
設定 **應用程式 ID 資料**。

ImqString applicationOriginData() const ;
傳回 **應用程式原始資料**的副本。

void setApplicationOriginData(const char * *data* = 0);
設定 **應用程式原始資料**。

MQLONG backoutCount() const ;
傳回 **取消計數**。

MQLONG characterSet() const ;
傳回 **字集**。

void setCharacterSet(const MQLONG *ccsid* = MQCCSI_Q_MGR);
設定 **字集**。

MQLONG encoding() const ;
傳回 **encoding**。

void setEncoding(const MQLONG *encoding* = MQENC_NATIVE);
設定 **編碼**。

MQLONG expiry() const ;
傳回 **expiry**。

void setExpiry(const MQLONG *expiry*);
設定 **expiry**。

ImqString format() const ;
傳回 **格式**的副本，包括尾端空白。

ImqBoolean formatIs(const char * *format-to-test*) const ;
如果 **format** 與 *format-to-test* 相同，則傳回 TRUE。

void setFormat(const char * *name* = 0);
設定 **格式**，以尾端空白填補到八個字元。

MQLONG messageFlags() const ;
傳回 **訊息旗標**。

void setMessageFlags(const MQLONG *flags*);
設定 **訊息旗標**。

MQLONG messageType() const ;
傳回 **訊息類型**。

void setMessageType(const MQLONG *type*);
設定 **訊息類型**。

MQLONG offset() const ;
傳回 **offset**。

void setOffset(const MQLONG *offset*);
設定 **偏移**。

MQLONG originalLength() const ;
傳回 **原始長度**。

void setOriginalLength(const MQLONG *length*);
設定 **原始長度**。

MQLONG persistence() const ;
傳回 **持續性**。

void setPersistence(const MQLONG *persistence*);
設定 **持續性**。

MQLONG priority() const ;
傳回 **priority**。

void setPriority(const MQLONG *priority*);
設定 **優先順序**。

ImqString putApplicationName() const ;
傳回 **put 應用程式名稱**的副本。

void setPutApplicationName(const char * *name* = 0);
設定 **put 應用程式名稱**。

MQLONG putApplication 類型() const ;
傳回 **put 應用程式類型**。

void setPutApplicationType(const MQLONG *type* = MQAT_NO_CONTEXT);
設定 **put 應用程式類型**。

ImqString putDate() const ;
傳回 **放置日期**的副本。

void setPutDate(const char * *date* = 0);
設定 **放置日期**。

ImqString putTime() const ;
傳回 **放置時間**的副本。

void setPutTime(const char * *time* = 0);
設定 **放置時間**。

ImqBoolean readItem(ImqItem & 項目);
使用 ImqItem **pasteIn** 方法，從訊息緩衝區讀取至 *item* 物件。 如果成功，它會傳回 TRUE。

ImqString replyToQueueManagerName() const ;
傳回 **reply-to 佇列管理程式名稱**的副本。

void setReplyToQueueManagerName(const char * *name* = 0);
設定 **回覆目的地佇列管理程式名稱**。

ImqString replyToQueueName() const ;
傳回 **reply-to 佇列名稱**的副本。

void setReplyToQueueName(const char * *name* = 0);
設定 **回覆目的地佇列名稱**。

MQLONG 報告() const ;
傳回 **報告**。

void setReport(const MQLONG *report*);
設定 **報告**。

MQLONG sequenceNumber() const ;
傳回 **序號**。

void setSequenceNumber(const MQLONG *number*);
設定 **序號**。

size_t totalMessageLength() const ;
傳回 **訊息長度總計**。

ImqString userId() const ;
傳回 **使用者 ID** 的副本。

void setUserId(const char * *id* = 0);
設定 **使用者 ID**。

ImqBoolean writeItem(ImqItem & 項目);
使用 ImqItem **copyOut** 方法，從 *item* 物件寫入訊息緩衝區。 撰寫可以採用插入、取代或附加的形式: 這取決於 *item* 物件的類別。 如果成功，此方法會傳回 TRUE。

物件方法 (protected)

static void setVersionSupported(const MQLONG);
設定 **MQMD 版本**。 預設為 **MQMD_VERSION_2**。

物件資料 (受保護)

MQMD1 *omqmd*
(僅限 WebSphere MQ for z/OS 。) MQMD 資料結構。

MQMD2 *omqmd*
(z/OS 以外的平台。) MQMD 資料結構。

ImqMessageTracker C++ 類別

此類別會封裝可與任一物件相關聯的 ImqMessage 或 ImqQueue 物件的那些屬性。

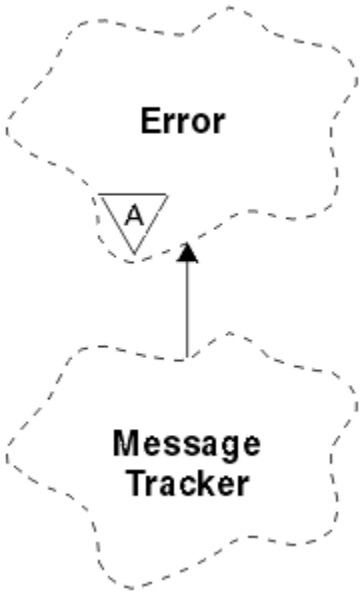


圖 *59: ImqMessage* 追蹤器類別

此類別與 第 1183 頁的『ImqMessage 追蹤器交互參照』 中列出的 MQI 呼叫相關。

- 第 1229 頁的『物件屬性』
- 第 1230 頁的『建構子』
- 第 1230 頁的『物件方法 (public)』
- 第 1231 頁的『原因碼』

物件屬性

帳戶記號
長度 MQ_ACCOUNTING_TOKEN_LENGTH 的二進位值 (MQBYTE32)。 起始值為 MQACT_NONE。

相關性 ID
您指派以與訊息產生關聯的二進位值 (MQBYTE24) , 長度為 MQ_CORREL_ID_LENGTH。 起始值為 MQCI_NONE。 可以使用其他值 MQCI_NEW_SESSION。

意見
要隨訊息一起傳送的意見資訊。 起始值為 MQFB_NONE。 下列是可能的其他值:

- 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
- MQ fb_data_length_too_big
- MQ fb_BUFFER_溢位

- 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_XX_ENCODE_CASE_ONE 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

您也可以使用您選擇的應用程式特定字串。 如需此作業的相關資訊，請參閱訊息描述子 (MQMD) 的 第 372 頁的『回饋 (MQLONG)』 欄位。

群組 ID
佇列中長度 MQ_GROUP_ID_LENGTH 唯一的二進位值 (MQBYTE24)。 起始值為 MQGI_NONE。

訊息 ID
佇列中長度 MQ_MSG_ID_LENGTH 唯一的二進位值 (MQBYTE24)。 起始值為 MQMI_NONE。

建構子

ImqMessageTracker ();
預設建構子。

ImqMessageTracker (const ImqMessageTracker & 追蹤器);
複製建構子。 如需詳細資料，請參閱 **operator =** 方法。

物件方法 (public)

void operator = (const ImqMessageTracker & 追蹤器);
從 追蹤器複製實例資料，並取代現有實例資料。

ImqBinary accountingToken() const ;
傳回 **帳戶記號**的副本。

ImqBoolean setAccounting 記號(const ImqBinary & 記號);
設定 **帳戶記號**。 *token* 的 **資料長度** 必須是零或 MQ_ACCOUNTING_TOKEN_LENGTH。 如果成功，此方法會傳回 TRUE。

void setAccountingToken(const MQBYTE32 *token* = 0);
設定 **帳戶記號**。 *token* 可以是零，這與指定 MQACT_NONE 相同。 如果 *token* 非零，則必須解決二進位資料的 MQ_ACCOUNTING_TOKEN_LENGTH 個位元組。 使用預定值 (例如 MQACT_NONE) 時，您可能需要進行強制轉型以確保簽章相符; 例如 (MQBYTE *) MQACT_NONE。

ImqBinary correlationId() const ;
傳回 **相關性 ID** 的副本。

ImqBoolean setCorrelationID(const ImqBinary & 記號);
設定 **相關性 ID**。 *token* 的 **資料長度** 必須是零或 MQ_CORREL_ID_LENGTH。 如果成功，此方法會傳回 TRUE。

void setCorrelationId(const MQBYTE24 *id* = 0);
設定 **相關性 ID**。 *id* 可以是零，這與指定 MQCI_NONE 相同。 如果 *id* 不是零，則必須解決二進位資料的 MQ_CORREL_ID_LENGTH 個位元組。 使用預定值 (例如 MQCI_NONE) 時，您可能需要進行強制轉型以確保簽章相符; 例如 (MQBYTE *) MQCI_NONE。

MQLONG feedback() const ;
傳回 **意見**。

void setFeedback(const MQLONG *feedback*);
設定 **意見**。

ImqBinary groupId() const ;
傳回 **群組 ID** 的副本。

ImqBoolean setGroupID(const ImqBinary & 記號);
設定 **群組 ID**。 *token* 的 **資料長度** 必須是零或 MQ_GROUP_ID_LENGTH。 如果成功，此方法會傳回 TRUE。

void setGroupId(const MQBYTE24 *id* = 0);
設定 **群組 ID**。 *id* 可以是零，這與指定 MQGI_NONE 相同。 如果 *id* 不是零，則必須解決二進位資料的 MQ_GROUP_ID_LENGTH 個位元組。 使用預定值 (例如 MQGI_NONE) 時，您可能需要進行強制轉型以確保簽章相符，例如 (MQBYTE *) MQGI_NONE。

ImqBinary messageId() const ;
傳回 **訊息 ID** 的副本。

ImqBoolean setMessageID(const ImqBinary & 記號);
設定 **訊息 ID**。 *token* 的 **資料長度** 必須是零或 MQ_MSG_ID_LENGTH。 如果成功，此方法會傳回 TRUE。

void setMessageId(const MQBYTE24 *id* = 0);
設定 **訊息 ID**。 *id* 可以是零，這與指定 MQMI_NONE 相同。 如果 *id* 非零，則必須解決二進位資料的 MQ_MSG_ID_LENGTH 個位元組。 使用預定值 (例如 MQMI_NONE) 時，您可能需要進行強制轉型以確保簽章相符，例如 (MQBYTE *) MQMI_NONE。

原因碼

- MQRC_BINARY_DATA_LENGTH_ERROR

ImqNamelist C++ 類別

這個類別會封裝名稱清單。

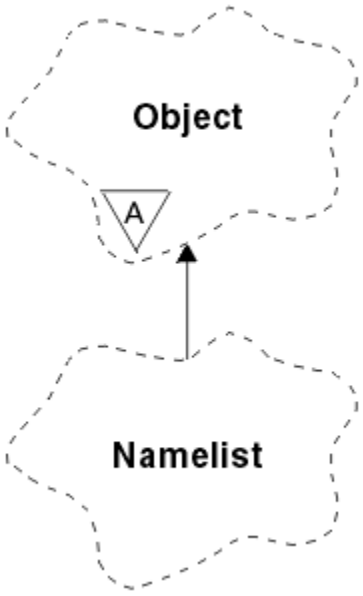


圖 *60: ImqNamelist* 類別

此類別與 第 1184 頁的『ImqNamelist 交互參照』中列出的 MQI 呼叫相關。

- 第 1232 頁的『物件屬性』
- 第 1232 頁的『建構子』
- 第 1232 頁的『物件方法 (public)』
- 第 1233 頁的『原因碼』

物件屬性

名稱計數
名單名稱中的物件名稱數。 這個屬性是唯讀的。

名單名稱
物件名稱，其數目由 **名稱計數**指出。 這個屬性是唯讀的。

建構子

ImqNamelist( );
預設建構子。

ImqNamelist(const ImqNamelist & 清單);
複製建構子。 ImqObject **開啟狀態** 為 false。

ImqNamelist(const char * *name*);
將 ImqObject 名稱設為 **name**。

物件方法 (public)

void operator = (const ImqNamelist & 清單);
從 *list* 複製實例資料，以取代現有的實例資料。 ImqObject **開啟狀態** 為 false。

ImqBoolean nameCount(MQLONG & 計數);
提供 **名稱計數**的副本。 如果成功，它會傳回 TRUE。

MQLONG nameCount ();
傳回 **名稱計數** ，且不指出任何可能的錯誤。

ImqBoolean namelistName (const MQLONG 索引, ImqString & 名);
依零型索引提供一個 **名單名稱** 的副本。 如果成功，它會傳回 TRUE。

ImqString namelistName (const MQLONG *index*);
依從零開始的索引傳回其中一個 **namelist 名稱**，不指出任何可能的錯誤。

原因碼

- MQRC_INDEX_ERROR
- MQRC_INDEX_NOT_PRESENT

ImqObject C++ 類別

這個類別是抽象的。 當這個類別的物件毀損時，會自動關閉它，且會切斷它的 ImqQueue 管理程式連線。

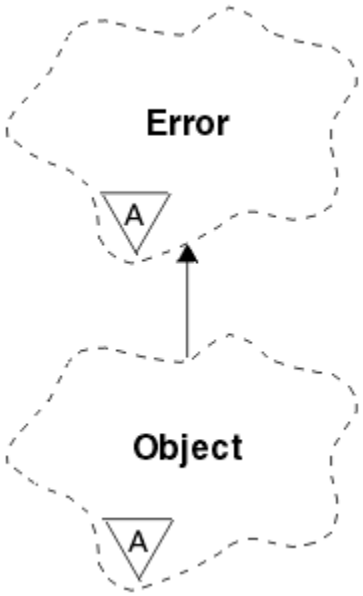


圖 *61: ImqObject* 類別

此類別與 第 1184 頁的『ImqObject 交互參照』中列出的 MQI 呼叫相關。

- 第 1233 頁的『類別屬性』
- 第 1234 頁的『物件屬性』
- 第 1235 頁的『建構子』
- 第 1235 頁的『類別方法 (public)』
- 第 1235 頁的『物件方法 (public)』
- 第 1237 頁的『物件方法 (protected)』
- 第 1237 頁的『物件資料 (受保護)』
- 第 1237 頁的『原因碼』

類別屬性

行為 (behavior)
控制隱含開啟的行為。

IMQ_IMPL_OPEN (8L)
容許隱含的開啟。 這是預設值。

物件屬性

變更日期
變更日期。 這個屬性是唯讀的。

變更時間
變更時間。 這個屬性是唯讀的。

替代使用者 ID
替代使用者 ID ，最多為 MQ_USER_ID_length 個字元。 起始值是空字串。

替代安全 ID
替代安全 ID。 長度為 MQ_SECURITY_ID_LENGTH 的二進位值 (MQBYTE40)。 起始值是 MQSID_NONE。

關閉選項
關閉物件時套用的選項。 起始值為 MQCO_NONE。 在隱含重新開啟作業期間會忽略此屬性，其中一律使用 MQCO_NONE 值。

連線參照
ImqQueueManager 物件的參照，提供與 (本端) 佇列管理程式的必要連線。 對於 ImqQueueManager 物件，它是物件本身。 起始值為零。

註: 請勿將此名稱與識別具名佇列之佇列管理程式 (可能遠端) 的 **佇列管理程式名稱** 混淆。

description
佇列管理程式、佇列、名單或程序的敘述性名稱 (最多 64 個字元)。 這個屬性是唯讀的。

name
佇列管理程式、佇列、名稱清單或處理程序的名稱 (最多 48 個字元)。 起始值是空字串。 在 **開啟** 之後，模型佇列的名稱會變更為產生的動態佇列名稱。

註: ImqQueue 管理程式可以具有空值名稱，代表預設佇列管理程式。 順利 **開啟**之後，名稱會變更為實際佇列管理程式。 「ImqDistribution 清單」是動態的，且必須具有空值名稱。

下一個受管理物件
這是此類別的下一個物件，沒有特定順序，具有與此物件相同的 **連線參照** 。 起始值為零。

開啟選項
開啟物件時套用的選項。 起始值為 MQOO_INQUIRE。 有兩種方法可以設定適當的值:

1. 請勿設定 **開啟選項** ，且不要使用 **open** 方法。 WebSphere MQ 會自動調整 **開啟選項** ，並視需要自動開啟、重新開啟及關閉物件。 這可能會導致不必要的重新開啟作業，因為 WebSphere MQ 會使用 **openFor** 方法，而且這只會漸進式地新增 **開啟選項** 。
2. 在使用任何導致 MQI 呼叫的方法之前，請先設定 **開啟選項** (請參閱 第 1178 頁的『C++ 及 MQI 交互參照』)。 這可確保不會發生不必要的重新開啟作業。 如果可能發生任何可能的重新開啟問題，請明確設定開啟選項 (請參閱 重新開啟)。

   如果您使用 **open** 方法，則 **必須** 確保 **open options** 首先適當。 不過，使用 **open** 方法並非必要; WebSphere MQ 仍會顯示與案例 1 相同的行為，但在此情況下，該行為是有效的。

零不是有效值; 請在嘗試開啟物件之前設定適當的值。 這可以使用 **setOpen 選項**(*lOpen* 選項) 來完成。 後面接著 **open**() 或 **openFor**(*lRequiredOpenOption*)。

註:

1. 在發佈清單的 **open** 方法期間， MQOO_OUTPUT 會替代 MQOO_INQUIRE ，因為 MQOO_OUTPUT 目前是唯一有效的 **open 選項** 。 不過，最好一律在使用 **open** 方法的應用程式中明確設定 MQOO_OUTPUT。
2. 如果您要使用類別的 **resolve_resolve_names** 及 **resolved queue name** 屬性，請指定 MQOO_RESOLVE_NAMES。

開啟狀態
物件是開啟 (TRUE) 還是關閉 (FALSE)。 起始值為 FALSE。 這個屬性是唯讀的。

前一個受管理物件
此類別的前一個物件 (無特定順序) 具有與此物件相同的 **連線參照** 。 起始值為零。

佇列管理程式 ID
佇列管理程式 ID。 這個屬性是唯讀的。

建構子

ImqObject( );
預設建構子。

ImqObject(const ImqObject & 物件);
複製建構子。 **開啟狀態** 將為 FALSE。

類別方法 (public)

static MQLONG behavior ();
傳回 **行為**。

void setBehavior(const MQLONG *behavior* = 0);
設定 **行為**。

物件方法 (public)

void operator = (const ImqObject & 物件);
必要的話，執行關閉，並從 物件複製實例資料。 **開啟狀態** 將為 FALSE。

ImqBoolean alterationDate(ImqString & 日期);
提供 **變更日期**的副本。 如果成功，它會傳回 TRUE。

ImqString alterationDate( );
傳回 **變更日期** ，不指出任何可能的錯誤。

ImqBoolean alterationTime(ImqString & 時間);
提供 **變更時間**的副本。 如果成功，它會傳回 TRUE。

ImqString alterationTime( );
傳回 **變更時間** ，不指出任何可能的錯誤。

ImqString alternateUserId() const ;
傳回 **替代使用者 ID** 的副本。

ImqBoolean setAlternateUserId(const char * *id*);
設定 **替代使用者 ID**。 只有在 **開啟狀態** 為 FALSE 時，才能設定 **替代使用者 ID** 。 如果成功，此方法會傳回 TRUE。

ImqBinary alternateSecurityId () const;
傳回 **替代安全 ID** 的副本。

ImqBoolean setAlternateSecurityId(const ImqBinary & 記號);
設定 **替代安全 ID**。 只有在 **開啟狀態** 為 FALSE 時，才能設定 **替代安全 ID** 。 *token* 的資料長度必須是零或 MQ_SECURITY_ID_LENGTH。 如果成功，它會傳回 TRUE。

ImqBoolean setAlternateSecurityId(const MQBYTE* *token* = 0);
設定 **替代安全 ID**。 *token* 可以是零，這與指定 MQSID_NONE 相同。 如果 *token* 不是零，則必須解決二進位資料的 MQ_SECURITY_ID_LENGTH 個位元組。 使用預定值 (例如 MQSID_NONE) 時，您可能需要進行強制轉型以確保簽章相符; 例如 (MQBYTE *) MQSID_NONE。

只有在 **開啟狀態** 為 TRUE 時，才能設定 **替代安全 ID** 。 如果成功，它會傳回 TRUE。

ImqBoolean setAlternateSecurityId(const unsigned char * *id* = 0);
設定 **替代安全 ID**。

ImqBoolean close();
將 **開啟狀態** 設為 FALSE。 如果成功，它會傳回 TRUE。

MQLONG closeOptions() const ;
傳回 **關閉選項**。

void setCloseOptions(const MQLONG *options*);
設定 **關閉選項**。

ImqQueueManager * connectionReference() const ;
傳回 **連線參照**。

void setConnection 參照(ImqQueue 管理員與 經理);
設定 **連線參照**。

void setConnectionReference(ImqQueueManager * ***manager*** **= 0);**
設定 **連線參照**。

虛擬 ImqBoolean 描述(ImqString & 描述) = 0;
提供 **說明**的副本。 如果成功，它會傳回 TRUE。

ImqString description();
傳回 **description** 的副本，不含任何可能錯誤的指示。

虛擬 ImqBoolean 名(ImqString & 名);
提供 **名稱**的副本。 如果成功，它會傳回 TRUE。

ImqString name();
傳回 **name** 的副本，但不指出可能的錯誤。

ImqBoolean setName(const char * ***name*** **= 0);**
設定 **名稱**。 只有在 **開啟狀態** 為 FALSE 時，且對於 ImqQueue 管理程式，當 **連線狀態** 為 FALSE 時，才能設定 **名稱** 。 如果成功，它會傳回 TRUE。

ImqObject * nextManagedObject() const ;
傳回 **下一個受管理物件**。

ImqBoolean open();
視需要開啟物件，並在其他屬性之間使用 **開啟選項** 及 **名稱**，將 **開啟狀態** 變更為 TRUE。 必要的話，此方法會使用 **連線參照** 資訊及 ImqQueueManager **connect** 方法，以確保 ImqQueueManager **連線狀態** 為 TRUE。 它會傳回 **開啟狀態**。

ImqBoolean openFor(const MQLONG ***required-options*** **= 0);**
嘗試使用 **open options** 或 **open options** (保證 *required-options* 參數值所隱含的行為) 來確保開啟物件。

如果 *required-options* 為零，則需要輸入，且任何輸入選項都足夠。 因此，如果 **開啟選項** 已包含下列其中一項:

- MQOO_INPUT_AS_Q_DEF
- MQOO_INPUT_SHARED
- MQOO_INPUT_EXCLUSIVE

開啟選項 已令人滿意且未變更; 如果 **開啟選項** 尚未包含任何這些選項，則 MQOO_INPUT_AS_Q_DEF 會設定在 **開啟選項**中。

如果 *required-options* 不是零，則會將必要選項新增至 **開啟選項**; 如果 *required-options* 是任何這些選項，則會重設其他選項。

如果任何 **開啟選項** 已變更且物件已開啟，則會暫時關閉物件並重新開啟，以調整 **開啟選項**。

如果成功，它會傳回 TRUE。 成功指出以適當的選項開啟物件。

MQLONG openOptions() const ;
傳回 **開啟選項**。

ImqBoolean setOpen 選項(const MQLONG 選項);
設定 **開啟選項**。 只有在 **開啟狀態** 為 FALSE 時，才能設定 **開啟選項** 。 如果成功，它會傳回 TRUE。

ImqBoolean openStatus() const ;
傳回 **開啟狀態**。

ImqObject * previousManagedObject() const ;
傳回 **前一個受管理物件**。

ImqBoolean queueManagerIdentifier(ImqString & ***ID*** **);**
提供 **佇列管理程式 ID** 的副本。 如果成功，它會傳回 TRUE。

ImqString queueManagerID ();
傳回 **佇列管理程式 ID**，但不指出任何可能的錯誤。

物件方法 (protected)

virtual ImqBoolean closeTemporarily();
重新開啟之前安全地關閉物件。 如果成功，它會傳回 TRUE。 此方法假設 **開啟狀態** 為 TRUE。

MQHCONN connectionHandle() const ;
傳回與 **連線參照**相關聯的 MQHCONN。 如果沒有 **連線參照**，或如果「管理程式」未連接，則此值為零。

ImqBoolean 查詢(連線 MQLONG *int-attr*, MQLONG 及 價值);
傳回整數值，其索引是 MQIA_ * 值。 如果發生錯誤，則值會設為 MQIAV_UNDEFINED。

ImqBoolean 查詢(連線 MQLONG *char-attr*, char * & 緩衝, 常數大小 _t 長度);
傳回字串，其索引是 MQCA_ * 值。

註: 這兩種方法都只會傳回單一屬性值。 如果需要多個值的 *Snapshot*，其中這些值在瞬間彼此一致，則 WebSphere MQ C++ 不會提供此機能，而且您必須搭配使用 MQINQ 呼叫與適當的參數。

virtual void openInformation 離散();
在 MQOPEN 呼叫之後立即從 MQOD 資料結構的變數區段分散資訊。

virtual ImqBoolean openInformationPrepare();
在 MQOPEN 呼叫之前立即準備 MQOD 資料結構的變數區段資訊，如果成功，則傳回 TRUE。

ImqBoolean set(const MQLONG *int-attr*, const MQLONG *value*);
設定 WebSphere MQ 整數屬性。

ImqBoolean set(const MQLONG *char-attr*, const char * *buffer*, const size_t *required-length*);
設定 WebSphere MQ 字元屬性。

void setNextManagedObject(const ImqObject * *object* = 0);
設定 **下一個受管理物件**。

注意: 只有在您確定不會岔斷受管理物件清單時，才使用此功能。

void setPreviousManagedObject(const ImqObject * *object* = 0);
設定 **前一個受管理物件**。

注意: 只有在您確定不會岔斷受管理物件清單時，才使用此功能。

物件資料 (受保護)

MQHOBJ *ohobj*
WebSphere MQ 物件控點 (只有在 **開啟狀態** 為 TRUE 時才有效)。

MQOD *omqod*
內嵌的 MQOD 資料結構。 配置給此資料結構的儲存體量是 MQOD 第 2 版所需要的。 檢查版本號碼 *(omqod.Version)* 並存取其他欄位，如下所示:

MQOD_VERSION_1
可以存取 *omqod* 中的所有其他欄位。

MQOD_VERSION_2
可以存取 *omqod* 中的所有其他欄位。

MQOD_VERSION_3
omqod.pmqod 是動態配置的較大 MQOD 的指標。 無法存取 *omqod* 中的其他欄位。 可以存取 *omqod.pmqod* 所說明的所有欄位。

註: *omqod.pmqod.Version* 可以小於 *omqod.Version*，指出 WebSphere MQ MQI 用戶端比 WebSphere MQ 伺服器具有更多功能。

原因碼

- MQRC_ATTRIBUTE_LOCKED

- MQRC_INCONSISTENT_OBJECT_STATE
- MQRC_NO_CONNECTION_REFERENCE
- MQRC_STORAGE_NOT_AVAILABLE
- MQRC_REOPEN_SAVED_CONTEXT_ERR
- (來自 MQCLOSE 的原因碼)
- (來自 MQCONN 的原因碼)
- (來自 MQINQ 的原因碼)
- (來自 MQOPEN 的原因碼)
- (來自 MQSET 的原因碼)

ImqProcess C++ 類別

此類別封裝可由觸發監視器觸發的應用程式程序 (類型 MQOT_PROCESS 的 WebSphere MQ 物件)。

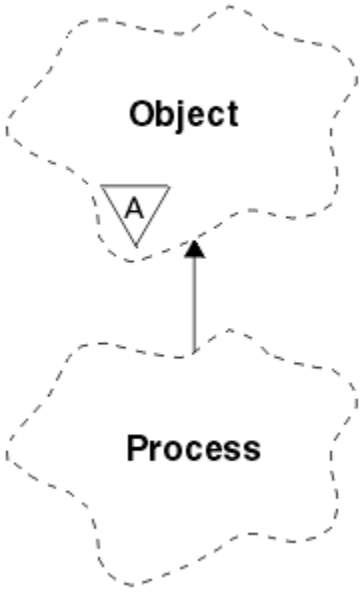


圖 *62: ImqProcess* 類別

- 第 1238 頁的『物件屬性』
- 第 1239 頁的『建構子』
- 第 1239 頁的『物件方法 (public)』

物件屬性

應用程式 ID
應用程式程序的身分。 這個屬性是唯讀的。

應用程式類型
應用程式程序的類型。 這個屬性是唯讀的。

環境資料
處理程序的環境資訊。 這個屬性是唯讀的。

使用者資料
處理程序的使用者資料。 這個屬性是唯讀的。

建構子

ImqProcess( );
預設建構子。

ImqProcess(const ImqProcess & 處理程序);
複製建構子。 ImqObject **開啟狀態** 為 FALSE。

ImqProcess(const char * *name*);
設定 ImqObject **名稱**。

物件方法 (public)

void operator = (const ImqProcess & 處理程序);
必要的話，執行關閉，然後從 處理程序複製實例資料。 ImqObject **開啟狀態** 將為 FALSE。

ImqBoolean applicationId(ImqString & *ID*);
提供 **應用程式 ID** 的副本。 如果成功，它會傳回 TRUE。

ImqString applicationId( );
傳回 **應用程式 ID** ，但不指出任何可能的錯誤。

ImqBoolean applicationType(MQLONG 及 類型);
提供 **應用程式類型**的副本。 如果成功，它會傳回 TRUE。

MQLONG applicationType();
傳回 **應用程式類型** ，但不指出任何可能的錯誤。

ImqBoolean environmentData(ImqString & 資料);
提供 **環境資料**的副本。 如果成功，它會傳回 TRUE。

ImqString environmentData( );
傳回 **環境資料** ，而不指出任何可能的錯誤。

ImqBoolean userData(ImqString & 資料);
提供 **使用者資料**的副本。 如果成功，它會傳回 TRUE。

ImqString userData( );
傳回 **使用者資料** ，而不指出任何可能的錯誤。

ImqPutMessageOptions C++ 類別

此類別會封裝 MQPMO 資料結構。

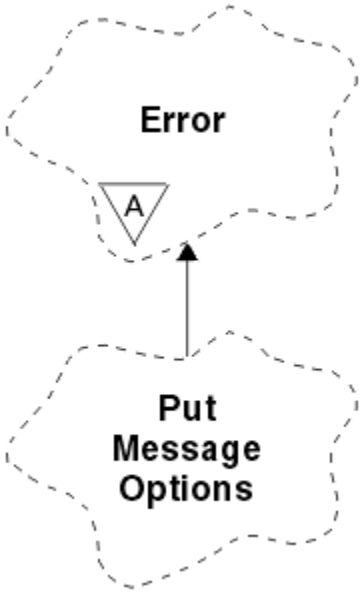


圖 *63: ImqPutMessageOptions* 類別

- 第 1240 頁的『物件屬性』
- 第 1240 頁的『建構子』
- 第 1241 頁的『物件方法 (public)』
- 第 1241 頁的『物件資料 (受保護)』
- 第 1241 頁的『原因碼』

物件屬性

環境定義參照
提供訊息環境定義的 ImqQueue 。 一開始沒有參照。

選項
放置訊息選項。 起始值為 MQPMO_NONE。 下列是可能的其他值:

- 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

記錄欄位
放置訊息時控制併入放置訊息記錄的旗標。 起始值為 MQPMRF_NONE。 下列是可能的其他值:

- MQPMRF_MSG_ID
- MQPMRF_CORREL_ID
- MQPMRF_GROUP_ID
- MQPMRF_FEEDBACK
- MQPMRF_ACCOUNTING_TOKEN

ImqMessage 追蹤器屬性取自所指定之任何欄位的物件。 ImqMessage 追蹤器屬性取自 未 指定之任何欄位的 ImqMessage 物件。

解析的佇列管理程式名稱
在放置期間決定的目的地佇列管理程式名稱。 起始值是空值。 這個屬性是唯讀的。

解析的佇列名稱
在放置期間決定的目的地佇列名稱。 起始值是空值。 這個屬性是唯讀的。

同步點參與
當訊息置於同步點控制下時為 TRUE。

建構子

ImqPutMessageOptions( );
預設建構子。

ImqPutMessageOptions(const ImqPutMessageOptions & 普莫);
複製建構子。

物件方法 (public)

void operator = (const ImqPutMessageOptions & 普莫);
從 *pmo* 複製實例資料，以取代現有實例資料。

ImqQueue * contextReference() const ;
傳回 **環境定義參照**。

void setContext 參照(const ImqQueue & 佇列);
設定 **環境定義參照**。

void setContextReference(const ImqQueue * *queue* = 0);
設定 **環境定義參照**。

MQLONG 選項() const ;
傳回 **選項**。

void setOptions(const MQLONG *options*);
設定 **選項**，包括 **同步點參與** 值。

MQLONG recordFields() const ;
傳回 **記錄欄位**。

void setRecordFields(const MQLONG *fields*);
設定 **記錄欄位**。

ImqString resolvedQueueManagerName() const ;
傳回 **已解析佇列管理程式名稱**的副本。

ImqString resolvedQueueName() const ;
傳回 **已解析佇列名稱**的副本。

ImqBoolean syncPointParticipation() const ;
傳回 **同步點參與** 值，如果 **選項** 包括 MQPMO_SYNCPOINT，則為 TRUE。

void setSyncPointParticipation(const ImqBoolean *sync*);
設定 **同步點參與** 值。 如果 *sync* 為 TRUE，則 **選項** 會變更為包括 MQPMO_SYNCPOINT，以及排除 MQPMO_NO_SYNCPOINT。 如果 *sync* 為 FALSE，則會變更 **選項** 以包括 MQPMO_NO_SYNCPOINT 及排除 MQPMO_SYNCPOINT。

物件資料 (受保護)

MQPMO *omqpmo*
MQPMO 資料結構。

原因碼

- MQRC_STORAGE_NOT_AVAILABLE

ImqQueue C++ 類別

此類別會封裝訊息佇列 (類型為 MQOT_Q 的 WebSphere MQ 物件)。

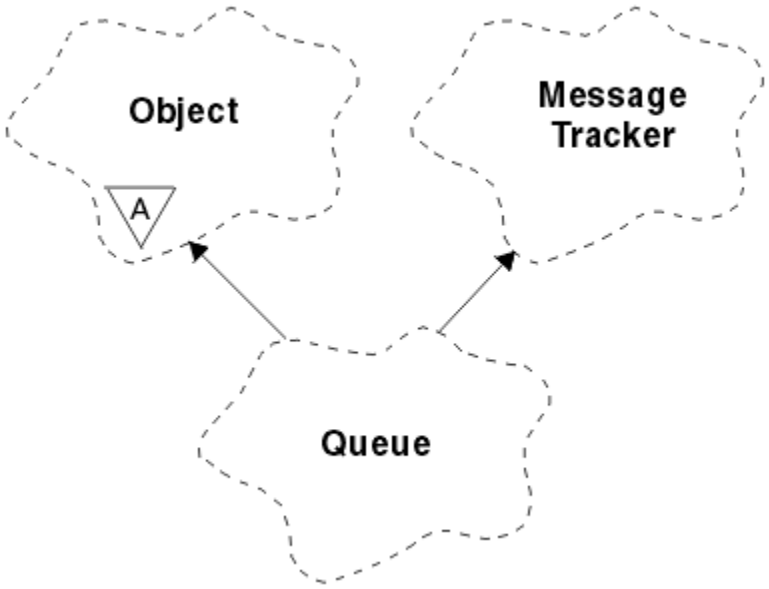


圖 *64: ImqQueue* 類別

此類別與 第 1185 頁的表 612 中列出的 MQI 呼叫相關。

- 第 1242 頁的『物件屬性』
- 第 1245 頁的『建構子』
- 第 1245 頁的『物件方法 (public)』
- 第 1251 頁的『物件方法 (protected)』
- 第 1251 頁的『原因碼』

物件屬性

取消/重入佇列名稱
取消重新排入佇列的名稱過多。 這個屬性是唯讀的。

取消臨界值
取消臨界值。 這個屬性是唯讀的。

基本佇列名稱
別名解析成的佇列名稱。 這個屬性是唯讀的。

叢集名稱
叢集名稱。 這個屬性是唯讀的。

叢集名單名稱
叢集名單名稱。 這個屬性是唯讀的。

叢集工作量等級
叢集工作量等級。 這個屬性是唯讀的。

叢集工作量優先順序
叢集工作量優先順序。 這個屬性是唯讀的。

叢集工作量使用佇列
叢集工作量使用佇列值。 這個屬性是唯讀的。

建立日期
佇列建立資料。 這個屬性是唯讀的。

建立時間
佇列建立時間。 這個屬性是唯讀的。

現行深度
佇列上的訊息數。 這個屬性是唯讀的。

預設連結
預設連結。 這個屬性是唯讀的。

預設的輸入開啟選項
預設 open-for-input 選項。 這個屬性是唯讀的。

預設持續性
預設訊息持續性。 這個屬性是唯讀的。

預設優先順序
預設訊息優先順序。 這個屬性是唯讀的。

定義類型
佇列定義類型。 這個屬性是唯讀的。

深度高事件
佇列深度高事件的控制屬性。 這個屬性是唯讀的。

深度高限制
佇列深度的高限制。 這個屬性是唯讀的。

深度低事件
佇列深度低事件的控制屬性。 這個屬性是唯讀的。

深度下限
佇列深度的下限。 這個屬性是唯讀的。

深度事件上限
佇列深度事件數上限的控制屬性。 這個屬性是唯讀的。

配送清單參照
ImqDistribution 清單的選用參照，可用來將訊息配送至多個佇列 (包括此佇列)。 起始值是空值。

註: 當開啟 ImqQueue 物件時，它所參照的任何開啟 ImqDistribution 清單物件都會自動關閉。

配送清單
傳輸佇列支援配送清單的功能。 這個屬性是唯讀的。

動態佇列名稱
動態佇列名稱。 起始值為 AMQ.* 適用於所有 Windows、 UNIX 及 Linux 平台。

強制取消
是否要強化取消計數。 這個屬性是唯讀的。

索引類型
索引類型。 這個屬性是唯讀的。

禁止取得
是否容許取得作業。 起始值取決於佇列定義。 此屬性僅適用於別名或本端佇列。

禁止放置
是否容許放置作業。 起始值取決於佇列定義。

起始佇列名稱
起始佇列的名稱。 這個屬性是唯讀的。

深度上限
佇列上容許的訊息數目上限。 這個屬性是唯讀的。

訊息長度上限
此佇列上任何訊息的長度上限，可以小於相關聯佇列管理程式所管理之任何佇列的長度上限。 這個屬性是唯讀的。

訊息遞送順序
訊息優先順序是否相關。 這個屬性是唯讀的。

下一個分散式佇列
此類別的下一個物件 (無特定順序) 具有與此物件相同的 **配送清單參照** 。 起始值為零。

如果刪除鏈中的物件，則會更新前一個物件及下一個物件，使其分散式佇列鏈結不再指向已刪除的物件。

非持續訊息類別
將非持續訊息放入此佇列的可靠性層次。 這個屬性是唯讀的。

開啟輸入計數
開啟以供輸入的 ImqQueue 物件數。 這個屬性是唯讀的。

開啟輸出計數
開啟以供輸出的 ImqQueue 物件數。 這個屬性是唯讀的。

前一個分散式佇列
此類別的前一個物件 (無特定順序) 與此物件具有相同的 **配送清單參照** 。 起始值為零。

如果刪除鏈中的物件，則會更新前一個物件及下一個物件，使其分散式佇列鏈結不再指向已刪除的物件。

程序名稱
程序定義的名稱。 這個屬性是唯讀的。

佇列計數
佇列的帳戶資訊層次。 這個屬性是唯讀的。

佇列管理程式名稱
佇列所在的佇列管理程式名稱 (可能是遠端)。 請勿混淆這裡所指名的佇列管理程式與 ImqObject **連線參照**，後者參照提供連線的 (本端) 佇列管理程式。 起始值是空值。

監視佇列
佇列的監視資料收集層次。 這個屬性是唯讀的。

佇列統計資料
佇列的統計資料層次。 這個屬性是唯讀的。

佇列類型
佇列類型。 這個屬性是唯讀的。

遠端佇列管理程式名稱
遠端佇列管理程式的名稱。 這個屬性是唯讀的。

遠端佇列名稱
遠端佇列管理程式上已知的遠端佇列名稱。 這個屬性是唯讀的。

解析的佇列管理程式名稱
已解析佇列管理程式名稱。 這個屬性是唯讀的。

解析的佇列名稱
已解析佇列名稱。 這個屬性是唯讀的。

保留間隔
佇列保留間隔。 這個屬性是唯讀的。

範圍
佇列定義的範圍。 這個屬性是唯讀的。

服務間隔 (service interval)
服務間隔。 這個屬性是唯讀的。

服務間隔事件 (service interval event)
服務間隔事件的控制屬性。 這個屬性是唯讀的。

共用性
是否可以共用佇列。 這個屬性是唯讀的。

儲存類別 (storage class)
儲存類別。 這個屬性是唯讀的。

傳輸佇列名稱
傳輸佇列的名稱。 這個屬性是唯讀的。

觸發控制
觸發控制。 起始值視佇列定義而定。 此屬性僅適用於本端佇列。

觸發資料
觸發資料。 起始值視佇列定義而定。 此屬性僅適用於本端佇列。

觸發深度
觸發深度。 起始值視佇列定義而定。 此屬性僅適用於本端佇列。

觸發訊息優先順序
觸發程式的臨界值訊息優先順序。 起始值視佇列定義而定。 此屬性僅適用於本端佇列。

觸發函式類型
觸發程式類型。 起始值視佇列定義而定。 此屬性僅適用於本端佇列。

用法
用法。 這個屬性是唯讀的。

建構子

ImqQueue( );
預設建構子。

ImqQueue(const ImqQueue & 佇列);
複製建構子。 ImqObject **開啟狀態** 將為 FALSE。

ImqQueue(const char * *name*);
設定 ImqObject **名稱**。

物件方法 (public)

void operator = (const ImqQueue & 佇列);
必要的話，執行關閉，然後從 佇列複製實例資料。 ImqObject **開啟狀態** 將為 FALSE。

ImqBoolean backoutRequeue 名稱(ImqString & 名);
提供 **取消重新排入佇列名稱**的副本。 如果成功，它會傳回 TRUE。

ImqString backoutRequeueName();
傳回 **取消重新排入佇列名稱** ，且沒有任何可能錯誤的指示。

ImqBoolean backoutThreshold(MQLONG 及 臨界值);
提供 **取消臨界值**的副本。 如果成功，它會傳回 TRUE。

MQLONG backoutThreshold();
傳回 **取消臨界值** 值，不含任何可能錯誤的指示。

ImqBoolean baseQueue 名稱(ImqString & 名);
提供 **基本佇列名稱**的副本。 如果成功，它會傳回 TRUE。

ImqString baseQueueName();
傳回 **基本佇列名稱** ，不指出任何可能的錯誤。

ImqBoolean clusterName(ImqString & 名);
提供 **叢集名稱**的副本。 如果成功，它會傳回 TRUE。

ImqString clusterName( );
傳回 **叢集名稱** ，但不指出任何可能的錯誤。

ImqBoolean clusterNamelistName(ImqString & 名);
提供 **叢集名單名稱**的副本。 如果成功，它會傳回 TRUE。

ImqString clusterNamelistName ();
傳回 **叢集名單名稱** ，且沒有任何錯誤指示。

ImqBoolean clusterWorkLoadPriority (MQLONG 及優先順序);
提供叢集工作量優先順序值的副本。 如果成功，它會傳回 TRUE。

MQLONG clusterWorkLoadPriority ();
傳回叢集工作量優先順序值，但不指出任何可能的錯誤。

ImqBoolean clusterWorkLoadRank (MQLONG 及等級);
提供叢集工作量等級值的副本。 如果成功，它會傳回 TRUE。

MQLONG clusterWorkLoadRank ();
傳回叢集工作量等級值，不含任何可能錯誤的指示。

ImqBoolean clusterWorkLoadUseQ (MQLONG 及 useq);
提供叢集工作量使用佇列值的副本。 如果成功，它會傳回 TRUE。

MQLONG clusterWorkLoadUseQ ();
傳回叢集工作量使用佇列值，不指出任何可能的錯誤。

ImqBoolean creationDate(ImqString & 日期);
提供 **建立日期**的副本。 如果成功，它會傳回 TRUE。

ImqString creationDate( );
傳回 **建立日期** ，不指出任何可能的錯誤。

ImqBoolean creationTime(ImqString & 時間);
提供 **建立時間**的副本。 如果成功，它會傳回 TRUE。

ImqString creationTime( );
傳回 **建立時間** ，不指出任何可能的錯誤。

ImqBoolean currentDepth(MQLONG 及 深度);
提供 **現行深度**的副本。 如果成功，它會傳回 TRUE。

MQLONG currentDepth();
傳回 **現行深度** ，不指出任何可能的錯誤。

ImqBoolean defaultInputOpenOption(MQLONG 及 選項);
提供 **預設輸入開啟選項**的副本。 如果成功，它會傳回 TRUE。

MQLONG defaultInputOpenOption();
傳回 **預設輸入開啟選項** ，不指出任何可能的錯誤。

ImqBoolean defaultPersistence(MQLONG 及 持續性);
提供 **預設持續性**的副本。 如果成功，它會傳回 TRUE。

MQLONG defaultPersistence();
傳回 **預設持續性** ，不指出任何可能的錯誤。

ImqBoolean defaultPriority(MQLONG 及 優先);
提供 **預設優先順序**的副本。 如果成功，它會傳回 TRUE。

MQLONG defaultPriority();
傳回 **預設優先順序** ，不指出任何可能的錯誤。

ImqBoolean defaultBind(MQLONG 及 捆綁);
提供 **預設連結**的副本。 如果成功，它會傳回 TRUE。

MQLONG defaultBind();
傳回 **預設連結** ，不指出任何可能的錯誤。

ImqBoolean definitionType(MQLONG 及 類型);
提供 **定義類型**的副本。 如果成功，它會傳回 TRUE。

MQLONG definitionType();
傳回 **定義類型** ，但不指出任何可能的錯誤。

ImqBoolean depthHigh 事件(MQLONG 及 事件);
提供 **深度高事件**的啟用狀態副本。 如果成功，它會傳回 TRUE。

MQLONG depthHighEvent();
傳回 **深度高事件** 的啟用狀態，不指出任何可能的錯誤。

ImqBoolean depthHigh 限制(MQLONG 及 限制);
提供 **深度上限**的副本。 如果成功，它會傳回 TRUE。

MQLONG depthHighLimit();
傳回 **depth high limit** 值，沒有任何可能錯誤的指示。

ImqBoolean depthLow 事件(MQLONG 及 事件);
提供 **深度低事件**的啟用狀態副本。 如果成功，它會傳回 TRUE。

MQLONG depthLowEvent();
傳回 **深度低事件** 的啟用狀態，不指出任何可能的錯誤。

ImqBoolean depthLow 限制(MQLONG 及 限制);
提供 **深度下限**的副本。 如果成功，它會傳回 TRUE。

MQLONG depthLowLimit();
傳回 **depth low limit** 值，沒有任何可能的錯誤指示。

ImqBoolean depthMaximum 事件(MQLONG 及 事件);
提供 **深度上限事件**的啟用狀態副本。 如果成功，它會傳回 TRUE。

MQLONG depthMaximumEvent();
傳回 **深度上限事件** 的啟用狀態，不指出任何可能的錯誤。

ImqDistributionList * distributionListReference() const ;
傳回 **配送清單參照**。

void setDistributionListReference(ImqDistribution 清單及 清單);
設定 **配送清單參照**。

void setDistributionListReference(ImqDistributionList * *list* = 0);
設定 **配送清單參照**。

ImqBoolean distributionLists(MQLONG 及 支援);
提供 **配送清單** 值的副本。 如果成功，它會傳回 TRUE。

MQLONG distributionLists();
傳回 **配送清單** 值，但不指出可能的錯誤。

ImqBoolean setDistributionLists(const MQLONG 支援);
設定 **配送清單** 值。 如果成功，它會傳回 TRUE。

ImqString dynamicQueueName() const ;
傳回 **動態佇列名稱**的副本。

ImqBoolean setDynamicQueueName(const char * *name*);
設定 **動態佇列名稱**。 只有在 ImqObject **開啟狀態** 為 FALSE 時，才能設定 **動態佇列名稱** 。 如果成功，它會傳回 TRUE。

ImqBoolean 取得(ImqMessage & 訊息, ImqGetMessageOptions & 選項);
使用指定的 選項從佇列中擷取訊息。 必要的話，呼叫 ImqObject **openFor** 方法，以確保 ImqObject **開啟選項** 包括 MQOO_INPUT_ * 值或 MQOO_BROWSE 值 (視 選項而定)。 如果 *msg* 物件具有 ImqCache **自動緩衝區**，則緩衝區會擴增以容納任何擷取的訊息。 在擷取之前，會對 *msg* 物件呼叫 **clearMessage** 方法。

如果成功，此方法會傳回 TRUE。

註: 如果 ImqObject **原因碼** 為 MQRC_TRUNCATED_MSG_FAILED ，則方法呼叫的結果為 FALSE ，即使此 **原因碼** 分類為警告。 如果接受截斷的訊息， ImqCache **訊息長度** 會反映截斷的長度。 在任一事件中， ImqMessage **訊息長度總計** 指出可用的位元組數。

ImqBoolean 取得(ImqMessage & 訊息);
至於先前的方法，除了使用預設取得訊息選項之外。

ImqBoolean 取得(ImqMessage & 訊息, ImqGetMessageOptions & 選項, 常數大小 _t 緩衝區大小);
至於前兩種方法，除了指出置換 *buffer-size* 之外。 如果 *msg* 物件使用 ImqCache **自動緩衝區**，在擷取訊息之前，會在 *msg* 物件上呼叫 **resizeBuffer** 方法，且緩衝區不會進一步成長以容納任何較大的訊息。

ImqBoolean 取得(ImqMessage & 訊息, 常數大小 _t 緩衝區大小);
至於先前的方法，除了使用預設取得訊息選項之外。

ImqBoolean hardenGet 取消(MQLONG 及 硬);
提供 **harden get backout** 值的副本。 如果成功，它會傳回 TRUE。

MQLONG hardenGetBackout();
傳回 **harden get backout** 值，不含任何可能錯誤的指示。

ImqBoolean indexType(MQLONG & 類型);
提供 **索引類型**的副本。 如果成功，它會傳回 TRUE。

MQLONG indexType();
傳回 **索引類型** ，但不指出任何可能的錯誤。

ImqBoolean inhibitGet(MQLONG 及 抑制);
提供 **禁止取得** 值的副本。 如果成功，它會傳回 TRUE。

MQLONG inhibitGet();
傳回 **禁止取得** 值，而不指出任何可能的錯誤。

ImqBoolean setInhibitGet(const MQLONG 禁止);
設定 **禁止取得** 值。 如果成功，它會傳回 TRUE。

ImqBoolean inhibitPut(MQLONG 及 抑制);
提供 **禁止放置** 值的副本。 如果成功，它會傳回 TRUE。

MQLONG inhibitPut();
傳回 **禁止放置** 值，且不指出任何可能的錯誤。

ImqBoolean setInhibitPut(const MQLONG 禁止);
設定 **禁止放置** 值。 如果成功，它會傳回 TRUE。

ImqBoolean initiationQueue 名稱(ImqString & 名);
提供 **起始佇列名稱**的副本。 如果成功，它會傳回 TRUE。

ImqString initiationQueueName();
傳回 **起始佇列名稱** ，不指出任何可能的錯誤。

ImqBoolean maximumDepth(MQLONG 及 深度);
提供 **深度上限**的副本。 如果成功，它會傳回 TRUE。

MQLONG maximumDepth();
傳回 **maximum depth** ，不含任何可能錯誤的指示。

ImqBoolean maximumMessage 長度(MQLONG 及 長度);
提供 **訊息長度上限**的副本。 如果成功，它會傳回 TRUE。

MQLONG maximumMessageLength();
傳回 **訊息長度上限** ，不指出任何可能的錯誤。

ImqBoolean messageDelivery 順序(MQLONG 及 序列);
提供 **訊息遞送順序**的副本。 如果成功，它會傳回 TRUE。

MQLONG messageDeliverySequence();
傳回 **訊息遞送順序** 值，不指出任何可能的錯誤。

ImqQueue * nextDistributedQueue() const ;
傳回 **下一個分散式佇列**。

ImqBoolean nonPersistentMessageClass (MQLONG & monq);
提供非持續訊息類別值的副本。 如果成功，它會傳回 TRUE。

MQLONG nonPersistentMessageClass ();
傳回非持續訊息類別值，且沒有任何可能錯誤的指示。

ImqBoolean openInput 計數(MQLONG 及 計數);
提供 **開啟輸入計數**的副本。 如果成功，它會傳回 TRUE。

MQLONG openInputCount();
傳回 **open input count** ，不含任何可能的錯誤指示。

ImqBoolean openOutput 計數(MQLONG 及 計數);
提供 **開啟輸出計數**的副本。 如果成功，它會傳回 TRUE。

MQLONG openOutputCount();
傳回 **開啟輸出計數** ，沒有任何可能錯誤的指示。

ImqQueue * previousDistributedQueue() const ;
傳回 **前一個分散式佇列**。

ImqBoolean processName(ImqString & 名);
提供 **程序名稱**的副本。 如果成功，它會傳回 TRUE。

ImqString processName( );
傳回 **處理程序名稱** ，但不指出任何可能的錯誤。

ImqBoolean 放(ImqMessage & 訊息);
使用預設放置訊息選項，將訊息放置在佇列上。 必要的話，使用 ImqObject **openFor** 方法，以確保 ImqObject **開啟選項** 包括 MQOO_OUTPUT。

如果成功，此方法會傳回 TRUE。

ImqBoolean 放(ImqMessage & 訊息, ImqPutMessageOptions & 普莫);
使用指定的 *pmo* 將訊息放入佇列。 視需要使用 ImqObject **openFor** 方法，以確保 ImqObject **開啟選項** 包括 MQOO_OUTPUT ，以及 (如果 *pmo* **選項** 包括任何 MQPMO_PASS_IDENTITY_CONTEXT、MQPMO_PASS_ALL_CONTEXT、MQPMO_SET_IDENTITY_CONTEXT 或 MQPMO_SET_ALL_CONTEXT) 對應的 MQOO_ * _CONTEXT 值。

如果成功，此方法會傳回 TRUE。

註: 如果 *pmo* 包括 **環境定義參照**，則必要的話，會開啟參照物件以提供環境定義。

ImqBoolean queueAccounting (MQLONG 及 acctq);
提供佇列帳戶值的副本。 如果成功，它會傳回 TRUE。

MQLONG queueAccounting ();
傳回佇列帳戶值，但不指出任何可能的錯誤。

ImqString queueManagerName() const ;
傳回 **佇列管理程式名稱**。

ImqBoolean setQueueManagerName(const char * *name*);
設定 **佇列管理程式名稱**。 只有在 ImqObject **開啟狀態** 為 FALSE 時，才能設定 **佇列管理程式名稱** 。 如果成功，此方法會傳回 TRUE。

ImqBoolean queueMonitoring (MQLONG & monq);
提供佇列監視值的副本。 如果成功，它會傳回 TRUE。

MQLONG queueMonitoring ();
傳回佇列監視值，但不指出任何可能的錯誤。

ImqBoolean queueStatistics (MQLONG & statq);
提供佇列統計資料值的副本。 如果成功，它會傳回 TRUE。

MQLONG queueStatistics ();
傳回佇列統計資料值，但不指出任何可能的錯誤。

ImqBoolean queueType(MQLONG 及 類型);
提供 **佇列類型** 值的副本。 如果成功，它會傳回 TRUE。

MQLONG queueType();
傳回 **佇列類型** ，不指出任何可能的錯誤。

ImqBoolean remoteQueueManagerName(ImqString & 名);
提供 **遠端佇列管理程式名稱**的副本。 如果成功，它會傳回 TRUE。

ImqString remoteQueueManagerName( );
傳回 **遠端佇列管理程式名稱** ，但不指出任何可能的錯誤。

ImqBoolean remoteQueue 名稱(ImqString & 名);
提供 **遠端佇列名稱**的副本。 如果成功，它會傳回 TRUE。

ImqString remoteQueueName();
傳回 **遠端佇列名稱** ，不指出任何可能的錯誤。

ImqBoolean resolvedQueueManagerName(ImqString & 名);
提供 **已解析佇列管理程式名稱**的副本。 如果成功，它會傳回 TRUE。

註: 除非 MQOO_RESOLVE_NAMES 在 ImqObject **開啟選項**中，否則此方法會失敗。

ImqString resolvedQueueManagerName( ) ;
傳回 **已解析佇列管理程式名稱**，沒有任何可能錯誤的指示。

ImqBoolean resolvedQueueName(ImqString & 名);
提供 **已解析佇列名稱**的副本。 如果成功，它會傳回 TRUE。

註: 除非 MQOO_RESOLVE_NAMES 在 ImqObject **開啟選項**中，否則此方法會失敗。

ImqString resolvedQueueName ();
傳回 **已解析佇列名稱**，且沒有任何可能錯誤的指示。

ImqBoolean retentionInterval(MQLONG 及 間隔);
提供 **保留間隔**的副本。 如果成功，它會傳回 TRUE。

MQLONG retentionInterval();
傳回 **保留間隔** ，不指出任何可能的錯誤。

ImqBoolean 範圍(MQLONG 及 範圍);
提供 **範圍**的副本。 如果成功，它會傳回 TRUE。

MQLONG scope();
傳回 **scope** ，不指出任何可能的錯誤。

ImqBoolean serviceInterval(MQLONG 及 間隔);
提供 **服務間隔**的副本。 如果成功，它會傳回 TRUE。

MQLONG serviceInterval();
傳回 **服務間隔** ，但不指出任何可能的錯誤。

ImqBoolean serviceInterval 事件(MQLONG 及 事件);
提供 **服務間隔事件**的啟用狀態副本。 如果成功，它會傳回 TRUE。

MQLONG serviceInterval 事件();
傳回 **服務間隔事件** 的啟用狀態，沒有任何可能錯誤的指示。

ImqBoolean 可共用性(MQLONG 及 可共用性);
提供 **shareability** 值的副本。 如果成功，它會傳回 TRUE。

MQLONG 可共用性();
傳回 **shareability** 值，不含任何可能錯誤的指示。

ImqBoolean storageClass(ImqString & 班);
提供 **儲存類別**的副本。 如果成功，它會傳回 TRUE。

ImqString storageClass( );
傳回 **儲存類別** ，但不指出任何可能的錯誤。

ImqBoolean transmissionQueue 名稱(ImqString & 名);
提供 **傳輸佇列名稱**的副本。 如果成功，它會傳回 TRUE。

ImqString transmissionQueueName();
傳回 **傳輸佇列名稱** ，不指出任何可能的錯誤。

ImqBoolean triggerControl(MQLONG 及 控制);
提供 **觸發控制** 值的副本。 如果成功，它會傳回 TRUE。

MQLONG triggerControl();
傳回 **觸發控制** 值，不指出任何可能的錯誤。

ImqBoolean setTriggerControl(const MQLONG *control*);
設定 **觸發控制** 值。 如果成功，它會傳回 TRUE。

ImqBoolean triggerData(ImqString & 資料);
提供 **觸發程式資料**的副本。 如果成功，它會傳回 TRUE。

ImqString triggerData( );
傳回 **觸發程式資料** 的副本，且沒有任何可能錯誤的指示。

ImqBoolean setTriggerData(const char * *data*);
設定 **觸發程式資料**。 如果成功，它會傳回 TRUE。

ImqBoolean triggerDepth(MQLONG 及 深度);
提供 **觸發深度**的副本。 如果成功，它會傳回 TRUE。

MQLONG triggerDepth();
傳回 **觸發程式深度** ，不指出任何可能的錯誤。

ImqBoolean setTrigger 深度(const MQLONG *depth*);
設定 **觸發程式深度**。 如果成功，它會傳回 TRUE。

ImqBoolean triggerMessage 優先順序(MQLONG 及 優先);
提供 **觸發訊息優先順序**的副本。 如果成功，它會傳回 TRUE。

MQLONG triggerMessagePriority();
傳回 **觸發訊息優先順序** ，不指出任何可能的錯誤。

ImqBoolean setTriggerMessagePriority(const MQLONG *priority*);
設定 **觸發訊息優先順序**。 如果成功，它會傳回 TRUE。

ImqBoolean triggerType(MQLONG 及 類型);
提供 **觸發類型**的副本。 如果成功，它會傳回 TRUE。

MQLONG triggerType();
傳回 **觸發程式類型** ，但不指出任何可能的錯誤。

ImqBoolean setTriggerType(const MQLONG *type*);
設定 **觸發程式類型**。 如果成功，它會傳回 TRUE。

ImqBoolean 使用情形(MQLONG 及 使用情形);
提供 **usage** 值的副本。 如果成功，它會傳回 TRUE。

MQLONG 用法();
傳回 **usage** 值，不指出任何可能的錯誤。

物件方法 (protected)

void setNextDistributedQueue(ImqQueue * *queue* = 0);
設定 **下一個分散式佇列**。

注意: 只有在您確定不會岔斷分散式佇列清單時，才使用此功能。

void setPreviousDistributedQueue(ImqQueue * *queue* = 0);
設定 **前一個分散式佇列**。

注意: 只有在您確定不會岔斷分散式佇列清單時，才使用此功能。

原因碼

- MQRC_ATTRIBUTE_LOCKED
- MQRC_CONTEXT_OBJECT_NOT_VALID
- MQRC_CONTEXT_OPEN_ERROR
- MQRC_CURSOR_NOT_VALID
- MQRC_NO_BUFFER
- MQRC_REOPEN_EXCEL_INPUT_ERROR
- MQRC_REOPEN_INQUIRE_ERROR
- MQRC_REOPEN_TEMPORARY_Q_ERROR
- (來自 MQGET 的原因碼)
- (來自 MQPUT 的原因碼)

ImqQueue 管理程式 C++ 類別

此類別封裝佇列管理程式 (MQOT_Q_MGR 類型的 WebSphere MQ 物件)。

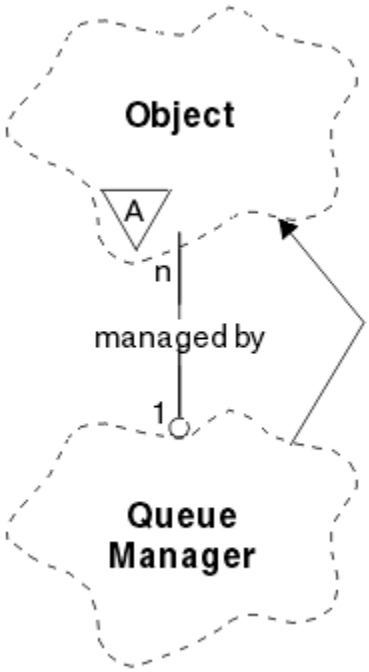


圖 *65: ImqQueue* 管理程式類別

此類別與 第 1187 頁的『ImqQueue 管理程式交互參照』 中列出的 MQI 呼叫相關。 並非所有列出的方法都適用於所有平台; 如需詳細資料，請參閱 ALTER QMGR 。

- 第 1252 頁的『類別屬性』
- 第 1253 頁的『物件屬性』
- 第 1257 頁的『建構子』
- 第 1257 頁的『解構子』
- 第 1257 頁的『類別方法 (public)』
- 第 1257 頁的『物件方法 (public)』
- 第 1266 頁的『物件方法 (protected)』
- 第 1266 頁的『物件資料 (受保護)』
- 第 1266 頁的『原因碼』

類別屬性

行為 (behavior)
控制隱含連線和斷線的行為。

IMQ_EXPL_DISC_BACKOUT (0L)
明確呼叫 **disconnect** 方法意味著取消。 此屬性與 IMQ_EXPL_DISC_COMMIT 互斥。

IMQ_EXPL_DISC_COMMIT (1L)
明確呼叫 **disconnect** 方法意味著確定 (預設值)。 此屬性與 IMQ_EXPL_DISC_BACKOUT 互斥。

IMQ_IMPL_CONN (2L)
容許隱含連線 (預設值)。

IMQ_IMPL_DISC_BACKOUT (0L)
對 **disconnect** 方法的隱含呼叫 (可能在物件毀損期間發生) 意味著取消。 此屬性與 IMQ_IMPL_DISC_COMMIT 互斥。

IMQ_IMPL_DISC_COMMIT (4L)
對 **disconnect** 方法的隱含呼叫 (在物件毀損期間可能發生) 意味著確定 (預設值)。 此屬性與 IMQ_IMPL_DISC_BACKOUT 互斥。

在 WebSphere MQ V7.0 及以上版本中，使用隱含連線的 C++ 應用程式需要在類別 ImqQueueManager 的物件上指定 IMQ_IMPL_CONN 以及 setBehavior() 方法中提供的任何其他選項。 如果您的應用程式未使用 setBehavior() 方法來明確設定行為選項，例如:

```
ImqQueueManager_object.setBehavior(IMQ_IMPL_DISC_COMMIT)
```

此變更不會影響您，因為依預設會啟用 MQ_IMPL_CONN。

如果您的應用程式明確設定行為選項，例如:

```
ImqQueueManager_object.setBehavior(IMQ_IMPL_DISC_COMMIT)
```

您需要在 setBehavior() 方法中包含 IMQ_IMPL_CONN ，如下所示，以容許您的應用程式完成隱含連線:

```
ImqQueueManager_object.setBehavior(IMQ_IMPL_CONN | IMQ_IMPL_DISC_COMMIT)
```

物件屬性

帳戶連線置換
容許應用程式置換 MQI 結算及佇列結算 values.This 屬性是唯讀的。

計數間隔時間
寫入中間統計記錄之前的時間 (以秒為單位)。 這個屬性是唯讀的。

活動記錄中
控制活動報告的產生。 這個屬性是唯讀的。

採用新的 MCA 檢查
當偵測到新的入埠通道與已在作用中的 MCA 同名時，會檢查元素以判斷是否應該採用 MCA。 這個屬性是唯讀的。

採用新的 MCA 類型
當偵測到符合採用新 MCA 檢查參數的新入埠通道要求時，是否應自動重新啟動特定通道類型 MCA 的孤立實例。 這個屬性是唯讀的。

鑑別類型
指出正在執行的鑑別類型。

權限事件
控制權限事件。 這個屬性是唯讀的。

開始選項
適用於 **begin** 方法的選項。 起始值為 MQBO_NONE。

橋接器事件
是否產生 IMS Bridge 事件。 這個屬性是唯讀的。

通道自動定義
通道自動定義值。 這個屬性是唯讀的。

通道自動定義事件
通道自動定義事件值。 這個屬性是唯讀的。

通道自動定義結束程式
通道自動定義結束程式名稱。 這個屬性是唯讀的。

通道事件 (channel event)
是否產生頻道事件。 這個屬性是唯讀的。

通道起始程式配接器
用於處理 WebSphere MQ 呼叫的配接器子作業數目。 這個屬性是唯讀的。

通道起始程式控制
是否應在啟動「佇列管理程式」時自動啟動「通道起始程式」。 這個屬性是唯讀的。

通道起始程式分派器
用於通道起始程式的分派器數目。 這個屬性是唯讀的。

通道起始程式追蹤自動啟動
通道起始程式追蹤是否應該自動啟動。 這個屬性是唯讀的。

通道起始程式追蹤表格大小
通道起始程式的追蹤資料空間大小 (MB)。 這個屬性是唯讀的。

監視通道
控制通道線上監視資料的收集。 這個屬性是唯讀的。

通道參照
通道定義的參照，在用戶端連線期間使用。 連接時，此屬性可以設為空值，但無法變更為任何其他值。 起始值是空值。

通道統計資料
控制通道統計資料的收集。 這個屬性是唯讀的。

字元集
編碼字集 ID (CCSID)。 這個屬性是唯讀的。

叢集傳送端監視
控制自動定義叢集傳送端通道的線上監視資料收集。 這個屬性是唯讀的。

叢集傳送端統計資料
控制自動定義叢集傳送端通道的統計資料收集。 這個屬性是唯讀的。

叢集工作量資料
叢集工作量結束程式資料。 這個屬性是唯讀的。

叢集工作量結束程式
叢集工作量結束程式名稱。 這個屬性是唯讀的。

叢集工作量長度
叢集工作量長度。 這個屬性是唯讀的。

叢集工作量 mru
叢集工作量最近使用的通道值。 這個屬性是唯讀的。

叢集工作量使用佇列
叢集工作量使用佇列值。 這個屬性是唯讀的。

指令事件 (command event)
是否產生指令事件。 這個屬性是唯讀的。

指令輸入佇列名稱
系統指令輸入佇列名稱。 這個屬性是唯讀的。

指令層次
佇列管理程式支援的指令層次。 這個屬性是唯讀的。

指令伺服器控制
啟動「佇列管理程式」時是否應自動啟動「指令伺服器」。 這個屬性是唯讀的。

連線選項
適用於 **connect** 方法的選項。 起始值為 MQCNO_NONE。 視平台而定，可能會有下列其他值:

- MQCN_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

連線 ID
可讓 MQ 可靠地識別應用程式的唯一 ID。

連線狀態
連接至佇列管理程式時為 TRUE。 這個屬性是唯讀的。

連線標記
要與連線相關聯的標籤。 只有在未連接時，才能設定此屬性。 起始值是空值。

加密硬體
加密硬體的配置詳細資料。 適用於 MQ MQI 用戶端連線。

無法傳送郵件的佇列名稱
無法傳送郵件的佇列名稱。 這個屬性是唯讀的。

預設傳輸佇列名稱
預設傳輸佇列名稱。 這個屬性是唯讀的。

配送清單
佇列管理程式支援配送清單的功能。

dns 群組
使用「工作量管理程式動態網域名稱服務」支援時，處理佇列共用群組之入埠傳輸的 TCP 接聽器應該加入的群組名稱。 這個屬性是唯讀的。

dns wlm
處理佇列共用群組之入埠傳輸的 TCP 接聽器是否應該向「動態網域名稱服務」的「工作量管理程式」登錄。 這個屬性是唯讀的。

第一個鑑別記錄
類別 ImqAuthentication 記錄的第一個以上物件 (無特定順序)，其中「ImqAuthentication 記錄」連線參照處理此物件。 適用於 MQ MQI 用戶端連線。

第一個受管理物件
類別 ImqObject 的第一個以上物件 (無特定順序)，其中 ImqObject **連線參照** 會處理此物件。 起始值為零。

禁止事件
控制項禁止事件。 這個屬性是唯讀的。

IP 位址版本
要用於通道連線的 IP 通訊協定 (IPv4 或 IPv6)。 這個屬性是唯讀的。

金鑰儲存庫 (key repository)
金鑰資料庫檔的位置，金鑰和憑證儲存在其中。 對於 WebSphere MQ MQI 用戶端連線。

金鑰重設計數
在重新協議秘密金鑰之前，在 SSL 交談內傳送及接收的未加密位元組數。 此屬性僅適用於使用 MQCONNX 的用戶端連線。 另請參閱 ssl 金鑰重設計數。

接聽器計時器
如果發生 APPC 或 TCP/IP 失敗， WebSphere MQ 嘗試重新啟動接聽器的時間間隔 (以秒為單位)。 這個屬性是唯讀的。

本端事件
控制本端事件。 這個屬性是唯讀的。

日誌程式事件
控制是否產生回復日誌事件。 這個屬性是唯讀的。

LU 群組名稱
處理佇列共用群組之入埠傳輸的 LU 6.2 接聽器應該使用的一般 LU 名稱。 這個屬性是唯讀的。

LU 名稱
用於出埠 LU 6.2 傳輸的 LU 名稱。 這個屬性是唯讀的。

lu62 臂字尾
SYS1.PARMLIB 成員 APPCPMxx，指定此通道起始程式的 LUADD。 這個屬性是唯讀的。

lu62 通道
可以是現行或可以連接且使用 LU 6.2 傳輸通訊協定的通道數上限。 這個屬性是唯讀的。

作用中通道數上限
任何時間都會處於作用中狀態的通道數目上限。 這個屬性是唯讀的。

通道數上限
可視為現行的通道數目上限（包括帶有已連接用戶端的伺服器連線通道）。 這個屬性是唯讀的。

控點數目上限
控點數目上限。 這個屬性是唯讀的。

訊息長度上限
此佇列管理程式所管理之任何佇列上任何訊息的可能長度上限。 這個屬性是唯讀的。

最大優先順序
訊息優先順序上限。 這個屬性是唯讀的。

未確定的訊息數上限
單元或工作中未確定的訊息數上限。 這個屬性是唯讀的。

MQI 計數
控制 MQI 資料的帳戶資訊收集。 這個屬性是唯讀的。

MQI 統計資料
控制收集佇列管理程式的統計資料監視資訊。 這個屬性是唯讀的。

出埠連接埠上限
連結送出通道時要使用之埠號範圍的較高端。 這個屬性是唯讀的。

出埠連接埠下限
連結送出通道時要使用之埠號範圍的下端。 這個屬性是唯讀的。

密碼
與使用者 ID 相關聯的密碼

效能事件 (performance event)
控制效能事件。 這個屬性是唯讀的。

platform
佇列管理程式所在的平台。 這個屬性是唯讀的。

佇列計數
控制佇列的帳戶資訊收集。 這個屬性是唯讀的。

監視佇列
控制收集佇列的線上監視資料。 這個屬性是唯讀的。

佇列統計資料
控制收集佇列的統計資料。 這個屬性是唯讀的。

接收逾時
在回到非作用中狀態之前，大約 TCP/IP 訊息通道等待從其友機接收資料 (包括活動訊號) 的時間長度。 這個屬性是唯讀的。

接收逾時下限
在回到非作用中狀態之前， TCP/IP 通道等待從其友機接收資料 (包括活動訊號) 的最短時間。 這個屬性是唯讀的。

接收逾時類型
套用至 **receive timeout** 的限定元。 這個屬性是唯讀的。

遠端事件
控制遠端事件。 這個屬性是唯讀的。

REPOSITORY NAME
儲存庫名稱。 這個屬性是唯讀的。

儲存庫名單
儲存庫名單名稱。 這個屬性是唯讀的。

共用佇列管理程式名稱
其中「 ObjectQMgr 名稱」是佇列共用群組中另一個佇列管理程式的共用佇列 MQOPENs 是否應該解析為開啟本端佇列管理程式上的共用佇列。 這個屬性是唯讀的。

ssl 事件
是否產生 SSL 事件。 這個屬性是唯讀的。

需要 SSL FIPS
在 WebSphere MQ 軟體中執行加密法時，是否只應使用 FIPS 認證的演算法。 這個屬性是唯讀的。

SSL 金鑰重設計數
在重新協議秘密金鑰之前，在 SSL 交談內傳送及接收的未加密位元組數。 這個屬性是唯讀的。

開始-停止事件
控制啟動/停止事件。 這個屬性是唯讀的。

統計時間間隔
統計資料監視資料寫入監視佇列的頻率。 這個屬性是唯讀的。

同步點可用性
同步點參與的可用性。 這個屬性是唯讀的。

註: IBM i 平台不支援佇列管理程式協調的廣域工作單元。

tcp channels
可以是現行或可連接且使用 TCP/IP 傳輸通訊協定之用戶端的通道數上限。 這個屬性是唯讀的。

TCP 持續作用
是否要使用 TCP KEEPALIVE 機能來檢查連線的另一端是否仍然可用。 這個屬性是唯讀的。

TCP 名稱
要使用的唯一或預設 TCP/IP 系統名稱，視 **tcp stack type** 的值而定。 這個屬性是唯讀的。

TCP 堆疊類型
通道起始程式是否只允許使用 **tcp name** 中指定的 TCP/IP 位址空間，或可以連結至任何選取的 TCP/IP 位址。 這個屬性是唯讀的。

追蹤路徑記錄
控制路徑追蹤資訊的記錄。 這個屬性是唯讀的。

觸發間隔
觸發間隔。 這個屬性是唯讀的。

使用者 ID
在 UNIX and Linux 平台上，這是應用程式的實際使用者 ID。 在 Windows 平台上，這是應用程式的使用者 ID。

建構子

ImqQueue 管理程式 ();
預設建構子。

ImqQueue 管理程式 (const ImqQueueManager & 經理);
複製建構子。 **連線狀態** 將為 FALSE。

ImqQueueManager (const char * *name*);
將 ImqObject **名稱** 設為 *name*。

解構子

當「ImqQueue 管理程式」物件毀損時，它會自動斷線。

類別方法 (public)

static MQLONG behavior ();
傳回 **行為**。

void setBehavior(const MQLONG *behavior* = 0);
設定 **行為**。

物件方法 (public)

void operator = (const ImqQueueManager & 經理);
必要的話，中斷連接，並從 *mgr* 複製實例資料。 **連線狀態** 為 FALSE。

ImqBoolean accountingConnOverride (MQLONG & statint);
提供帳戶連線置換值的副本。 如果成功，它會傳回 TRUE。

MQLONG accountingConnOverride ();
傳回帳戶連線置換值，而不指出任何可能的錯誤。

ImqBoolean accountingInterval (MQLONG & statint);
提供帳戶間隔值的副本。 如果成功，它會傳回 TRUE。

MQLONG accountingInterval ();
傳回帳戶間隔值，不含任何可能錯誤的指示。

ImqBoolean activityRecording (MQLONG & rec);
提供活動記錄值的副本。 如果成功，它會傳回 TRUE。

MQLONG activityRecording ();
傳回活動記錄值，不含任何可能錯誤的指示。

ImqBoolean adoptNewMCACheck (MQLONG & check);
提供採用新 MCA 檢查值的副本。 如果成功，它會傳回 TRUE。

MQLONG adoptNewMCACheck ();
傳回採用新的 MCA 檢查值，且沒有任何可能錯誤的指示。

ImqBoolean adoptNewMCAType (MQLONG & type);
提供採用新 MCA 類型的副本。 如果成功，它會傳回 TRUE。

MQLONG adoptNewMCAType ();
傳回採用新的 MCA 類型，且沒有任何可能錯誤的指示。

QLONG authenticationType () const;
傳回鑑別類型。

void setAuthenticationType (const MQLONG type = MQCSP_AUTH_NONE);
設定鑑別類型。

ImqBoolean authorityEvent(MQLONG & 事件);
提供 **權限事件**的啟用狀態副本。 如果成功，它會傳回 TRUE。

MQLONG authorityEvent();
傳回 **權限事件** 的啟用狀態，不指出任何可能的錯誤。

ImqBoolean backout ();
取消未確定的變更。 如果成功，它會傳回 TRUE。

ImqBoolean begin ();
開始工作單元。 **開始選項** 會影響此方法的行為。 如果成功，它會傳回 TRUE ，但也會傳回 TRUE ，即使基礎 MQBEGIN 呼叫會傳回 MQRC_NO_EXTERNAL_PARTICIPANTS 或 MQRC_PARTICIPANT_NOT_AVAILABLE (兩者都與 MQCC_WARNING 相關聯)。

MQLONG beginOptions() const;
傳回 **begin options**。

void setBeginOptions (const MQLONG *options* = MQBO_NONE);
設定 **開始選項**。

ImqBoolean bridgeEvent (MQLONG 及事件);
提供橋接器事件值的副本。 如果成功，它會傳回 TRUE。

MQLONG bridgeEvent ();
傳回橋接器事件值，但不指出任何可能的錯誤。

ImqBoolean channelAutoDefinition(MQLONG & 價值);
提供 **通道自動定義** 值的副本。 如果成功，它會傳回 TRUE。

MQLONG channelAutoDefinition ();
傳回 **通道自動定義** 值，不含任何可能錯誤的指示。

ImqBoolean channelAutoDefinitionEvent(MQLONG & 價值);
提供 **通道自動定義事件** 值的副本。 如果成功，它會傳回 TRUE。

MQLONG channelAutoDefinitionEvent();
傳回 **通道自動定義事件** 值，不含任何可能錯誤的指示。

ImqBoolean channelAutoDefinitionExit(ImqString & 名);
提供 **通道自動定義結束程式** 名稱的副本。 如果成功，它會傳回 TRUE。

ImqString channelAutoDefinitionExit( );
傳回 **通道自動定義結束程式** 名稱，不指出任何可能的錯誤。

ImqBoolean channelEvent (MQLONG 及事件);
提供通道事件值的副本。 如果成功，它會傳回 TRUE。

MQLONG channelEvent();
傳回通道事件值，不指出任何可能的錯誤。

MQLONG channelInitiatorAdapters ();
傳回通道起始程式配接卡值，不指出任何可能的錯誤。

ImqBoolean channelInitiator 配接器 (MQLONG 及配接器);
提供通道起始程式配接卡值的副本。 如果成功，它會傳回 TRUE。

MQLONG channelInitiatorControl ();
傳回通道起始程式啟動值，但不指出任何可能的錯誤。

ImqBoolean channelInitiator 控制 (MQLONG 及 init);
提供通道起始程式控制啟動值的副本。 如果成功，它會傳回 TRUE。

MQLONG channelInitiatorDispatchers ();
傳回通道起始程式分派器值，不含任何可能錯誤的指示。

ImqBoolean channelInitiator 分派器 (MQLONG 及分派器);
提供通道起始程式分派器值的副本。 如果成功，它會傳回 TRUE。

MQLONG channelInitiatorTraceAutoStart ();
傳回通道起始程式追蹤自動啟動值，不指出任何可能的錯誤。

ImqBoolean channelInitiatorTraceAutoStart (MQLONG & auto);
提供通道起始程式追蹤自動啟動值的副本。 如果成功，它會傳回 TRUE。

MQLONG channelInitiatorTraceTable 大小 ();
傳回通道起始程式追蹤表格大小值，不指出任何可能的錯誤。

ImqBoolean channelInitiatorTraceTable 大小 (MQLONG 及大小);
提供通道起始程式追蹤表格大小值的副本。 如果成功，它會傳回 TRUE。

ImqBoolean channelMonitoring (MQLONG & monchl);
提供通道監視值的副本。 如果成功，它會傳回 TRUE。

MQLONG channelMonitoring ();
傳回通道監視值，不含任何可能錯誤的指示。

ImqBoolean channelReference(ImqChannel * & ***pchannel*** **);**
提供 **通道參照**的副本。 如果 **通道參照** 無效，請將 *pchannel* 設為空值。 如果成功，此方法會傳回 TRUE。

ImqChannel * channelReference( );
傳回 **通道參照** ，但不指出可能的錯誤。

ImqBoolean setChannelReference(ImqChannel & ***channel*** **);**
設定 **通道參照**。 如果成功，此方法會傳回 TRUE。

ImqBoolean setChannelReference (ImqChannel * ***channel*** **= 0);**
設定或重設 **通道參照**。 如果成功，此方法會傳回 TRUE。

ImqBoolean channelStatistics (MQLONG & statchl);
提供通道統計資料值的副本。 如果成功，它會傳回 TRUE。

MQLONG channelStatistics ();
傳回通道統計資料值，不含任何可能錯誤的指示。

ImqBoolean characterSet(MQLONG & ***ccsid*** **);**
提供 **字集**的副本。 如果成功，它會傳回 TRUE。

MQLONG characterSet();
傳回 **字集**的副本，不指出任何可能的錯誤。

MQLONG clientSslKeyReset 計數 () const;
傳回用戶端連線上使用的 SSL 金鑰重設計數值。

void setClientSslKeyResetCount(const MQLONG count);
設定用戶端連線上使用的 SSL 金鑰重設計數。

ImqBoolean clusterSenderMonitoring (MQLONG 及 monacls);
提供叢集傳送端監視預設值的副本。 如果成功，它會傳回 TRUE。

MQLONG clusterSenderMonitoring ();
傳回叢集傳送端監視預設值，不含任何可能錯誤的指示。

ImqBoolean clusterSender 統計資料 (MQLONG 及 statacls);
提供叢集傳送端統計資料值的副本。 如果成功，它會傳回 TRUE。

MQLONG clusterSender 統計資料 ();
傳回叢集傳送端統計資料值，不指出任何可能的錯誤。

ImqBoolean clusterWorkloadData(ImqString & 資料);
提供 **叢集工作量結束程式資料**的副本。 如果成功，它會傳回 TRUE。

ImqString clusterWorkload 資料 ();
傳回 **叢集工作量結束程式資料** ，沒有任何可能錯誤的指示。

ImqBoolean clusterWorkloadExit(ImqString & 名);
提供 **叢集工作量結束程式名稱**的副本。 如果成功，它會傳回 TRUE。

ImqString clusterWorkloadExit ();
傳回 **叢集工作量結束程式名稱** ，但不指出任何可能的錯誤。

ImqBoolean clusterWorkloadLength(MQLONG & 長度);
提供 **叢集工作量長度**的副本。 如果成功，它會傳回 TRUE。

MQLONG clusterWorkload 長度 ();
傳回 **叢集工作量長度** ，且沒有任何可能錯誤的指示。

ImqBoolean clusterWorkLoadMRU (MQLONG & mru);
提供叢集工作量最近使用的通道值的副本。 如果成功，它會傳回 TRUE。

MQLONG clusterWorkLoadMRU ();
傳回叢集工作量最近使用的通道值，不含任何可能錯誤的指示。

ImqBoolean clusterWorkLoadUseQ (MQLONG 及 useq);
提供叢集工作量使用佇列值的副本。 如果成功，它會傳回 TRUE。

MQLONG clusterWorkLoadUseQ ();
傳回叢集工作量使用佇列值，不指出任何可能的錯誤。

ImqBoolean commandEvent (MQLONG 及事件);
提供指令事件值的副本。 如果成功，它會傳回 TRUE。

MQLONG commandEvent ();
傳回指令事件值，但不指出任何可能的錯誤。

ImqBoolean commandInputQueueName(ImqString & 名);
提供 **指令輸入佇列名稱**的副本。 如果成功，它會傳回 TRUE。

ImqString commandInputQueueName( );
傳回 **指令輸入佇列名稱** ，不指出任何可能的錯誤。

ImqBoolean commandLevel(MQLONG & 水平);
提供 **指令層次**的副本。 如果成功，它會傳回 TRUE。

MQLONG commandLevel();
傳回 **指令層次** ，不指出任何可能的錯誤。

MQLONG commandServerControl ();
傳回指令伺服器啟動值，但不指出任何可能的錯誤。

ImqBoolean commandServer 控制 (MQLONG 及伺服器);
提供指令伺服器控制啟動值的副本。 如果成功，它會傳回 TRUE。

ImqBoolean commit ();
確定未確定的變更。 如果成功，它會傳回 TRUE。

ImqBoolean connect ();
連接至具有給定 ImqObject **名稱**的佇列管理程式，預設值為本端佇列管理程式。 如果您要連接至特定佇列管理程式，請在連線之前使用 ImqObject **setName** 方法。 如果有 **通道參照**，則會用來將通道定義的相關資訊傳遞至 MQCD 中的 MQCONNX。 MQCD 中的 ChannelType 設定為 MQCHT_CLNTCONN。 對於伺服器連線，會忽略 **通道參照** 資訊 (僅對用戶端連線有意義)。 **連接選項** 會影響此方法的行為。 如果成功，此方法會將 **連線狀態** 設為 TRUE。 它會傳回新的連線狀態。

如果有第一個鑑別記錄，則會使用鑑別記錄鏈來鑑別安全用戶端通道的數位憑證。

您可以將多個 ImqQueue 管理程式物件連接至相同的佇列管理程式。 所有人都使用相同的 MQHCONN 連線控點，並針對與執行緒相關聯的連線共用 UOW 功能。 第一個連接的 ImqQueue 管理程式會取得 MQHCONN 控點。 最後一個中斷連線的 ImqQueue 管理程式會執行 MQDISC。

對於多執行緒程式，建議對每一個執行緒使用個別「ImqQueue 管理程式」物件。

ImqBinary connectionId () const;
傳回連線 ID。

ImqBinary connectionTag () const;
傳回 **連線標籤**。

ImqBoolean setConnectionTag (const MQBYTE128 *tag* = 0);
設定 **連線標籤**。 如果 標籤 為零，則清除 **連線標籤**。 如果成功，此方法會傳回 TRUE。

ImqBoolean setConnectionTag (const ImqBinary & 標籤);
設定 **連線標籤**。 標籤 的 **資料長度** 必須為零 (以清除 **連線標籤**) 或 MQ_CONN_TAG_LENGTH。 如果成功，此方法會傳回 TRUE。

MQLONG connectOptions() const;
傳回 **連接選項**。

void setConnectOptions (const MQLONG *options* = MQCNO_NONE);
設定 **連接選項**。

ImqBoolean connectionStatus() const;
傳回 **連線狀態**。

ImqString cryptographicHardware ( );
傳回 **加密硬體**。

ImqBoolean setCryptographicHardware (const char * *hardware* = 0);
設定 **加密硬體**。 如果成功，此方法會傳回 TRUE。

ImqBoolean deadLetterQueueName(ImqString & 名);
提供 **無法傳送郵件的佇列名稱**的副本。 如果成功，它會傳回 TRUE。

ImqString deadLetterQueueName( );
傳回 **無法傳送郵件的佇列名稱**的副本，沒有任何可能錯誤的指示。

ImqBoolean defaultTransmissionQueueName(ImqString & 名);
提供 **預設傳輸佇列名稱**的副本。 如果成功，它會傳回 TRUE。

ImqString defaultTransmissionQueueName( );
傳回 **預設傳輸佇列名稱** ，不指出任何可能的錯誤。

ImqBoolean disconnect ();
中斷與佇列管理程式的連線，並將 **連線狀態** 設為 FALSE。 在斷線之前，關閉所有與此物件相關聯的 ImqProcess 和 ImqQueue 物件，並關閉它們的 **連線參照** 。 如果有多個「ImqQueue 管理程式」物件連接至相同的佇列管理程式，則只有最後一個斷線會執行實體斷線; 其他會執行邏輯斷線。 只有在實體斷線時才會確定未確定的變更。

如果成功，此方法會傳回 TRUE。 如果在沒有現有連線時呼叫它，則回覆碼也是 true。

ImqBoolean distributionLists(MQLONG & 支援);
提供 **配送清單** 值的副本。 如果成功，它會傳回 TRUE。

MQLONG distributionLists();
傳回 **配送清單** 值，但不指出可能的錯誤。

ImqBoolean dnsGroup (ImqString & group);
提供 DNS 群組名稱的副本。 如果成功，它會傳回 TRUE。

ImqString dnsGroup ( );
傳回 DNS 群組名稱，但不指出任何可能的錯誤。

ImqBoolean dnsWlm (MQLONG & wlm);
提供 DNS WLM 值的副本。 如果成功，它會傳回 TRUE。

MQLONG dnsWlm ();
傳回 DNS WLM 值，但不指出任何可能的錯誤。

ImqAuthenticationRecord * firstAuthenticationRecord () const;
傳回 **第一個鑑別記錄**。

void setFirstAuthenticationRecord (const ImqAuthenticationRecord * *air* **= 0);**
設定 **第一個鑑別記錄**。

ImqObject * firstManagedObject () const;
傳回 **第一個受管理物件**。

ImqBoolean inhibitEvent(MQLONG & 事件);
提供 **禁止事件**的啟用狀態副本。 如果成功，它會傳回 TRUE。

MQLONG inhibitEvent();
傳回 **禁止事件** 的啟用狀態，而不指出任何可能的錯誤。

ImqBoolean ipAddressVersion (MQLONG & version);
提供 IP 位址版本值的副本。 如果成功，它會傳回 TRUE。

MQLONG ipAddress 版本 ();
傳回 IP 位址版本值，不指出任何可能的錯誤。

ImqBoolean keepAlive (MQLONG 及 keepalive);
提供「保持作用中」值的副本。 如果成功，它會傳回 TRUE。

MQLONG keepAlive ();
傳回保持作用中值，不指出任何可能的錯誤。

ImqString keyRepository ( );
傳回 **金鑰儲存庫**。

ImqBoolean setKeyRepository (const char * *repository* **= 0);**
設定 **金鑰儲存庫**。 如果成功，它會傳回 TRUE。

ImqBoolean listenerTimer (MQLONG 及計時器);
提供接聽器計時器值的副本。 如果成功，它會傳回 TRUE。

MQLONG listenerTimer ();
傳回接聽器計時器值，不指出任何可能的錯誤。

ImqBoolean localEvent(MQLONG & 事件);
提供 **本端事件**的啟用狀態副本。 如果成功，它會傳回 TRUE。

MQLONG localEvent();
傳回 **本端事件** 的啟用狀態，而不指出任何可能的錯誤。

ImqBoolean loggerEvent (MQLONG 及計數);
提供日誌程式事件值的副本。 如果成功，它會傳回 TRUE。

MQLONG loggerEvent ();
傳回日誌程式事件值，但不指出任何可能的錯誤。

ImqBoolean luGroupName (ImqString & name);.
提供 LU 群組名稱的副本。 如果成功，它會傳回 TRUE

ImqString luGroupName ();
傳回 LU 群組名稱，但不指出任何可能的錯誤。

ImqBoolean lu62ARMSuffix (ImqString & suffix);
提供 LU62 ARM 字尾的副本。 如果成功，它會傳回 TRUE。

ImqString lu62ARMSuffix ( );
傳回 LU62 ARM 字尾，沒有任何可能錯誤的指示

ImqBoolean luName (ImqString & name);
提供 LU 名稱的副本。 如果成功，它會傳回 TRUE。

ImqString luName ( );
傳回 LU 名稱，但不指出任何可能的錯誤。

ImqBoolean maximumActive 通道 (MQLONG 及通道);
提供作用中通道數上限值的副本。 如果成功，它會傳回 TRUE。

MQLONG maximumActive 通道 ();
傳回作用中通道數上限值，不指出任何可能的錯誤。

ImqBoolean maximumCurrent 通道 (MQLONG 及通道);
提供現行通道值上限的副本。 如果成功，它會傳回 TRUE。

MQLONG maximumCurrent 通道 ();
傳回現行通道數目上限值，而不指出任何可能的錯誤。

ImqBoolean maximumHandles(MQLONG & 數字);
提供 **控點數目上限**的副本。 如果成功，它會傳回 TRUE。

MQLONG maximumHandles();
傳回 **控點數目上限** ，且不指出任何可能的錯誤。

ImqBoolean maximumLu62Channels (MQLONG 及通道);
提供 LU62 通道數上限值的副本。 如果成功，它會傳回 TRUE。

MQLONG maximumLu62Channels ();。
傳回沒有任何可能錯誤指示的 LU62 通道數上限值

ImqBoolean maximumMessageLength(MQLONG & 長度);
提供 **訊息長度上限**的副本。 如果成功，它會傳回 TRUE。

MQLONG maximumMessage 長度 ();
傳回 **訊息長度上限** ，不指出任何可能的錯誤。

ImqBoolean maximumPriority(MQLONG & 優先);
提供 **優先順序上限**的副本。 如果成功，它會傳回 TRUE。

MQLONG maximumPriority();
傳回 **maximum priority** 的副本，沒有任何可能錯誤的指示。

ImqBoolean maximumTcp 通道 (MQLONG 及通道);
提供 TCP 通道值上限的副本。 如果成功，它會傳回 TRUE。

MQLONG maximumTcp 通道 ();
傳回 TCP 通道數上限值，不指出任何可能的錯誤。

ImqBoolean maximumUncommittedMessages(MQLONG & 數字);
提供 **未確定的訊息數上限**的副本。 如果成功，它會傳回 TRUE。

MQLONG maximumUncommitted 訊息 ();
傳回 **未確定的訊息數上限** ，沒有任何可能的錯誤指示。

ImqBoolean mqiAccounting (MQLONG & statint);
提供 MQI 帳戶值的副本。 如果成功，它會傳回 TRUE。

MQLONG mqiAccounting ();
傳回 MQI 帳戶值，不指出任何可能的錯誤。

ImqBoolean mqiStatistics (MQLONG & statmqi);
提供 MQI 統計資料值的副本。 如果成功，它會傳回 TRUE。

MQLONG mqiStatistics ();
傳回 MQI 統計資料值，不指出任何可能的錯誤。

ImqBoolean outboundPortMax (MQLONG & max);
提供出埠埠值上限的副本。 如果成功，它會傳回 TRUE。

MQLONG outboundPortMax ();
傳回出埠連接埠值上限，不指出任何可能的錯誤。

ImqBoolean outboundPort 最小值 (MQLONG 及最小值);
提供出埠埠值下限的副本。 如果成功，它會傳回 TRUE。

MQLONG outboundPortMin ();
傳回最小出埠連接埠值，不指出任何可能的錯誤。

ImqBinary 密碼 () const;
傳回用戶端連線上使用的密碼。

ImqBoolean setPassword (const ImqString & password);
設定用於用戶端連線的密碼。

ImqBoolean setPassword (const char * = 0 password);
設定用於用戶端連線的密碼。

ImqBoolean setPassword (const ImqBinary & password);
設定用於用戶端連線的密碼。

ImqBoolean performanceEvent(MQLONG & 事件);
提供 **效能事件**的啟用狀態副本。 如果成功，它會傳回 TRUE。

MQLONG performanceEvent();
傳回 **效能事件** 的啟用狀態，不指出任何可能的錯誤。

ImqBoolean platform(MQLONG & 平臺);
提供 **平台**的副本。 如果成功，它會傳回 TRUE。

MQLONG 平台 ();
傳回 **platform** ，不指出任何可能的錯誤。

ImqBoolean queueAccounting (MQLONG 及 acctq);
提供佇列帳戶值的副本。 如果成功，它會傳回 TRUE。

MQLONG queueAccounting ();
傳回佇列帳戶值，但不指出任何可能的錯誤。

ImqBoolean queueMonitoring (MQLONG & monq);
提供佇列監視值的副本。 如果成功，它會傳回 TRUE。

MQLONG queueMonitoring ();
傳回佇列監視值，但不指出任何可能的錯誤。

ImqBoolean queueStatistics (MQLONG & statq);
提供佇列統計資料值的副本。 如果成功，它會傳回 TRUE。

MQLONG queueStatistics ();
傳回佇列統計資料值，但不指出任何可能的錯誤。

ImqBoolean receiveTimeout (MQLONG 及逾時);
提供接收逾時值的副本。 如果成功，它會傳回 TRUE。

MQLONG receiveTimeout ();
傳回接收逾時值，不指出任何可能的錯誤。

ImqBoolean receiveTimeout 最小值 (MQLONG 及最小值);
提供接收逾時值下限的副本。 如果成功，它會傳回 TRUE。

MQLONG receiveTimeoutMin ();
傳回接收逾時值下限，不含任何可能錯誤的指示。

ImqBoolean receiveTimeout 類型 (MQLONG 及類型);
提供接收逾時類型的副本。 如果成功，它會傳回 TRUE。

MQLONG receiveTimeout 類型 ();
傳回接收逾時類型，不指出任何可能的錯誤。

ImqBoolean remoteEvent(MQLONG & 事件);
提供 **遠端事件**的啟用狀態副本。 如果成功，它會傳回 TRUE。

MQLONG remoteEvent();
傳回 **遠端事件** 的啟用狀態，不指出任何可能的錯誤。

ImqBoolean repositoryName(ImqString & 名);
提供 **儲存庫名稱**的副本。 如果成功，它會傳回 TRUE。

ImqString repositoryName( );
傳回 **儲存庫名稱** ，但不指出任何可能的錯誤。

ImqBoolean repositoryNamelistName(ImqString & 名);
提供 **儲存庫名單名稱**的副本。 如果成功，它會傳回 TRUE。

ImqString repositoryNamelistName ();
傳回 **儲存庫名單名稱** 的副本，且不會指出任何可能的錯誤。

ImqBoolean sharedQueueQueueManagerName (MQLONG & name);
提供共用佇列管理程式名稱值的副本。 如果成功，它會傳回 TRUE。

MQLONG sharedQueueQueueManager 名稱 ();
傳回共用佇列管理程式名稱值，且不指出任何可能的錯誤。

ImqBoolean sslEvent (MQLONG 及事件);
提供 SSL 事件值的副本。 如果成功，它會傳回 TRUE。

MQLONG sslEvent ();
傳回 SSL 事件值，但不指出任何可能的錯誤。

ImqBoolean sslFips (MQLONG 及 sslfips);
提供 SSL FIPS 值的副本。 如果成功，它會傳回 TRUE。

MQLONG sslFips ();
傳回 SSL FIPS 值，但不指出任何可能的錯誤。

ImqBoolean sslKeyResetCount (MQLONG 及計數);
提供 SSL 金鑰重設計數值的副本。 如果成功，它會傳回 TRUE。

MQLONG sslKeyResetCount ();
傳回 SSL 金鑰重設計數值，且沒有任何可能錯誤的指示。

ImqBoolean startStopEvent(MQLONG & 事件);
提供 **開始-停止事件**的啟用狀態副本。 如果成功，它會傳回 TRUE。

MQLONG startStopEvent ();
傳回 **啟動-停止事件** 的啟用狀態，不指出任何可能的錯誤。

ImqBoolean statisticsInterval (MQLONG & statint);
提供統計資料間隔值的副本。 如果成功，它會傳回 TRUE。

MQLONG statisticsInterval ();
傳回統計資料間隔值，不含任何可能錯誤的指示。

ImqBoolean syncPointAvailability(MQLONG & 同步);
提供 **同步點可用性** 值的副本。 如果成功，它會傳回 TRUE。

MQLONG syncPoint 可用性 ();
傳回 **同步點可用性** 值的副本，不指出任何可能的錯誤。

ImqBoolean tcpName (ImqString & name);
提供 TCP 系統名稱的副本。 如果成功，它會傳回 TRUE。

ImqString tcpName ( );
傳回 TCP 系統名稱，但不指出任何可能的錯誤。

ImqBoolean tcpStack 類型 (MQLONG 及類型);
提供 TCP 堆疊類型的副本。 如果成功，它會傳回 TRUE。

MQLONG tcpStack 類型 ();
傳回 TCP 堆疊類型，不指出任何可能的錯誤。

ImqBoolean traceRoute 記錄 (MQLONG 及 routerec);
提供追蹤路徑記錄值的副本。 如果成功，它會傳回 TRUE。

MQLONG traceRoute 記錄 ();
傳回追蹤路徑記錄值，不指出任何可能的錯誤。

ImqBoolean triggerInterval(MQLONG & 間隔);
提供 **觸發間隔**的副本。 如果成功，它會傳回 TRUE。

MQLONG triggerInterval();
傳回 **觸發間隔** ，不指出任何可能的錯誤。

ImqBinary userId () const;
傳回用於用戶端連線的使用者 ID。

ImqBoolean setUserId (const ImqString & id);
設定用於用戶端連線的使用者 ID。

ImqBoolean setUserId (const char * = 0 id);
設定用於用戶端連線的使用者 ID。

ImqBoolean setUserId (const ImqBinary & id);
設定用於用戶端連線的使用者 ID。

物件方法 (protected)

void setFirstManagedObject(const ImqObject * *object* = 0);
設定 **第一個受管理物件**。

物件資料 (受保護)

MQHCONN 連線
WebSphere MQ 連線控點 (只有在 **連線狀態** 為 TRUE 時才有意義)。

原因碼

- MQRC_ATTRIBUTE_LOCKED
- MQRC_ENVIRONMENT_ERROR
- MQRC_FUNCTION_NOT_SUPPORTED
- MQRC_REFERENCE_ERROR
- (MQBACK 的原因碼)
- (MQBEGIN 的原因碼)
- (MQCMIT 的原因碼)
- (MQCONNX 的原因碼)
- (MQDISC 的原因碼)
- (MQCONN 的原因碼)

ImqReference 標頭 C++ 類別

此類別封裝 MQRMH 資料結構的特性。

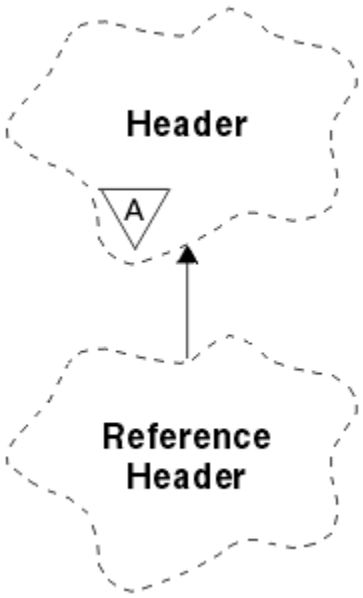


圖 *66: ImqReference* 標頭類別

此類別與 第 1190 頁的『ImqReference 標頭交互參照』中列出的 MQI 呼叫相關。

- 第 1267 頁的『物件屬性』
- 第 1267 頁的『建構子』
- 第 1267 頁的『超載 ImqItem 方法』
- 第 1267 頁的『物件方法 (public)』
- 第 1268 頁的『物件資料 (受保護)』
- 第 1268 頁的『原因碼』

物件屬性

目的地環境
目的地的環境。 起始值是空字串。

目的地名稱
資料目的地的名稱。 起始值是空字串。

實例 ID
實例 ID。 長度為 MQ_OBJECT_INSTANCE_ID_LENGTH 的二進位值 (MQBYTE24)。 起始值為 MQOII_NONE。

邏輯長度
此標頭後面的訊息資料的邏輯或預期長度。 起始值為零。

邏輯偏移
在最終目的地，要在資料整體環境定義中解譯之下列訊息資料的邏輯偏移。 起始值為零。

邏輯偏移 2
邏輯偏移的高階延伸。 起始值為零。

參照類型
參照類型。 起始值是空字串。

來源環境
來源的環境。 起始值是空字串。

來源名稱
資料來源的名稱。 起始值是空字串。

建構子

ImqReferenceHeader ();
預設建構子。

ImqReference 標頭 (const ImqReferenceHeader & 標頭);
複製建構子。

超載 ImqItem 方法

虛擬 ImqBoolean copyOut(ImqMessage & 訊息);
在開始時將 MQRMH 資料結構插入訊息緩衝區，進一步移動現有的訊息資料，並將 *msg* **格式** 設為 MQFMT_REF_MSG_HEADER。

如需進一步詳細資料，請參閱 第 1216 頁的『ImqHeader C++ 類別』 上的 ImqHeader 類別方法說明。

虛擬 ImqBoolean pasteIn(ImqMessage & 訊息);
從訊息緩衝區讀取 MQRMH 資料結構。

若要成功， ImqMessage **格式** 必須是 MQFMT_REF_MSG_HEADER。

如需進一步詳細資料，請參閱 第 1216 頁的『ImqHeader C++ 類別』 上的 ImqHeader 類別方法說明。

物件方法 (public)

void operator = (const ImqReferenceHeader & 標頭);
從 標頭複製實例資料，以取代現有的實例資料。

ImqString destinationEnvironment() const ;
傳回 **目的地環境**的副本。

void setDestinationEnvironment(const char * *environment* = 0);
設定 **目的地環境**。

ImqString destinationName() const ;
傳回 **目的地名稱**的副本。

void setDestinationName(const char * *name* = 0);
設定 **目的地名稱**。

ImqBinary instanceId() const ;
傳回 **實例 ID** 的副本。

ImqBoolean setInstanceID(const ImqBinary & *ID*);
設定 **實例 ID**。 *token* 的 **資料長度** 必須是 0 或 MQ_OBJECT_INSTANCE_ID_LENGTH。 如果成功，此方法會傳回 TRUE。

void setInstanceId(const MQBYTE24 *id* = 0);
設定 **實例 ID**。 *id* 可以是零，這與指定 MQOII_NONE 相同。 如果 *id* 不是零，則必須解決二進位資料的 MQ_OBJECT_INSTANCE_ID_LENGTH 個位元組。 使用預先定義的值 (例如 MQOII_NONE) 時，您可能需要進行強制轉型以確保簽章相符，例如 (MQBYTE *) MQOII_NONE。

MQLONG logicalLength() const ;
傳回 **邏輯長度**。

void setLogicalLength(const MQLONG *length*);
設定 **邏輯長度**。

MQLONG logicalOffset() const ;
傳回 **邏輯偏移**。

void setLogicalOffset(const MQLONG *offset*);
設定 **邏輯偏移**。

MQLONG logicalOffset2() const ;
傳回 **邏輯偏移 2**。

void setLogicalOffset2(const MQLONG *offset*);
設定 **邏輯偏移 2**。

ImqString referenceType() const ;
傳回 **參照類型**的副本。

void setReferenceType(const char * *name* = 0);
設定 **參照類型**。

ImqString sourceEnvironment() const ;
傳回 **來源環境**的副本。

void setSourceEnvironment(const char * *environment* = 0);
設定 **來源環境**。

ImqString sourceName() const ;
傳回 **來源名稱**的副本。

void setSourceName(const char * *name* = 0);
設定 **來源名稱**。

物件資料 (受保護)

MQRMH *omqrmh*
MQRMH 資料結構。

原因碼

- MQRC_BINARY_DATA_LENGTH_ERROR
- MQRC_STRUC_LENGTH_ERROR

- MQRC_STRUC_ID_ERROR
- MQRC_INSUFFICIENT_DATA
- MQRC_INCONSISTENT_FORMAT
- MQRC_ENCODING_ERROR

ImqString C++ 類別

此類別為以空值結尾的字串提供字串儲存及操作。

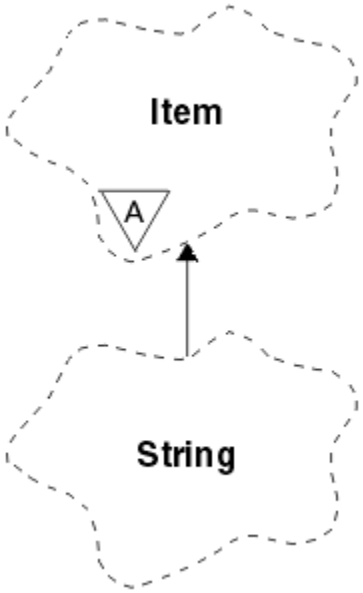


圖 *67: ImqString* 類別

在參數呼叫 **char *** 的大部分情況下，請使用 ImqString 來取代 **char *** 。

- 第 1269 頁的『物件屬性』
- 第 1269 頁的『建構子』
- 第 1270 頁的『類別方法 (public)』
- 第 1270 頁的『超載 ImqItem 方法』
- 第 1270 頁的『物件方法 (public)』
- 第 1273 頁的『物件方法 (protected)』
- 第 1273 頁的『原因碼』

物件屬性

個字元後
儲存體 中位於尾端空值之前的字元。

長度
字元中的位元組數。 如果沒有 **儲存體**，則 **長度** 為零。 起始值為零。

儲存體
任意大小的暫時位元組陣列。 尾端空值必須一律出現在 **儲存體** 中的 **字元**之後，以便可以偵測到 **字元** 的結尾。 方法可確保維護此狀況，但在直接設定陣列中的位元組時，請確保修改之後存在尾端空值。 一開始，沒有 **storage** 屬性。

建構子

ImqString( );
預設建構子。

ImqString(const ImqString & 弦);
複製建構子。

ImqString(const char *c*);
字元 包含 *c*。

ImqString(const char * *text*);
從 *text* 中複製 **字元** 。

ImqString(const void * *buffer*, const size_t *length*);
從 緩衝區 開始複製 長度 位元組，並將它們指派給 **字元**。 對複製的任何空值字元進行替代。 替代字元是句點 (.)。 對於複製的任何其他不可列印或不可顯示的字元，不會有特殊考量。

類別方法 (public)

static ImqBoolean copy (char * *destination-buffer*, const size_t *length*, const char * *source-buffer*, const char *pad* = 0);
將最多 長度 個位元組從 *source-buffer* 複製到 *destination-buffer*。 如果 *source-buffer* 中的字元數不足，請在 *destination-buffer* 的剩餘空間中填入 *pad* 字元。 *source-buffer* 可以是零。 如果 *length* 也是零，則 *destination-buffer* 可以是零。 任何錯誤碼都會遺失。 如果成功，此方法會傳回 TRUE。

static ImqBoolean copy (char * 目的地緩衝區, const size_t 長度, const char * *source-buffer*, ImqError & 錯誤-物件, const char 墊 = 0);
將最多 長度 個位元組從 *source-buffer* 複製到 *destination-buffer*。 如果 *source-buffer* 中的字元數不足，請在 *destination-buffer* 的剩餘空間中填入 *pad* 字元。 *source-buffer* 可以是零。 如果 *length* 也是零，則 *destination-buffer* 可以是零。 任何錯誤碼都設定在 *error-object* 中。 如果成功，此方法會傳回 TRUE。

超載 ImqItem 方法

虛擬 ImqBoolean copyOut(ImqMessage & 訊息);
將 **字元** 複製到訊息緩衝區，以取代任何現有的內容。 將 *msg* **格式** 設為 MQFMT_STRING。

如需進一步詳細資料，請參閱母類別方法說明。

虛擬 ImqBoolean pasteIn(ImqMessage & 訊息);
從訊息緩衝區傳送剩餘資料，並取代現有的 **字元**，以設定 **字元** 。

若要成功， *msg* 物件的 **編碼** 必須是 MQENC_NATIVE。 擷取具有 MQGMO_CONVERT 至 MQENC_NATIVE 的訊息。

若要成功， ImqMessage **格式** 必須是 MQFMT_STRING。

如需進一步詳細資料，請參閱母類別方法說明。

物件方法 (public)

char & operator [ ] (const size_t 偏移) const ;
在 **儲存體**中，參照偏移 *offset* 的字元。 請確定相關位元組存在且可定址。

ImqString 運算子 () (const size_t *offset*, const size_t *length* = 1) const;
從 *offset* 開始，複製 **字元** 中的位元組，以傳回子字串。 如果 *length* 為零，則會傳回其餘 **字元**。 如果 *offset* 與 *length* 的組合未在 **字元**內產生參照，則會傳回空的 ImqString。

void operator = (const ImqString & 弦);
從 *string* 複製實例資料，並取代現有的實例資料。

ImqString operator + (const char *c*) const;
傳回將 *c* 附加至 **字元**的結果。

ImqString operator + (const char * *text*) const;
傳回將 *text* 附加至 **字元**的結果。 這也可以顛倒。 例如：

```
strOne + "string two" ;
"string one" + strTwo ;
```

註: 雖然大部分編譯器都接受 **strOne + "string 2";** Microsoft Visual C++ 需要 **strOne + (char *) "string 2";**

ImqString operator + (const ImqString & *string1*) const ;
傳回將 *string1* 附加至 **字元**的結果。

ImqString operator + (const double *number*) const;
傳回在轉換為文字之後，將 *number* 附加至 **字元** 的結果。

ImqString operator + (const long *number*) const;
傳回在轉換為文字之後，將 *number* 附加至 **字元** 的結果。

void operator + = (const char *c*);
將 *c* 附加至 **字元**。

void operator + = (const char * *text*);
將 *text* 附加至 **字元**。

void operator += (const ImqString & 弦);
將 *string* 附加至 **字元**。

void operator + = (const double *number*);
在轉換為文字之後，將 數字 附加至 **字元** 。

void operator + = (const long *number*);
在轉換為文字之後，將 數字 附加至 **字元** 。

operator char * () const;
傳回 **儲存體**中第一個位元組的位址。 此值可以是零，並且是暫時的。 此方法僅用於唯讀目的。

ImqBoolean operator < (const ImqString & 弦) const ;
使用 **compare** 方法來比較 **字元** 與 *string* 的字元。 如果小於，則結果為 TRUE ，如果大於或等於，則結果為 FALSE。

ImqBoolean operator > (const ImqString & 弦) const ;
使用 **compare** 方法來比較 **字元** 與 *string* 的字元。 如果大於，則結果為 TRUE ，如果小於或等於，則結果為 FALSE。

ImqBoolean operator <= (const ImqString & 弦) const ;
使用 **compare** 方法來比較 **字元** 與 *string* 的字元。 如果小於或等於，則結果為 TRUE ，如果大於，則結果為 FALSE。

ImqBoolean operator >= (const ImqString & 弦) const ;
使用 **compare** 方法來比較 **字元** 與 *string* 的字元。 如果大於或等於，則結果為 TRUE ，如果小於，則結果為 FALSE。

ImqBoolean operator == (const ImqString & 弦) const ;
使用 **compare** 方法來比較 **字元** 與 *string* 的字元。 它會傳回 TRUE 或 FALSE。

ImqBoolean operator != (const ImqString & 弦) const ;
使用 **compare** 方法來比較 **字元** 與 *string* 的字元。 它會傳回 TRUE 或 FALSE。

short compare(const ImqString & 弦) const ;
比較 **字元** 與 *string* 的字元。 如果 **字元** 相等，則結果為零; 如果小於，則為負數; 如果大於，則為正數。 比較區分大小寫。 空值 ImqString 被視為小於非空值 ImqString。

ImqBoolean copyOut(char * *buffer*, const size_t *length*, const char *pad* = 0);
將最多 長度 個位元組從 **字元** 複製到 緩衝區。 如果 **字元數** 不足，請以 填補 字元填入 *buffer* 中的剩餘空間。 如果 *length* 也是零，則 *buffer* 可以是零。 如果成功，它會傳回 TRUE。

size_t copyOut(long & 數字) const ;
從文字轉換之後，從 **字元** 中設定 數字 ，並傳回轉換所涉及的字元數。 如果此值為零，則未執行任何轉換，且未設定 *number* 。 可轉換字元順序必須以下列值開頭:

```
<blank(s)>
<+|->
digit(s)
```

size_t copyOut(ImqString & 記號, const char *c* = ' ') const ;
如果 **字元** 包含一或多個不同於 *c* 的字元，請將記號識別為這類字元的第一個連續序列。 在此情況下，*token* 會設為該順序，而傳回的值是前導字元數 *c* 與順序中位元組數的總和。 否則，會傳回零，且不會設定 *token*。

size_t cutOut(long & 數字);
將 *number* 設定為適用於 **copy** 方法，但也會從 **字元** 中移除回覆值所指示的位元組數。 例如，透過使用 **cutOut**(*number*) ，可以將下列範例中顯示的字串剪下成三個數字 三次:

```
strNumbers = "-1 0     +55 "

while ( strNumbers.cutOut( number ) );
number becomes -1, then 0, then 55
leaving strNumbers == " "
```

size_t cutOut(ImqString & 記號, const char *c* = ' ' '
將 *token* 設定為 **copyOut** 方法，並從 **字元** 移除 *strToken* 字元，以及 記號 字元之前的任何字元 *c* 。 如果 *c* 不是空白，則移除直接繼承 記號 字元的字元 *c* 。 傳回已移除的字元數。 例如，透過使用 **cutOut**(*token*) ，可以將下列範例中顯示的字串剪下成三個記號 三次:

```
strText = "  Program Version 1.1  "

while ( strText.cutOut( token ) );

// token becomes "Program", then "Version",
// then "1.1" leaving strText == "   "
```

下列範例顯示如何剖析 DOS 路徑名稱:

```
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 & 弦);
在 **字元**內的任何位置搜尋 *string* 的完全相符項。 如果找不到相符項，則會傳回 FALSE。 否則會傳回 TRUE。 如果 *string* 是空值，則會傳回 TRUE。

ImqBoolean find(const ImqString & 弦, size_t & 偏移);
從偏移 *offset* 開始，搜尋 *string* 在 **字元** 內的某個位置。 如果 *string* 是空值，則會傳回 TRUE ，而不會更新 *offset*。 如果找不到相符項，則會傳回 FALSE (*offset* 的值可能已增加)。 如果找到相符項，則會傳回 TRUE 並將 *offset* 更新為 **字元**內 *string* 的偏移。

size_t length () const;
傳回 **length**。

ImqBoolean pasteIn(const double *number*, const char * *format* = "%f");
在轉換為文字之後，將 數字 附加至 **字元** 。 如果成功，它會傳回 TRUE。

使用規格 *format* 來格式化浮點轉換。 如果指定的話，它必須是適用於 **printf** 和浮點數字的項目，例如 **%.3f**。

ImqBoolean pasteIn(const long *number*);
在轉換為文字之後，將 數字 附加至 **字元** 。 如果成功，它會傳回 TRUE。

ImqBoolean pasteIn(const void * *buffer*, const size_t *length*);
將 *buffer* 中的 *length* 個位元組附加至 **字元**，並新增最終尾端空值。 替換所複製的任何空值字元。 替代字元是句點 (.)。 對於任何其他複製的不可列印或不可顯示字元，不會特別考慮。 如果成功，此方法會傳回 TRUE。

ImqBoolean set (const char * *buffer*, const size_t *length*);
從固定長度字元欄位 (可能包含空值) 設定 **字元** 。 必要的話，將空值附加至固定長度欄位中的字元。 如果成功，此方法會傳回 TRUE。

ImqBoolean setStorage(const size_t *length*);
配置 (或重新配置) **儲存體**。 保留任何原始 **字元**，包括任何尾端空值 (如果仍有空間) ，但未起始設定任何其他儲存體。

如果成功，此方法會傳回 TRUE。

size_t 儲存體 () const;
傳回 **儲存體**中的位元組數。

size_t stripLeading(const char *c* = ');
從 **字元** 中刪除前導字元 *c* ，並傳回移除的數字。

size_t stripTrailing(const char *c* = ');
從 **字元** 中除去尾端字元 *c* ，並傳回已移除的數字。

ImqString upperCase() const;
傳回 **字元**的大寫副本。

物件方法 (protected)

ImqBoolean 分配(const ImqString & 弦);
相當於對等的 **operator =** 方法，但非虛擬。 如果成功，它會傳回 TRUE。

原因碼

- MQRC_DATA_TRUNCATED
- MQ RC_NULL_POINTER
- MQRC_STORAGE_NOT_AVAILABLE
- MQRC_BUFFER_ERROR
- MQRC_INCONSISTENT_FORMAT

ImqTrigger C++ 類別

此類別會封裝 MQTM (觸發訊息) 資料結構。

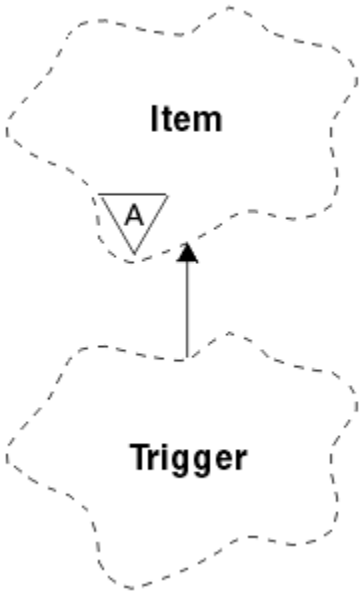


圖 *68: ImqTrigger* 類別

此類別的物件通常由觸發監視器程式使用。 觸發監視器程式的作業是等待這些特定訊息並採取行動，以確保在訊息等待它們時，會啟動其他 WebSphere MQ 應用程式。

如需用法範例，請參閱 IMQSTRG 範例程式。

- 第 1274 頁的『物件屬性』
- 第 1274 頁的『建構子』
- 第 1275 頁的『超載 ImqItem 方法』
- 第 1275 頁的『物件方法 (public)』
- 第 1275 頁的『物件資料 (受保護)』
- 第 1276 頁的『原因碼』

物件屬性

應用程式 ID
傳送訊息的應用程式身分。 起始值是空字串。

應用程式類型
傳送訊息的應用程式類型。 起始值為零。 下列是可能的其他值:

- MQAT_AIX
- MQAT_CICS
- MQAT_DOS
- MQAT_IMS
- MQAT_MVS
- MQAT_NOTES_AGENT
- MQAT_OS2
- MQAT_OS390
- MQAT_OS400
- MQAT_UNIX
- MQ 視窗
- MQAT_WINDOWS_NT
- MQAT_USER_FIRST
- MQAT_USER_LAST

環境資料
處理程序的環境資料。 起始值是空字串。

程序名稱
處理程序名稱。 起始值是空字串。

佇列名稱
要啟動的佇列名稱。 起始值是空字串。

觸發資料
程序的觸發資料。 起始值是空字串。

使用者資料
處理程序的使用者資料。 起始值是空字串。

建構子

ImqTrigger( );
預設建構子。

ImqTrigger(const ImqTrigger & 觸發程式);
複製建構子。

超載 ImqItem 方法

虛擬 ImqBoolean copyOut(ImqMessage & 訊息);
將 MQTM 資料結構寫入訊息緩衝區，以取代任何現有的內容。 將 *msg* **格式** 設為 MQFMT_TRIGGER。

如需進一步詳細資料，請參閱 ImqItem 類別方法說明，網址為 第 1220 頁的『ImqItem C++ 類別』 。

虛擬 ImqBoolean pasteIn(ImqMessage & 訊息);
從訊息緩衝區讀取 MQTM 資料結構。

若要成功， ImqMessage **格式** 必須為 MQFMT_TRIGGER。

如需進一步詳細資料，請參閱 ImqItem 類別方法說明，網址為 第 1220 頁的『ImqItem C++ 類別』 。

物件方法 (public)

void operator = (const ImqTrigger & 觸發程式);
從 觸發程式複製實例資料，以取代現有的實例資料。

ImqString applicationId() const ;
傳回 **應用程式 ID** 的副本。

void setApplicationId(const char * *id*);
設定 **應用程式 ID**。

MQLONG applicationType() const ;
傳回 **應用程式類型**。

void setApplicationType(const MQLONG *type*);
設定 **應用程式類型**。

ImqBoolean copyOut(MQTMC2 * *ptmc2*);
封裝 MQTM 資料結構，這是在起始佇列上收到的資料結構。 填寫呼叫者所提供的對等 MQTMC2 資料結構，並將 QMgrName 欄位 (不在 MQTM 資料結構中) 設為所有空白。 MQTMC2 資料結構傳統上用作觸發監視器所啟動應用程式的參數。 如果成功，此方法會傳回 TRUE。

ImqString environmentData() const ;
傳回 **環境資料**的副本。

void setEnvironmentData(const char * *data*);
設定 **環境資料**。

ImqString processName() const ;
傳回 **程序名稱**的副本。

void setProcessName(const char * *name*);
將 **處理程序名稱**設定為 48 個字元，並以空格填補。

ImqString queueName() const ;
傳回 **佇列名稱**的副本。

void setQueueName(const char * *name*);
設定 **佇列名稱**，並填補空白至 48 個字元。

ImqString triggerData() const ;
傳回 **觸發程式資料**的副本。

void setTriggerData(const char * *data*);
設定 **觸發程式資料**。

ImqString userData() const ;
傳回 **使用者資料**的副本。

void setUserData(const char * *data*);
設定 **使用者資料**。

物件資料 (受保護)

MQTM *omqtm*
MQTM 資料結構。

原因碼

- MQ RC_NULL_POINTER
- MQRC_INCONSISTENT_FORMAT
- MQRC_ENCODING_ERROR
- MQRC_STRUC_ID_ERROR

ImqWork 標頭 C++ 類別

此類別封裝 MQWIH 資料結構的特定特性。

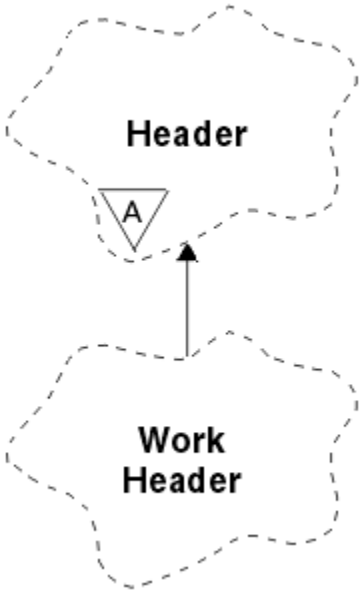


圖 *69: ImqWork* 標頭類別

應用程式會使用此類別的物件，將訊息放入 z/OS 「工作量管理程式」所管理的佇列。

- 第 1276 頁的『物件屬性』
- 第 1276 頁的『建構子』
- 第 1277 頁的『超載 ImqItem 方法』
- 第 1277 頁的『物件方法 (public)』
- 第 1277 頁的『物件資料 (受保護)』
- 第 1277 頁的『原因碼』

物件屬性

訊息記號 (message token)
z/OS 工作量管理程式的訊息記號，長度為 MQ_MSG_TOKEN_LENGTH。 起始值為 MQMTOK_NONE。

服務名稱
處理程序的 32 個字元名稱。 名稱最初為空白。

服務步驟
處理程序內步驟的 8 個字元名稱。 名稱最初為空白。

建構子

ImqWork 標頭 ();
預設建構子。

ImqWorkHeader(const ImqWorkHeader & 標頭);
複製建構子。

超載 ImqItem 方法

virtual ImqBoolean copyOut(ImqMessage & 訊息);
將 MQWIH 資料結構插入訊息緩衝區的開頭，進一步移動現有的訊息資料，並將 *msg* **格式** 設為 MQFMT_WORK_INFO_HEADER。

如需詳細資料，請參閱母類別方法說明。

virtual ImqBoolean pasteIn(ImqMessage & 訊息);
從訊息緩衝區讀取 MQWIH 資料結構。

若要成功， *msg* 物件的編碼必須是 MQENC_NATIVE。 擷取具有 MQGMO_CONVERT 至 MQENC_NATIVE 的訊息。

ImqMessage 格式必須為 MQFMT_WORK_INFO_HEADER。

如需詳細資料，請參閱母類別方法說明。

物件方法 (public)

void operator = (const ImqWorkHeader & 標頭);
從 標頭複製實例資料，以取代現有的實例資料。

ImqBinary messageToken () const;
傳回 **訊息記號**。

ImqBoolean setMessageToken(const ImqBinary & 記號);
設定 **訊息記號**。 *token* 的資料長度必須是零或 MQ_MSG_TOKEN_LENGTH。 如果成功，它會傳回 TRUE。

void setMessageToken (const MQBYTE16 *token* = 0);
設定 **訊息記號**。 *token* 可以是零，這與指定 MQMTOK_NONE 相同。 如果 *token* 不是零，則必須解決二進位資料的 MQ_MSG_TOKEN_LENGTH 位元組。

使用預定值 (例如 MQMTOK_NONE) 時，您可能需要進行強制轉型以確保簽章相符; 例如 (MQBYTE *) MQMTOK_NONE。

ImqString serviceName () const;
傳回 **服務名稱**，包括尾端空白。

void setServiceName (const char * *name*);
設定 **服務名稱**。

ImqString serviceStep () const;
傳回 **服務步驟**，包括尾端空白。

void setServiceStep (const char * *step*);
設定 **服務步驟**。

物件資料 (受保護)

MQWIH omqwih
MQWIH 資料結構。

原因碼

- MQRC_BINARY_DATA_LENGTH_ERROR

Java 程式庫的 IBM WebSphere MQ 類別

Java 程式庫的 IBM WebSphere MQ 類別位置會因平台而異。 當您啟動應用程式時，請指定此位置。

若要指定「Java 原生介面 (JNI)」程式庫的位置，請使用具有下列格式的 **java** 指令來啟動應用程式:

```
java -Djava.library.path=library_path application_name
```

其中 *library_path* 是適用於 Java 程式庫 (包括 JNI 程式庫) 的 WebSphere MQ 類別的路徑。 第 1278 頁的表 615 顯示每一個平台的 WebSphere MQ 類別 (適用於 Java 程式庫) 的位置。

表 *615:* 每一個平台之 *Java* 程式庫的 *WebSphere MQ* 類別位置

平台	**包含 Java 程式庫的 WebSphere MQ 類別的目錄**
AIX	*MQ_INSTALLATION_PATH*/java/lib (32 位元程式庫) *MQ_INSTALLATION_PATH*/java/lib64 (64 位元程式庫)
HP-UX Linux (POWER， x86-64 和 zSeries s390x 平台) Solaris (x86-64 及 SPARC 平台)	*MQ_INSTALLATION_PATH*/java/lib (32 位元程式庫) *MQ_INSTALLATION_PATH*/java/lib64 (64 位元程式庫)
Linux (x86 平台)	*MQ_INSTALLATION_PATH*/java/lib
Windows	*MQ_INSTALLATION_PATH*\Java\lib (32 位元程式庫) *MQ_INSTALLATION_PATH*\Java\lib64 (64 位元程式庫)
MQ_INSTALLATION_PATH 代表 WebSphere MQ 安裝所在的高階目錄。	

註:

1. 在 AIX、 HP-UX、 Linux (Power 平台) 或 Solaris 上，請使用 32 位元程式庫或 64 位元程式庫。 只有在 64 位元平台上的 64 位元 Java 虛擬機器 (JVM) 中執行應用程式時，才使用 64 位元程式庫。 否則，請使用 32 位元程式庫。
2. 在 Windows 上，您可以使用 PATH 環境變數來指定適用於 Java 程式庫的 WebSphere MQ 類別的位置，而不是在 **java** 指令上指定它們的位置。
3. 若要在 IBM i 上以連結模式使用 WebSphere MQ for Java 類別，請確定程式庫清單中有程式庫 QMQMJAVA。

相關工作

使用適用於 Java 的 WebSphere MQ 類別

IBM WebSphere MQ classes for JMS 物件的內容

IBM WebSphere MQ classes for JMS 中的所有物件都具有內容。 不同的內容會套用至不同的物件類型。 不同的內容具有不同的容許值，且符號內容值在管理工具與程式程式碼之間不同。

IBM WebSphere MQ classes for JMS 提供使用 WebSphere MQ JMS 管理工具 WebSphere MQ 探險家或在應用程式中設定及查詢物件內容的機能。 許多內容僅與物件類型的特定子集相關。

如需如何使用 WebSphere MQ JMS 管理工具的相關資訊，請參閱 使用 WebSphere MQ JMS 管理工具 。

第 1279 頁的表 616 提供每一個內容的簡要說明，並針對每一個內容顯示它套用至哪些物件類型。 物件類型是使用關鍵字來識別; 如需這些物件類型的說明，請參閱 JMS 物件類型 。

數字是指表格結尾的附註。 另請參閱 第 1326 頁的『JMS 物件的 WebSphere MQ 類別內容之間的相依關係』。

內容由下列格式的名稱/值配對組成:

```
PROPERTY_NAME(property_value)
```

本節中的主題列出每一個內容、內容名稱及簡要說明，並顯示管理工具中使用的有效內容值。 以及設定方法，用於在應用程式中設定內容的值。 這些主題也會顯示每一個內容的有效內容值，以及工具中使用的符號內容值與其可程式化對等項目之間的對映。

內容名稱不區分大小寫，且僅限於這些主題中顯示的可辨識名稱集。

表 *616:* 內容名稱及適用物件類型

內容	簡短形式	物件類型							
		CF	QCF	TCF	Q	T	XACF	XAQCF	XATCF
第 1282 頁的『APPLICATIONNAME』	APPNAME	Y	Y	Y			Y	Y	Y
第 1282 頁的『ASYNCEXCEPTION』	AEX	Y	Y	Y			Y	Y	Y
第 1283 頁的『BROKERCCDURSUBQ』[1]	CCDSUB					Y			
第 1284 頁的『BROKERCCSUBQ』[1]	CCSUB	Y		Y			Y		Y
第 1284 頁的『BROKERCONQ』[1]	BCON	Y		Y			Y		Y
第 1285 頁的『BROKERDURSUBQ』[1]	BDSUB					Y			
第 1285 頁的『BROKERPUBQ』[1]	BPUB	Y		Y		Y	Y		Y
第 1286 頁的『BROKERPUBQMGR』[1]	BPQM					Y			
第 1286 頁的『BROKERQMGR』[1]	BQM	Y		Y			Y		Y
第 1286 頁的『BROKERSUBQ』[1]	BSUB	Y		Y			Y		Y
第 1287 頁的『BROKERVER』[1]	BVER	Y[2]		Y[2]		Y	Y		Y
第 1287 頁的『CCDTURL』[3]	CCDT	Y	Y	Y			Y	Y	Y
第 1288 頁的『CCSID』	CCS	Y	Y	Y	Y	Y	Y	Y	Y
第 1288 頁的『CHANNEL』[3]	CHAN	Y	Y	Y			Y	Y	Y
第 1289 頁的『CLEANUP』[1]	CL	Y		Y			Y		Y
第 1289 頁的『CLEANUPINT』[1]	CLINT	Y		Y			Y		Y
第 1290 頁的『ConnectionNameList』	CNLIST	Y	Y	Y					
第 1290 頁的『CLIENTRECONNECTOPTIONS』	CROPT	Y	Y	Y					
第 1291 頁的『CLIENTRECONNECTTIMEOUT』	CRT	Y	Y	Y					
第 1291 頁的『CLIENTID』	CID	Y[2]	Y	Y[2]			Y	Y	Y
第 1292 頁的『CLONESUPP』	CLS	Y		Y			Y		Y
第 1292 頁的『COMPHDR』	HC	Y		Y			Y		Y
第 1293 頁的『COMPMSG』	MC	Y	Y	Y			Y	Y	Y
第 1293 頁的『CONNOPT』	CNOPT	Y	Y	Y			Y	Y	Y
第 1294 頁的『CONNTAG』	CNTAG	Y	Y	Y			Y	Y	Y
第 1294 頁的『說明』	降冪	Y[2]	Y	Y[2]	Y	Y	Y	Y	Y
第 1295 頁的『DIRECTAUTH』	DAUTH	Y[2]		Y[2]					

表 616: 內容名稱及適用物件類型 (繼續)

內容	簡短形式	物件類型							
		CF	QCF	TCF	Q	T	XACF	XAQCF	XATCF
第 1295 頁的『ENCODING』	ENC				Y	Y			
第 1296 頁的『EXPIRY』	EXP				Y	Y			
第 1297 頁的『FAILIFQUIESCE』	FIQ	Y	Y	Y	Y	Y	Y	Y	Y
第 1297 頁的『HOSTNAME』	HOST	Y[2]	Y	Y[2]			Y	Y	Y
第 1298 頁的『LOCALADDRESS』	LA	Y[2]	Y	Y[2]			Y	Y	Y
第 1299 頁的『MAPNAMESTYLE』	MNST	Y	Y	Y			Y	Y	Y
第 1299 頁的『MAXBUFFSIZE』	MBSZ	Y[2]		Y[2]					
第 1299 頁的『MDREAD』	MDR				Y	Y			
第 1300 頁的『MDWRITE』	MDW				Y	Y			
第 1300 頁的『MDMSGCTX』	MDCTX				Y	Y			
第 1301 頁的『MSGBATCHSZ』[1]	MBS	Y	Y	Y			Y	Y	Y
第 1301 頁的『MSGBODY』	MBODY				Y	Y			
第 1302 頁的『MSGRETENTION』	MRET	Y	Y				Y	Y	
第 1303 頁的『MSGSELECTION』[1]	MSEL	Y		Y			Y		Y
第 1303 頁的『MULTICAST』	MCAST	Y[2]		Y[2]		Y			
第 1304 頁的『OPTIMISTICPUBLICATION』[1]	OPTPUB	Y		Y					
第 1304 頁的『OUTCOMENOTIFICATION』[1]	NOTIFY	Y		Y					
第 1305 頁的『PERSISTENCE』	PER				Y	Y			
第 1305 頁的『POLLINGINT』[1]	PINT	Y	Y	Y			Y	Y	Y
第 1306 頁的『PORT』	PORT	Y[2]	Y	Y[2]			Y	Y	Y
第 1306 頁的『PRIORITY』	PRI				Y	Y			
第 1307 頁的『PROCESSDURATION』[1]	PROCDUR	Y		Y					
第 1307 頁的『PROVIDERVERSION』	PVER	Y	Y	Y			Y	Y	Y
第 1308 頁的『PROXYHOSTNAME』	PHOST	Y[2]		Y[2]					
第 1309 頁的『PROXYPORT』	PPORT	Y[2]		Y[2]					
第 1309 頁的『PUBACKINT』[1]	PAI	Y		Y			Y		Y
第 1310 頁的『PUTASYNCALLOWED』	PAALD				Y	Y			
第 1310 頁的『QMANAGER』	QMGR	Y	Y	Y	Y		Y	Y	Y
第 1311 頁的『佇列』	QU				Y				
第 1311 頁的『READAHEADALLOWED』	RAALD				Y	Y			

表 616: 內容名稱及適用物件類型 (繼續)

內容	簡短形式	物件類型							
		CF	QCF	TCF	Q	T	XACF	XAQCF	XATCF
第 1312 頁的『READAHEADCLOSEPOLICY』	RACP				Y	Y			
第 1312 頁的『RECEIVECCSID』	RCCS				Y	Y			
第 1313 頁的『RECEIVECONVERSION』	RCNV				Y	Y			
第 1313 頁的『RECEIVEISOLATION』[1]	RCVISOL	Y		Y					
第 1313 頁的『RECEXIT』	RCX	Y	Y	Y			Y	Y	Y
第 1314 頁的『RECEXITINIT』	RCXI	Y	Y	Y			Y	Y	Y
第 1314 頁的『REPLYTOSTYLE』	RTOST				Y	Y			
第 1315 頁的『RESCANINT』[1]	RINT	Y	Y				Y	Y	
第 1315 頁的『SECEXIT』	SCX	Y	Y	Y			Y	Y	Y
第 1316 頁的『SECEXITINIT』	SCXI	Y	Y	Y			Y	Y	Y
第 1316 頁的『SENDCHECKCOUNT』	SCC	Y	Y	Y			Y	Y	Y
第 1317 頁的『SENDEXIT』	SDX	Y	Y	Y			Y	Y	Y
第 1317 頁的『SENDEXITINIT』	SDXI	Y	Y	Y			Y	Y	Y
第 1318 頁的『SHARECONVALLOWED』	SCALD	Y	Y	Y			Y	Y	Y
第 1318 頁的『SPARSESUBS』[1]	SSUBS	Y		Y					
第 1319 頁的『SSLCIPHERSUITE』	SCPHS	Y	Y	Y			Y	Y	Y
第 1319 頁的『SSLCRL』	SCRL	Y	Y	Y			Y	Y	Y
第 1320 頁的『SSLFIPSREQUIRED』	SFIPS	Y	Y	Y			Y	Y	Y
第 1320 頁的『SSLPEERNAME』	SPEER	Y	Y	Y			Y	Y	Y
第 1320 頁的『SSLRESETCOUNT』	SRC	Y	Y	Y			Y	Y	Y
第 1321 頁的『STATREFRESHINT』[1]	SRI	Y		Y			Y		Y
第 1321 頁的『SUBSTORE』[1]	不銹鋼	Y		Y			Y		Y
第 1322 頁的『SYNCPOINTALLGETS』	SPAG	Y	Y	Y			Y	Y	Y
第 1322 頁的『TARGCLIENT』	TC				Y	Y			
第 1323 頁的『TARGCLIENTMATCHING』	TCM	Y	Y				Y	Y	
第 1323 頁的『TEMPMODEL』	TM	Y	Y				Y	Y	
第 1324 頁的『TEMPQPREFIX』	TQP	Y	Y				Y	Y	
第 1324 頁的『TEMPTOPICPREFIX』	TTP	Y		Y			Y		Y
第 1325 頁的『TOPIC』	TOP					Y			

表 616: 內容名稱及適用物件類型 (繼續)

內容	簡短形式	物件類型							
		CF	QCF	TCF	Q	T	XACF	XAQCF	XATCF
第 1325 頁的『TRANSPORT』	TRAN	Y[2]	Y	Y[2]			Y	Y	Y
第 1326 頁的『WILDCARDFORMAT』	WCFMT	Y		Y			Y		Y

註:

1. 此內容可以與適用於 JMS 的 WebSphere MQ 類別 7.0 版搭配使用，但對於連接至 7.0 版佇列管理程式的應用程式沒有任何作用，除非 Connection Factory 的 PROVIDERVERSION 內容設為小於 7 的版本號碼。
2. 當使用分配管理系統的即時連線時， ConnectionFactory 或 TopicConnectionFactory 物件只支援 BROKERVER、CLIENTID、DESCRIPTION、DIRECTAUTH、HOSTNAME、LOCALADDRESS、MAXBUFFSIZE、MULTICAST、PORT、PROXYHOSTNAME、PROXYPORT 及 TRANSPORT 內容。
3. 物件的 CCDTURL 和 CHANNEL 內容不能同時設定。

APPLICATIONNAME

應用程式可以設定名稱來識別其與佇列管理程式的連線。 此應用程式名稱由 **DISPLAY CONN MQSC/PCF** 指令 (其中欄位稱為 **APPLTAG**) 或在「 IBM WebSphere MQ 瀏覽器 **應用程式連線** 」顯示畫面中 (其中欄位稱為 **App name**) 顯示。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :APPLICATIONNAME

JMS 管理工具簡稱 :APPNAME

以編程方式存取

setter/getter

- MQConnectionFactory.setApp 名稱 ()
- MQConnectionFactory.getApp 名稱 ()

值

長度不超過 28 個字元的任何有效字串。 必要的話，會移除前導套件名稱，以調整較長的名稱以符合需要。 例如，如果呼叫端類別是 `com.example.MainApp`，則會使用完整名稱，但如果呼叫端類別是 `com.example.dictionaryAndThesaurus.multilingual.mainApp`，則會使用名稱 `multilingual.mainApp` ，因為它是最適合可用長度的類別名稱與最右邊套件名稱的最長組合。

如果類別名稱本身長度超過 28 個字元，則會截斷以適合。 例如，`com.example.mainApplicationForSecondTestCase` 會變成 `mainApplicationForSecondTest`。

ASYNCEXCEPTION

此內容決定適用於 JMS 的 WebSphere MQ 類別是否僅在連線中斷或 JMS API 呼叫非同步發生任何異常狀況時才通知 ExceptionListener 。 這適用於從這個已登錄 ExceptionListener 的 ConnectionFactory 建立的所有連線。

適用的物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :AXX_ENCODE_CASE_ONE syncexception

JMS 管理工具簡稱 :AEX

以編程方式存取

setter/Getter

- MQConnectionFactory.setAsync 異常狀況 ()
- MQConnectionFactory.getAsync 異常狀況 ()

值

ASYNC_EXCEPTIONS_ALL

非同步偵測到的任何異常狀況、同步 API 呼叫的範圍之外，以及所有連線中斷異常狀況都會傳送至 ExceptionListener。

環境	值
JMS 管理工具	ALL
程式化	WMQCONSTANTS.ASYNC_EXCEPTIONS_ALL = -1
WebSphere MQ 探險家	全部

ASYNC_EXCEPTIONS_CONNECTIONBROKEN

只有指出連線中斷的異常狀況才會傳送至 ExceptionListener。 在非同步處理期間發生的任何其他異常狀況都不會向 ExceptionListener 報告，因此應用程式不會收到這些異常狀況的通知。 V 7.5.0.8 這是 IBM WebSphere MQ Version 7.5.0Fix Pack 8 中的預設值 (請參閱 JMS: 7.5 版中的異常狀況接聽器變更)。

環境	值
JMS 管理工具	CONNECTIONBROKEN
程式化	WMQCONSTANTS.ASYNC_EXCEPTIONS_CONNECTIONBROKEN = 1
WebSphere MQ 探險家	連線已中斷

已定義下列其他常數: V 7.5.0.8

- 從 Version 7.5.0, Fix Pack 8: WMQCONSTANTS.ASYNC_EXCEPTIONS_DEFAULT = ASYNC_EXCEPTIONS_CONNECTIONBROKEN
- 在 Version 7.5.0, Fix Pack 8 之前: WMQCONSTANTS.ASYNC_EXCEPTIONS_DEFAULT = ASYNC_EXCEPTIONS_ALL

相關概念

適用於 JMS 的 WebSphere MQ 類別中的異常狀況

BROKERCCDURSUBQ

針對 ConnectionConsumer 從中擷取可延續訂閱訊息的佇列名稱。

適用的物件

主題

JMS 管理工具完整名稱 :XX_ENCODE_CASE_ONE brokerccdursubq

JMS 管理工具簡稱 :CCDSUB

以編程方式存取

setter/getter

- MQTopic.setBrokerCCDurSubQueue()
- MQTopic.getBrokerCCDurSubQueue()

值

SYSTEM.JMS.D.CC.SUBSCRIBER.QUEUE
這是預設值。

任何有效字串

BROKERCCSUBQ

針對 ConnectionConsumer 從中擷取不可延續訂閱訊息的佇列名稱。

適用的物件

ConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: BROKERCCSUBQ

JMS 管理工具簡稱 :CCSUB

以編程方式存取

setter/getter

- MQConnectionFactory.setBrokerCCSubQueue()
- MQConnectionFactory.getBrokerCCSubQueue()

值

SYSTEM.JMS.ND.CC.SUBSCRIBER.QUEUE
這是預設值。

任何有效字串

BROKERCONQ

分配管理系統的控制佇列名稱。

適用物件

ConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :XX_ENCODE_CASE_ONE brokerconq

JMS 管理工具簡稱 :BCON

以編程方式存取

setter/getter

- MQConnectionFactory.setBrokerControlQueue()
- MQConnectionFactory.getBrokerControlQueue()

值

SYSTEM.BROKER.CONTROL.QUEUE
這是預設值。

任何有效字串

BROKERDURSUBQ

當在 WebSphere MQ 傳訊提供者移轉模式中使用適用於 JMS 的 WebSphere MQ 類別時，這個內容指定從中擷取可延續訂閱訊息的佇列名稱。

適用的物件

主題

JMS 管理工具完整名稱 :XX_ENCODE_CASE_ONE brokerdursubq

JMS 管理工具簡稱 :BDSUB

以編程方式存取

setter/getter

- MQTopic.setBrokerDurSubQueue()
- MQTopic.getBrokerDurSubQueue()

值

SYSTEM.JMS.D.SUBSCRIBER.QUEUE
這是預設值。

任何有效字串
從 SYSTEM.JMS.D

相關概念
選取 WebSphere MQ 傳訊提供者模式的規則

BROKERPUBQ

傳送已發佈訊息的佇列名稱 (串流佇列)。

適用物件

ConnectionFactory、 TopicConnectionFactory、Topic、 XAConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: BROKERPUBQ

JMS 管理工具簡稱 :BPUB

以編程方式存取

setter/getter

- MQConnectionFactory.setBrokerPubQueue
- MQConnectionFactory.getBrokerPubQueue

值

SYSTEM.BROKER.DEFAULT.STREAM
這是預設值。

任何有效字串

BROKERPUBQMGR

擁有佇列的佇列管理程式名稱，在此佇列中傳送針對主題發佈的訊息。

適用物件

主題

JMS 管理工具完整名稱 :XX_ENCODE_CASE_ONE brokerpubqmgr

JMS 管理工具簡稱 :BPQM

以編程方式存取

setter/getter

- MQTopic.setBrokerPubQueueManager()
- MQTopic.getBrokerPubQueueManager()

值

空值
這是預設值。

任何有效字串

BROKERQMGR

分配管理系統執行所在的佇列管理程式名稱。

適用物件

ConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :XX_ENCODE_CASE_ONE brokerqmgr

JMS 管理工具簡稱 :BQM

以編程方式存取

Setter/getter

- MQConnectionFactory.setBrokerQueueManager()
- MQConnectionFactory.getBrokerQueueManager()

值

空值
這是預設值。

任何有效字串

BROKERSUBQ

在 WebSphere MQ 傳訊提供者移轉模式中使用適用於 JMS 的 WebSphere MQ 類別時，此內容指定從中擷取不可延續訂閱訊息的佇列名稱。

適用物件

ConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :BROKERSUBQ

JMS 管理工具簡稱 :BSUB

以編程方式存取

setter/getter

- MQConnectionFactory.setBrokerSubQueue()
- MQConnectionFactory.getBrokerSubQueue()

值

SYSTEM.JMS.ND.SUBSCRIBER.QUEUE
這是預設值。

任何有效字串
從 SYSTEM.JMS.ND

相關概念
選取 WebSphere MQ 傳訊提供者模式的規則

BROKERVER

正在使用的分配管理系統版本。

適用物件

ConnectionFactory、 TopicConnectionFactory、Topic、 XAConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: BROKERVER

JMS 管理工具簡稱 :BVER

以編程方式存取

setter/getter

- MQConnectionFactory.setBroker 版本 ()
- MQConnectionFactory.getBroker 版本 ()

值

V1
使用 WebSphere MQ 發佈/訂閱分配管理系統，或以相容模式使用 WebSphere MQ Integrator、 WebSphere Event Broker、 WebSphere Business Integration Event Broker 或 WebSphere Business Integration Message Broker 的分配管理系統。 如果 TRANSPORT 設為 BIND 或 CLIENT ，則這是預設值。

V2
若要使用 WebSphere MQ Integrator 的分配管理系統，請以原生模式 WebSphere Event Broker、 WebSphere Business Integration Event Broker 或 WebSphere Business Integration Message Broker。 如果 TRANSPORT 設為 DIRECT 或 DIRECTHTTP ，則這是預設值。

未指定
在分配管理系統從 V6 移轉至 V7 之後，請設定此內容，以便不再使用 RFH2 標頭。 移轉之後，此內容不再相關。

CCDTURL

統一資源定址器（URL）識別包含用戶端通道定義表的檔案之名稱和位置，並指定如何存取該檔案。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :CCDTURL

JMS 管理工具簡稱 :CCDT

以編程方式存取

setter/getter

- MQConnectionFactory.setCCDTURL()
- MQConnectionFactory.getCCDTURL()

值

空值
這是預設值。

統一資源定址器 (URL)

CCSID

用於連線或目的地的編碼字集 ID。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、Queue、Topic、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :CCSID

JMS 管理工具簡稱 :CCS

以編程方式存取

setter/getter

- MQConnectionFactory.setCCSID()
- MQConnectionFactory.getCCSID()

值

819
這是 Connection Factory 的預設值。

1208
這是目的地的預設值。

任何正整數

CHANNEL

正在使用的用戶端連線通道名稱。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: CHANNEL

JMS 管理工具簡稱 :CHAN

以編程方式存取

setter/getter

- MQConnectionFactory.setChannel()
- MQConnectionFactory.getChannel()

值

SYSTEM.DEF.SVRCONN
這是預設值。

任何有效字串

CLEANUP

BROKER 或 MIGRATE 訂閱儲存庫的清除層次。

適用物件

ConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: CLEANUP

JMS 管理工具簡稱 :CL

以編程方式存取

setter/getter

- MQConnectionFactory.setCleanup 層次 ()
- MQConnectionFactory.getCleanup 層次 ()

值

安全
使用安全清理。 這是預設值。

ASPROP
根據 Java 指令行上的內容集，使用 safe、strong 或 no cleanup。

無
不使用清理。

進階安全
使用強型態清理。

CLEANUPINT

發佈/訂閱清理公用程式的背景執行間隔 (毫秒)。

適用物件

ConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :CLEANUPINT

JMS 管理工具簡稱 :CLINT

以編程方式存取

setter/getter

- MQConnectionFactory.setCleanupInterval ()
- MQConnectionFactory.getCleanup 間隔 ()

值

3600000
這是預設值。

任何正整數

ConnectionNameList

TCP/IP 連線名稱清單。 會依序嘗試清單，每次重試重新連線一次。

適用物件

ConnectionFactory， QueueConnectionFactory， TopicConnectionFactory

JMS 管理工具完整名稱 :CONNECTIONNAMELIST

JMS 管理工具簡稱 :CNLIST

以編程方式存取

setter/getter

- MQConnectionFactory.setconnectionNameList ()
- MQConnectionFactory.getconnectionNameList ()

值

HOSTNAME (PORT) 的逗點區隔清單。 HOSTNAME 可以是 DNS 名稱或 IP 位址。

PORT 預設為 1414。

CLIENTRECONNECTOPTIONS

控管重新連線的選項。

適用物件

ConnectionFactory， QueueConnectionFactory， TopicConnectionFactory

JMS 管理工具完整名稱 :CLIENTRECONNECTOPTIONS

JMS 管理工具簡稱 :CROPT

以編程方式存取

Setter/getter

- MQConnectionFactory.setClientReconnectOptions()
- MQConnectionFactory.getClientReconnectOptions()

值

QMGR

應用程式可以重新連接，但只能重新連接原來連接的佇列管理程式。

如果應用程式可以重新連接，但 JMS 應用程式的 WebSphere MQ 類別與其第一次建立連線的佇列管理程式之間存在親緣性，請使用此值。

如果您想要應用程式自動重新連接至高可用性佇列管理程式的待命實例，請選擇此值。

若要以程式化方式使用此值，請使用常數 WMQConstants.WMQ_CLIENT_RECONNECT_Q_MGR。

ANY

應用程式可以重新連接到任何的佇列管理程式。

只有在 JMS 應用程式的 WebSphere MQ 類別與其起始建立連線的佇列管理程式之間沒有親緣性時，才使用重新連接選項。

若要從程式使用此值，請使用常數 WMQConstants.WMQ_CLIENT_RECONNECT。

已停用

應用程式將不會重新連接。

若要以程式化方式使用此值，請使用常數 WMQConstants.WMQ_CLIENT_RECONNECT_DISABLED。

ASDEF

應用程式是否會自動重新連接，取決於 WebSphere MQ 通道屬性 DefReconnect 的值。

若要從程式使用此值，請使用常數 WMQConstants.WMQ_CLIENT_RECONNECT_AS_DEF。

CLIENTRECONNECTTIMEOUT

停止重新連線重試之前的時間。

適用物件

ConnectionFactory， QueueConnectionFactory， TopicConnectionFactory

JMS 管理工具完整名稱 :CLIENTRECONNECTTIMEOUT

JMS 管理工具簡稱 :CRT

以編程方式存取

setter/getter

- MQConnectionFactory.setClientReconnectTimeout()
- MQConnectionFactory.setClientReconnectTimeout()

值

間隔 (以秒為單位)。 預設 1800 (30 分鐘)。

CLIENTID

用戶端 ID 是可延續訂閱的應用程式連線的唯一識別方式。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :CLIENTID

JMS 管理工具簡稱 :CID

以編程方式存取

setter/getter

- MQConnectionFactory.setClientId ()
- MQConnectionFactory.getClientId ()

值

空值
這是預設值。

任何有效字串

CLONESUPP

相同可延續主題訂閱者的兩個以上實例是否可以同步執行。

適用物件

ConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :CLONESUPP

JMS 管理工具簡稱 :CLS

以編程方式存取

setter/getter

- MQConnectionFactory.setClone 支援 ()
- MQConnectionFactory.getClone 支援 ()

值

已停用
一次只能執行一個可延續主題訂閱者的實例。 這是預設值。

ENABLED
相同可延續主題訂閱者的兩個以上實例可以同步執行，但每一個實例都必須在個別 Java 虛擬機器 (JVM) 中執行。

COMPHDR

可用來壓縮連線上標頭資料的技術清單。

適用物件

ConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :COMPHDR

JMS 管理工具簡稱 :HC

以編程方式存取

setter/getter

- MQConnectionFactory.setHdrCompList()
- MQConnectionFactory.getHdrCompList()

值

無
這是預設值。

系統
執行 RLE 訊息標頭壓縮。

COMPMSG

可用來壓縮連線上訊息資料的技術清單。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :COMPMSG

JMS 管理工具簡稱 :MC

以編程方式存取

setter/getter

- MQConnectionFactory.setMsgCompList()
- MQConnectionFactory.getMsgCompList()

值

無
這是預設值。

下列一或多個值的清單，以空白字元區隔:
RLE ZLIBFAST ZLIBHIGH

CONNOPT

控制使用連結傳輸之 JMS 應用程式的 WebSphere MQ 類別如何連接至佇列管理程式。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :CONNOPT

JMS 管理工具簡稱 :CNOPT

以編程方式存取

setter/getter

- MQConnectionFactory.setMQConnection 選項 ()
- MQConnectionFactory.getMQConnection 選項 ()

值

標準
應用程式與佇列管理程式之間的連結本質取決於佇列管理程式的 *DefaultBind* 類型 屬性值， STANDARD 值會對映至 WebSphere MQ *ConnectOption* MQCN_STANDARD_BINDING

共用
應用程式及本端佇列管理程式代理程式以個別執行單元執行，但共用部分資源。 此值對映至 WebSphere MQ *ConnectOption* MQCNO_SHARED_BINDING。

隔離
應用程式及本端佇列管理程式代理程式在個別執行單元中執行，且不共用任何資源。 隔離值對映至 WebSphere MQ *ConnectOption* MQCNO_ISOLATED_BINDING。

Fastpath
應用程式和本端佇列管理程式代理程式在相同的執行單元中執行。 此值對映至 WebSphere MQ *ConnectOption* MQCNO_FASTPATH_BINDING。

SERIALQM
應用程式要求在佇列管理程式的範圍內專用連線標籤。 此值對映至 WebSphere MQ *ConnectOption* MQCNO_SERIALIZE_CONN_TAG_Q_MGR。

SERIALQSG
應用程式要求在佇列管理程式所屬的佇列共用群組範圍內，專用連線標籤。 SERIALQSG 值對映至 WebSphere MQ *ConnectOption* MQCNO_SERIALIZE_CONN_TAG_QSG。

RESTRICTQM
應用程式要求共用連線標籤，但在佇列管理程式的範圍內共用連線標籤有一些限制。 此值對映至 WebSphere MQ *ConnectOption* MQCNO_RESTRICT_CONN_TAG_Q_MGR。

RESTRICTQSG
應用程式要求共用連線標籤，但在佇列管理程式所屬的佇列共用群組範圍內共用連線標籤有一些限制。 此值對映至 WebSphere MQ *ConnectOption* MQCNO_RESTRICT_CONN_TAG_QSG。

如需 WebSphere MQ 連線選項的進一步資訊，請參閱 使用 MQCONNX 呼叫連接至佇列管理程式

CONNTAG

當應用程式連接至佇列管理程式時，佇列管理程式與應用程式在工作單元內所更新的資源相關聯的標籤。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :CONNTAG

JMS 管理工具簡稱 :CNTAG

以編程方式存取

setter/getter

- MQConnectionFactory.setConn 標籤 ()
- MQConnectionFactory.getConn 標籤 ()

值

128 個元素的位元組陣列，其中每一個元素都是 0
這是預設值。

任何字串
如果值超過 128 個位元組，則會截斷該值。

說明

儲存物件的說明。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、Queue、Topic、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :XX_ENCODE_CASE_ONE description

JMS 管理工具簡稱 :DESC

以編程方式存取

setter/getter

- MQConnectionFactory.setDescription()
- MQConnectionFactory.getDescription()

值

空值
這是預設值。

任何有效字串
`

DIRECTAUTH

是否在與分配管理系統的即時連線上使用 SSL 鑑別。

適用物件

ConnectionFactory， TopicConnectionFactory

JMS 管理工具完整名稱 :DIRECTTAUTH

JMS 管理工具簡稱 :DAUTH

以編程方式存取

setter/getter

- MQConnectionFactory.setDirectAuth ()
- MQConnectionFactory.getDirectAuth ()

值

基本
無鑑別、使用者名稱鑑別或密碼鑑別。 這是預設值。

憑證
公開金鑰憑證鑑別。

ENCODING

當訊息傳送至這個目的地時，如何代表訊息內文中的數值資料。 此內容指定二進位整數、聚集十進位整數及浮點數字的表示法。

適用物件

佇列，主題

JMS 管理工具完整名稱 :ENCODING

JMS 管理工具簡稱 :ENC

以編程方式存取

setter/getter

- MQDestination.setEncoding()
- MQDestination.getEncoding()

值

ENCODING 內容

ENCODING 內容可以採用的有效值是從三個子內容建構而來:

整數編碼
正常或反向

小數編碼
正常或反向

浮點數編碼
IEEE 正常、IEEE 反轉或 z/OS

ENCODING 內容以三個字元的字串表示，語法如下:

```
{N|R}{N|R}{N|R|3}
```

在此字串中:

- N 表示正常
- R 表示已反轉
- 3 表示 z/OS
- 第一個字元代表 整數編碼
- 第二個字元代表 十進位編碼
- 第三個字元代表 浮點數編碼

這會為 ENCODING 內容提供一組 12 個可能的值。

另外還有一個值，即字串 NATIVE，可針對 Java 平台設定適當的編碼值。

下列範例顯示 ENCODING 的有效組合:

```
     ENCODING(NNR)
     ENCODING(NATIVE)
     ENCODING(RR3)
```

EXPIRY

目的地訊息到期之前的時間。

適用物件

佇列，主題

JMS 管理工具完整名稱 :EXPIRY

JMS 管理工具簡稱 :EXP

以編程方式存取

setter/getter

- MQDestination.setExpiry()
- MQDestination.getExpiry()

值

應用程式
期限可以由 JMS 應用程式定義。 這是預設值。

UNLIM
未發生期限。

0
未發生期限。

任何正整數，代表期限 (毫秒)。

FAILIFQUIESCE

如果佇列管理程式處於靜止狀態，或應用程式正在使用 CLIENT 傳輸連接至佇列管理程式，且應用程式使用的通道已進入靜止狀態 (例如，使用 **STOP CHANNEL** 或 **STOP CHANNEL MODE(QUIESCE)** MQSC 指令) ，則此內容會決定對特定方法的呼叫是否失敗。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、Queue、Topic、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :FAILIFQUIESCE

JMS 管理工具簡稱 :FIQ

以編程方式存取

setter/getter

- MQConnectionFactory.setFailIfQuiesce()
- MQConnectionFactory.getFailIfQuiesce()

值

YES
如果佇列管理程式處於靜止狀態，或用來連接至佇列管理程式的通道處於靜止狀態，則對某些方法的呼叫會失敗。 如果應用程式偵測到其中任一狀況，則應用程式可以完成其立即作業並關閉連線，讓佇列管理程式或通道實例停止。 這是預設值。

NO
沒有方法呼叫失敗，因為佇列管理程式或用來連接佇列管理程式的通道處於靜止狀態。 如果您指定此值，則應用程式無法偵測到佇列管理程式或通道正在靜止。 應用程式可能會繼續對佇列管理程式執行作業，因此阻止佇列管理程式停止。

HOSTNAME

若為佇列管理程式的連線，則為執行佇列管理程式之系統的主機名稱或 IP 位址，若為分配管理系統的即時連線，則為執行分配管理系統之系統的主機名稱或 IP 位址。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :HOSTNAME

JMS 管理工具簡稱 :HOST

以編程方式存取

setter/getter

- MQConnectionFactory.setHostName ()
- MQConnectionFactory.getHostName ()

值

本端主機
這是預設值。

任何有效字串

LOCALADDRESS

對於佇列管理程式的連線，此內容指定要使用的本端網路介面，或要使用的本端埠或本端埠範圍。 對於分配管理系統的即時連線，此內容僅在使用多重播送時相關，並指定要使用的本端網路介面。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :LOCALADDRESS

JMS 管理工具簡稱 :LA

以編程方式存取

setter/getter

- MQConnectionFactory.setLocalAddress ()
- MQConnectionFactory.getLocalAddress ()

值

"" (空字串)
這是預設值。

格式為 [ip-addr] [(low-port [， high-port])] 的字串
這裡是一些範例:

192.0.2.0

通道會在本端連結至位址 192.0.2.0 。

192.0.2.0(1000)

通道會在本端連結至位址 192.0.2.0 ， 並使用埠 1000。

192.0.2.0(1000,2000)

通道會在本端連結至位址 192.0.2.0 ， 並使用 1000 至 2000 範圍內的埠。

(1000)

通道會在本端連結至埠 1000。

(1000,2000)

通道在本端連結至 1000 至 2000 範圍內的埠。

您可以指定主機名稱而非 IP 位址。 對於與分配管理系統的即時連線，此內容僅在使用多重播送時相關，且內容的值不得包含埠號或埠號範圍。 在此情況下，內容的唯一有效值是空值、IP 位址或主機名稱。

MAPNAMESTYLE

容許 MapMessage 元素名稱使用相容性樣式。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :MAPNAMESTYLE

JMS 管理工具簡稱 :MNST

以編程方式存取

setter/getter

- MQConnectionFactory.setMapNameStyle()
- MQConnectionFactory.getMapNameStyle()

值

標準
將使用標準 com.ibm.jms.JMSMapMessage 元素命名格式。 這是預設值，容許使用不合法的 Java ID 作為元素名稱。

相容
將使用較舊的 com.ibm.jms.JMSMapMessage 元素命名格式。 只能使用合法的 Java ID 作為元素名稱。 只有在將對映訊息傳送至使用 5.3 版之前 IBM WebSphere MQ classes for JMS 版本的應用程式時，才需要這樣做。

MAXBUFFSIZE

等待應用程式處理時，可以儲存在內部訊息緩衝區中的已接收訊息數上限。 僅當 TRANSPORT 具有值 DIRECT 或 DIRECTHTTP 時，此內容才適用。

適用物件

ConnectionFactory， TopicConnectionFactory

JMS 管理工具完整名稱 :MAXBUFFSIZE

JMS 管理工具簡稱 :MBSZ

以編程方式存取

setter/getter

- MQConnectionFactory.setMaxBufferSize()
- MQConnectionFactory.getMaxBufferSize()

值

1000
這是預設值。

任何正整數

MDREAD

此內容決定 JMS 應用程式是否可以擷取 MQMD 欄位的值。

適用物件

JMS 管理工具完整名稱 :MDREAD

JMS 管理工具簡稱 :MDR

以編程方式存取

setter/getter

- MQDestination.setMQMDReadEnabled()
- MQDestination.getMQMDReadEnabled()

值

NO
傳送訊息時，不會更新已傳送訊息上的 JMS_IBM_MQMD* 內容，以反映 MQMD 中已更新的欄位值。 接收訊息時，接收的訊息上沒有任何可用的 JMS_IBM_MQMD* 內容，即使寄件者已設定部分或全部內容都一樣。 這是管理工具的預設值。

對於程式，請使用 False。

是
傳送訊息時，會更新已傳送訊息上的所有 JMS_IBM_MQMD* 內容 (包括寄件者未明確設定的內容) ，以反映 MQMD 中已更新的欄位值。 接收訊息時，所有 JMS_IBM_MQMD* 內容 (包括寄件者未明確設定的內容) 可用於接收的訊息。

對於程式，請使用 True。

MDWRITE

此內容決定 JMS 應用程式是否可以設定 MQMD 欄位的值。

適用物件

佇列，主題

JMS 管理工具完整名稱 :MDWRITE

JMS 管理工具簡稱 :MDR

以編程方式存取

setter/getter

- MQDestination.setMQMDWriteEnabled()
- MQDestination.getMQMDWriteEnabled()

值

NO
會忽略所有 JMS_IBM_MQMD* 內容，且其值不會複製到基礎 MQMD 結構。 這是管理工具的預設值。

對於程式，請使用 False。

YES
會處理 JMS_IBM_MQMD* 內容。 其值會複製到基礎 MQMD 結構。

對於程式，請使用 True。

MDMSGCTX

JMS 應用程式要設定的訊息環境定義層次。 應用程式必須以適當的環境定義權限執行，才能使此內容生效。

適用物件

JMS 管理工具完整名稱 :MDMSGCTX

JMS 管理工具簡稱 :MDCTX

以編程方式存取

setter/getter

- MQDestination.setMQMDMessageContext()
- MQDestination.getMQMDMessageContext()

值

預設值
MQOPEN API 呼叫和 MQPMO 結構未指定明確訊息環境定義選項。 這是管理工具的預設值。

若為程式，請使用 WMQ_MDCTX_DEFAULT。

SET_IDENTITY_CONTEXT
MQOPEN API 呼叫會指定訊息環境定義選項 MQOO_SET_IDENTITY_CONTEXT ， 而 MQPMO 結構會指定 MQPMO_SET_IDENTITY_CONTEXT。

若為程式，請使用 WMQ_MDCTX_SET_IDENTITY_CONTEXT。

環境 _ALL_CONTEXT
MQOPEN API 呼叫指定訊息環境定義選項 MQOO_SET_ALL_CONTEXT ， MQPMO 結構指定 MQPMO_SET_ALL_CONTEXT。

若為程式，請使用 WMQ_MDCTX_SET_ALL_CONTEXT。

MSGBATCHSZ

使用非同步訊息遞送時，從一個封包中的佇列取得的訊息數上限。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :MAXBUFFSIZE

JMS 管理工具簡稱 :MBSZ

以編程方式存取

setter/getter

- MQConnectionFactory.setMsgBatchSize()
- MQConnectionFactory.getMsgBatchSize()

值

10
這是預設值。

任何正整數

MSGBODY

決定 JMS 應用程式是否在訊息有效負載中存取 IBM WebSphere MQ 訊息的 MQRFH2 。

適用物件

佇列，主題

JMS 管理工具完整名稱 :WMQ_MESSAGE_Body

JMS 管理工具簡稱 :MBODY

以編程方式存取

setter/getter

- MQConnectionFactory.setMessageBodyStyle()
- MQConnectionFactory.getMessageBodyStyle()

值

未指定
傳送時，視 WMQ_TARGET_CLIENT 的值而定， IBM WebSphere MQ classes for JMS 不一定會產生並包含 MQRFH2 標頭。 接收時，充當值 JMS。

JMS
傳送時， IBM WebSphere MQ classes for JMS 會自動產生 MQRFH2 標頭，並將它包含在 WebSphere MQ 訊息中。

接收時， IBM WebSphere MQ classes for JMS 會根據 MQRFH2 (如果有的話) 中的值來設定 JMS 訊息內容; 它不會將 MQRFH2 呈現為 JMS 訊息內文的一部分。

MQ
傳送時， IBM WebSphere MQ classes for JMS 不會產生 MQRFH2。

接收時， IBM WebSphere MQ classes for JMS 會將 MQRFH2 呈現為 JMS 訊息內文的一部分。

MSGRETENTION

連線消費者是否在輸入佇列中保留未遞送的訊息。

適用物件

ConnectionFactory、 QueueConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、

JMS 管理工具完整名稱 :MSGRETENTION

JMS 管理工具簡稱 :MRET

以編程方式存取

setter/getter

- MQConnectionFactory.setMessage 保留 ()
- MQConnectionFactory.getMessage 保留 ()

值

是
未遞送的訊息仍留在輸入佇列中。 這是預設值。

否
未遞送的訊息會根據其處置選項來處理。

MSGSELECTION

決定訊息選擇是由適用於 JMS 的 WebSphere MQ 類別或分配管理系統來完成。 如果 TRANSPORT 具有值 DIRECT，則訊息選擇一律由分配管理系統完成，且會忽略 MSGSELECTION 的值。 當 BROKERVER 具有值 V1 時，不支援分配管理系統選取訊息。

適用物件

ConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :MSGSELECTION

JMS 管理工具簡稱 :MSEL

以編程方式存取

setter/getter

- MQConnectionFactory.setMessageSelection ()
- MQConnectionFactory.getMessageSelection ()

值

用戶端
訊息選擇由適用於 JMS 的 WebSphere MQ 類別完成。 這是預設值。

分配管理系統
訊息選擇由分配管理系統完成。

MULTICAST

在與分配管理系統的即時連線上啟用多重播送，並指定使用多重播送將訊息從分配管理系統遞送至訊息消費者的精確方式 (如果已啟用)。 此內容不會影響訊息產生者將訊息傳送至分配管理系統的方式。

適用物件

ConnectionFactory， TopicConnectionFactory，主題

JMS 管理工具完整名稱: 多重播送

JMS 管理工具簡稱 :MCAST

以編程方式存取

setter/getter

- MQConnectionFactory.setMulticast()
- MQConnectionFactory.getMulticast()

值

已停用
訊息不會使用多重播送傳輸遞送至訊息消費者。 這是 ConnectionFactory 和 TopicConnectionFactory 物件的預設值。

ASCF
根據與訊息消費者相關聯的 Connection Factory 的多重播送設定，將訊息遞送至訊息消費者。 建立訊息消費者時，會記下 Connection Factory 的多重播送設定。 此值僅對「主題」物件有效，且是「主題」物件的預設值。

ENABLED
如果在分配管理系統中配置主題以進行多重播送，則會使用多重播送傳輸將訊息遞送至訊息消費者。 如果主題配置為可靠的多重播送，則會使用可靠的服務品質。

可靠
如果主題在分配管理系統中配置為可靠的多重播送，則會使用具有可靠服務品質的多重播送傳輸，將訊息遞送至訊息消費者。 如果主題未配置為可靠多重播送，則無法建立主題的訊息消費者。

NOTR
如果在分配管理系統中配置主題進行多重播送，則會使用多重播送傳輸將訊息遞送至訊息消費者。 即使主題配置為可靠的多重播送，也不會使用可靠的服務品質。

OPTIMISTICPUBLICATION

此內容決定適用於 JMS 的 WebSphere MQ 類別是否立即將控制項傳回給已發佈訊息的發佈者，或者它是否只在完成與呼叫相關聯的所有處理且可以向發佈者報告結果之後才傳回控制項。

適用物件

ConnectionFactory， TopicConnectionFactory

JMS 管理工具完整名稱 :XX_ENCODE_CASE_ONE Optimisticpublication

JMS 管理工具簡稱 :OPTPUB

以編程方式存取

setter/getter

- MQConnectionFactory.setOptimisticPublication ()
- MQConnectionFactory.getOptimisticPublication ()

值

NO
當發佈者發佈訊息時，適用於 JMS 的 WebSphere MQ 類別在完成與呼叫相關聯的所有處理並可向發佈者報告結果之前不會將控制權傳回給發佈者。 這是預設值。

YES
當發佈者發佈訊息時， WebSphere 適用於 JMS 的 MQ 類別會立即將控制權交還給發佈者，然後它才會完成與呼叫相關聯的所有處理程序，並可將結果報告給發佈者。 只有在發佈者確定訊息時，適用於 JMS 的 WebSphere MQ 類別才會報告結果。

OUTCOMENOTIFICATION

此內容決定用於 JMS 的 WebSphere MQ 類別是否立即將控制傳回給剛確認或已確定訊息的訂閱者，或者它是否僅在完成與呼叫相關聯的所有處理且可以向訂閱者報告結果之後才傳回控制。

適用物件

ConnectionFactory， TopicConnectionFactory

JMS 管理工具完整名稱: OUTCOMENOTIFICATION

JMS 管理工具簡稱: NOTIFY

以編程方式存取

setter/getter

- MQConnectionFactory.setOutcome 通知 ()
- MQConnectionFactory.getOutcome 通知 ()

值

YES
當訂閱者確認或確定訊息時，適用於 JMS 的 WebSphere MQ 類別不會將控制權交還給訂閱者，除非它已完成與呼叫相關聯的所有處理程序，並且可以向訂閱者報告結果。 這是預設值。

NO
當訂閱者確認或確定訊息時， WebSphere 適用於 JMS 的 MQ 類別會在訂閱者完成與呼叫相關聯的所有處理並可將結果報告給訂閱者之前，立即將控制權交還給訂閱者。

PERSISTENCE

傳送至目的地之訊息的持續性。

適用物件

佇列，主題

JMS 管理工具完整名稱: 持續性

JMS 管理工具簡稱 :PER

以編程方式存取

setter/getter

- MQDestination.setPersistence()
- MQDestination.getPersistence()

值

應用程式
持續性由 JMS 應用程式定義。 這是預設值。

qdef
持續性採用佇列預設值。

PERS
訊息持續存在。

非
訊息是非持續性。

高
如需使用此值的進一步相關資訊，請參閱 JMS 持續訊息 。

POLLINGINT

如果階段作業內的每一個訊息接聽器在其佇列上都沒有適當的訊息，則這是在每一個訊息接聽器再次嘗試從其佇列取得訊息之前所經歷的間隔上限 (毫秒)。 如果經常發生階段作業中的任何訊息接聽器都沒有可用的適當訊息，請考量增加此內容的值。 只有在 TRANSPORT 具有 BIND 或 CLIENT 值時，此內容才相關。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :POLLINGINT

JMS 管理工具簡稱 :PINT

以編程方式存取

setter/getter

- MQConnectionFactory.setPollingInterval ()
- MQConnectionFactory.getPollingInterval ()

值

5000
這是預設值。

任何正整數

PORT

若為佇列管理程式的連線，則為佇列管理程式接聽所在的埠號; 若為分配管理系統的即時連線，則為分配管理系統接聽即時連線所在的埠號。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :PORT

JMS 管理工具簡稱 :PORT

以編程方式存取

setter/getter

- MQConnectionFactory.setPort()
- MQConnectionFactory.getPort()

值

1414
如果 TRANSPORT 設為 CLIENT，則這是預設值。

1506
如果 TRANSPORT 設為 DIRECT 或 DIRECTHTTP，則這是預設值。

任何正整數

PRIORITY

傳送至目的地之訊息的優先順序。

適用物件

佇列，主題

JMS 管理工具完整名稱 :XX_ENCODE_CASE_ONE priority

JMS 管理工具簡稱 :PRI

以編程方式存取

setter/getter

- MQDestination.setPriority()
- MQDestination.getPriority()

值

應用程式
優先順序由 JMS 應用程式定義。 這是預設值。

qdef
優先順序採用佇列預設值。

0-9 範圍內的任何整數
從最低到最高。

PROCESSDURATION

這個內容決定訂閱者是否保證在將控制權交還給適用於 JMS 的 WebSphere MQ 類別之前，會快速處理它所收到的任何訊息。

適用物件

ConnectionFactory， TopicConnectionFactory

JMS 管理工具完整名稱 :PROCESSDURATION

JMS 管理工具簡稱 :PROCDUR

以編程方式存取

setter/getter

- MQConnectionFactory.setProcess 持續時間 ()
- MQConnectionFactory.getProcess 持續時間 ()

值

不明
訂閱者無法保證它可以多快處理它所接收的任何訊息。 這是預設值。

SHORT
訂閱者保證在將控制權交還給適用於 JMS 的 WebSphere MQ 類別之前，會快速處理它所收到的任何訊息。

PROVIDERVERSION

這個內容會區分兩個 WebSphere MQ 傳訊作業模式: WebSphere MQ 傳訊提供者標準模式和 WebSphere MQ 傳訊提供者移轉模式。

WebSphere MQ 傳訊提供者標準模式會使用 WebSphere MQ 7.0 佇列管理程式的所有特性來實作 JMS。 此模式僅用於連接至 WebSphere MQ 佇列管理程式，並且可以在用戶端或連結模式下連接至 WebSphere MQ 7.0 版佇列管理程式。 此模式已最佳化為使用新的 WebSphere MQ 7.0 功能。 如果您不是使用 WebSphere MQ 即時傳輸，則使用的作業模式主要由 Connection Factory 的 PROVIDERVERSION 內容來決定。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnection Factory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :PROVIDERVERSION

JMS 管理工具簡稱 :PVER

以編程方式存取

setter/getter

- MQConnectionFactory.setProvider 版本 ()

- MQConnectionFactory.getProvider 版本 ()

值

您可以將 **PROVIDERVERSION** 設為可能的值: 7、 6 或 未指定。 不過， **PROVIDERVERSION** 可以是下列任何一種格式的字串:

- V.R.M.F
- V.R.M
- V.R
- V

其中 V、R、M 和 F 是大於或等於零的整數值。

7

使用 WebSphere MQ 傳訊提供者標準模式。

如果您將 PROVIDERVERSION 設為 7 ，則只能使用 WebSphere MQ 傳訊提供者標準作業模式。 如果因 Connection Factory 中的其他設定而連接的佇列管理程式不是 7.0 版佇列管理程式，則 createConnection() 方法會因異常狀況而失敗。

WebSphere MQ 傳訊提供者標準模式使用共用交談特性，可共用的交談數由伺服器連線通道上的 SHARECNV () 內容控制。 如果這個內容設為 0 ， 您無法使用 WebSphere MQ 傳訊提供者標準模式，且 createConnection() 方法會因異常狀況而失敗。

6

使用 WebSphere MQ 傳訊提供者移轉模式。

適用於 JMS 的 WebSphere MQ 類別使用 WebSphere MQ 6.0 所提供的特性和演算法。 如果您想要使用 WebSphere MQ Enterprise Transport 來連接至 WebSphere Event Broker 或 WebSphere Message Broker ， 您必須使用此模式。 您可以使用此模式連接至 WebSphere MQ 7.0 版佇列管理程式，但不會使用 7.0 版佇列管理程式的新增特性，例如先讀或串流。

未指定

這是預設值，實際文字為「未指定」。

當 Connection Factory 與新版 WebSphere MQ for JMS 類別搭配使用時，使用 JNDI 中適用於 JMS 的舊版 WebSphere MQ 類別所建立的 Connection Factory 會採用此值。 下列演算法是用來判定所使用的作業模式。 當呼叫 createConnection() 方法並使用 Connection Factory 的其他方面來判斷是否需要 WebSphere MQ 傳訊提供者標準模式或 WebSphere MQ 傳訊提供者移轉模式時，會使用這個演算法。

- 首先，嘗試使用 WebSphere MQ 傳訊提供者標準模式。
- 如果連接的佇列管理程式不是 WebSphere MQ 7.0 版，則會關閉連線，並改用 WebSphere MQ 傳訊提供者移轉模式。
- 如果伺服器連線通道上的 SHARECNV () 內容設為 0 ，則會關閉連線，並改用 WebSphere MQ 傳訊提供者移轉模式。
- 如果 BROKERVER 設為 1 或新的預設「未指定」值，則會繼續使用 WebSphere MQ 傳訊提供者標準模式，因此任何發佈/訂閱作業都會使用新的 WebSphere MQ V7.0 特性。 如果在相容模式下使用 WebSphere Event Broker 或 WebSphere Message Broker (且您想要使用 6.0 發佈/訂閱功能而非 WebSphere MQ 第 7 版發佈/訂閱功能) ， 請將 PROVIDERVERSION 設為 6 ， 以確保使用 WebSphere MQ 傳訊提供者移轉模式。

PROXYHOSTNAME

使用透過 Proxy 伺服器與分配管理系統的即時連線時， Proxy 伺服器執行所在之系統的主機名稱或 IP 位址。

適用物件

ConnectionFactory， TopicConnectionFactory

JMS 管理工具完整名稱 :PROXYHOSTNAME

JMS 管理工具簡稱 :PHOST

以編程方式存取

setter/getter

- MQConnectionFactory.setProxyHostName()
- MQConnectionFactory.getProxyHostName()

值

空值
Proxy 伺服器的主機名稱。 這是預設值。

PROXYPORT

當透過 Proxy 伺服器使用與分配管理系統的即時連線時， Proxy 伺服器正在接聽的埠號。

適用物件

ConnectionFactory， TopicConnectionFactory

JMS 管理工具完整名稱 :PROXYPORT

JMS 管理工具簡稱 :PPORT

以編程方式存取

setter/getter

MQConnectionFactory.setProxy 埠 ()

MQConnectionFactory.getProxy 埠 ()

值

443
Proxy 伺服器的埠號。 這是預設值。

PUBACKINT

在 WebSphere 適用於 JMS 的 MQ 類別向分配管理系統要求確認通知之前，發佈者已發佈的訊息數。

當您降低此內容的值時， JMS 要求的 WebSphere MQ 類別會更常確認，因此發佈者的效能會降低。 當您提高此值時，如果分配管理系統失敗， JMS 的 WebSphere MQ 類別會花費較長時間來擲出異常狀況。 只有在 TRANSPORT 具有 BIND 或 CLIENT 值時，此內容才相關。

適用物件

ConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :PROXYPORT

JMS 管理工具簡稱 :PPORT

以編程方式存取

setter/getter

MQConnectionFactory.setPubAckInterval()

MQConnectionFactory.getPubAckInterval()

值

25 GB
任何正整數都可以是預設值。

PUTASYNCALLOWED

此內容決定是否容許訊息產生者使用非同步放置，以傳送訊息至此目的地。

適用物件

佇列，主題

JMS 管理工具完整名稱: PUTASYNCALLOWED

JMS 管理工具簡稱 :PAALD

以編程方式存取

setter/getter

MQDestination.setPutAsyncAllowed()

MQDestination.getPutAsyncAllowed()

值

AS_DEST
透過參照佇列或主題定義來判定是否容許非同步放置。 這是預設值。

AS_Q_DEF
請參照佇列定義來判斷是否容許非同步放置。

AS_TOPIC_DEF
請參照主題定義來判斷是否容許非同步放置。

NO
不容許非同步放置。

YES
容許非同步放置。

QMANAGER

要連接的佇列管理程式的名稱。

不過，如果您的應用程式使用用戶端通道定義表來連接佇列管理程式，請參閱 搭配使用用戶端通道定義表與適用於 JMS 的 WebSphere MQ 類別。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、Queue、 XAConnectionFactory、 XAQueueConnection Factory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :QMANAGER

JMS 管理工具簡稱 :QMGR

以編程方式存取

setter/getter

- MQConnectionFactory.setQueueManager ()
- MQConnectionFactory.getQueueManager ()

值

"" (空字串)
任何字串都可以是預設值。

佇列

JMS 佇列目的地的名稱。 這符合佇列管理程式所使用的佇列名稱。

適用物件

佇列

JMS 管理工具完整名稱 :QUEUE

JMS 管理工具簡稱 :QU

值

任何字串
任何有效的 IBM WebSphere MQ 佇列名稱。

相關概念
IBM WebSphere MQ 物件的命名規則

READAHEADALLOWED

這個內容決定是否容許訊息消費者和佇列瀏覽器在接收非持續訊息之前，使用先讀來從這個目的地取得非持續訊息到內部緩衝區。

適用物件

佇列，主題

JMS 管理工具完整名稱 :READAheadALLOWED

JMS 管理工具簡稱 :RAALD

以編程方式存取

setter/getter

- MQDestination.setReadAheadAllowed()
- MQDestination.getReadAheadAllowed()

值

AS_DEST
請參照佇列或主題定義來決定是否容許先讀。 這是管理工具中的預設值。
在程式中使用 WMQConstants.WMQ_READ_AHEAD_ALLOWED_AS_DEST 。

AS_Q_DEF
請參照佇列定義來決定是否容許先讀。
在程式中使用 WMQConstants.WMQ_READ_AHEAD_ALLOWED_AS_Q_DEF 。

AS_TOPIC_DEF
請參閱主題定義來判斷是否容許先讀。
在程式中使用 WMQConstants.WMQ_READ_AHEAD_ALLOWED_AS_TOPIC_DEF 。

NO
不容許先讀。
在程式中使用 WMQConstants.WMQ_READ_AHEAD_ALLOWED_DISABLED 。

YES
容許先讀。

在程式中使用 WMQConstants.WMQ_READ_AHEAD_ALLOWED_ENABLED 。

READAHEADCLOSEPOLICY

對於遞送至非同步訊息接聽器的訊息，當訊息消費者關閉時，內部先讀緩衝區中的訊息會發生什麼情況。

適用物件

佇列，主題

JMS 管理工具完整名稱 :READAHHEADCLOSEPOLICY

JMS 管理工具簡稱 :RACP

以編程方式存取

setter/getter

- MQDestination.setReadAheadClosePolicy()
- MQDestination.getReadAheadClosePolicy()

值

全部遞送
在傳回之前，內部先讀緩衝區中的所有訊息都會遞送至應用程式的訊息接聽器。 這是管理工具中的預設值。

在程式中使用 WMQConstants.WMQ_READ_AHEAD_DELIVERALL 。

遞送 _ 現行
只有現行訊息接聽器呼叫在傳回之前完成，可能會將訊息留在內部先讀緩衝區中，然後捨棄這些訊息。

在程式中使用 WMQConstants.WMQ_READ_AHEAD_DELIVERCURRENT 。

RECEIVECCSID

設定佇列管理程式訊息轉換的目標 CCSID 的目的地內容。 除非 RECEIVECONVERSION 設為 `WMQ_RECEIVE_CONVERSION_QMGR` ，否則會忽略此值

適用物件

佇列，主題

JMS 管理工具完整名稱 :RECEIVECCSID

JMS 管理工具簡稱 :RCCS

以編程方式存取

Setter/Getter

- `MQDestination.setReceiveCCSID`
- `MQDestination.getReceiveCCSID`

值

WMQConstants.WMQ_RECEIVE_CCSID_JVM_DEFAULT
0 -使用 JVM `Charset.defaultCharset`

1208
UTF-8

ccsid
支援的編碼字集 ID。

RECEIVECONVERSION

此目的地內容決定佇列管理程式是否要執行資料轉換。

適用物件

佇列，主題

JMS 管理工具完整名稱 :RECEIVECONVERSION

JMS 管理工具簡稱 :RCNV

以編程方式存取

Setter/Getter

- `MQDestination.setReceiveConversion`
- `MQDestination.getReceiveConversion`

值

WMQConstants.WMQ_RECEIVE_CONVERSION_CLIENT_MSG
1 -僅在 JMS 用戶端上執行資料轉換。 預設值最多從 V7.0 開始，以及從 7.0.1.5 開始 (含)。

WMQConstants.WMQ_RECEIVE_CONVERSION_QMGR
2 -將訊息傳送至用戶端之前，在佇列管理程式上執行資料轉換。 從 V7.0 到 V7.0.1.4 (含) 的預設 (僅限) 值，除非已套用 APAR IC72897 。

RECEIVEISOLATION

此內容決定訂閱者是否可能接收訂閱者佇列上尚未確定的訊息。

適用物件

ConnectionFactory， TopicConnectionFactory

JMS 管理工具完整名稱 :RECEIVEISOLATION

JMS 管理工具簡稱 :RCVISOL

值

已確定
訂閱者只會接收訂閱者佇列上已確定的那些訊息。 這是管理工具中的預設值。

在程式中使用 WMQConstants.WMQ_RCVISOL_COMMITTED 。

未確定
訂閱者可以接收訂閱者佇列上尚未確定的訊息。

在程式中使用 WMQConstants.WMQ_RCVISOL_UNCOMMITTED 。

RECEXIT

識別通道接收結束程式，或一連串要連續執行的接收結束程式。

可能需要其他配置， IBM WebSphere MQ classes for JMS 才能找到接收結束程式。 如需相關資訊，請參閱配置適用於 JMS 的 IBM WebSphere MQ 類別以使用通道結束程式。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :RECEXIT

JMS 管理工具簡稱 :RCX

以編程方式存取

setter/getter

- MQConnectionFactory.setReceiveExit ()
- MQConnectionFactory.getReceiveExit ()

值

空值
包含一個以上以逗點區隔的項目的字串，其中每一個項目為:

- 實作 WMQReceiveExit 介面 (適用於以 Java 撰寫的通道接收結束程式) 的類別名稱。
- 格式為 *libraryName(entryPointName)* 的字串 (適用於未以 Java 撰寫的通道接收結束程式)。

這是預設值。

RECEXITINIT

當呼叫通道接收結束程式時，傳給通道接收結束程式的使用者資料。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :RECEXITINIT

JMS 管理工具簡稱 :RCXI

以編程方式存取

setter/getter

- MQConnectionFactory.setReceiveExitInit()
- MQConnectionFactory.getReceiveExitInit()

值

空值
包含一或多個使用者資料項目的字串，以逗點區隔。 這是預設值。

REPLYTOSTYLE

決定如何建構所接收訊息中的 JMSReplyTo 欄位。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: REPLYTOSTYLE

JMS 管理工具簡稱 :RTOST

以編程方式存取

setter/getter

- MQConnectionFactory.setReplyToStyle()
- MQConnectionFactory.getReplyToStyle()

值

預設值
相當於 MQMD。

RFH2
使用 RFH2 標頭中提供的值。 如果已在傳送端應用程式中設定 JMSReplyTo 值，請使用該值。

MQMD
請使用 MQMD 提供的值。 此行為相當於 WebSphere MQ 6.0.2.4 版和 6.0.2.5 版的預設行為。

如果傳送端應用程式所設定的 JMSReplyTo 值未包含佇列管理程式名稱，則接收端佇列管理程式會在 MQMD 中插入自己的名稱。 如果您將此參數設為 MQMD，則您使用的回覆目的地佇列位於接收端佇列管理程式上。 如果您將此參數設為 RFH2，則您使用的回覆目的地佇列位於傳送訊息的 RFH2 中最初由傳送應用程式設定的佇列管理程式上。

如果傳送端應用程式所設定的 JMSReplyTo 值包含佇列管理程式名稱，則此參數的值並不重要，因為 MQMD 和 RFH2 都包含相同的值。

RESCANINT

當點對點網域中的訊息消費者使用訊息選取器來選取要接收的訊息時， WebSphere MQ for JMS 類別會以佇列的 MsgDeliverySequence 屬性所決定的順序搜尋 WebSphere MQ 佇列中的適當訊息。

在適用於 JMS 的 WebSphere MQ 類別找到適當的訊息並將它遞送給消費者之後，適用於 JMS 的 WebSphere MQ 類別會從其在佇列中的現行位置回復搜尋下一個適當的訊息。 WebSphere 適用於 JMS 的 MQ 類別會以這種方式繼續搜尋佇列，直到它到達佇列結尾，或直到此內容值所決定的時間間隔 (毫秒) 過期為止。 在每一種情況下， JMS 的 WebSphere MQ 類別會回到佇列開頭以繼續搜尋，並開始新的時間間隔。

適用物件

ConnectionFactory、 QueueConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory

JMS 管理工具完整名稱 :RESCANINT

JMS 管理工具簡稱 :RINT

以編程方式存取

setter/getter

- MQConnectionFactory.setRescanInterval ()
- MQConnectionFactory.getRescanInterval ()

值

5000
任何正整數都可以是預設值。

SECEXIT

識別通道安全結束程式。

可能需要其他配置， IBM WebSphere MQ classes for JMS 才能找到安全結束程式。 如需相關資訊，請參閱配置適用於 JMS 的 IBM WebSphere MQ 類別以使用通道結束程式。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: SECEXIT

JMS 管理工具簡稱: SXC

以編程方式存取

setter/getter

- MQConnectionFactory.setSecurityExit ()
- MQConnectionFactory.getSecurityExit ()

值

空值
實作 WMQSecurityExit 介面 (適用於以 Java 撰寫的通道安全結束程式) 的類別名稱。
格式為 *libraryName(entryPointName)* 的字串 (適用於未以 Java 撰寫的通道安全結束程式)。

SECEXITINIT

通道安全結束程式在被呼叫時收到的使用者資料。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: SECEXITINIT

JMS 管理工具簡稱: SCXI

以編程方式存取

setter/getter

- MQConnectionFactory.setSecurityExitInit()
- MQConnectionFactory.getSecurityExitInit()

值

空值
任何字串都可以是預設值。

SENDCHECKCOUNT

在單一非交易式 JMS 階段作業內，檢查非同步放置錯誤之間容許的傳送呼叫數。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: SENDCHECKCOUNT

JMS 管理工具簡稱: SCC

以編程方式存取

setter/getter

- MQConnectionFactory.setSendCheckCount()
- MQConnectionFactory.getSendCheckCount()

值

空值
任何字串都可以是預設值。

SENDEXIT

識別通道傳送結束程式，或一連串要連續執行的傳送結束程式。

可能需要其他配置， IBM WebSphere MQ classes for JMS 才能找到傳送結束程式。 如需相關資訊，請參閱配置適用於 JMS 的 IBM WebSphere MQ 類別以使用通道結束程式。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: SENDEXIT

JMS 管理工具簡稱: SDX

以編程方式存取

setter/getter

- MQConnectionFactory.setSendExit ()
- MQConnectionFactory.getSendExit ()

值

空值
包含一個以上以逗點區隔之項目的任何字串，其中每一個項目都是:

- 實作 WMQSendExit 介面的類別名稱 (適用於以 Java 撰寫的通道傳送結束程式)。
- 格式為 *libraryName(entryPointName)* 的字串 (適用於未以 Java 撰寫的通道傳送結束程式)。
-

這是預設值。

SENDEXITINIT

傳送給所呼叫之通道傳送結束程式的使用者資料。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: SENDEXITINIT

JMS 管理工具簡稱: SDXI

以編程方式存取

setter/getter

- MQConnectionFactory.setSendExitInit()
- MQConnectionFactory.getSendExitInit()

值

空值
任何包含一或多個使用者資料項目 (以逗點區隔) 的字串都可以是預設值。

SHARECONVALLOWED

此內容決定如果通道定義相符，用戶端連線是否可以與從相同處理程序到相同佇列管理程式的其他最上層 JMS 連線共用其 Socket。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: SHARERELOWED

JMS 管理工具簡稱: SCALD

以編程方式存取

setter/getter

- MQConnectionFactory.setShareConvAllowed()
- MQConnectionFactory.getShareConvAllowed()

值

YES
這是管理工具的預設值。
若為程式，請使用 WMQConstants.WMQ_SHARE_CONV_ALLOWED_YES。

NO
此值適用於管理工具。
若為程式，請使用 WMQConstants.WMQ_SHARE_CONV_ALLOWED_NO。

SPARSESUBS

控制 TopicSubscriber 物件的訊息擷取原則。

適用物件

ConnectionFactory， TopicConnectionFactory

JMS 管理工具完整名稱: SPARSESUBS

JMS 管理工具簡稱: SSUBS

以編程方式存取

setter/getter

- MQConnectionFactory.setSparse 訂閱 ()
- MQConnectionFactory.getSparse 訂閱 ()

值

NO
訂閱會接收經常相符的訊息。 這是管理工具的預設值。

對於程式，請使用 false。

YES
訂閱接收不頻繁的相符訊息。 此值需要可以開啟訂閱佇列以進行瀏覽。

對於程式，請使用 true。

SSLCIPHERSUITE

用於 SSL 連線的 CipherSuite 。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: SSLCIPHERSUITE

JMS 管理工具簡稱: SCPHS

以編程方式存取

setter/getter

- MQConnectionFactory.setSSLCipherSuite ()
- MQConnectionFactory.getSSLCipherSuite ()

值

空值
這是預設值。 如需相關資訊，請參閱 JMS 物件的 SSL 內容。

SSLCRL

用來檢查 SSL 憑證撤銷的 CRL 伺服器。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: SSLCRL

JMS 管理工具簡稱: SCRL

以編程方式存取

setter/getter

- MQConnectionFactory.setSSLCertStores ()
- MQConnectionFactory.getSSLCert 商店 ()

值

空值
以空格區隔的 LDAP URL 清單。 這是預設值。 如需相關資訊，請參閱 JMS 物件的 SSL 內容。

SSLFIPSREQUIRED

此內容決定 SSL 連線是否必須使用 IBM Java JSSE FIPS 提供者 (IBMJSSEFIPS) 支援的 CipherSuite 。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: SSLFIPSREQUIRED

JMS 管理工具簡稱: SFIPS

以編程方式存取

setter/getter

- MQConnectionFactory.setSSLFips 必要 ()
- MQConnectionFactory.getSSLFips 必要 ()

值

NO
SSL 連線可以使用 IBM Java JSSE FIPS 提供者 (IBMJSSEFIPS) 不支援的任何 CipherSuite 。
這是預設值。 在程式中，使用 false。

YES
SSL 連線必須使用 IBMJSSEFIPS 支援的 CipherSuite 。
在程式中，使用 true。

SSLPEERNAME

對於 SSL， *distinguished name* 架構必須符合佇列管理程式所提供的架構。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: SSLPEERNAME

JMS 管理工具簡稱: SPEER

以編程方式存取

setter/getter

- MQConnectionFactory.setSSLPeer 名稱 ()
- MQConnectionFactory.getSSLPeer 名稱 ()

值

空值
這是預設值。 如需相關資訊，請參閱 JMS 物件的 SSL 內容。

SSLRESETCOUNT

若為 SSL，在重新協議用於加密的秘密金鑰之前，連線所傳送及接收的位元組總數。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: SSLRESETCOUNT

JMS 管理工具簡稱: SRC

以編程方式存取

setter/getter

- MQConnectionFactory.setSSLReset 計數 ()
- MQConnectionFactory.getSSLReset 計數 ()

值

0
零，或任何小於或等於 999、999、999 的正整數。 這是預設值。 如需相關資訊，請參閱 JMS 物件的 SSL 內容。

STATREFRESHINT

長時間執行交易的重新整理間隔 (毫秒) ， 會偵測訂閱者何時失去與佇列管理程式的連線。

僅當 SUBSTORE 具有值 QUEUE 時，此內容才相關。

適用物件

ConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: STATREFRESHINT

JMS 管理工具簡稱: SRI

以編程方式存取

setter/getter

- MQConnectionFactory.setStatusRefreshInterval()
- MQConnectionFactory.getStatusRefreshInterval()

值

6000
任何正整數都可以是預設值。 如需相關資訊，請參閱 JMS 物件的 SSL 內容。

SUBSTORE

其中 WebSphere MQ classes for JMS 儲存與作用中訂閱相關的持續資料。

適用物件

ConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: SUBSTORE

JMS 管理工具簡稱: SS

以編程方式存取

setter/getter

- MQConnectionFactory.setSubscriptionStore ()
- MQConnectionFactory.getSubscriptionStore ()

值

分配管理系統
使用分配管理系統型訂閱儲存庫來保留訂閱的詳細資料。 這是管理工具的預設值。

若為程式，請使用 WMQConstants.WMQ_SUBSTORE_BROKER。

移轉
將訂閱資訊從佇列型訂閱儲存庫傳送至分配管理系統型訂閱儲存庫。

若為程式，請使用 WMQConstants.WMQ_SUBSTORE_MIGRATE。

佇列
使用佇列型訂閱儲存庫來保留訂閱的詳細資料。

若為程式，請使用 WMQConstants.WMQ_SUBSTORE_QUEUE。

SYNCPOINTALLGETS

此內容決定是否在同步點下執行所有取得。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :SYNCPOINTALLGETS

JMS 管理工具簡稱: SPAG

以編程方式存取

setter/getter

- MQConnectionFactory.setSyncpointAllGets()
- MQConnectionFactory.getSyncpointAllGets()

值

否
這是預設值。

是

TARGCLIENT

此內容決定是否使用 WebSphere MQ RFH2 格式與目標應用程式交換資訊。

適用物件

佇列，主題

JMS 管理工具完整名稱: TARGCLIENT

JMS 管理工具簡稱 :TC

以編程方式存取

setter/getter

- MQDestination.setTargetClient()

- MQDestination.getTargetClient()

值

JMS
訊息的目標是 JMS 應用程式。 這是管理工具的預設值。

若為程式，請使用 WMQConstants.WMQ_CLIENT_JMS_COMPLIANT。

MQ
訊息的目標是非 JMS WebSphere MQ 應用程式。

若為程式，請使用 WMQConstants.WMQ_CLIENT_NONJMS_MQ。

TARGCLIENTMATCHING

此內容決定傳送至送入訊息的 JMSReplyTo 標頭欄位所識別佇列的回覆訊息，是否只有在送入訊息具有 MQRFH2 標頭時，才會有 MQRFH2 標頭。

適用物件

ConnectionFactory、 QueueConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory

JMS 管理工具完整名稱: TARGCLIENTMATCHING

JMS 管理工具簡稱 :TCM

以編程方式存取

setter/getter

- MQConnectionFactory.setTargetClientMatching()
- MQConnectionFactory.getTargetClientMatching()

值

YES
如果送入訊息沒有 MQRFH2 標頭，則從訊息的 JMSReplyTo 標頭欄位衍生之「佇列」物件的 TARGCLIENT 內容會傳送至 MQ。 如果訊息具有 MQRFH2 標頭，則會改為將 TARGCLIENT 內容設為 JMS。 這是管理工具的預設值。

對於程式，請使用 true。

NO
從送入訊息的 JMSReplyTo 標頭欄位衍生之「佇列」物件的 TARGCLIENT 內容一律設為 JMS。

對於程式，請使用 false。

TEMPMODEL

從中建立 JMS 暫時佇列的模型佇列名稱。

適用物件

ConnectionFactory、 QueueConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory

JMS 管理工具完整名稱: TEMPMODEL

JMS 管理工具簡稱 :TM

以編程方式存取

setter/getter

- MQConnectionFactory.setTemporaryModel ()

- MQConnectionFactory.getTemporary 模型 ()

值

SYSTEM.DEFAULT.MODEL.QUEUE
任何字串都可以是預設值。

TEMPQPREFIX

用來形成 WebSphere MQ 動態佇列名稱的字首。

適用物件

ConnectionFactory、 QueueConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory

JMS 管理工具完整名稱: TEMPQPREFIX

JMS 管理工具簡稱: TQP

以編程方式存取

setter/getter

- MQConnectionFactory.setTempQPrefix ()
- MQConnectionFactory.getTempQPrefix ()

值

'' '' (空字串)
所使用的字首是 CSQ.* (在 z/OS 上) 及 AMQ.* (在所有其他平台上)。 這些是預設值。

佇列字首
佇列字首是符合在 WebSphere MQ 物件描述子 (結構 MQOD) 中形成 *DynamicQName* 欄位內容之規則的任何字串，但最後一個非空白字元必須是星號。

TEMPTOPICPREFIX

建立暫時主題時， JMS 會產生 '' TEMP/*TEMPTOPICPREFIX/unique_id*'' 格式的主題字串，或者如果此內容保留預設值，則只會產生 '' TEMP/*unique_id*''。 指定非空的 TEMPTICPREFIX 可讓您定義特定的模型佇列，以便為訂閱者建立在這個連線之下所建立之暫時主題的受管理佇列。

適用物件

ConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱 :TEMPTOPICPREFIX

JMS 管理工具簡稱: TTP

以編程方式存取

setter/getter

- MQConnectionFactory.setTempTopicPrefix()
- MQConnectionFactory.getTempTopicPrefix()

值

任何非空值字串，只包含 WebSphere MQ 主題字串的有效字元。 預設值為 '' '' (空字串)。

TOPIC

JMS 主題目的地的名稱，佇列管理程式會使用此值作為發佈或訂閱的主題字串。

適用物件

主題

JMS 管理工具完整名稱: TOPIC

JMS 管理工具簡稱 :TOP

值

任何字串
形成有效 IBM WebSphere MQ 主題字串的字串。 當使用 IBM WebSphere MQ 作為 WebSphere Application Server 的傳訊提供者時，請指定符合名稱的值，以在 WebSphere Application Server 內用來識別主題以進行管理。

相關概念
主題字串

TRANSPORT

佇列管理程式或分配管理系統的連線本質。

適用物件

ConnectionFactory、 QueueConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XAQueueConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: TRANSPORT

JMS 管理工具簡稱: TRAN

以編程方式存取

setter/getter

- MQConnectionFactory.setTransport 類型 ()
- MQConnectionFactory.getTransport 類型 ()

值

BIND
對於以連結模式連接至佇列管理程式的連線。 這是管理工具的預設值。
若為程式，請使用 WMQConstants.WMQ_CM_BINDINGS。

用戶端
對於用戶端模式下的佇列管理程式連線。
若為程式，請使用 WMQConstants.WMQ_CM_CLIENT。

直接
對於不使用 HTTP 通道的分配管理系統即時連線。
若為程式，請使用 WMQConstants.WMQ_CM_DIRECT_TCPIP。

DIRECTHTTP
針對使用 HTTP 通道的分配管理系統即時連線。 僅支援 HTTP 1.0 。
若為程式，請使用 WMQConstants.WMQ_CM_DIRECT_HTTP。

WILDCARDFORMAT

此內容決定要使用的萬用字元語法版本。

適用物件

ConnectionFactory、 TopicConnectionFactory、 XAConnectionFactory、 XATopicConnectionFactory

JMS 管理工具完整名稱: WILDCARDFORMAT

JMS 管理工具簡稱 :WCFMT

以編程方式存取

setter/getter

- MQConnectionFactory.setWildCardFormat()
- MQConnectionFactory.getWildCardFormat()

值

TOPIC_ONLY
僅辨識分配管理系統第 2 版中使用的主題層次萬用字元。 這是管理工具的預設值。
若為程式，請使用 WMQConstants.WMQ_WILDCARD_TOPIC_ONLY。

CHAR_ONLY
只能辨識分配管理系統第 1 版中使用的字元萬用字元。
若為程式，請使用 WMQConstants.WMQ_WILDCARD_CHAR_ONLY。

JMS 物件的 WebSphere MQ 類別內容之間的相依關係

部分內容的有效性取決於其他內容的特定值。

此相依關係可以在下列內容群組中發生:

- 用戶端內容
- 與分配管理系統之即時連線的內容
- 結束起始設定字串

用戶端內容
對於佇列管理程式的連線，僅當 TRANSPORT 具有值 CLIENT 時，下列內容才相關:

- HOSTNAME
- PORT
- CHANNEL
- LOCALADDRESS
- CCDTURL
- CCSID
- COMPHDR
- COMPMSG
- RECEXIT
- RECEXITINIT
- SECEXIT
- SECEXITINIT
- SENDEXIT

- SENDEXITINIT
- SHARECONVALLOWED
- SSLCIPHERSUITE
- SSLCRL
- SSLFIPSREQUIRED
- SSLPEERNAME
- SSLRESETCOUNT
- APPLICATIONNAME

如果 TRANSPORT 具有 BIND 值，則無法使用管理工具來設定這些內容的值。

如果 TRANSPORT 具有值 CLIENT，則 BROKERVER 內容的預設值為 V1，且 PORT 內容的預設值為 1414。 如果您明確設定 BROKERVER 或 PORT 的值，則稍後對 TRANSPORT 值所做的變更不會置換您的選擇。

與分配管理系統之即時連線的內容
如果 TRANSPORT 值為 DIRECT 或 DIRECTHTTP，則只有下列內容相關:

- BROKERVER
- CLIENTID
- 說明
- DIRECTAUTH
- HOSTNAME
- LOCALADDRESS
- MAXBUFFSIZE
- 多重播送 (僅 DIRECT 支援)
- PORT
- PROXYHOSTNAME (僅 DIRECT 支援)
- PROXYPORT (僅 DIRECT 支援)

如果 TRANSPORT 具有值 DIRECT 或 DIRECTTP，則 BROKERVER 內容的預設值為 V2，且 PORT 內容的預設值為 1506。 如果您明確設定 BROKERVER 或 PORT 的值，則稍後對 TRANSPORT 值所做的變更不會置換您的選擇。

結束起始設定字串
如果沒有提供對應的結束程式名稱，請勿設定任何結束程式起始設定字串。 結束程式起始設定內容如下:

- RECEXITINIT
- SECEXITINIT
- SENDEXITINIT

例如，指定 `RECEXITINIT(myString)` 而不指定 `RECEXIT(some.exit.classname)` 會導致錯誤。

ENCODING 內容

ENCODING 內容包含 12 個可能的組合中的 3 個子內容。

ENCODING 內容可以採用的有效值是從三個子內容建構而來:

整數編碼
正常或反向

小數編碼
正常或反向

浮點數編碼
IEEE 正常、IEEE 反轉或 z/OS

ENCODING 內容以三個字元的字串表示，語法如下:

```
{N|R}{N|R}{N|R|3}
```

在此字串中:

- N 表示正常
- R 表示已反轉
- 3 表示 z/OS
- 第一個字元代表 整數編碼
- 第二個字元代表 十進位編碼
- 第三個字元代表 浮點數編碼

這會為 ENCODING 內容提供一組 12 個可能的值。

另外還有一個值，即字串 NATIVE，可針對 Java 平台設定適當的編碼值。

下列範例顯示 ENCODING 的有效組合:

```
ENCODING(NNR)
ENCODING(NATIVE)
ENCODING(RR3)
```

JMS 物件的 SSL 內容

使用 SSLCIPHERSUITE 內容啟用 Secure Sockets Layer (SSL) 加密。 然後，您可以使用數個其他內容來變更 SSL 加密的性質。

當您指定 TRANSPORT (CLIENT) 時，可以使用 SSLCIPHERSUITE 內容來啟用 Secure Sockets Layer (SSL) 加密通訊。 將此內容設為 JSSE 提供者所提供的有效 CipherSuite ; 它必須符合 CHANNEL 內容所指定 SVRCONN 通道上指定的 CipherSpec 。

不過， CipherSpecs (在 SVRCONN 通道上指定) 和 CipherSuites (在 ConnectionFactory 物件上指定) 會使用不同的命名方法來代表相同的 SSL 加密演算法。 如果在 SSLCIPHERSUITE 內容上指定可辨識的 CipherSpec 名稱， JMSAdmin 會發出警告，並將 CipherSpec 對映至其對等 CipherSuite。 請參閱 JMS 中的 SSL CipherSpecs 和 CipherSuites ，以取得 WebSphere MQ 和 JMSAdmin 所辨識的 CipherSpecs 清單。

如果您需要連線才能使用 IBM Java JSSE FIPS 提供者 (IBMJSSEFIPS) 支援的 CipherSuite ，請將 Connection Factory 的 SSLFIPSREQUIRED 內容設為 YES。 此內容的預設值為 NO ，表示連線可以使用任何支援的 CipherSuite。 如果未設定 SSLCIPHERSUITE ，則會忽略此內容。

SSLPEERNAME 符合可在通道定義上設定之 SSLPEER 參數的格式。 它是以逗點或分號區隔的屬性名稱/值配對清單。 例如:

```
SSLPEERNAME(CN=QMGR.*, OU=IBM, OU=WEBSPHERE)
```

名稱和值集組成 識別名稱。 如需識別名稱及其與 WebSphere MQ 搭配使用的詳細資料，請參閱 安全。

給定的範例會檢查伺服器在連接時所呈現的識別憑證。 若要連線成功，憑證必須具有以 QMGR. 開頭的「通用名稱」。 且必須至少有兩個組織單位名稱，第一個名稱是 IBM ，第二個名稱是 WEBSPHERE。 檢查不區分大小寫。

如果未設定 SSLPEERNAME ，則不會執行這類檢查。 如果未設定 SSLCIPHERSUITE ，則會忽略 SSLPEERNAME。

SSLCRL 內容指定零個以上 CRL (憑證撤銷清冊) 伺服器。 使用此內容需要 Java 2 v1.4 的 JVM。 這是以空格定界的項目清單，格式如下:

```
ldap://hostname:[port]
```

選擇性地後接單一/。 如果省略 *port* ，則會採用預設 LDAP 埠 389。 在連接時，會根據指定的 CRL 伺服器來檢查伺服器所呈現的 SSL 憑證。 如需 CRL 安全的相關資訊，請參閱 安全 。

如果未設定 SSLCRL，則不會執行這類檢查。 如果未設定 SSLCIPHERSUITE，則會忽略 SSLCRL。

SSLRESETCOUNT 內容代表在重新協議用於加密的秘密金鑰之前，連線所傳送及接收的位元組總數。 傳送的位元組數是加密之前的數目，而接收的位元組數是解密之後的數目。 位元組數也包括 WebSphere MQ for JMS 類別所傳送及接收的控制資訊。

比方說，如果要配置 ConnectionFactory 物件，以用來透過啟用 SSL 的 MQI 通道建立連線，且該通道具有在傳送 4 MB 資料之後重新協議的秘密金鑰，請向 JMSAdmin 發出下列指令:

```
ALTER CF(my.cf) SSLRESETCOUNT(4194304)
```

如果 SSLRESETCOUNT 的值為零 (這是預設值)，則永不重新協議秘密金鑰。 如果未設定 SSLCIPHERSUITE，則會忽略 SSLRESETCOUNT 內容。

注意事項

本資訊係針對 IBM 在美國所提供之產品與服務所開發。

在其他國家中，IBM 可能不會提供本書中所提的各項產品、服務或功能。 請洽當地 IBM 業務代表，以取得當地目前提供的產品和服務之相關資訊。 這份文件在提及 IBM 的產品、程式或服務時，不表示或暗示只能使用 IBM 的產品、程式或服務。 只要未侵犯 IBM 的智慧財產權，任何功能相當的產品、程式或服務都可以取代 IBM 的產品、程式或服務。 不過，任何非 IBM 的產品、程式或服務，使用者必須自行負責作業的評估和驗證責任。

本文件所說明之主題內容，IBM 可能擁有其專利或專利申請案。 提供本文件不代表提供這些專利的授權。 您可以書面提出授權查詢，來函請寄到：

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

如果是有關雙位元組 (DBCS) 資訊的授權查詢，請洽詢所在國的 IBM 智慧財產部門，或書面提出授權查詢，來函請寄到：

智慧財產權授權
法務部與智慧財產權法律
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

下列段落不適用於英國，若與任何其他國家之法律條款抵觸，亦不適用於該國： International Business Machines Corporation 只依 "現況" 提供本出版品，不提供任何明示或默示之保證，其中包括且不限於不侵權、可商用性或特定目的之適用性的隱含保證。 有些地區在特定交易上，不允許排除明示或暗示的保證，因此，這項聲明不一定適合您。

這項資訊中可能會有技術上或排版印刷上的訛誤。 因此，IBM 會定期修訂；並將修訂後的內容納入新版中。 IBM 隨時會改進及/或變更本出版品所提及的產品及/或程式，不另行通知。

本資訊中任何對非 IBM 網站的敘述僅供參考，IBM 對該網站並不提供任何保證。 這些網站所提供的資料不是 IBM 本產品的資料內容，如果要使用這些網站的資料，您必須自行承擔風險。

IBM 得以各種適當的方式使用或散布由您提供的任何資訊，無需對您負責。

如果本程式的獲授權人為了 (i) 在個別建立的程式和其他程式（包括本程式）之間交換資訊，以及 (ii) 相互使用所交換的資訊，因而需要相關的資訊，請洽詢：

IBM Corporation
軟體交互作業能力協調程式，部門 49XA
3605 公路 52 N
Rochester， MN 55901
U.S.A.

在適當條款與條件之下，包括某些情況下（支付費用），或可使用此類資訊。

IBM 基於雙方之 IBM 客戶合約、IBM 國際程式授權合約或任何同等合約之條款，提供本資訊所提及的授權程式與其所有適用的授權資料。

本文件中所含的任何效能資料都是在受管制的環境下判定。 因此不同作業環境之下所得的結果，可能會有很大的差異。 有些測定已在開發階段系統上做過，不過這並不保證在一般系統上會出現相同結果。 甚至有部分的測量，是利用插補法而得的估計值， 實際結果可能有所不同。 本文件的使用者應驗證其特定環境適用的資料。

© Copyright IBM Corp. 2007, 2024

本文件所提及之非 IBM 產品資訊，取自產品的供應商，或其發佈的聲明或其他公開管道。 IBM 並未測試過這些產品，也無法確認這些非 IBM 產品的執行效能、相容性或任何對產品的其他主張是否完全無誤。 有關非 IBM 產品的性能問題應直接洽詢該產品供應商。

有關 IBM 未來方針或目的之所有聲明，僅代表 IBM 的目標與主旨，隨時可能變更或撤銷，不必另行通知。

這份資訊含有日常商業運作所用的資料和報告範例。 為了要使它們儘可能完整，範例包括個人、公司、 品牌和產品的名稱。 這些名稱全屬虛構，如與實際公司的名稱和住址雷同，純屬巧合。

著作權授權:

本資訊含有原始語言之範例應用程式，用以說明各作業平台中之程式設計技術。 您可以基於研發、使用、銷售或散布符合作業平台（撰寫範例程式的作業平台）之應用程式介面的應用程式等目的，以任何形式複製、修改及散布這些範例程式，而不必向 IBM 付費。 這些範例並未在所 有情況下完整測試。 因此，IBM 不保證或暗示這些程式的可靠性、有用性或功能。

若 貴客戶正在閱讀本項資訊的電子檔，可能不會有照片和彩色說明。

程式設計介面資訊

程式設計介面資訊 (如果有提供的話) 旨在協助您建立與此程式搭配使用的應用軟體。

本書包含預期程式設計介面的相關資訊，可讓客戶撰寫程式以取得 IBM WebSphere MQ 的服務。

不過，本資訊也可能包含診斷、修正和調整資訊。 提供診斷、修正和調整資訊，是要協助您進行應用軟體的除錯。

重要: 請勿使用此診斷、修改及調整資訊作為程式設計介面，因為它可能會變更。

商標

IBM、 IBM 標誌 ibm.com 是 IBM Corporation 在全球許多適用範圍的商標。 IBM 商標的最新清單可在 Web 的 "Copyright and trademark information"www.ibm.com/legal/copytrade.shtml 中找到。 其他產品和服務名稱，可能是 IBM 或其他公司的商標。

Microsoft 及 Windows 是 Microsoft Corporation 在美國及/或其他國家或地區的商標。

UNIX 是 The Open Group 在美國及/或其他國家/地區的註冊商標。

Linux 是 Linus Torvalds 在美國及/或其他國家或地區的註冊商標。

本產品包含 Eclipse Project (http://www.eclipse.org/) 所開發的軟體。

Java 和所有以 Java 為基礎的商標及標誌是 Oracle 及/或其子公司的商標或註冊商標。

產品編號:

(1P) P/N: